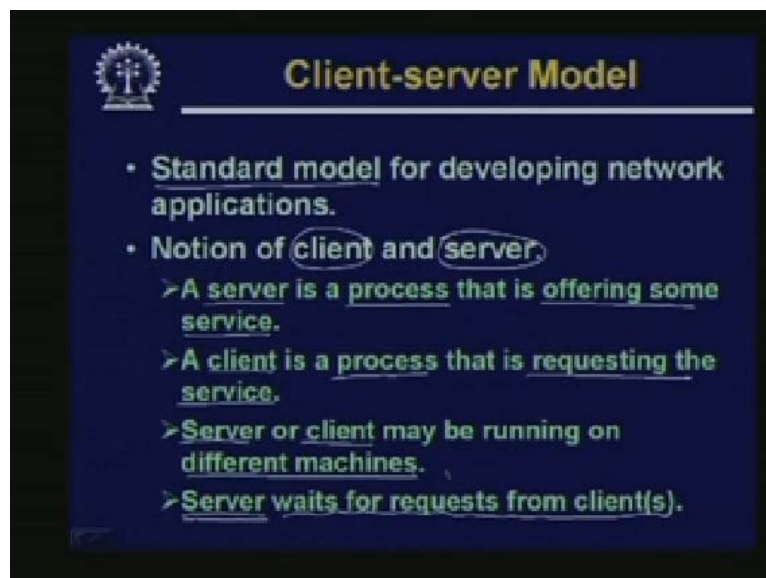


Internet Technology
Prof. Indranil Sengupta
Department of Computer Science and Engineering
Indian Institute of Technology, Kharagpur
Lecture No # 09
Client Server Concepts DNS, Telnet, FTP (01:00:04)

In this lecture I would like to touch up on a very important concept which is used widely in the internet. Nowadays you must have heard about the term client-server concept. Client server programming in fact most of the internet applications that we see today, that we use today, they are based this kind of client-server programming. So first I try to explain what this client-server programming concept is all about. Then we shall be specifically looking at some of the applications of the internet. So client server concepts and some of the applications are the topic of our discussion today.

(Refer Slide Time: 01:43)



The slide, titled "Client-server Model", features a logo in the top left corner. It contains the following text:

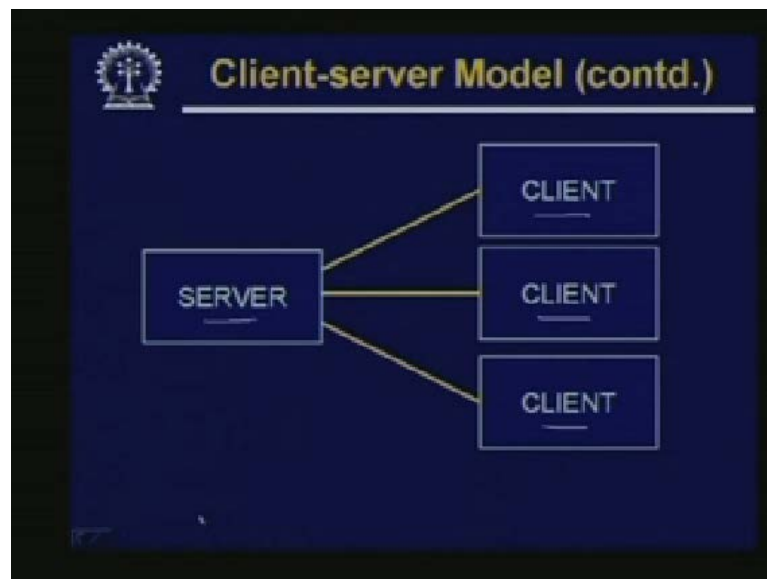
- Standard model for developing network applications.
- Notion of client and server.
 - > A server is a process that is offering some service.
 - > A client is a process that is requesting the service.
 - > Server or client may be running on different machines.
 - > Server waits for requests from client(s).

So first see what the client server model is all about. This is a standard model for developing network applications today. Now as the name implies there is the notion of a client, there is the notion of a server. The basic idea is that a server is a program or which is sometimes called a process. A program in execution is called a process. A server is a process which is continuously running and it wants to offer some services to clients. A client in contrast is another process which is requesting the service from a server. Well a simple example the server may simply return with the date and time; the current date and current time. The client can ask the server what is the date and time.

Please tell me the server can return back with the date and time. This can be a very simple example. Now in general a server can provide any kind of service and a client is somebody who is asking for that service. This is the difference between client and server and in general since we are talking about networking environment the server and client may not be running on the same machine. They may be on different computers altogether. Typically server is the program which starts first and server waits for requests from clients. So as and when the

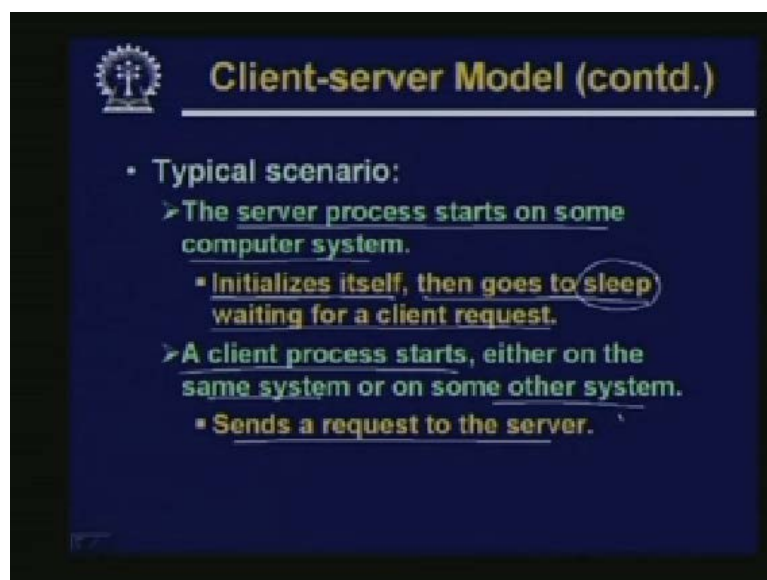
requests come from client the server will service and will again go back and wait for some more request from the clients.

(Refer Slide Time: 03:30)



So pictorially this looks like this. There can be one server; there can be several clients which need service. And we shall see later there are different ways of handling these requests. In case there are simultaneous requests coming.

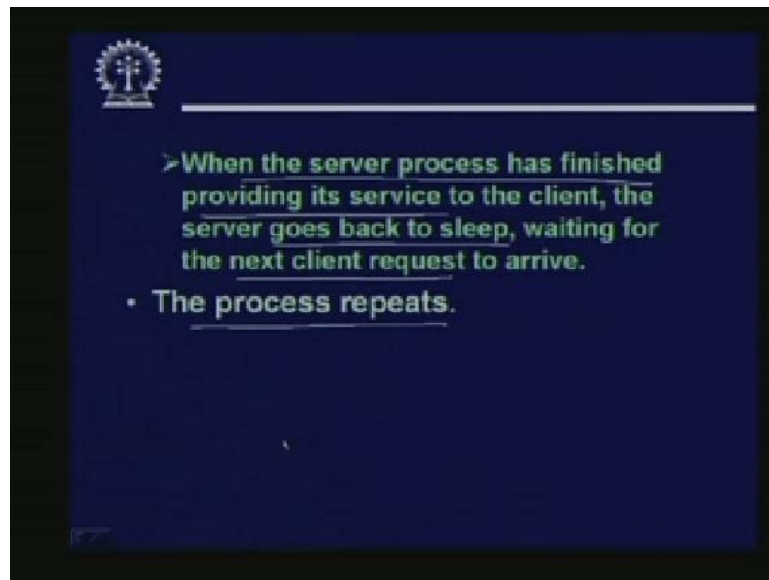
(Refer Slide Time: 03:47)



Typical scenario as I had mentioned that the server should be the first process to start. In the typical scenario, first the server process must start on some system, the server process must initialize itself and then it goes to sleep waiting for a client request. See sleep is a terminology which is used in operating system. It means that it basically goes to the waiting state. It is not executing anything right now. But it is waiting for a request from a client. If that event occurs that means a request from a client comes then the server program will wake up and it will

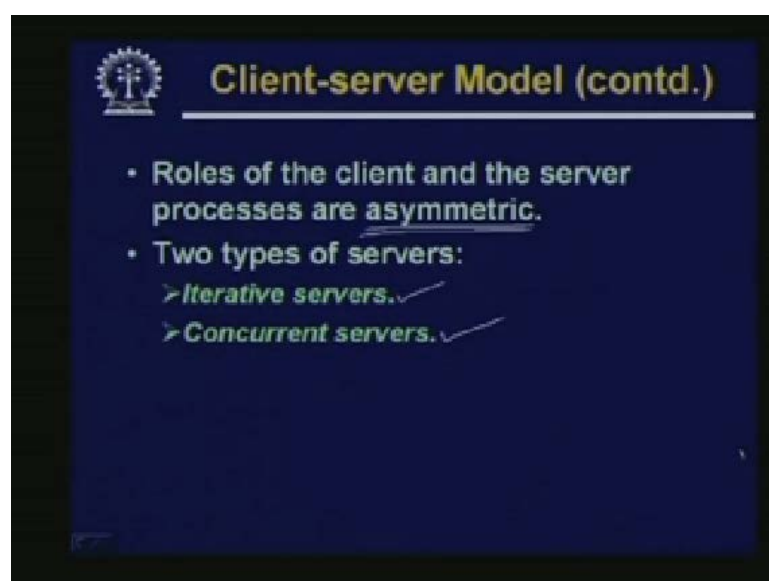
start executing. So if you remember or recall the typical state transition diagram of a process that are ready running block, then possibly the server process will be in the block state. It will be waiting for a request from the client to come in order to move into ready then a client process will start. Well it can be in the same system, it can be on some other system and it can send a request to the server.

(Refer Slide Time: 05:03)



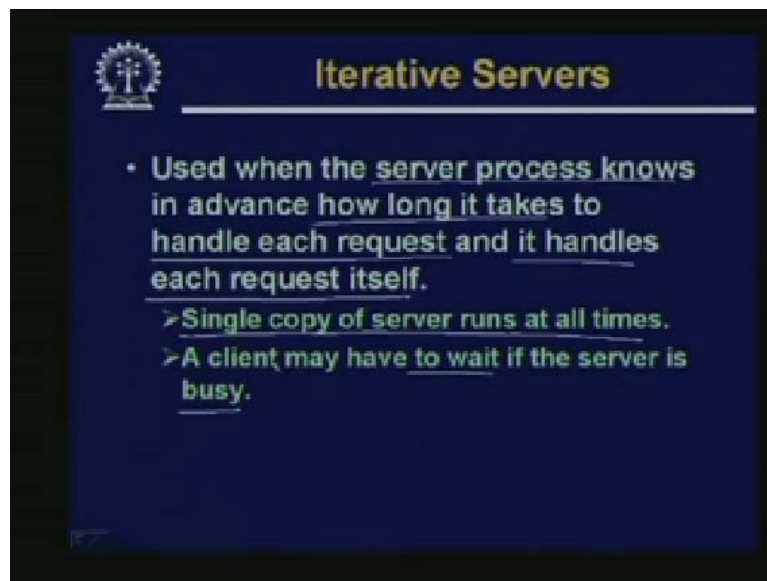
So the server will obviously process the request. So when the server has finished providing the server can go back to sleep waiting for the next client request and this process will repeat indefinitely. As long as you do not explicitly shut down the server this is a simple scenario. Server is waiting clients are sending requests, the server will be servicing the requests, it will be sending back the response and this will go on.

(Refer Slide Time: 05:38)



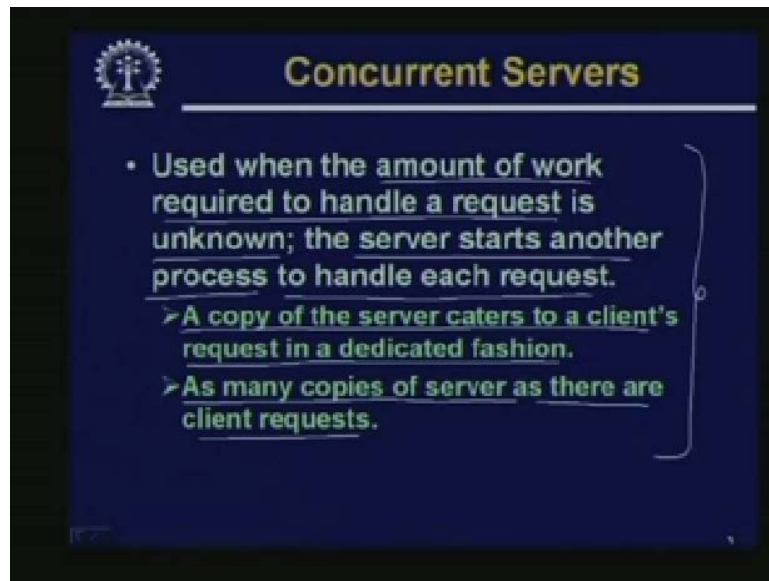
Now typically the roles of the client and server processes are asymmetric. See asymmetric in terms of roles, as well as when you write the actual client and server programs. This program you may be writing in a language like C or C++ or maybe in Java or any language. You will find that the way you write the client program, the way you write the server program, there are some places where they differ that is they are called asymmetric. They are not symmetrical if the client and server program look at exactly the same. Then they would have said to be symmetrical but they are asymmetric. Now in terms of the exact way the requests are processed by the server servers can be categorized as either Iterative or concurrent. Now let us see what these two mean.

(Refer Slide Time: 06:36)



A server is said to be Iterative when the first thing is that the server process knows in advance how long it takes to handle each request. Typically these kinds of server are used only when the service time is fairly small. If the request service time can be fairly long this is not a good solution, this is not a good alternative. Typically, iterative servers are used in cases where the service times for the client requests are fairly small in time. So single copy of the server will run at all times and the server will process one client request at a time. The server will handle each request itself. Since it is a single copy then only one request can be handled at a time. So if there are five clients which are sending requests at the same time, the first one that the server receives will be handled first the other four have to wait. As soon as the server is finished with the first is ready with the next. The next requesting queue will come and the others will wait. So in this way it will go on. So one request is being serviced or processed at a time and all other clients which have also sent request will be waiting. This is how it works. So the client may have to wait if the server is busy processing other requests.

(Refer Slide Time: 08:19)



Now in contrast, concurrent servers in a different ways. There are typically used when the amount of work required to handle a request is unknown. So you really do not know how long the client process will engage the server in providing the result of the request. Some request may take much longer time. Say I am giving a simple example, you can think of an application like a railway reservation system. Suppose whenever an operator turns on the system and connects to the server, say after three hours he shuts down the connection. Maybe for this entire 3 hour the client remains connected to the server and whatever request is going, the server is servicing immediately. So this, a long term connection or you can say request processing scenario. Here you cannot handle the request in an iterative way.

But here what is done is that whenever a client request comes, the server starts another process to handle each request. Which means that a copy of the server caters to a clients request in a dedicated fashion as many copies of server as there are client requests, see if there are, and see here this scenario is like this. There originally there was one server which was waiting for client request say client A sends a request. So what this server will do server will make a copy of itself and will allocate that copy to cater to the request of client A in a dedicated fashion and the server itself will go back and wait for another request. Now a request from client B comes the server makes a copy of it and asks that copy to service request from B and so on.

So in this way the thing will go on, if there are ten requests coming simultaneous, there will be 10 copies of the server. Now this may be seems to be a little complicated. That means how these copies are created. But if you know some of the details about process management, in operating system will know that almost in all operating systems there are some functions which you can use to create network processes. Like for example if you are using the UNIX or the Linux system there is a function called (11:03 word not audible) that is a system call. Suppose you are writing a program in C, if there is a function called (11:11 word not audible) in the C program, so whenever that is executed there is another copy of the process that gets created. So effective concurrent servers uses that Fork kind of calls to create multiple copies of itself.

(Refer Slide Time: 11:26)

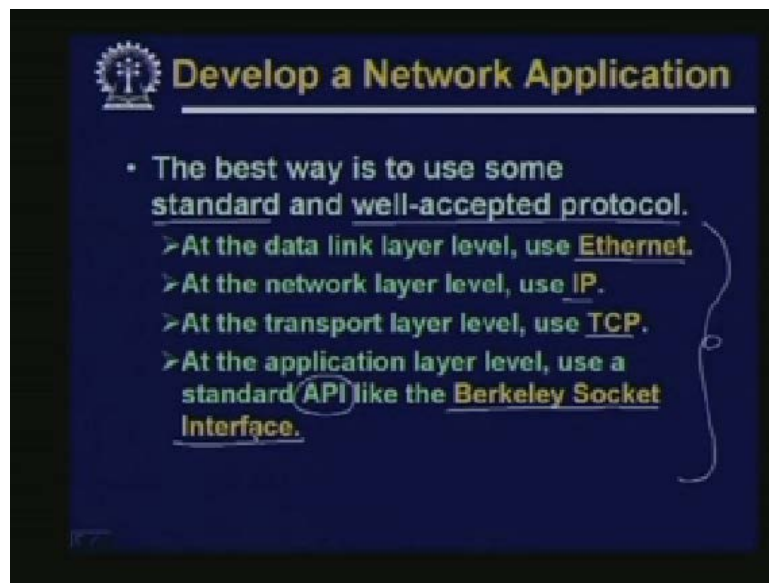
The slide is titled "Using TCP or UDP" and features a list of five components in a connection. To the right of the list is a diagram showing two boxes labeled 'S' and 'D' representing source and destination. A double-headed arrow connects the top of the boxes, and a single-headed arrow points from 'S' to 'D' at the bottom, indicating communication flow.

- Before start of communication, a connection has to be established between the two hosts.
- Five components in a connection:
 - > Protocol used ✓
 - > Source IP address ✓
 - > Source port number ✓
 - > Destination IP address ✓
 - > Destination port number ✓

Now whatever we have said is the basic concept behind client and server. Now how it actually works? Well the client server communication can use either TCP or UDP depending on your requirement. Now if it is TCP, then before start communication a connection has to establish. But if it is UDP of course you do not see when we talk about a connection, we must first understand what we mean by a connection. See suppose there are two computers on two side; one out here, one out here, this is your source and this is your destination. And you want to establish a connection between S and D, while this can be a two way communication in general. Now there are a few things you need to know about this communication, like which protocol you are using are you using, TCP are you using, UDP or something else, some new protocol.

What is the IP address of the source? What is the IP address of the destination and also as I had that in both the source and destination sides there can be multiple programs which are running. So the port number specifies exactly which program the packet is destined to or is coming from. So you can have the source port number and destination port number also. So when you have all these 5 components known to you, you know exactly what is going is on, you know exactly what protocol is being used, you know exactly which program out here is talking to which program out here. Now you can say that a connection has exactly been established actually been established between these two programs right. So these 5 components will constitute a connection.

(Refer Slide Time: 13:38)

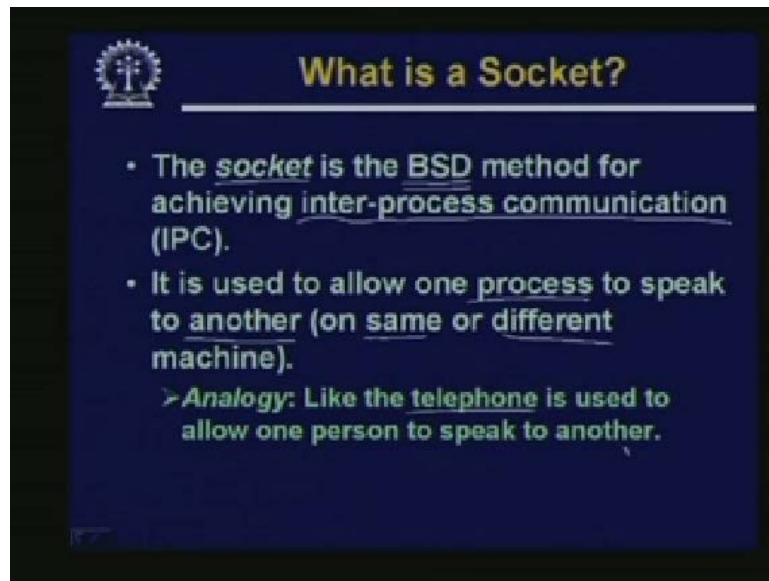


The slide features a dark blue background with a white gear icon in the top left corner. The title "Develop a Network Application" is written in a yellow, sans-serif font. Below the title, a white horizontal line separates it from a bulleted list. The list contains four items, each starting with a white bullet point. The first item is "The best way is to use some standard and well-accepted protocol." The following three items are indented and start with a white right-pointing arrow: "At the data link layer level, use Ethernet.", "At the network layer level, use IP.", and "At the transport layer level, use TCP.". The final item is "At the application layer level, use a standard API like the Berkeley Socket Interface." A white curly brace on the right side of the slide groups the last three items together.

- The best way is to use some standard and well-accepted protocol.
 - At the data link layer level, use Ethernet.
 - At the network layer level, use IP.
 - At the transport layer level, use TCP.
 - At the application layer level, use a standard API like the Berkeley Socket Interface.

Now typically when you develop a network application, it is always advisable to use some standard and well-accepted protocols. Because if you follow some you will often get support from the software tools that you use in developing the software. Now here I am showing a very typical scenario. Say whenever one wants to develop an internet application, usually at the data link layer level, you have the Ethernet protocol in most of the LAN's. We see in the network layer it has to be IP at transport layer it can be TCP. And when you are developing the application. If you are using TCP and IP then there are some standard application programming interfaces or APIs which are available one very popular API is called the so-called Berkeley Socket Interface. Now if you are using Berkeley Socket Interface, you will be having access to some functions which if you call using that you can actually establish a connection between a client and a server, you can actually send a packet, you can receive a packet, and you can do whatever you want. So you will have to understand what these functions are and how to use them. Once you know that you can very easily write such an application.

(Refer Slide Time: 15:05)

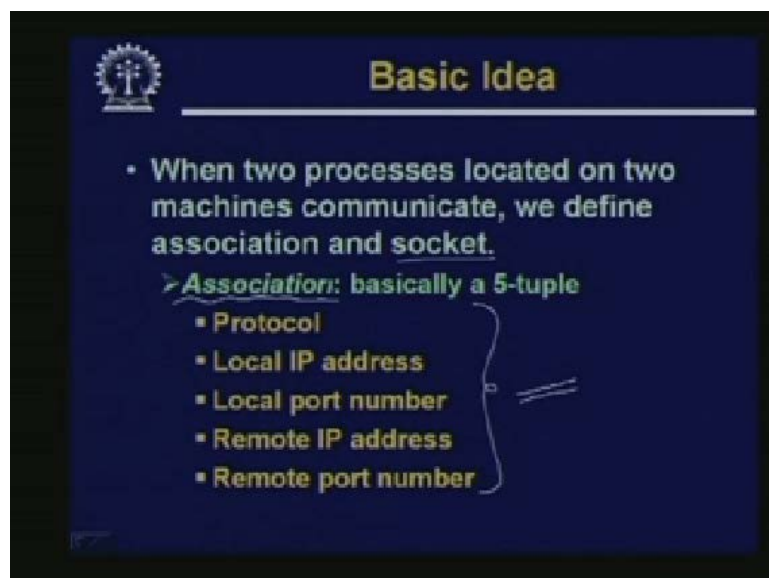


What is a Socket?

- The socket is the BSD method for achieving inter-process communication (IPC).
- It is used to allow one process to speak to another (on same or different machine).
 - > Analogy: Like the telephone is used to allow one person to speak to another.

Now the basic concept behind the Berkeley socket which is part of the Berkeley socket distribution or BSD or Berkeley standard distribution is called the socket. Socket is the basic concept behind inter-process communication over the network between client and server. Basically the concept of socket allows one process to communicate with another. Now this one process can be the client the other process can be the server. They maybe on the same machine they can be on different machines. Now analogy is like the telephone line you can establish a connection over the telephone to allow one person to talk to another. Similarly socket is an analogy the telephone connection using socket you can establish the connection then client can send something. The server receive the server can send back something the client will receive this way a two communication can start.

(Refer Slide Time: 16:13)

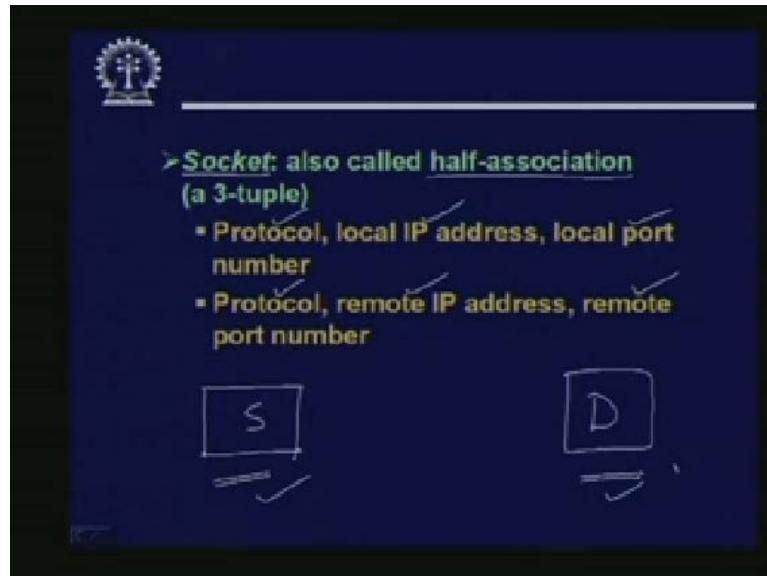


Basic Idea

- When two processes located on two machines communicate, we define association and socket.
 - > Association: basically a 5-tuple
 - Protocol
 - Local IP address
 - Local port number
 - Remote IP address
 - Remote port number

Now from the socket point of view, there is a term called full association which refers to that same 5 tuple I referred to earlier Protocol, Local IP address, Local port number, Remote IP address, Remote port number. All these 5 things taken together, is called an association.

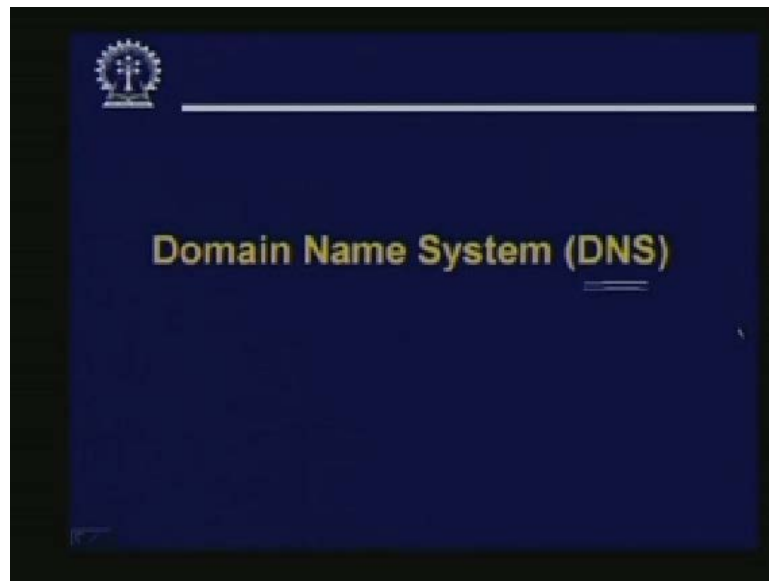
(Refer Slide Time: 16:38)



And technically speaking socket is defined as a so called half-association. Like you have the two sides; one is the source, other is the destination. Now if we have complete information about one of these sides that is called a socket. Like either if you have the protocol local IP address, local port number. That is also socket on one side on the side of the source. If you have protocol remote IP address, remote port numbers, then you have the socket on the side of destination. So the source opens a socket; the destination opens a socket, then there are some functions using which they communicate and establish a full-association among the sockets. This is how it actually works but in order to actually understand how it works you will have to look into this function in depth. And you will actually study how this client server programming can be done.

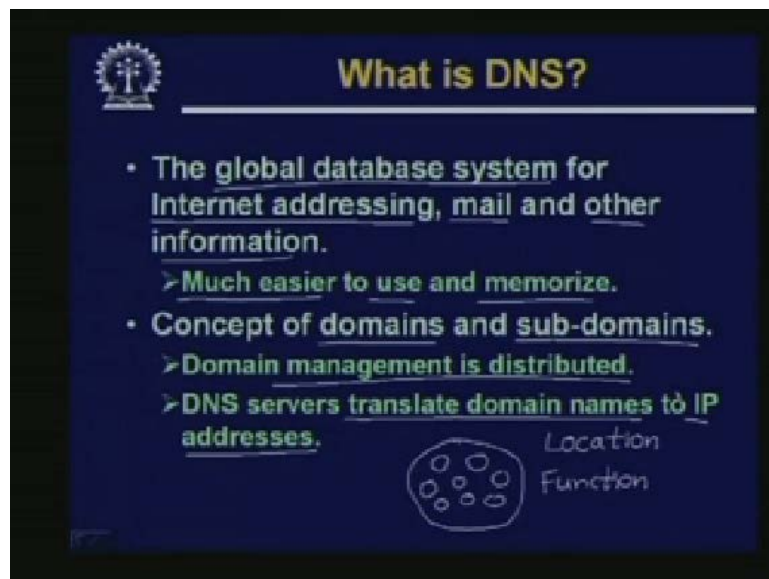
But I am not into details of the coding and details of the function because these are slightly out of the scope of the present structure. But if you are interested you can certainly into the details of the Berkeley socket interface. What kind of functions are available and how you can utilize, those functions to write client and server programs. Both in the connection oriented, that means using TCP or in the connection-less, which means the UDP environments. So my purpose of telling all these things was to give you a rough idea about what this client server concept is all about and I have told you all the internet applications that you see they are all based on the client server concept. Now I will start with one of the most important applications which will in itself is not an independent application. But is used by many other practical applications, that is, the so called Domain Name System or DNS.

(Refer Slide Time: 18:56)



This is you may have heard, this is a very important service which is provided in the internet in LAN's in different areas.

(Refer Slide Time: 19:01)



Basically the domain name system is a global database system for internet addressing mail. Other see, here this need not be restricted to internet and mail it can refer to World Wide Web, Telnet, FTP, any kind of application you use today. You typically use a name you do not always use an address. The purpose of the DNS essentially is to convert from a name into an address. So this is how it works. Now the advantage of using DNS is that see the using names rather than addresses is much easier to use easier to memorize. Like for example I am giving you, one practical I can say example, say the mail server of the computer science and engineering department of IIT Kharagpur. That can be referred to by a name cse, the full name is cse.iitkgp.ernet.in. Well you are familiar with names of this type. Now this kind of a

name is easier to memorize, but I also know that the IP address of the cse machine is 144.16.192.57.

Of course you cannot remember many such addresses with numbers you may tend to forget. But using names is much more easier because it is more English like more easy to memorize. Now in the domain name system there is a concept domains and sub domains actually domain refers to a set of computers which are related in some way. There are some computers which are related either by their location. Location means I am talking about geographical location, they may be I can say all computers in India. All computers in IIT Kharagpur, so these are some examples. These are some categorization of sub domains based on their location and I can also specify these computers based on the way they are used or their function. Like I can say that well all computers which are in the educational institutions all computers which refer to commercial organizations. These are example of distinction from the functional point of view.

So as I had mentioned that DNS servers basically translate domain names to IP addresses. This is the basic task and this thing is done in a distributed system because of this simple reason. Because one server in one place cannot have information about all the domain names in the whole world. There are millions and millions of domain names and the corresponding addresses you cannot expect that there will be one single server available in one place where you can get all the information. Rather this has to be some kind of distributed, you can say database, but information is spread across a number of different computers. So just using some systematic algorithm these computers can interact among themselves to get means whatever you can say result you want to get. Given a DNS, a name, says a name you can get an IP address by systematic query of these multiple server in a particular order. We shall see how these orders would be shortly.

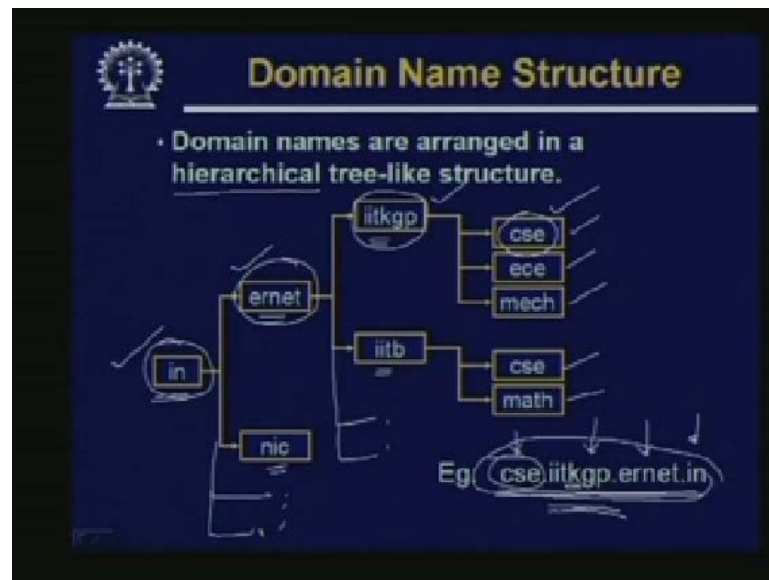
(Refer Slide Time: 23:14)



But before that let us talk about something called Top Level Domains. Well you may be aware that the addresses you write for example yahoo.com. Com say something, .gov something, .edu stanford .edu. In refers to India say ernet.in. See these are called Top Level Domain when you write an address for example when I write say www.yahoo.com, the last

part that I write that is called the Top Level Domain. Now Top Level Domain can be one of many I only shown a few out of them. Some of these are based on functional classification, com means Commercial or Commercial organization org means non-profit, net means Network server provider gov, gov traditionally stands for US government, mil Military, edu Educational. Now in addition to this there are two letter domains for each and every country of the world. For example pk stands for Pakistan, there is a code for each and every country, this in stands for India, this nz stands for New Zealand, tw stands for Taiwan and so on.

(Refer Slide Time: 24:53)



Now the domain name structure is a hierarchy I am giving an example. So I talked about the address of our mail server in the computer science and engineering department. Since address has four different parts; cse iitkgp ernet in. Well if you look from the other side from the side of in, this in is a sub domain. This in refers to all computers in India, this ernet is a sub class inside India. Now within there can be other sub classes there can be nic. There can be VSNL there can be others also. So ernet is a sub domain that is inside India. Again inside the ernet you have iitkgp. So within ernet you have iitkgp IIT Bombay you can have others as well I have not shown. All I have shown only two, then inside iitkgp you can have computer science engineering, electronics communication engineering, mechanical engineering.

Then within IIT Bombay you can have again have computer science engineering, math and so on. So you can see that we you give a domain name this is an example of a so called fully qualified domain name. Fully qualified means starting from the top level domain I am specifying the entire path of the sub domains that you should traverse to arrive at the particular place that you want to come I want to refer to cse I specify in ernet iitkgp and then cse. So these are examples of domains and sub domains. So I had mentioned that domain distribution is or domain name management is not in one palace. It is distributed for example iitkgp domain server can only keep information about the names that are under iitkgp like cse, ece, mech and so on.

Domain name system of ernet can only keep track of all the institutions that are under ernet like iitkgp IIT Bombay, IIT Delhi and other institutions. Similarly domain name India can they can keep track of all those ISP's that are inside India like ernet nic VSNL and so on. So

in the each of these tables you have only those entries that are immediately under it, there is a big domain, there are smaller sub domains under it and each of these sub domains. Again they will be having smaller sub domains under them. So each of these domains will contain information only about the sub domain that are immediately under it. This is how the domain name server information is distributed. So effectively DNS server will store the domain name and IP address they will store information only about the computers that are immediately under them.

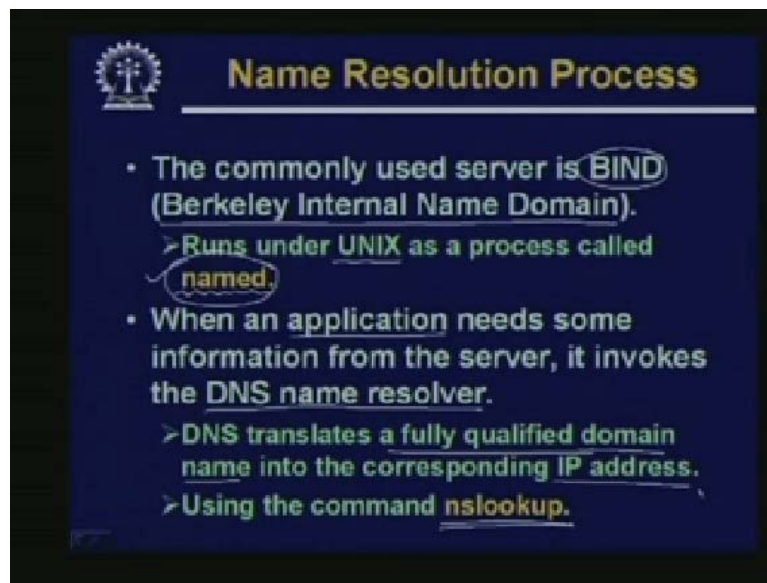
(Refer Slide Time: 28:10)



Year	COM	EDU	GOV	MIL	NET	ORG
1991	200	250	75	20	5	50
1992	350	450	100	20	10	50
1993	700	700	150	30	25	75
1994	1,200	1,075	250	100	125	175
1995	2,400	1,700	300	250	700	250

Just have a quick look at statistics. This is only up to the year 1995 you can just imagine how it has grown since. These are the figures in 1000s, the number of computers in the dot com domain in 1991 it was about 2 lakhs. In 1995 it is “2.4” million. Similarly educational institutions from 250 1000. It has gone to about “1.7” million. So you can just imagine the rate in which the number of computers are growing. So the number of such domain sub domain and these names are growing.

(Refer Slide Time: 28:55)



Now DNS server converts an IP address it converts domain name into an IP address. Let us see how this is actually done and how this domain name servers are called or where they are kept. Now the most common domain name server is Berkeley Internal Name Domain in short it is called BIND, B I N D. For instance if you are using a UNIX or a Linux system you will find that there is a process whose name is named the Berkeley Internal Name. Domain server runs as a process called named. This is the actually name server process. When an application it can be telnet, ftp, web mail, anything when it wants to get some information from the server. It can invoke the so called DNS name resolver. See the BIND is already running, that is the server program. The client program is separate the client program will be using to be used to send a request to the server program. So there is one popular client program. This is called nslookup.

So if you run nslookup on any system, it can be a Window system it can be Linux system. Whatever so that if you run this nslookup then what will happen? It will ask you to type in the domain name well it has to be the fully qualified domain name starting from the top level. And this nslookup will be actually contacting the name server and it will get translated into the IP address and the IP address will be returned back to you. So either in the UNIX or Linux scenario or in the Windows environment wherever if you give a command nslookup, say if you are using the Windows environment you give it, from that run prompt or through a command prompt. Then we will find that it will be asking you to type a domain name. You type a domain name www.yahoo.com return you will find that after some time it will return you back with the IP address of that yahoo.com. This is how it works.

(Refer Slide Time: 31:31)

> If the name server does not have the information locally, it asks its primary server, and so on.

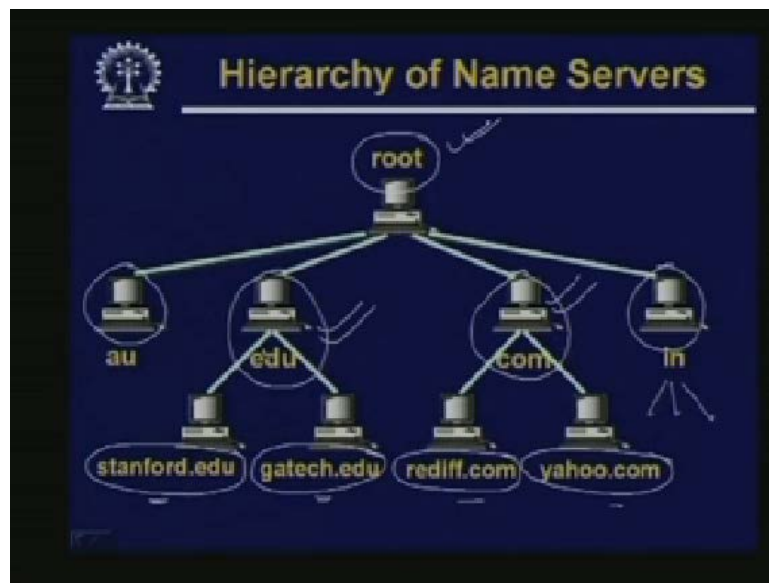
> For redundancy, each host may also have one or more secondary name servers which may be queried when the primary fails.

- How do name servers update themselves?

But as I had mentioned that the name server that is running, it cannot have all the information. Well and in general it will not have the information locally. In that case every name server has a connection with a so called primary name server which means if you have a name server out here this is your name server. If it does not have information then it will contact a primary name server. So you can configure that which will be the primary name server for your case and in case the primary name server is not available. You can also specify some secondary name server, some other name server also specify secondary name server. So if the primary is not accessible you can contact the secondary name server. But the point to note is that well whether you contact the primary or secondary that is not important. The important thing is that information is available in one place you have to look into other places if you do not have the information.

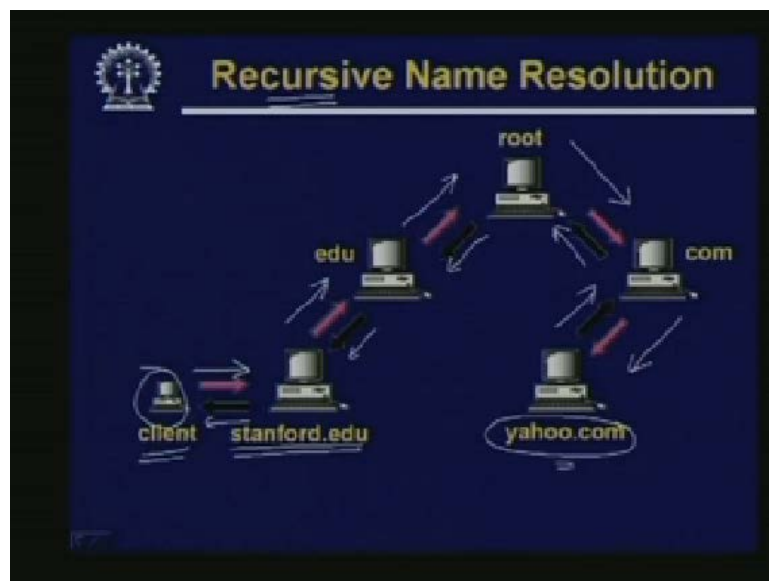
For example in your primary name server I am giving an example in your institute. There is one primary name server available. So in all our computers I have we have a set means I have also set it in my computer set the primary name server to be that particular machine. So whenever we type in some name, some domain name, that domain name gets transferred to that primary name server with a request for the IP address. If it available in its local table it immediately sends back. Otherwise it sends back to its primary name server. Because if you look at that primary name server that is also not knowing everything. So with respect to that particular machine there will be some other primary name server who, it will ask if it does not have the information there. So everyone will be having a link to some other name server till you reach the absolute top level name server. From there it can again come down, so I will give a small example.

(Refer Slide Time: 33:50)



This diagram gives you the so called hierarchy of name server. See at the top you have a virtual root level name server say I have a taken some example. I have taken two educational institutions Stanford and Georgia tech and two commercial sites rediff and yahoo, stanford and gatech will be connected to a name server under edu. They will be connected with a name server under com, if you have Indian companies they will be under the India in name server. If you have Australian companies they will be having under au so on. So this is how name servers are organised. This root will contain information about all the top domains. This edu will contain information about all the universities under this edu. This com will contain all the dot com companies. This is how the domain name information are distributed.

(Refer Slide Time: 35:00)



Now let us see how this name resolution takes place. Well here let us take an example. Suppose here we have a client and we want to get the IP address of yahoo.com say. And the client is a machine which is connected to the Stanford.edu site. So the client will first send a

request to the Stanford.edu name server. This Stanford.edu name server will find that the information is not available with Stanford.edu. So it will be sending in sending the request up to the edu name server. Well edu will also find that well yahoo.com is not in my table. So this edu web server will also send it one level up to the root name server. So root name server will tell well yes I know what com is, com is somewhere here. So it will forward the request to the com name server. Finally com name server will know well yes I know yahoo.com; yahoo dot com is here. So from here you can get back the actual IP address. Once the IP address is obtained this is the so called recursive name resolution the respond that means the actual IP address which we have found out.

This will be following the exact reverse path with respect to the original request. So you see that so many message exchanges are required to get back the IP address of the name that you have requested for. This is the requested name resolution process and here say with respect to the client if you send out one request exactly one response is been sent back. So just if I am the client if I am if I am a client means, I am on a computer, I am typing some name, say www.yahoo.com. So from my computer exactly one request will go out and exactly once response will come back. So beyond that what is happening that I am unable to see that I am not seeing? So but if it is recursive name resolution then there are lot of message transmission that are going on from one name server to other. In order to resolve the name.

(Refer Slide Time: 37:33)

The slide is titled "Iterative Name Resolution" and features a Stanford University logo in the top left corner. It contains the following text:

- Client sequentially sends queries to DNS servers and receives response.
 - > If response is negative, the DNS server to query next is also returned.
 - > Unlike recursive name resolution, where only one response is finally returned back to the client.

Below the text is a diagram showing a client (represented by a square) sending queries to two DNS servers: "stanford.edu" and "edu". Arrows indicate the direction of the queries from the client to the servers.

There is another alternative, it is called iterative; iterative says that the client will sequentially send queries to DNS server. Like for example, this is was the client. The client first sends a request to the say Stanford.edu name server. Stanford.edu name server will be sending back a response saying that well I do not have the information. But you can contact edu. So it will send back another request to the edu name server. The edu name server will be sending back another request saying that well I do not have root maybe having. So it will be sending another request to the root. So in this way the client has the responsibility of sending the requests getting the sending next request and so on. So the responses that the client is getting back, in this case will contain which name server I can contact next or in the final case I have got the IP address finally.

So this iterative name server is easier to implement, that is why most of the systems they implement this iterative name server process. So the client sequentially send request. So the response is negative it is not found, then the corresponding DNS server will be sending back the, which server to query next. So unlike recursive name server where only one response is sent back to the client. Here you can have multiple requests going out and multiple responses coming into the client. Now this name resolution or this DNS protocol is central to any internet application. We now talk about some of the internet applications which maybe using DNS for their use. Two of the protocols we would be discussing as part of today's lecture.

(Refer Slide Time: 39:55)

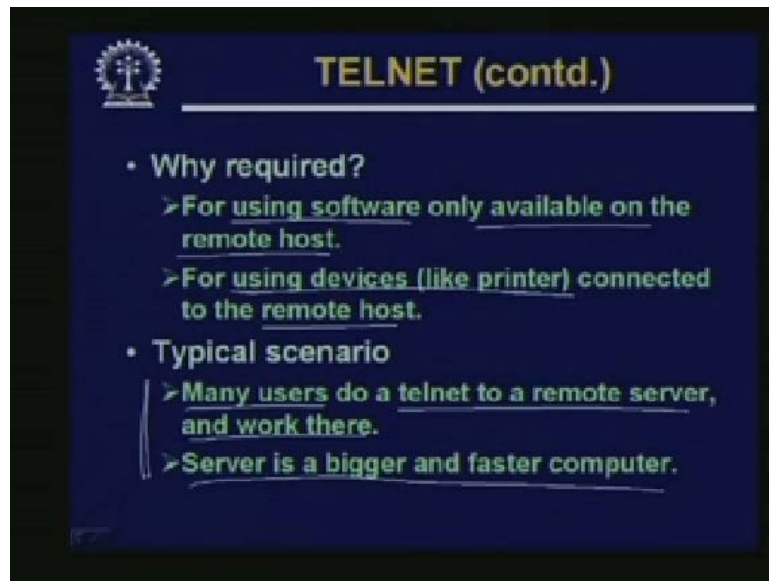
TELNET

- Allows a person sitting on one computer to work on another computer.
- Starts a remote session on another machine.
- Requires a person to supply login name and password to gain entry.
- Command:
>telnet <domain_name>
>telnet <ip_address>

The first is the so called protocol. Telnet is one of the oldest protocols that have been in use. Even before the internet took its shape this telnet command or the telnet facility was available. Subsequently telnet has become an internet standard. The idea of telnet is that it allows a person sitting on one computer to work on another computer. Like for example I am sitting on this computer A and I want to start using computer B over the network. What I do? I give a command from A the command looks like this telnet followed by the address of the computer in which I want to work. Well here I can either specify the name or if I remember the IP address I can also specify the IP address. Now once I do a telnet from A to B. Well obviously I am trying to start a remote session and usually there is some authentication where I am if I am sitting at A I will have to supply a valid login name and password corresponding to machine B to gain entry.

So sitting machine A I give a telnet B and on the screen I find there are some username password coming typing those things I gain entry into machine B. Now whatever I see is this screen corresponding to machine B, if I give a directory listing I can see the files which are there on machine B. If I run a prom if I say for example run a command like C compiler or anything if I run a command. Command is not running on my computer rather the command is running on computer B or what is happening. Whatever is the output of that command somehow that output is coming on my screen. So as if I am having an illusion that I am actually sitting on the computer B, this is a very useful facility particularly in a network environment where you do not have everything on your computer something maybe available on some other computer you can do a telnet and use it.

(Refer Slide Time: 42:26)

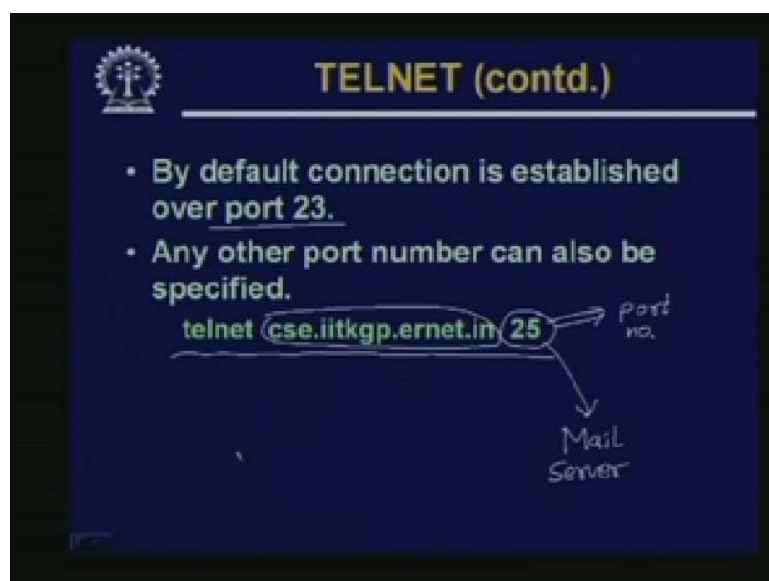


TELNET (contd.)

- Why required?
 - > For using software only available on the remote host.
 - > For using devices (like printer) connected to the remote host.
- Typical scenario
 - > Many users do a telnet to a remote server, and work there.
 - > Server is a bigger and faster computer.

So some typical applications, telnet can be used to use software which is only available on a central server. For example, another application suppose you have a printer attached to one computer. So you want taken some printout, so you can do a logarithm you can do a telnet to that remote machine, remote host and then you can do a print command. So a typical scenario in a laboratory environment is that many users who are typically students do a telnet to a remote server and work there. Server is actually a bigger and faster computer. Now the advantage of this is that instead of loading the software all these students are wanting to use on all their computers, you are loading them in only one central place. It also simplifies a lot the maintenance of the software. That means how they are using, what they are using and if you want change the software to a new version you need to this change only in one place. So telnet is a very useful command.

(Refer Slide Time: 43:32)



TELNET (contd.)

- By default connection is established over port 23.
- Any other port number can also be specified.

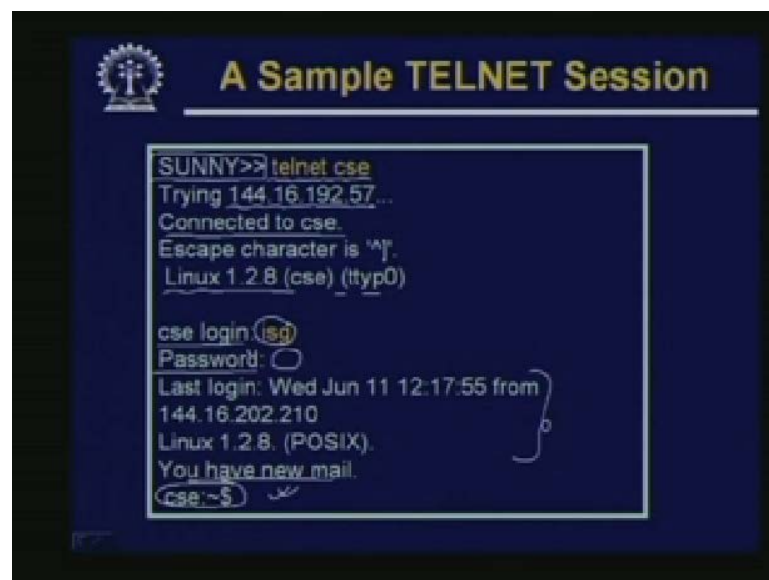
telnet cse.iitkgp.ernet.in 25

port no.

Mail Server

Now by default the telnet server runs on port number 23 which means whenever you give a command like telnet, something telnet, say some machine xyz, then a TCP connection is established to the machine TCP with the machine xyz over port number 23. 23 is a standard for telnet which is the default port number which is used. But in fact this telnet command can be used to specify any other port number as well as how like this. I can do a telnet to any machine by specifying the name. But at the end as an optional parameter I can also specify the port number. Now you will learn later that this port number 25 is actually port number which is used by the mail servers. So when I do a telnet to this machine over port number 25, what I am trying to do is that I am actually trying to contact the mail server directly. There are many cases are many application where you may want to do this. Well we shall be talking about this in more detail. When you talk about the mail email electronic mail system later there we will see that we can actually do our telnet to connect to a mail server and you can actually send and receive mails through that facility.

(Refer Slide Time: 45:20)



```
SUNNY>> telnet cse
Trying 144.16.192.57...
Connected to cse.
Escape character is '^X'.
Linux 1.2.8 (cse) (tty0)

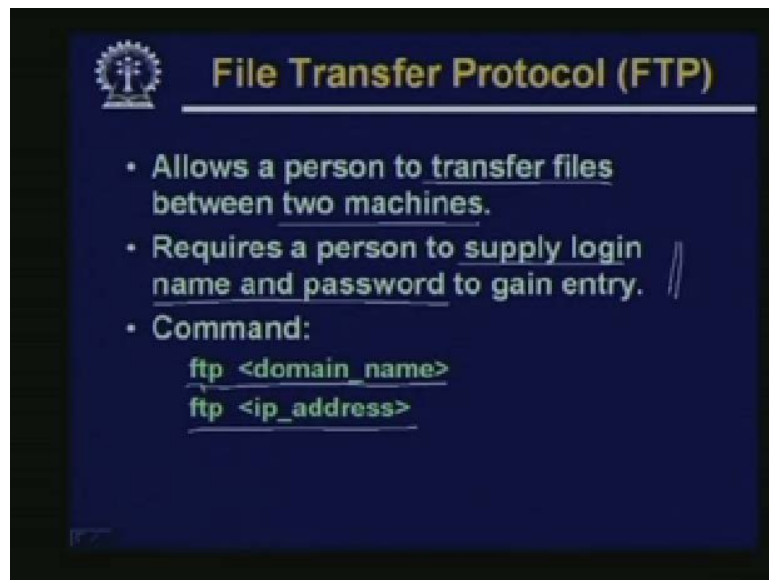
cse login (isd)
Password: ○
Last login: Wed Jun 11 12:17:55 from
144.16.202.210
Linux 1.2.8. (POSIX).
You have new mail.
cse:~$
```

Just to show you a sample telnet session, this is an example which we have taken from a Linux system. This was the original Linux prompt SUNNY. This is the command which was given telnet cse. See what this telnet cse was given, cse was a name. So automatically the domain name system was contacted and the names cse was translated into the corresponding IP address. But as a user I do not see that happening. It is happening in the background automatically. So even if I have given telnet cse, I am seeing that some IP address is coming trying to contact 144.16.192.57. This is the IP address of the cse. So finally after connection is done some message is coming connected to cse. This is the message connect which is coming from the telnet server. On the cse side some other message is coming you note these things. Some Linux version etcetera coming. Then a prompt is coming asking me to supply the login name. So I supply the login name here, a prompt is coming to supply the password.

Well I supply the password here, but this is not visible because passwords are normally not visible on screen. So if my user name and password, so if my user and password are correct some welcome messages come right here. Right here last login this Linux, this you have new mail and this is prompt, which is the prompt of this cse machine. So now I can give some commands where the response will be coming from this cse machine and not from this

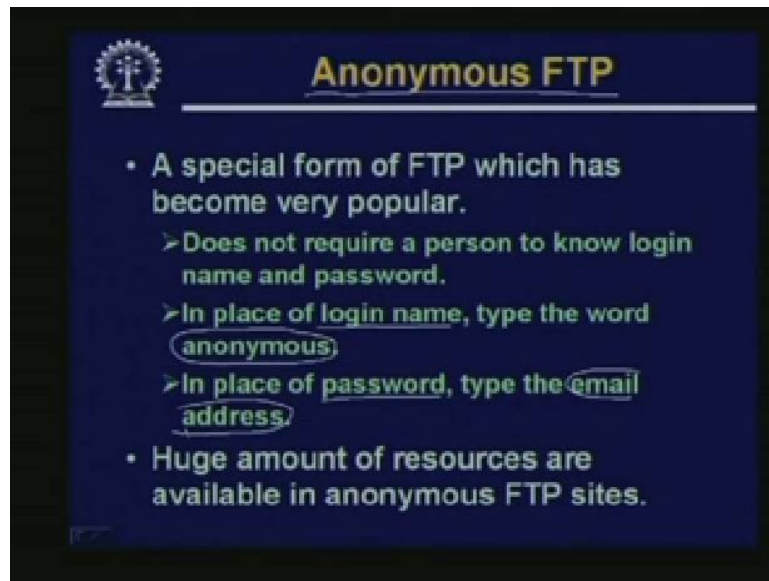
SUNNY machine. So this is just a simple example that I have shown that how we can use telnet to connect to a new or remote machine. We talk about another protocol called FTP, see using telnet we are able to logon to another system and use some facility out there. But what we cannot do in telnet is that we cannot physically transfer a file from one machine to another, we cannot bring a file from that machine to my machine or I cannot send a file from my machine to that machine. For that purpose you need to use a protocol called FTP or File Transfer Protocol. So let us see what FTP is.

(Refer Slide Time: 48:00)



Now in FTP allows a person to transfer file as I had said between two machines. Just exactly like telnet to provide authentication you need to supply login name and password. Because if this was not, so then anybody can transfer any file across any machines. That is not what you always want. There has to be some security also, some files you do not want anyone to access. Only persons who are having a valid account login name password can only access. So this kind of login name and password authentication has to be there and the command is very similar to telnet, ftp name or ftp ip address.

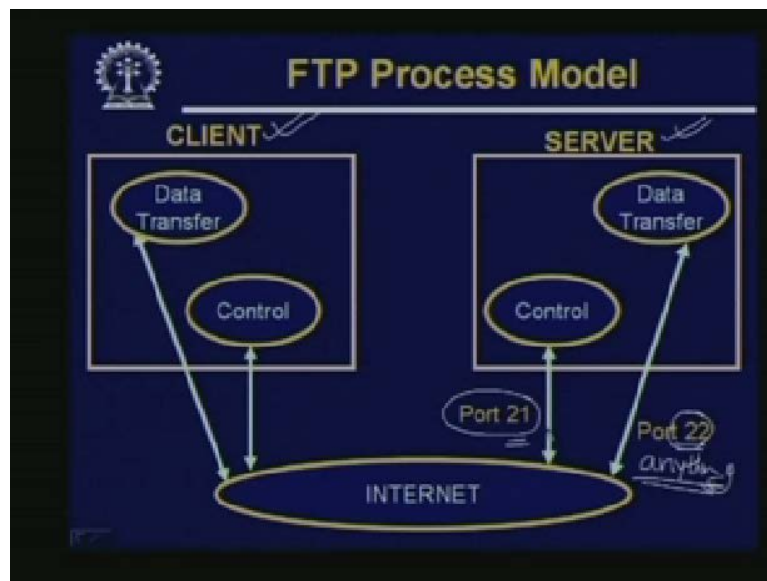
(Refer Slide Time: 48:54)



Anonymous FTP is a variation of FTP which had become very popular in the early days of the internet. Now see the conventional FTP requires the user to supply a user name and a password. But suppose on my machine I have kept a number of information useful files which many people may want. But how many people have access to my machine, they do not have a valid username and password many of them. In fact so this anonymous FTP is a mechanism using which anyone who is sitting anywhere in the world can come to my machine and take some files from me. Here the idea is very simple whenever the system asks for the login name you type. This string anonymous instead of a valid login name and when it asks for the password you type your email address.

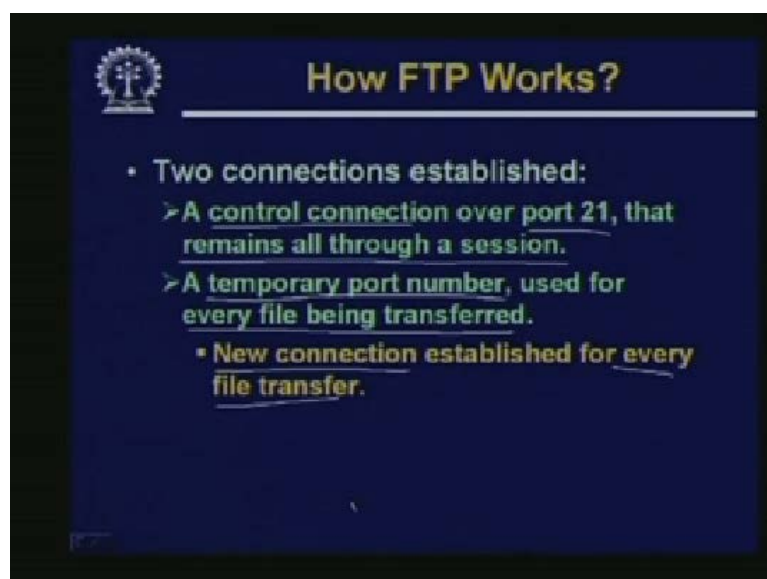
Then whenever you type the login name as anonymous and your password as your email address then you get logged in as an anonymous user in that FTP server. There may be a lot there may be a whole lot of files available for the anonymous user for downloading. So you can have a look at the files, you can list it, you can get the files that you need. In fact this anonymous FTP is in the early days of the internet was one reason that internet become so popular. There are many good peoples who wanted to share their information. They kept a lot of information on their servers and using that they allowed other people to share. So the huge amount of resources were available in anonymous FTP sites.

(Refer Slide Time: 50:51)



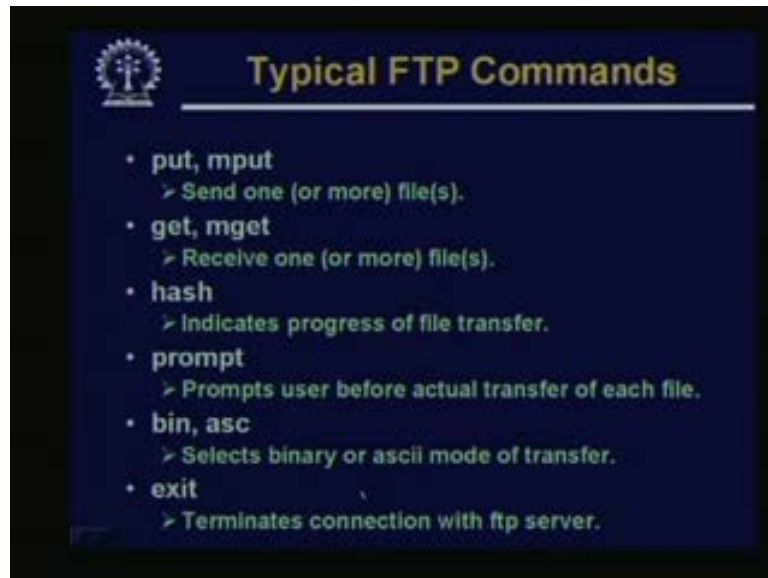
Now the FTP process model looks like this. Suppose this is a client which gives the FTP command and this is server on which the FTP server is work working. Now in fact FTP uses two port numbers; port number 21 is standard and this 22 can be in fact this can be anything. But port number 21 is must. There are two connections which are established; one is called the control connection through the control connect. Control connection is established over port number 21. So the FTP connection gets established after the control connection. Whenever you want to transfer a data, so a data connection will be established. Every time you transfer a data and that data connection can be opened over any port number. Here as an example I have shown 22. But it can 1000, 2000, 3000 anything. But this port number 21 is standard. So the idea is that if you do a telnet and if I want to transfer 5 files, there will be 5 data connections opened and closed one after the other. For every single file transfer data connection will be opened and after the transfer is complete it will again be closed.

(Refer Slide Time: 52:19)



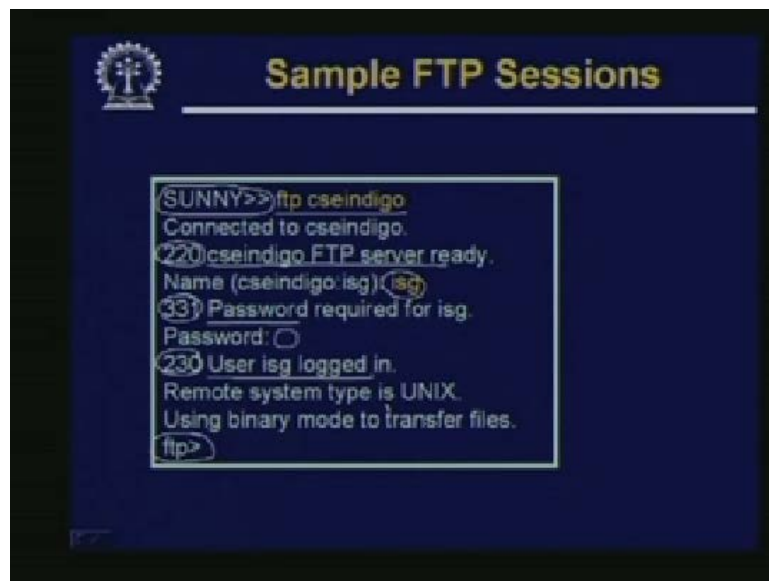
So this is what I have said, 2 connections is established. One is the control connection over port 21 and control connection remains all throughout a session until or unless you logout of that FTP. And the other port number is a temporary port number used for every file being transferred. As I have said for every file transfer you establish new connection.

(Refer Slide Time: 52:48)



And some typical FTP command, there are many commands. I have shown only a few put or mput. Put means send one file, mput means multiple put, get or mget receive a file or multiple file, hash if before put or mput. You give the hash command; you can see the progress of the file transfer some symbols are getting displayed one after the other. Now each symbol typically display shows either 4 kilo bytes or 8 kilo bytes of data transfer. So you can see the hashes coming one after the other. You can see that well my file transfer is in progress prompt. It prompt the user before the transfer whether you really want to transfer yes, no like that. It asks you whether the file you want to transfer is a binary file or a text file ASCII file, exit means it terminates exits the FTP server.

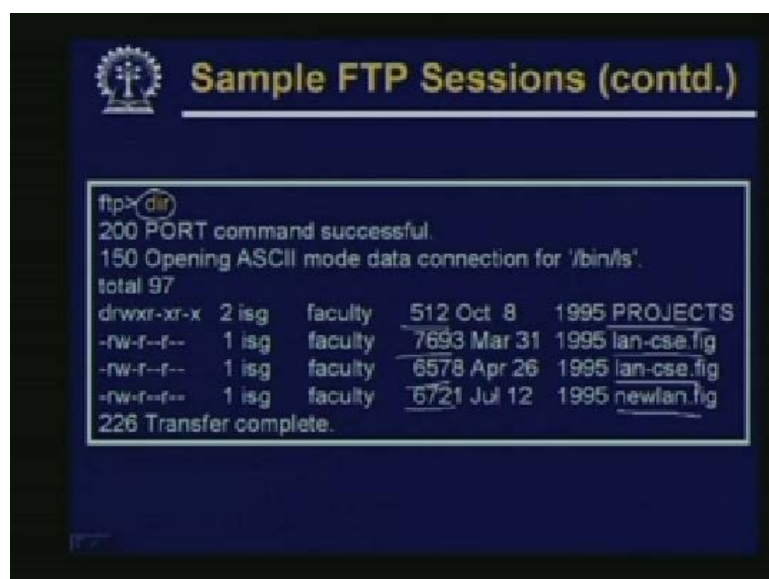
(Refer Slide Time: 53:47)



```
SUNNY>>ftp cseindigo
Connected to cseindigo.
220cseindigo FTP server ready.
Name (cseindigo:isg)(isg)
331 Password required for isg.
Password: 
230 User isg logged in.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp>
```

Well some example session, let us say, suppose here again we are sitting on a machine called SUNNY, from there I want to do a FTP into another machine whose name is say cseindigo. So connected to cseindigo, the connection over the control connection is established. Cs, see some response codes or you can say 220, some I say 331, 230. These are some intermediate response codes followed by the actual response. See when this connection is done, then this 220 means that the connection is successful. Then I ask for the username then ask for password. This 331 anything which ends with one, this means this is an intermediate completion reply. So this username password thing is not yet complete, it is half way through. So once this authentication is done it response to 230 it ends with 0. Means it is the end of one step. So now you see a prompt FTP comes. Now I can give the FTP commands.

(Refer Slide Time: 55:00)

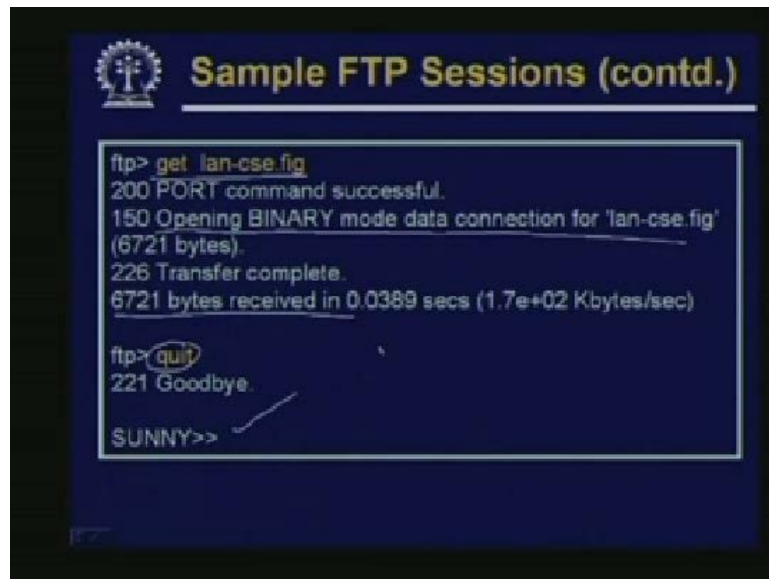


```
ftp>dir
200 PORT command successful.
150 Opening ASCII mode data connection for '/bin/lis'.
total 97
drwxr-xr-x  2 isg  faculty   512 Oct  8  1995 PROJECTS
-rw-r--r--  1 isg  faculty   7693 Mar 31  1995 lan-cse.fig
-rw-r--r--  1 isg  faculty   6578 Apr 26  1995 lan-cse.fig
-rw-r--r--  1 isg  faculty   6721 Jul 12  1995 newlan.fig
226 Transfer complete.
```

Like once small example I am giving. If I give a dir command, it means directory. So I can see a list of all the files which are there in the other computer. So I can see the list of all the

files their sizes, how big they are and so on. So if I choose to transfer any one of these files I can give a command.

(Refer Slide Time: 55:30)



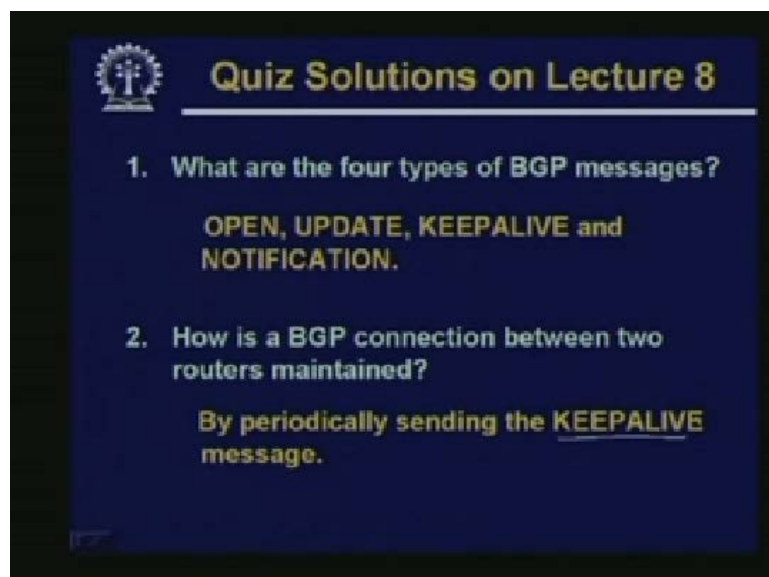
```
ftp> get lan-cse.fig
200 PORT command successful.
150 Opening BINARY mode data connection for 'lan-cse.fig'
(6721 bytes).
226 Transfer complete.
6721 bytes received in 0.0389 secs (1.7e+02 Kbytes/sec)

ftp> quit
221 Goodbye.

SUNNY>>
```

Say I want to transfer this file lan-cse fig get this file name. So it opens a new connection, it also tells you that 671 bytes received transfer complete. Then I can terminate the connection by this quit. So connection gets terminated. So this shows the sample. FTP session how you can use it, so this brings me to the end of today's lecture. Before ending today let me have a quick look at the solutions to the quiz questions of last lecture.

(Refer Slide Time: 56:01)

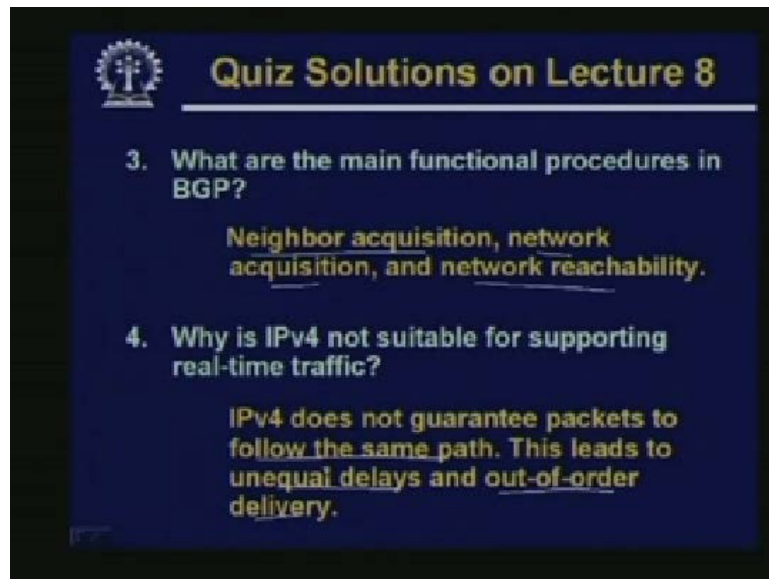
- 
- Quiz Solutions on Lecture 8**
1. What are the four types of BGP messages?
OPEN, UPDATE, KEEPALIVE and NOTIFICATION.
 2. How is a BGP connection between two routers maintained?
By periodically sending the KEEPALIVE message.

First question was: What are the four types of BGP messages?
They are OPEN, UPDATE, KEEPALIVE and NOTIFICATION.

How is a BGP connection between two routers maintained?

As I had mentioned by periodically exchanging KEEPALIVE messages.

(Refer Slide Time: 56:20)



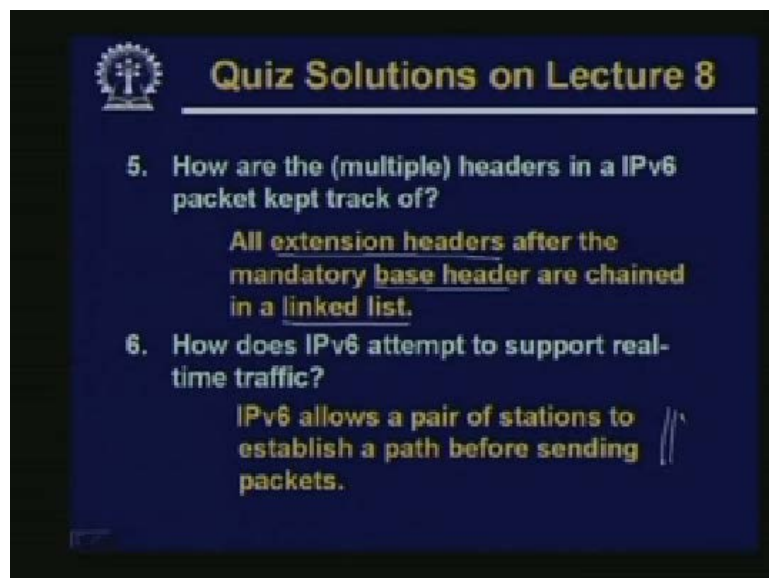
What are the main functional procedures in BGP?

Neighbour acquisition network acquisition network reachability.

Why is IPv4 not suitable for supporting real-time traffic?

Because they are based on datagram, they do not guarantee that packets will follow the same path. So there will unequal delays in the packet and may also they can also get out-of-order delivery for which real-time response is not possible.

(Refer Slide Time: 56:46)



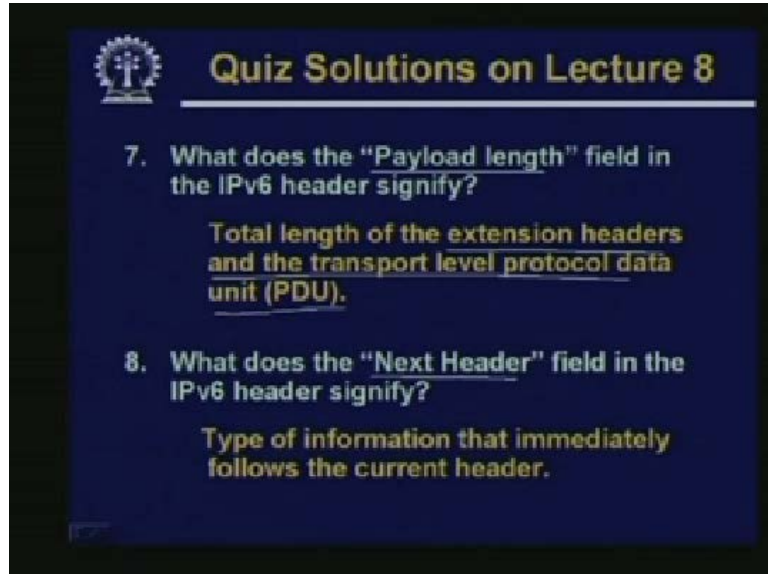
How are the headers in IPv6 packet kept track of?

So all extension headers after the base header are chained in a link list. This I had mentioned.

How does IPv6 attempt to support real-time traffic?

IPv6 allows a pair of stations to establish a path before sending packets. So this is an option you have in IPv6 to support real-time traffic.

(Refer Slide Time: 57:14)



The slide is titled "Quiz Solutions on Lecture 8" and features a university logo in the top left corner. It contains two quiz questions and their solutions. Question 7 asks about the "Payload length" field in the IPv6 header, with the solution being "Total length of the extension headers and the transport level protocol data unit (PDU)". Question 8 asks about the "Next Header" field, with the solution being "Type of information that immediately follows the current header".

Quiz Solutions on Lecture 8

7. What does the "Payload length" field in the IPv6 header signify?
Total length of the extension headers and the transport level protocol data unit (PDU).

8. What does the "Next Header" field in the IPv6 header signify?
Type of information that immediately follows the current header.

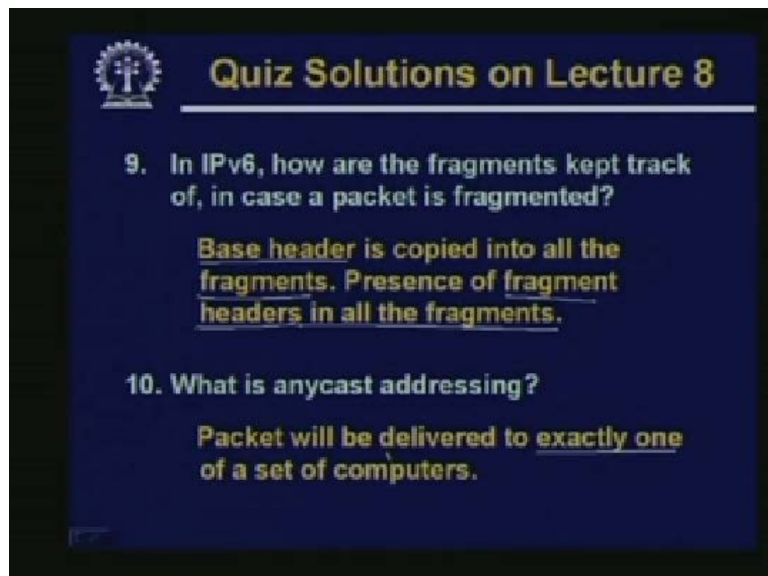
What does the Payload length field in the IPv6 header signify?

This specifies the length of the extension headers and the transport level protocol data in an IPv6 packet.

What does the Next Header field mean?

The type of information that immediately follows the current header. The current header, so there can be multiple headers or the PDU. So the Next Header will contain what is following the present header.

(Refer Slide Time: 57:41)



The slide is titled "Quiz Solutions on Lecture 8" and features a university logo in the top left corner. It contains two quiz questions and their solutions. Question 9 asks how fragments are tracked in IPv6, with the solution being "Base header is copied into all the fragments. Presence of fragment headers in all the fragments." Question 10 asks about anycast addressing, with the solution being "Packet will be delivered to exactly one of a set of computers."

Quiz Solutions on Lecture 8

9. In IPv6, how are the fragments kept track of, in case a packet is fragmented?
Base header is copied into all the fragments. Presence of fragment headers in all the fragments.

10. What is anycast addressing?
Packet will be delivered to exactly one of a set of computers.

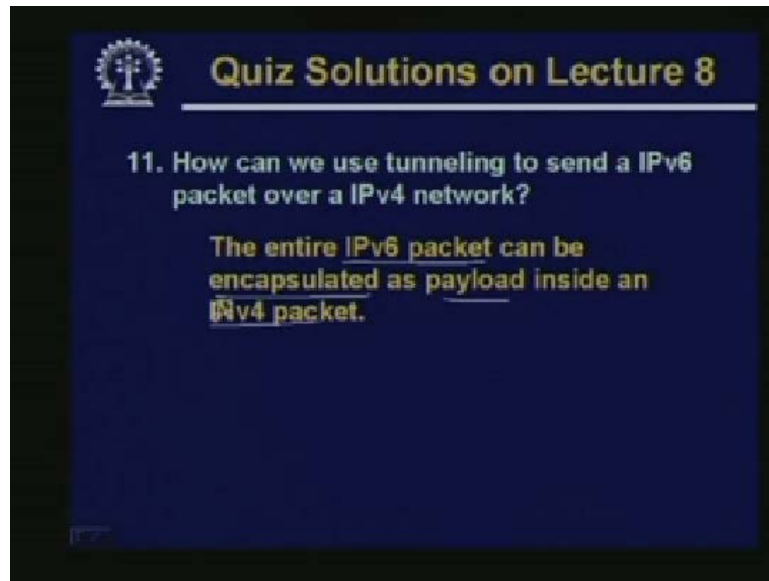
In IPv6 how are the fragments kept track of in case the packet is fragmented?

First thing is that base header is copied into all the fragments and if there is a fragment header in one packet it means that it is fragmented. So there has to be fragment header in all the fragments.

What is anycast addressing?

Packet will deliver to exactly one of a set of computers.

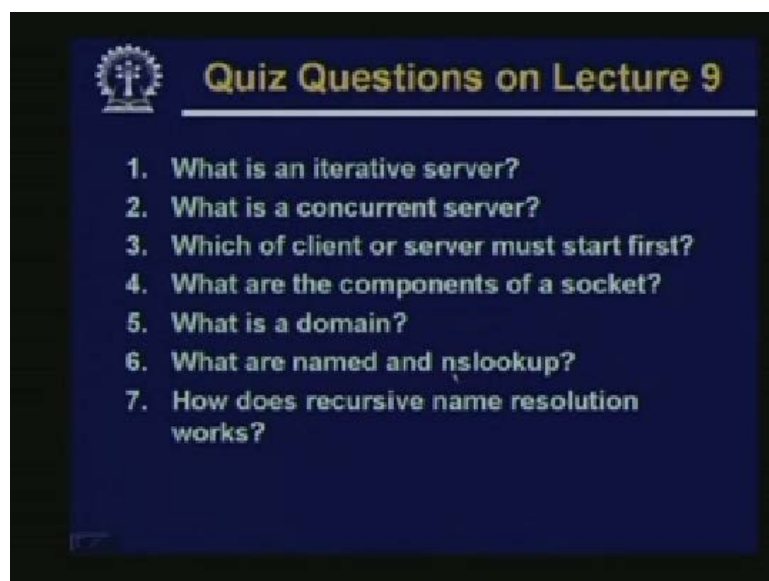
(Refer Slide Time: 58:09)



How can we use tunnelling to send an IPv6 over an IPv4?

The entire IPv6 packet is encapsulated as payload and IPv4 header will be added to it. So now some questions from today's lecture.

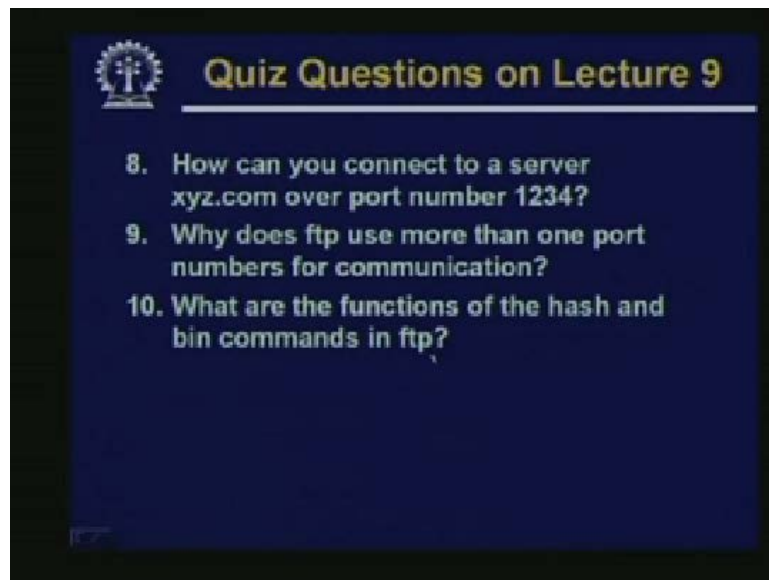
(Refer Slide Time: 58:27)



What is an iterative sever?

What is a concurrent server?
Which of client or server must start first?
What are the components of a socket?
What is a domain?
What are named and nslookup?
How does recursive name resolution works?

(Refer Slide Time: 58:49)



How can you connect to a server xyz.com over port number 1234?
Why does ftp use more than one port numbers for communication?
What are the functions of the hash and bin commands in ftp?
So with that we come to the end of today's lecture. Thank you.