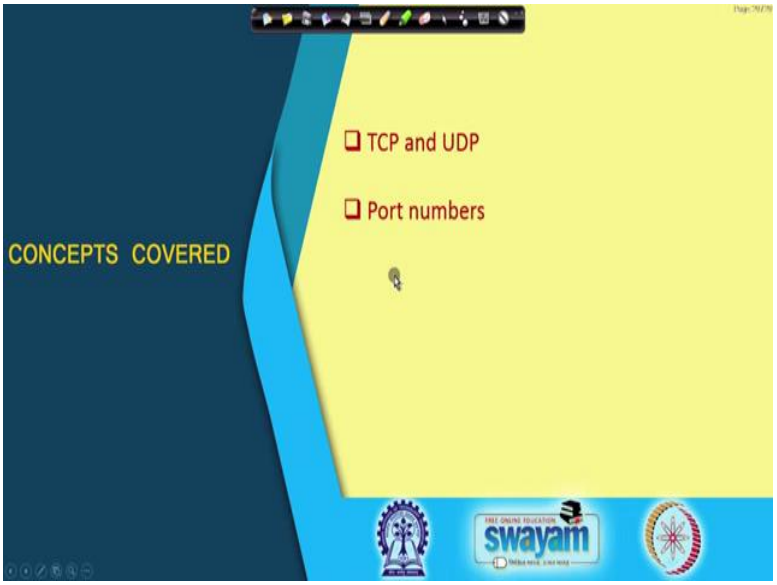
Ethical Hacking
Prof. Indranil Sengupta
Department of Computer Science and Engineering
Indian Institute of Technology, Kharagpur

Lecture - 08
TCP and UDP (Part I)

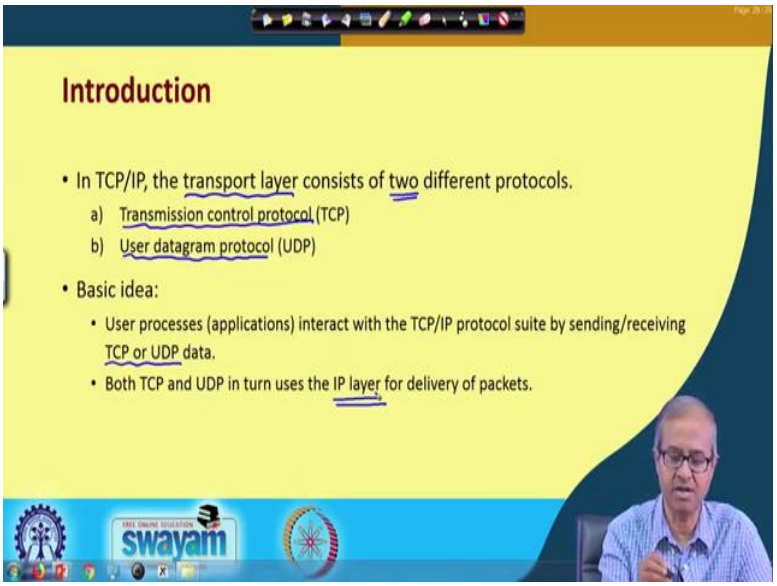
In this lecture, we start with some specific discussion on the TCP and the UDP protocols, what they are, and how they work in the context of the TCP/IP protocol suite. So, this is the first part of this lecture TCP and UDP.

(Refer Slide Time: 00:35)



In this lecture, we shall be talking about the basic functionalities and roles of TCP and UDP, in particular we shall be talking about something called port numbers and how they relate to data communication at the transport layer level, and how TCP and UDP uses port numbers ok. These are the few things we shall be covering in this lecture.

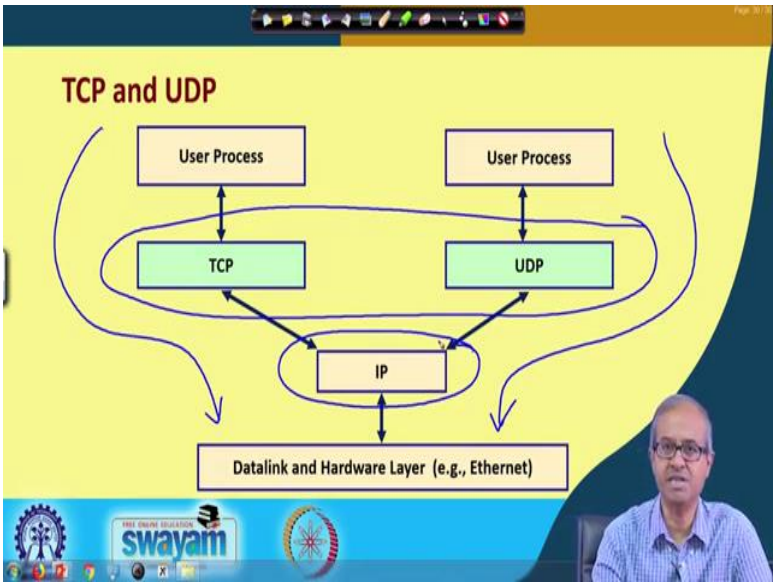
(Refer Slide Time: 01:00)



Now, we have already talked about the TCP/IP protocol suite. Now, in TCP/IP if you look at the transport layer level, at the transport layer level, there are two different protocols that are used, one is TCP which is the short form for transmission control protocol, other is UDP which is the short form for user datagram protocol. Now, the basic idea is whenever there is a user program, some program, which is trying to send or receive data from some other host on the Internet, it interacts with either the TCP or the UDP layer.

Suppose, I want to send the data, I can give my data to TCP, TCP in turn will give it to IP and IP will actually send or route the data via the data link layer which is below it ok. This is how it works. The user processes, they will interact with either TCP or UDP and TCP and UDP will be using the IP layer which is below it at the network layer level.

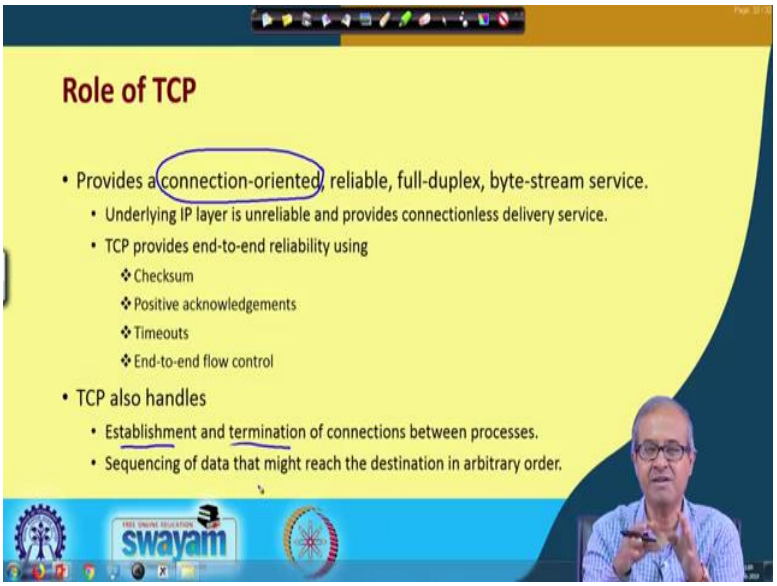
(Refer Slide Time: 02:28)



Now, this diagram shows what I just now said diagrammatically. So, this is where we are at the transport layer level. So, at the transport layer level, we have two alternatives, either TCP or UDP. So, whenever there has some user process which is running, there is a choice, it can either choose to use TCP or it can choose to use UDP. It depends on the application, what kind of facilities are the transport layer level the user process wants and accordingly it can do that.

But the point to note is that you see whatever you use TCP or UDP, at the network layer level, you have no choice; you have this IP and only IP. So, both TCP and UDP, they interact with the IP protocol at the network layer level which you recall is a datagram base service which is unreliable, datagrams might get lost, duplicates might get generated, order may not be maintained. These things I mentioned a number of times, ok, fine.

(Refer Slide Time: 03:47)



Let us now look at specifically what are the roles of TCP, what does the TCP protocol does. TCP provides a connection-oriented service, it is reliable. Reliable means if there is any data corruption during transmission, some data packets are getting lost or gets corrupted, TCP is supposed to keep track of that.

And if something is wrong the source or the sender will be requested to again send the relevant part of the data, so that the receiver can receive the data in a correct way that is reliability. Full-duplex means it is a two-way communication, a can send to b, b is also sending some data back to a. So, it is not a one-way communication.

Byte-stream service means whenever some message is being transmitted, both the sender and receiver keeps track of how much data have been sent and how much data is yet to be send in terms of the number of bytes. So, each byte has a number 1, 2, 3, 4, 5, 6, 7, 8, and both sender and the receiver keeps track of these byte numbers. How many bytes I have already sent, how many bytes I have not yet sent and similarly for the receiver, it sends back sometime some acknowledgement, it says that well I have received all bytes up to byte number 1050, you can now send me the bytes after that, something like that ok.

So, to provide this end to end reliability that TCP provides, you see because TCP provides reliability, the applications that use TCP, it does not have to care about anything. It assumes that my network is very reliable. There are no errors, everything that I sent will be received correctly by the other end, but it is actually the TCP layer which handles all

errors and tries to recover from the errors ok. Now, TCP provides the reliability using several mechanisms. There is a checksum where each packet is verified using a checksum mechanism whether there packet has become corrupted, whether there is an error in the packet or not.

Positive acknowledgement, the receiver sends back some acknowledgement to the sender that well I have received all bytes up to 1050, this is the positive acknowledgement. So, now, the sender knows I have to send byte number 1051 onwards now, rest have been received correctly. Timeouts, well if the sender sees that the acknowledgement is not coming from the receiver till some defined interval, there will be a timeout mechanism which will be enforced and the entire message or the segment will be resend.

It will assume that there was some network error, maybe the data did not receive the other end, let me send it again. And end-to-end flow controls, so this speed of the sender and receiver may not be the same. So, dynamically both the send and receive at the TCP layer adjusts the speed of data transmission and data receiver, recipient, receiving, sending and receiving, so that this speed mismatch is taken care of

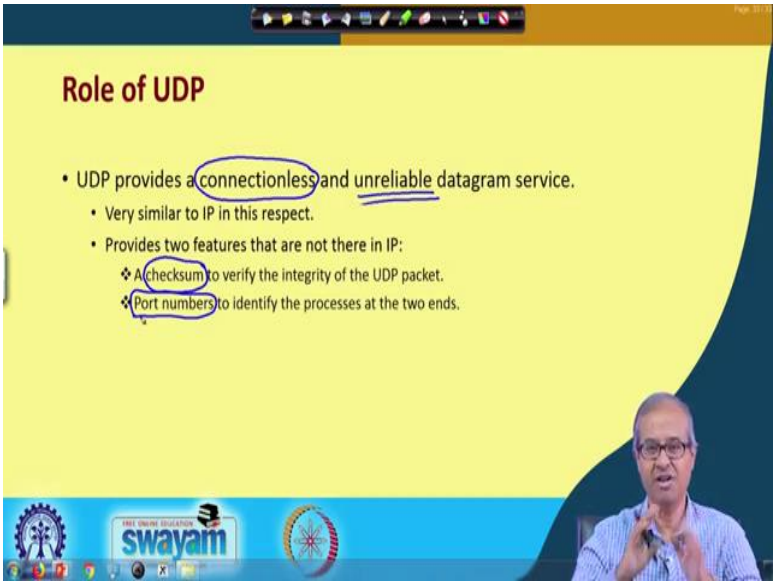
Well, now in addition to this, TCP also handles connection establishment and termination, because it is a connection-oriented service similar to virtual circuits we talked about earlier. There is a concept of connection establishment and termination, but the thing to note is that you are talking about connection at the TCP layer level, but down below, it is IP. Whatever you do at the TCP layer level ultimately IP will be sending out datagrams, they will be following different paths and they will be reaching the destination in any other order.

But the TCP layer gives that illusion to the user process that this is a connection-oriented service ok, this is how it works. And TCP also you see because of the datagram at the IP layer, the data might reach the destination in some arbitrary order. TCP will put together all received data in the correct order, which is called sequencing. So, that the user process at the receiving end will get the data in the correct order always, that is one of the main responsibilities of TCP and this is what we mean by connection-oriented.

There is a virtual connection which is maintained and the user process assumes or I mean, it gets a feeling that there is a connection and all data are coming through that connection

in the same order ok. But actually TCP has to work quite hard in order to have all these features incorporated.

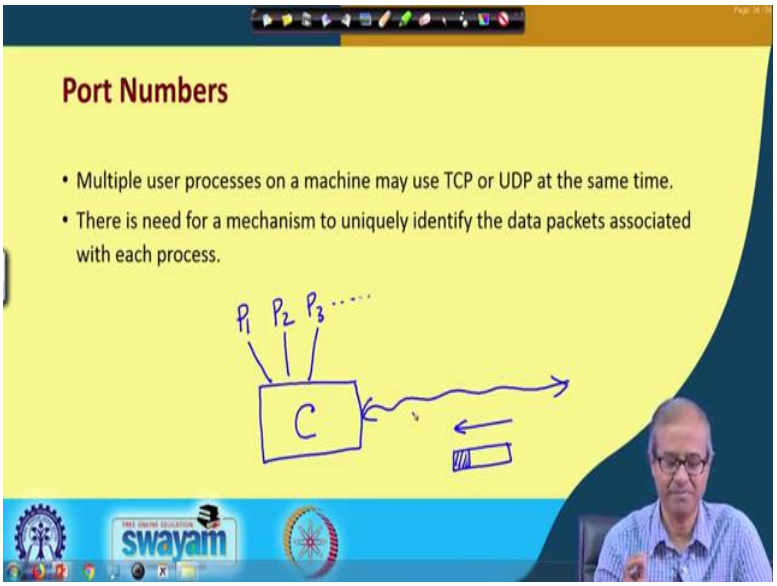
(Refer Slide Time: 09:34)



Well, in contrast UDP is very much similar to IP, it does not do much beyond that, it does not try to ensure any kind of reliability. It provides a connectionless kind of a service, it is a datagram service. TCP is a connection-oriented that is why, some connection establishment and connection termination are required. But UDP does not require any connection establishment, each packet is being sent independently. And because it is like a datagram service, this is also unreliable just like IP ok.

There I just two additional things which are there in UDP which IP does not have, at the transport layer level it incorporates a checksum to verify whether the UDP packet is correct or has been corrupted. If there is some corruption there will be a checksum error possibly. And there is something called port numbers which are also added. Port numbers actually identify the user processes that are interacting with UDP or TCP, we will talk about this.

(Refer Slide Time: 11:00)

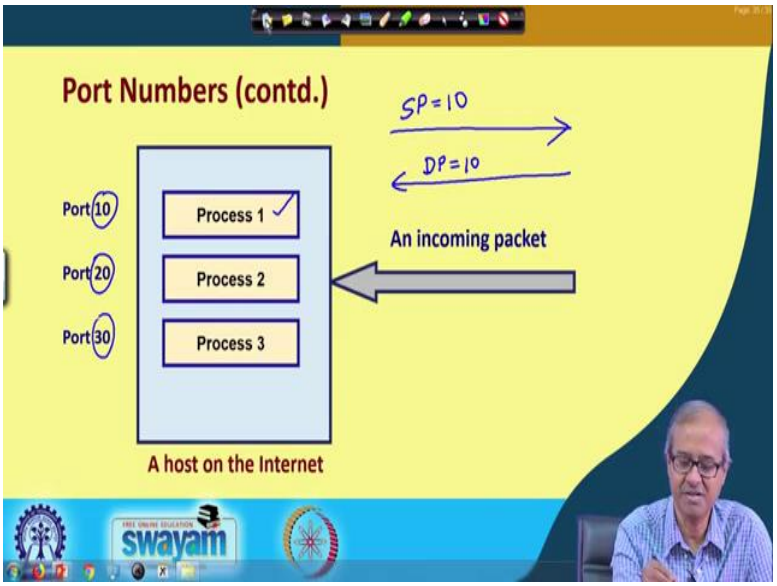


Yes, now you see you think of a computer, you think of a particular computer on the Internet. There can be several user programs which are running P_1, P_2, P_3, many user programs are running on this computer. And they may all be trying to communicate some data over the network, over the network it is sending and receiving some data ok. Suppose, P_1 has send some data to somebody and that somebody is again sending back a response, a packet is coming back.

So, when the packet reaches this computer, see how will this computer know that this packet has to be sent to P_1 and not P_2. There has to be something in this packet, in the header which must clearly identify that this packet has to be delivered to P_1, and that is this port number. On each computer every running process is identified with an unique port number.

And whenever at the transport layer level a message has been transmitted, either we using TCP or UDP in the header, there is a port number and that port number specifies which process is sending that message, and at the other end which process is supposed to receive that message. Those port numbers identify that; that is the role of port number mechanism, to uniquely identify the data packets associated with each process. So, each process will have a unique port number for every connection it is maintaining ok.

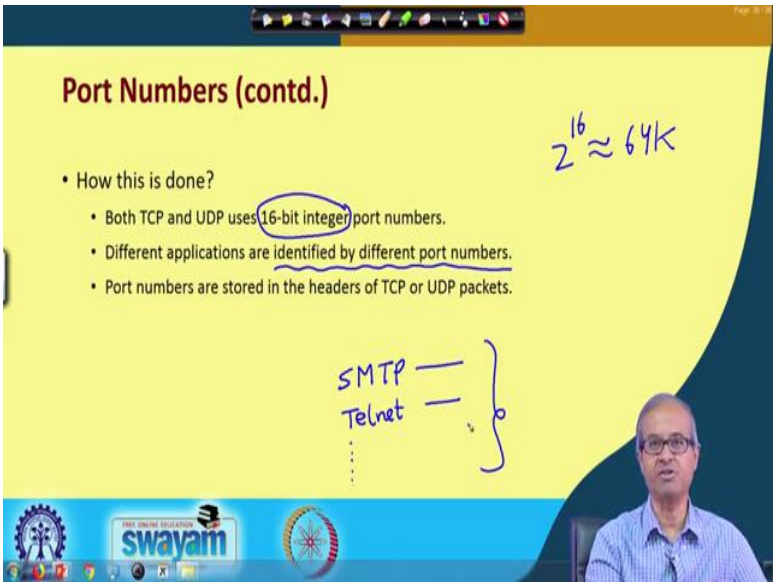
(Refer Slide Time: 12:57)



So, as this diagram shows, diagrammatically we will list the same thing. Suppose on a particular computer, there are three running processes. And when they are communicating over the Internet, over the network, they will be assigned some port numbers, let us say port number 10, 20 and 30. So, when process 1 is sending some data packet; when we sending out some data packet, it will specify the source port number which is 10.

And when a response comes back, in the response the destination port number will be set as 10, so that this computer will know that 10 is what, 10 is this process 1, this packet has to be delivered to process 1. So, this is how the transport layer in this computer will take care of sending and receiving of the packets and delivering them correctly to the appropriate application or process ok, this is how it works.

(Refer Slide Time: 14:17)

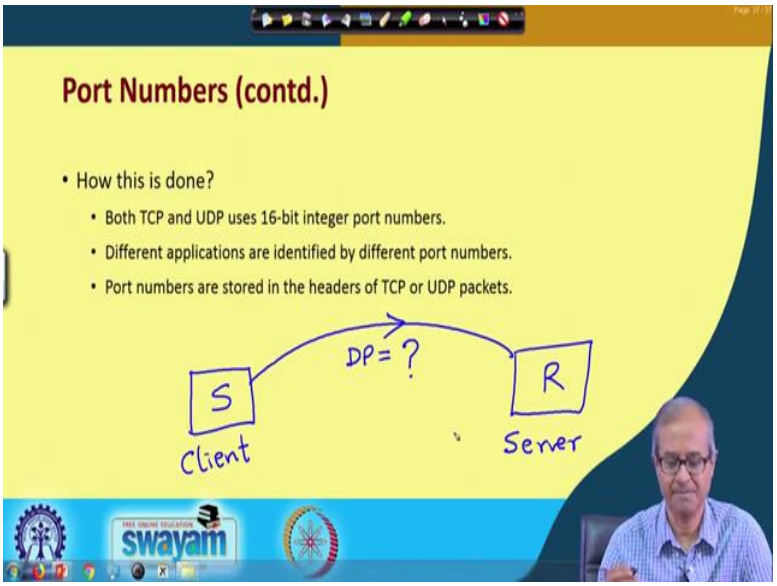


Now, in TCP and UDP, this port numbers are 16 bit quantities; well, 16 bit is quite large, you can have 2^{16} which is about 64000, 64 k. You can have that many active connections, active processes running and this quiet large. And you see, there are certain applications which are well known, they are the uniquely identifies by some port numbers, like you think of your mail, SMTP, SMTP will be having some particular port number. You think of some network application like telnet, telnet will be having some unique port number.

And if you think of any other kind of network application the common ones will be having some unique port numbers, so that suppose I am trying to send a mail, I know that is my mail server, I always know that my mail server will be working on port number so and so, port number 25 or something whatever. So, whenever I want to send a packet to my mail server that particular process, I will be sending a packet mentioning destination port number is equal to 25. So, it will always go to the mail server.

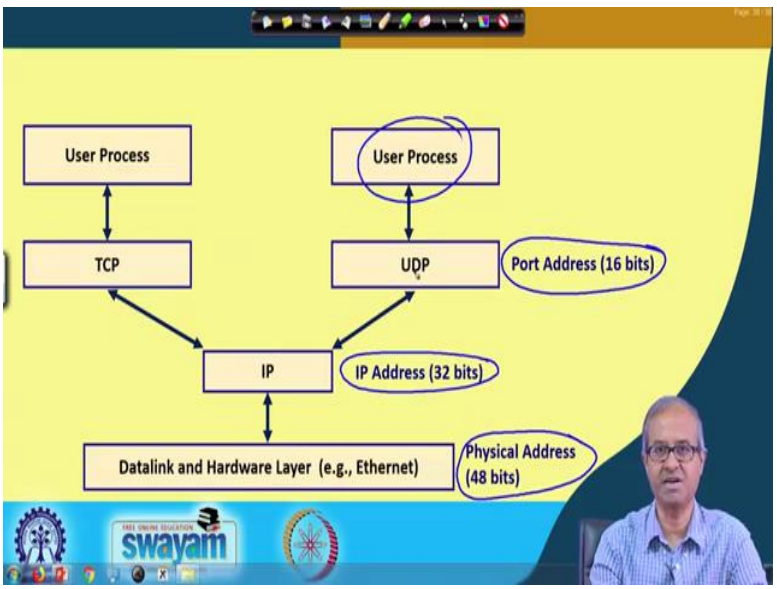
Similarly, when I want to send some request to a web server, may be the web server is, is working on port number 80, I will be sending a packet saying destination port number 80. So, my packet will automatically go to that web server, and web server processes it, and sends the request back to me. So, by this unique port numbers, I can uniquely identify and contact a particular process which is running on the other end right.

(Refer Slide Time: 16:32)



Just one thing I will tell you, well in an internet scenario, if you think of a situation like this, this is a sender, this is a receiver. Well, often we talk about client-server scenario, let us say this is a client and this is a server. The client will be sending some request to the server. With respect to the request, this is the sender, this is the receiver. Now, when it sends a request, the client is supposed to know which process at the other end, it is trying to contact to. So, the destination port number must be already known to the client ok. As I told you for all the common applications, the destination port number is well-defined, everyone knows about it.

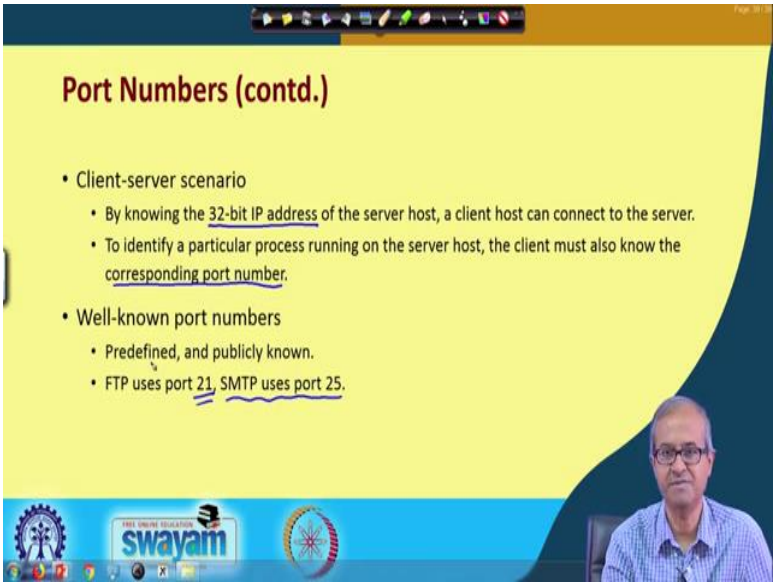
(Refer Slide Time: 17:32)



Let us look at it. This is the overall picture, again I am showing here the MAC address or the hardware address we already mentioned earlier, this is a 48-bit address, at the IP layer you have a 32-bit IP address. Now, at the TCP or UDP level, you have a 16-bit port number or port address. So, again just repeating physical address ensures uniqueness of the network interface cards, every network interface card you plug into your computer that will be having a unique 48-bit address.

At the IP address level, every computer you want to connect to the Internet must have a unique 32-bit IP address. Every connection to the Internet, to the router, to the network has to have a unique IP address. Port number or port address, every user process running on a computer must have a unique port address, these are the different levels of uniqueness that are maintained, so that process-to-process communication becomes very unambiguous and unique fine.

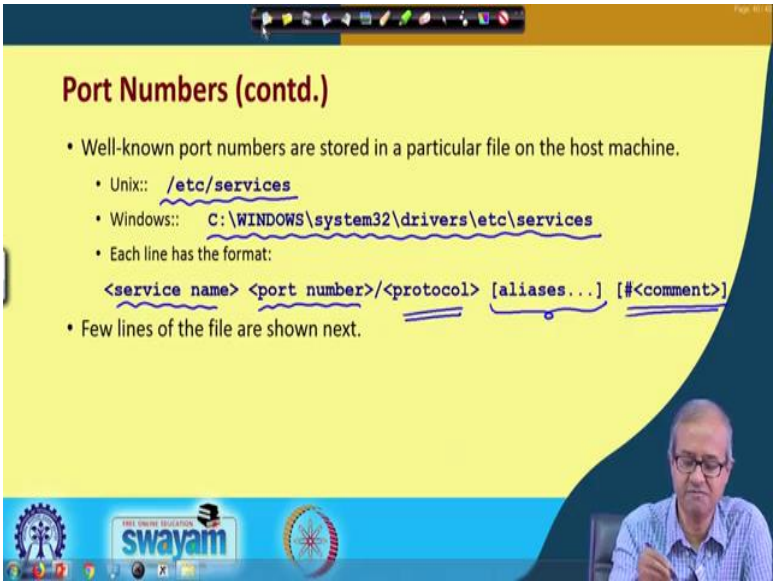
(Refer Slide Time: 18:58)



Talking about this port number, we already mention this briefly. In a client-server scenario, I was talking about this is a client, who is sending request to a server, it is wanting some service. Now, what does the client need to know, a client need to know, what is the IP address of the server, because it has to send an IP packet to the server, the request. So, the IP address is required, destination address, not only that it will also must know the corresponding port number of the application which I am trying to contact to.

Well, I have given an example already if it is a mail server which is running the SMTP protocol, I need to contact over port number 25 ok, SMTP uses port number 25. If I am using file transfer, if I am using the protocol FTP, FTP uses port number 21. So, these are all so called well-defined or well-known port numbers, these are predefined and these are some kind of standardized, everyone uses this numbers.

(Refer Slide Time: 20:23)

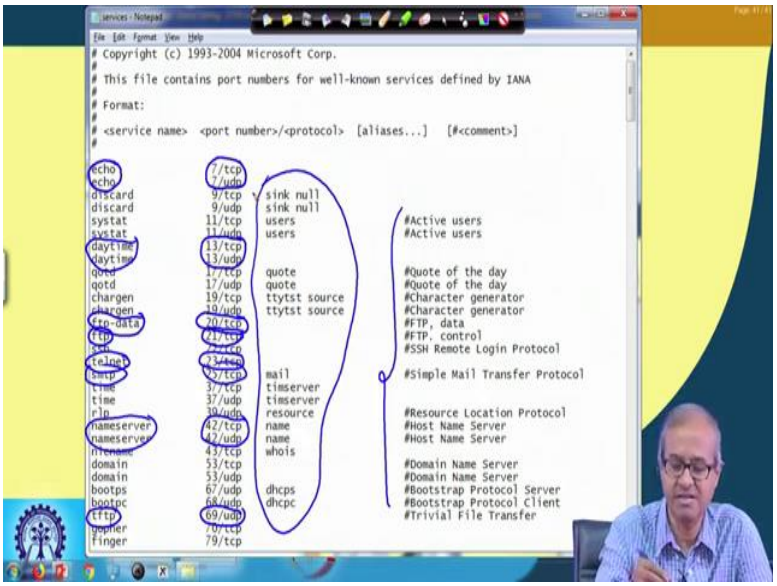


Now, on each system, each computer, these port numbers, well defined port numbers and other port numbers that you may like to define yourself, they are stored in a particular file. If you want you can have a look at them yourselves. Well, on any Unix or Linux system, there is a file which is located in under root, under the etc. directory, the name of the file is services. Well under the windows operating systems and various versions of windows are there.

Typically under this path, there is also a file called services. Now, this file is a text file, it contains several lines of information. Every line contains something like this. It starts with the name of a service. It specifies the port number a slash followed by a protocol, it can use either TCP or UDP and there can be some optional comments at the end, starting with a hash symbol. It explains what that service actually means and there can be some aliases also, you can give some alternate names also.

Well, here I am giving an example snapshot of a services file of one of the computers.

(Refer Slide Time: 21:57)



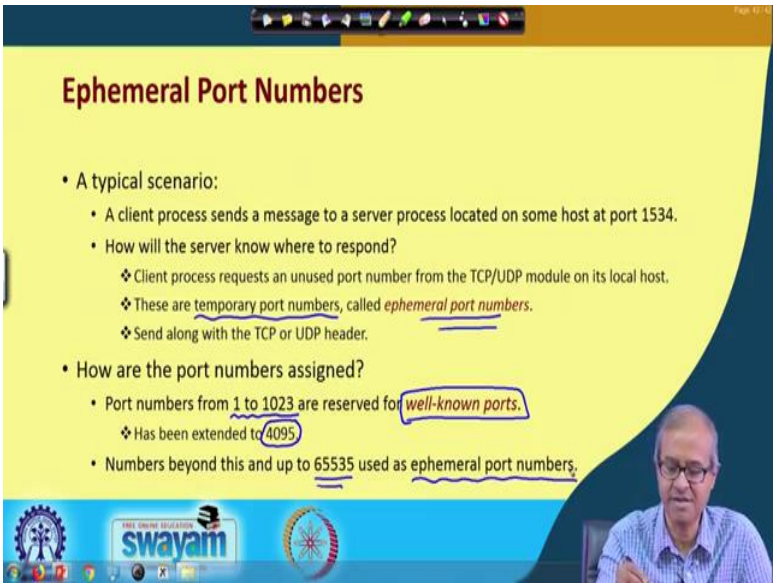
This is shown. Well, here the font size is small. I am sure you may not be able to see it very clearly. But if you open that file, I just mentioned which is there on your own computer, you can see this file for yourself. Now, well here I am just now, here I am just reading out some of the lines. There is a service called echo, it is running on port number 7 on tcp as well as udp. So, if you send some message to the echo server, it will send back the same message back to you; echo is used sometimes for the purpose of testing the communication ok.

Let us say daytime, daytime is another service which is again running on port number 13 on both tcp, udp. If you send a request, day and time will be sent back to you. Then you see ftp - file transfer protocol uses port number 21, under tcp this is for ftp connection establishment, but when the actual data is being transferred you use ftp data and that uses port number 20 of tcp.

Then telnet, telnet is used for remote login, it uses port number 23 of tcp. This smtp uses port number 25 of tcp. Name server, name server uses both tcp and udp port number 42 ok. There are so many tftp, trivial ftp uses port number 69 of udp. And you see on the right with this hash, some explanations are given, these are comment lines and these are aliases, this some alternate names are also given right.

This is the typical format of the services file which is there in each of the computers that are connected to the Internet. It contains a list of all the active services and the corresponding port numbers and protocols that are used there.

(Refer Slide Time: 24:23)



Now, there is something called ephemeral port numbers. Now, you imagine a situation like this I am trying to contact a mail server, I know mail servers port number. So, I specify that as a destination port number and send it to that.

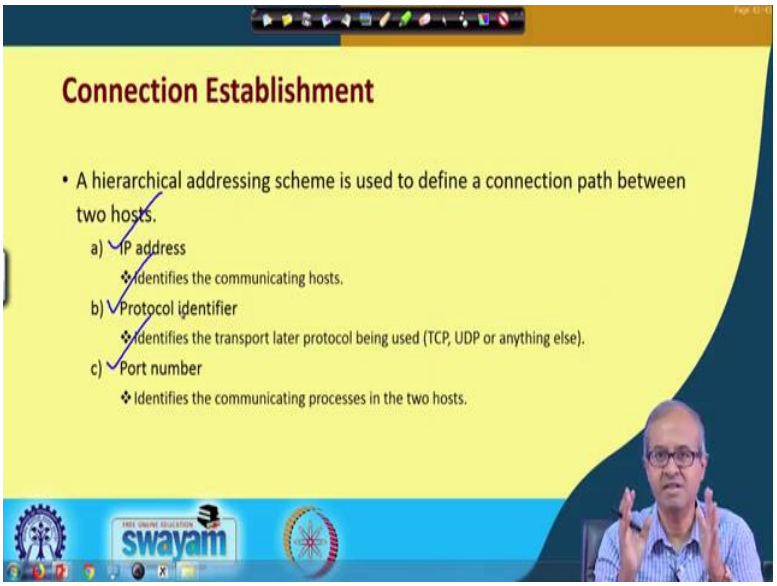
But when the mail server sends back a response to me, how will I get the packet because on my computer there may be several user processes running. So, I must also be having some kind of port number which I should tell the mail server as my source port, and when that packet comes back that source port will become the destination port, so that it comes to me.

But I do not have any unique port number means what I mean to say, I do not have any well defined port number, I am using it for sending a particular mail only. These are called temporary port numbers, which I can request on a temporary basis. And these temporary port numbers are called something called ephemeral port numbers right. So, here the same thing as I mentioned is explained here, this temporary port numbers are called ephemeral port numbers.

So, whenever such a communication is initiated by a client, the senders port number is some kind of random a temporary port number which is assigned. So, after that communication is over that is again discarded right. This port number will not appear on that services file ok.

Now, there is some conventions which are followed, port numbers starting from 1 up to 1023 are supposed to be reserved for the well-known ports. But recently this has been extended up to 4095, as the number of such well-known services are increasing, various kind of services are there. And numbers which are beyond 4095 and up to the maximum, they can be used as the temporary port numbers, the ephemeral port numbers right.

(Refer Slide Time: 26:50)

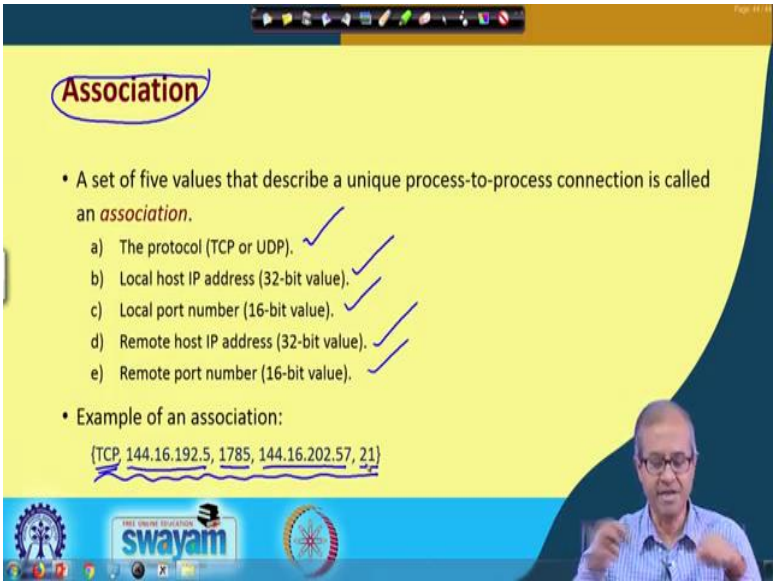


Now, talking about connection establishment whenever under TCP/IP, under with TCP, I want to establish a connection with another host. There are three things, I mentioned already you need to specify. You need to specify the, specify the IP address. You need to specify which protocol you are using TCP or UDP, and you need to specify the port numbers. So, whenever you specify all of these things, then only you can establish a connection.

You see for protocol, only for TCP, the question of connection is coming; for UDP there is no connection, it is like a datagram service ok. So, you have to specify all these things which are the IP addresses of the two ends, the protocol TCP, the port numbers at the two

ends. If you tell all these things, then only you can say that you are in a position to establish a connection.

(Refer Slide Time: 27:53)



And this kind of a connection, this information you are providing is sometimes referred to as an association. These five values you specify, the protocol you are using local IP address, remote IP address, local port number which can be an ephemeral port number and the remote port number.

So, as an example, this can be an example of an association where it using the protocol TCP/IP, local IP address, local port number, remote IP address, remote port number. So, whenever you are, do some kind of a communication over the Internet, this kind of association gets created automatically in the underlying networking software or the networking driver that is present.

So, with this we come to the end of this lecture. In the lecture, in the next lecture we shall be continuing with our discussion, we shall be looking at some of the more features of TCP and UDP, in particular we have not yet seen the header formats of TCP and UDP. In particular how TCP connections are made and terminated and so on and so forth. These we shall be discussing in our next lecture.

Thank you.