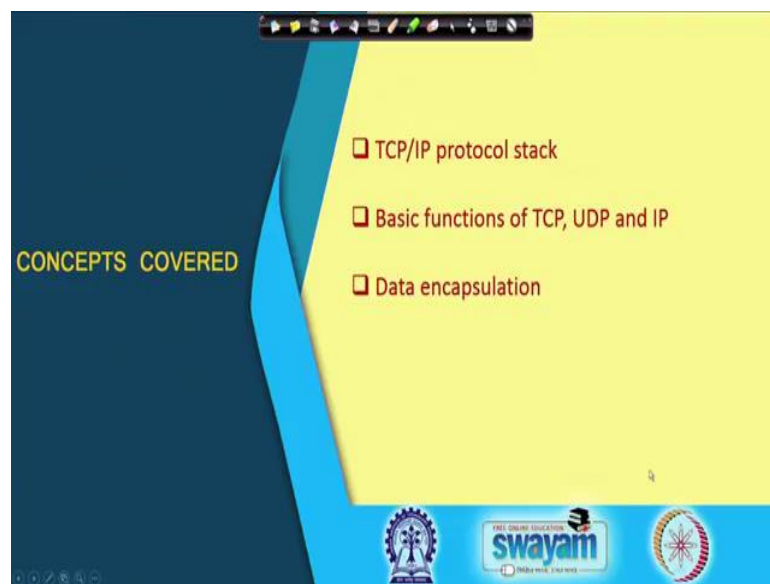


Ethical Hacking
Prof. Indranil Sengupta
Department of Computer Science and Engineering
Indian Institute of Technology, Kharagpur

Lecture - 04
TCP / IP Protocol Stack (Part I)

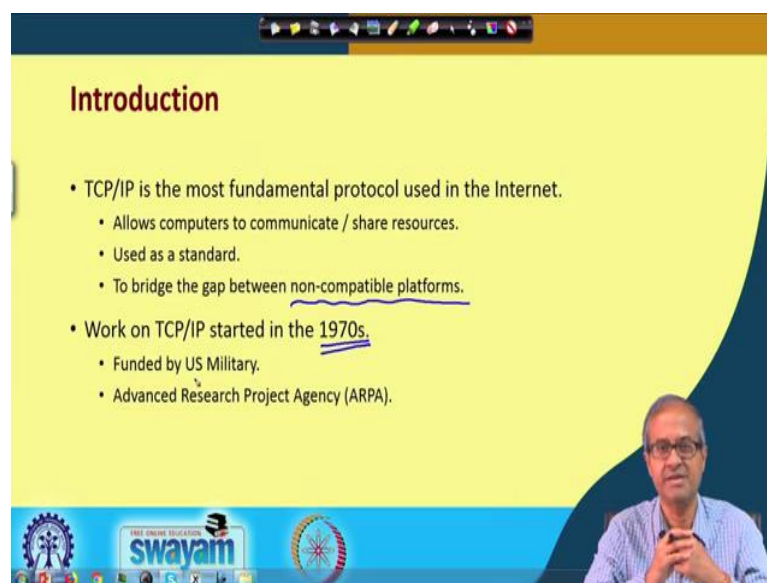
In this lecture, we shall be starting our discussion on something called TCP/IP Protocol Stack. So, the title of the lecture is TCP/IP protocol stack, the first part.

(Refer Slide Time: 00:28)



Now, in this lecture, I shall be talking about firstly, the TCP/IP protocol stack on which the internet is basically based on. We shall be looking at the basic functions of these TCP, UDP and IP, the most important three components of this TCP/IP stack. And with the help of an example we shall see the concept of something called data encapsulation.

(Refer Slide Time: 01:02)



Introduction

- TCP/IP is the most fundamental protocol used in the Internet.
 - Allows computers to communicate / share resources.
 - Used as a standard.
 - To bridge the gap between non-compatible platforms.
- Work on TCP/IP started in the 1970s.
 - Funded by US Military.
 - Advanced Research Project Agency (ARPA).

The slide features a blue sidebar on the left with the 'swayam' logo and text 'FREE ONLINE EDUCATION'. A presenter is visible in the bottom right corner of the slide frame.

So, let us start with TCP/IP. What it is actually? As I mentioned TCP/IP is the most fundamental protocol in the internet. Just to imagine, what is an internet; internet is a network of networks. There are so many networks, they are all connected together, inside a network there are so many different protocols that are being followed, computers are connected to it, but somehow they are all able to communicate with each other. So, there must be some common language, all these computers are speaking and it is this TCP/IP that is the common language.

So, if a node, you connect to the computer, if it understands this TCP/IP protocol, then it can communicate with the Internet to any other computer in the world right ok. Now, TCP/IP by virtue of its standardization, this allows computers to communicate and share resources over the internet all across the world. This is used as an universal standard in the Internet.

Now, this TCP/IP as a universal standard is used to bridge the gap between various different platforms. Well, when you say platforms, you can think about operating system to start with, you have Windows, you have Linux, you have Mac, then you can have different kind of networks also. You have Ethernet, you can have ATM, you can have so many other kinds of networks. Talking about wide area networks, there also so many standards are there.

So, there are various alternatives and options available at every step. So, how do you make them work together? It is this TCP/IP; TCP/IP is the common glove that binds everything together right. Now, this TCP/IP, the work, the standardization started in the early 1970s as part of a project which was initially funded by the US military. But subsequently, it spread to the different universities and finally, it became a de facto standard in the Internet. Nowadays everybody uses TCP/IP.

(Refer Slide Time: 03:31)

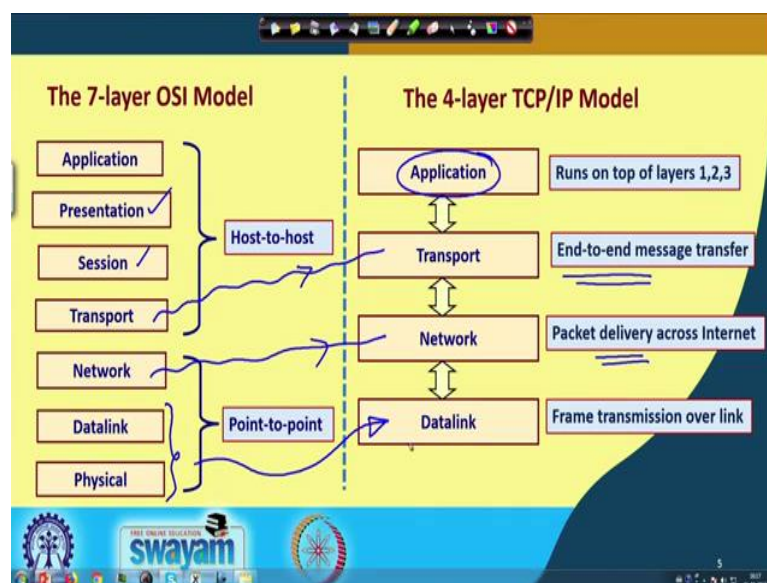
Network Layering in TCP/IP

- In 1978, International Standards Organization (ISO) proposed the 7-layer OSI reference model for network services and protocols.
 - TCP/IP does not strictly follow the OSI model.
 - It follows a simplified 4-layer model.

The slide is part of a presentation, as evidenced by the navigation icons at the top and the presenter's video feed in the bottom right corner. The bottom of the slide features logos for 'swayam' and other educational institutions.

Now, in TCP/IP, the network layering, well you have already seen in the previous lecture the 7-layer OSI model which was proposed by International Standards Organization. Therefore, TCP/IP, there is a similar kind of layering which is followed, but it is somewhat simplified, it does not contain all the 7-layers. In fact, it uses a very much simplified 4-layer model, there are only 4-layers here ok. So, in contrast to the 7-layer OSI model, TCP/IP has only 4-layers. Let us try to see this layering side by side and try to make a comparison ok.

(Refer Slide Time: 04:27)



Here you see the 7-layer model on the left, and the 4-layer TCP/IP model on the right. And the 7-layer model as I mentioned, the lower 3-layers are the point-to-point layers and the upper 4-layers are host-to-host layers. Then the TCP/IP, what they have done, they have simplified that the all upward four host-to-host layers, they have integrated in a single application layer ok.

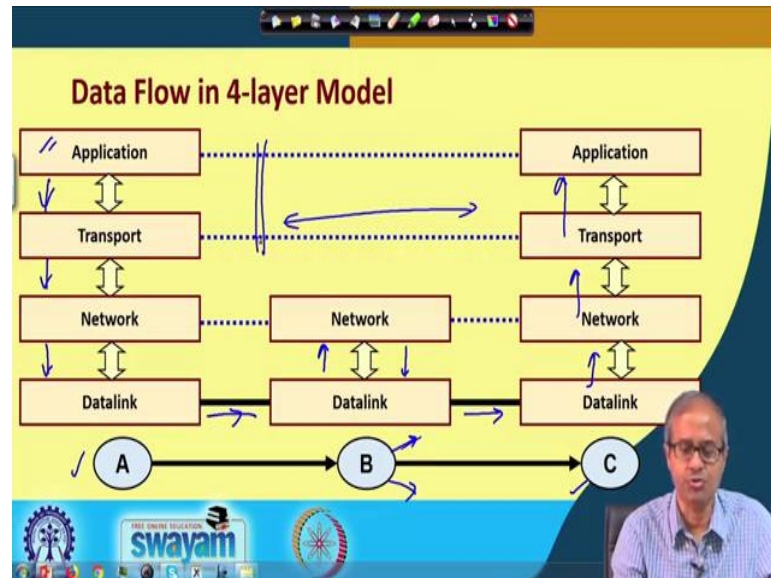
You see it depends on the application whether you actually require a session, whether we actually need to do some changes in your data presentation. So, it really depends. So, instead of keeping separate layers, the TCP/IP has integrated all of them in a single layer and calling it the application layer.

And this transport layer this of course, has a more or less correspondence with the transport layer here. This is a host-to-host layer, this is responsible for end-to-end message transfer. Network layer also has a correspondence. Network layer is mainly responsible for packet routing, delivery of packet across the Internet, across the network. And again this data link and the physical layers, they are merged together and they are referred to as a single link, sometimes this is referred to the as a data link or a physical link, they are called differently.

The point is that, in a computer system when you, when you want to connect it to a network, there is a some kind of a network card, you want to use network interface card, and the network interface card typically integrates both data link and the physical layers

together that is why in a simplified TCP/IP model, the two layers have been merged into a single layer ok. So, this is the simplified 4-layer version in TCP/IP.

(Refer Slide Time: 06:40)

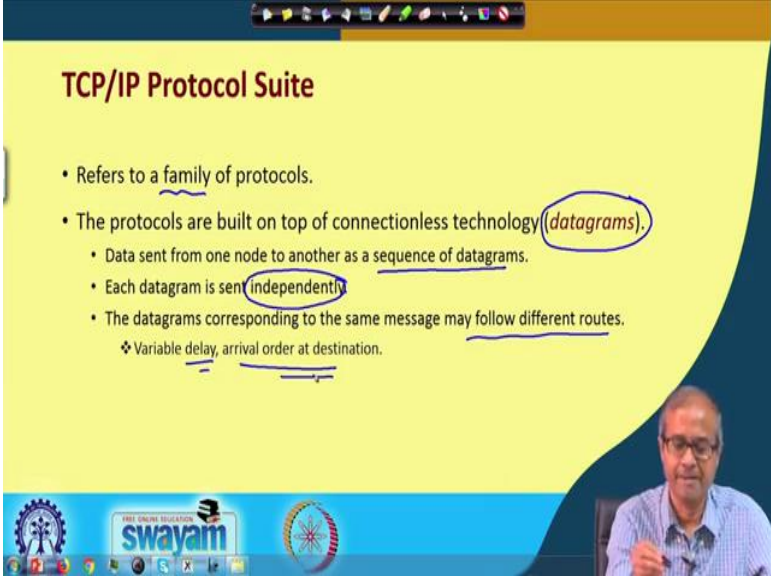


Now, in this 4-layer model, when you think of the data transmission, again it is somewhat simplified as you can see here. This A is the sender, C is the receiver, and B is an intermediate node. So, this A generates some data from some application; from application it goes down to the transport layer; from transport layer it goes down to the network layer, network layer to the data link layer and to the data link layer it sends the data actually over the data link or the point to point link.

Now, this intermediate node B receives the frame here, sense it up to the network layer; network layer takes a decision where to forward the packet next, and accordingly it sends it down to the data link layer of the outgoing link. This B may be having several outgoing links, there will be one data link corresponding to every outgoing link. So, it will be sending out to the data link corresponding to the correct outgoing link and the frame will be going there, received, and it will be traverse in the reverse direction, and finally, took some application in the destination C.

Now, you can see, this transport is a, this is a end to end layer or host-to-host layer as if A and C are communicating directly. This intermediate node B is not visualized at this layer, and also at the application layer ok. These are called end to end or host-to-host layers.

(Refer Slide Time: 08:27)



TCP/IP Protocol Suite

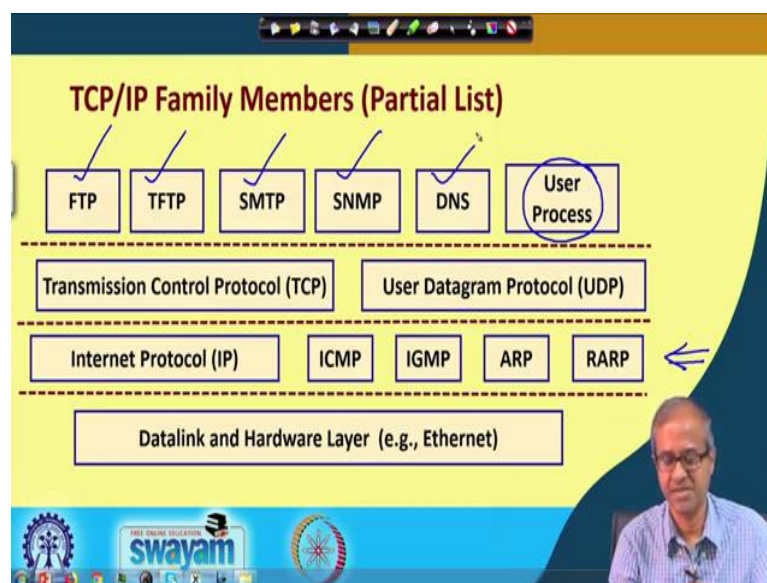
- Refers to a family of protocols.
- The protocols are built on top of connectionless technology (datagrams).
 - Data sent from one node to another as a sequence of datagrams.
 - Each datagram is sent independently.
 - The datagrams corresponding to the same message may follow different routes.
 - ❖ Variable delay, arrival order at destination.

The slide features a yellow background with a blue header and footer. The title 'TCP/IP Protocol Suite' is in bold red text. The bullet points are in black text, with some words underlined or circled in blue. A man is visible in the bottom right corner of the slide frame.

Now, talking about the TCP/IP protocol suite; TCP/IP protocol suite is not a single protocol, but rather it is a family of protocols or collection of protocols that is why it is called a family of protocols. And these protocols are all based on datagram mode of data communication, they do not rely on virtual circuits, they rely on datagrams.

So, with and all these things we have already discussed earlier. Datagrams are sent independently from one node to another as a sequence of datagrams, they are all independent ok. And the same message can be broken up into several packets, and each of this package or datagrams may follow different routes, so may follow different routes. So, delay may be variable, the arrival order at destination may be different, this already we have discussed earlier all right.

(Refer Slide Time: 09:39)



Now, this is a diagram which shows some of the TCP/IP family member, there are more in fact, so the most important ones I showed here shown here. So, in the lowest level this is the data link layer, sometimes it is called the data link and hardware layer as I said physical layer different names.

So, if you have an Ethernet connection, you will be using an Ethernet layer here, this will be at Ethernet network card that will constitute the lowest layer. This is your network layer, this is here you have the network layer. Now, at the network layer, the most important protocol is the internet protocol or IP. IP is the most important protocol that is responsible for routing of the packets right.

Now, in addition, there are some other protocols, I will briefly talk about this, after this ICMP, IGMP, ARP, RARP, I will talk about these. They all work at the network layer level. At the higher layer transport layer there are two alternate protocols TCP and UDP, Transmission Control Protocol and User Datagram Protocol.

And at the topmost layer, application layer, you can have any arbitrary user applications running. Now, in addition as part of the TCP/IP, there are some specific applications also which are also included in the family like file transfer protocol, trivial file transfer protocol, simple mail transfer protocol, all our mail email is send and receive using this SMTP protocol, Simple Network Management protocol, domain name system and so on, there are many others ok. So, this is just a very quick picture.

(Refer Slide Time: 11:48)

- **Address Resolution Protocol (ARP)**
 - Map IP addresses to hardware (MAC) addresses.
- **Reverse Address Resolution Protocol (RARP)**
 - Map hardware addresses to IP addresses.
- **Internet Control Message Protocol (ICMP)**
 - A network device can send error messages and other information.
- **Internet Group Management Protocol (IGMP)**
 - A node can send its multicast group membership to adjacent routers.

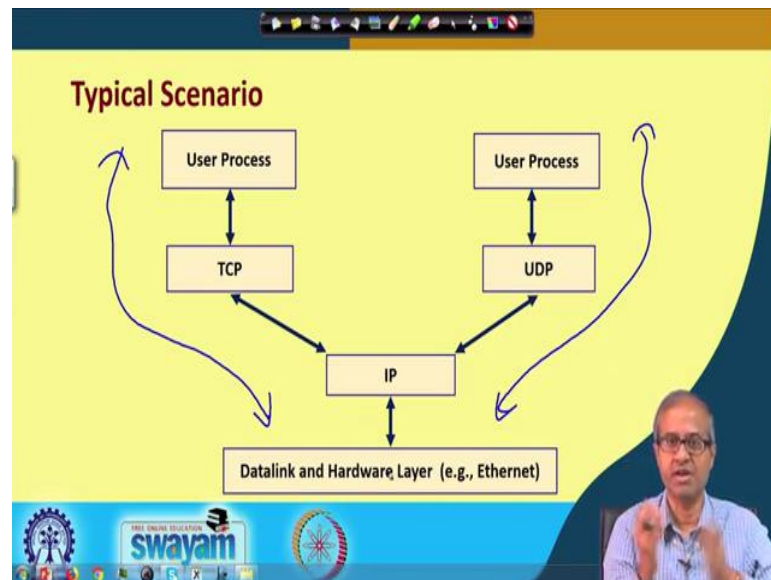
Now, talking about these four, I will very briefly talk about these. This Address Resolution Protocol or ARP, this is basically used to convert the IP address, well each computer is having an IP address, will have an IP address. This IP address is converted to an hardware address which is called a MAC address; MAC address is referred to as a hardware address.

The Reverse Address Resolution Protocol or RARP does the reverse, it converts the MAC address or hardware address to IP address. Now, whenever data are flowing in a network in a LAN this ARP and RARP protocols are very useful, they are used quite widely, we will be, I mean will be discussing this later again.

Now, then we have this ICMP or Internet Control Message Protocol which is mostly used for a device on the network to send error message to other members in the network ok, like some service not available, something not found, etc. And finally, Internet Group Management Protocol, IGMP.

So, here a node can communicate with its adjacent routers, some information about its multicast group membership. Let whenever it wants to do some broadcast or multicast selectively, it wants to send data to some set of nodes. So, it can inform the routers that this is the kind of multicast groups that you want to send data or receive data from. So, these protocols are roughly used for these purposes.

(Refer Slide Time: 13:45)



Now, a typically scenario I am showing here. Well, here I am showing two paths; one is a TCP path and other I am showing the UDP path. Well, here the application will be some user process, which will be using TCP at the transport layer level. TCP will be using IP at the network layer level and IP will be using the lowest layer hardware layer level or you can use UDP, the user process may be communicating with UDP, IP then hardware layer level.

So, what I mean to say is that when you are developing some application, you may either chose to, choose to use TCP or you may choose to use this UDP both options are available to you. TCP has some features, UDP has some features, depending on your suitability, what exactly you need, you can use either TCP or UDP. But the point to note is that below TCP and UDP, you have a common member IP. So, IP is the network layer, it is a basic datagram routing service that is common, but above IP, you have a choice, you can have either TCP or UDP.

(Refer Slide Time: 15:07)

What does IP do?

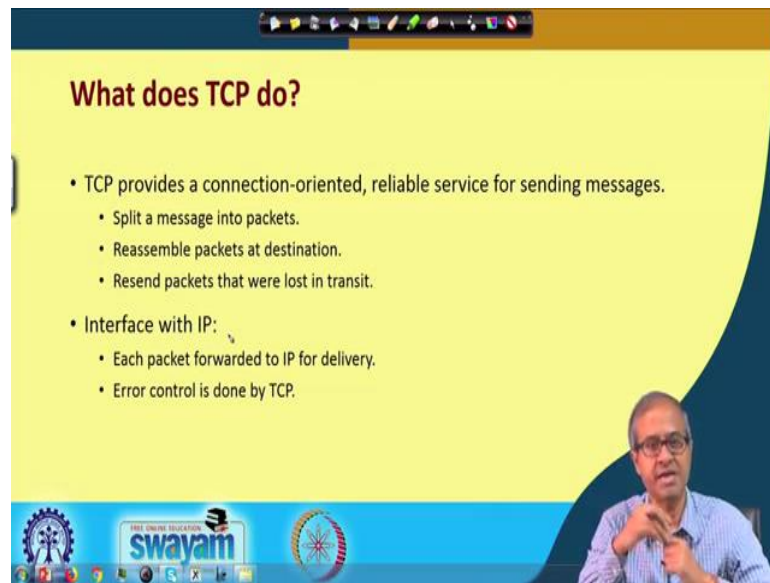
- IP transports datagrams (packets) from a source node to a destination node.
 - Responsible for routing the packets.
 - Breaks a packet into smaller packets, if required.
 - Unreliable service.
 - ❖ A packet may be lost in transit.
 - ❖ Packets may arrive out of order.
 - ❖ Duplicate packets may be generated.

The slide features a yellow background with a blue header and footer. The footer includes the 'swayam' logo and a video inset of a man with glasses speaking. A blue bracket groups the three sub-bullets under 'Unreliable service'.

Now, very briefly let us here, let us look at the functionality. Well, what does IP do, while IP is responsible for routing of the packets or the datagrams, it transports datagrams, sorry datagrams or the packets from a source to a destination. This is called routing. So, it will decide which node to forward it the packet next, that next node will again decide where to forward that packet next, this way the packet will finally reach the destination. So, the network layer or the IP layer in each of these nodes will be responsible for this routing or packet forwarding.

This IP has an additional responsibility will, if it finds that a packet is large, too large, it may break a packet into smaller packets, this is called fragmentation, this we shall see. The point to note is that IP uses datagrams and we mentioned earlier, when you talked about datagrams is that, datagram is an unreliable service; unreliable in the sense that some datagrams might get lost, some duplicates might get generated, datagrams maybe received out of order at the destination. So, such scenarios may happen. So, means if you are using IP, you must be, you must be aware of these problems that some packet may actually get lost fine.

(Refer Slide Time: 17:00)



What does TCP do?

- TCP provides a connection-oriented, reliable service for sending messages.
 - Split a message into packets.
 - Reassemble packets at destination.
 - Resend packets that were lost in transit.
- Interface with IP:
 - Each packet forwarded to IP for delivery.
 - Error control is done by TCP.

TCP understands that IP that is there below is not reliable. TCP tries to make the network connection reliable by taking some additional responsibility. How, TCP will explicitly check whether the data that is being sent is being correctly sent to the final destination, because the destination will be sending back an acknowledgement. So, the sender will know whether it was received correctly or not. If it sees that it was not received correctly, then the data will be transmitted again, so that some kind of reliability is maintained.

So, essentially TCP tries to provide a connection oriented reliable service. Reliable service, in this context if some packet gets lost, it will get retransmitted. Connection oriented in the sense that the application running on the receiving end will have an illusion that well as if I have some kind of a connection oriented connection, some kind of a virtual circuit or some kind of circuit switching is going on, data is coming and the data is being received in the same order, but as I said, IP does not guarantee that, IP may receive the packets out of order.

So, now it will be the responsibility of the TCP layer to order the packets in the correct order, and then forward it to the application, so that the application, we will feel that well everything is fine, the packets have coming in the same order, but it is TCP which is ensuring that. So, TCP will be splitting a message into packets; reassembling the packets at destination; and if some packets will lost as I said, it will be resending. So, all this services are done by TCP.

And for actual data transmission, I mean it interfaces with IP in the way that each packet it sends it to IP for delivery. And IP will be using its own rules, and its own routing tables to transmit and receive the packets. And if there are some errors happening, well IP will not handle any errors, it will be handled by TCP. TCP will check that whether all the packets are arriving correctly or not. And if some packets are missing, it will be requesting the sender to send it again ok, something like this is happening.

(Refer Slide Time: 19:54)

What does UDP do?

- UDP provides a connectionless, unreliable service for sending datagrams (packets).
 - Messages small enough to fit in a packet (e.g., DNS query).
 - Simpler (and faster) than TCP.
 - Never split data into multiple packets.
 - Does not care about error control.
- Interface with IP:
 - Each UDP packet sent to IP for delivery.

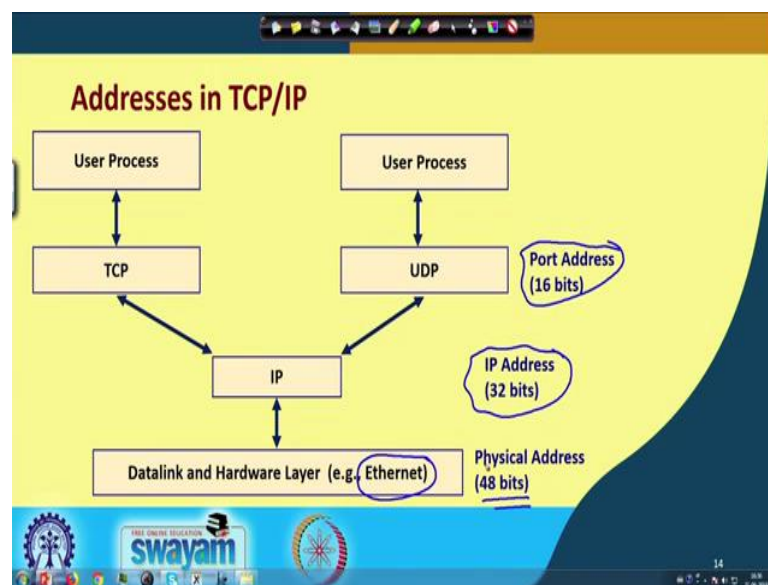
A hand-drawn diagram of a packet header is shown, consisting of a box divided into two sections. The left section contains the letter 'H' and the right section is empty.

A UDP is a simpler version of TCP, which is basically quite similar to what IP is, just a little thing extra is there beyond IP. You see UDP does not provide any kind of reliable connection. So, what UDP says is that it provides the connectionless just like a datagram, unreliable, just like a datagram for sending datagram that means you can see UDP and IP are very similar. It just provides the transport layer interface to directly interface with IP. It does not provide any reliability like TCP; it does not order the packets like what is done by TCP, it simply receives the packet as it comes.

There are some applications where even if some packets get lost, you do not care. Like periodically some network equipments are sending some status information to a central server that well I am good, I am good, or what is this status, well even if 1 or 2 packets get lost in between it does not matter, because anyway the next packet will be coming again very soon. So, I will know about the status.

So, there are some applications where reliability of communication is not important, and we can use UDP, because UDP is much simpler to use, their packet format is simpler and it is also faster. We shall see later that how the packet formats look like right. So, UDP is simpler, because it is simpler that is why it is faster. You see as I told you with each data some headers get added up. For UDP the header is small; for TCP the header is much bigger that is one way you can understand why it is faster ok, and it does not care about error control; if there is an error, let it be. Some packets are lost, fine; so it does not care about it, all right fine, ok.

(Refer Slide Time: 22:15)



Now, the point to note is that in this TCP/IP, the simplified this 3-layers, there are several addresses which are coming into play. At the lowest layer hardware layer, well most commonly we see something called Ethernet networks that is there at the physical layer level in our local area networks. So, at the lowest layer, this is the most common kind of network we encounter is called Ethernet. Yes, this is the Ethernet. Now, in this, in Ethernet there is a concept of Ethernet address which is 48-bit address. The Ethernet cards which you plug in your computer, which is plug in your laptop, they all have an Ethernet interface and they have a 48-bit, so called MAC address or Ethernet address that is unique. Each such Ethernet card in the world that is manufactured that has a unique 48-bit address.

At the network layer level, at the IP, you talk about something called IP address or internet protocol address that is a 32-bit address that identifies a supposedly unique address of a

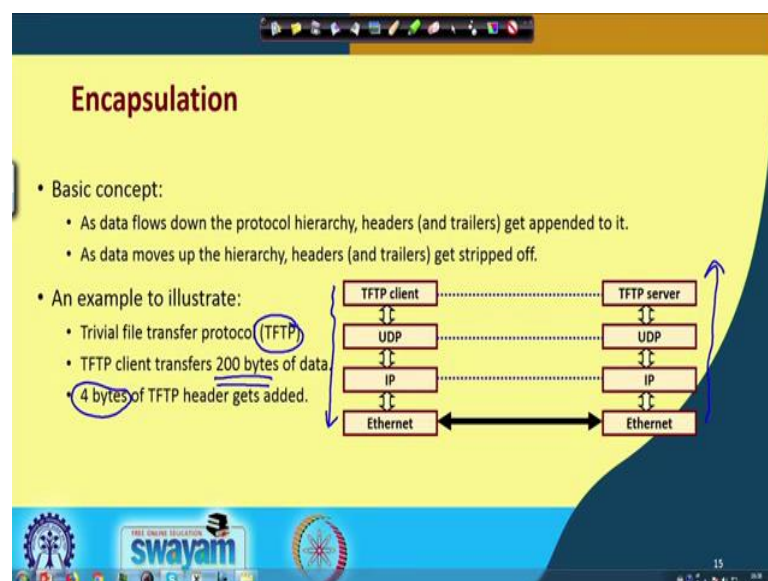
network interface. Whenever you connect a computer to a network, that network interface will be having an IP address associated with it; and that IP address also is supposed to be unique, but that is a 32-bit address.

And at the transport layer level you see there are so many applications or user processes which are using the network. How to distinguish that which user process is sending data, which user process is supposed to receive data? There is a concept of a port number or a port address. Port address actually uniquely identifies a particular application or a user process running on your machine.

So, whenever a process sends a data, a data packet, the corresponding port number is carried with the packet. And whenever say a packet comes back, it is received, there is a destination port number, that destination port number will tell you where to forward that packets. There may be many applications or processes, user programs which program to forward it to.

So, this port address basically uniquely identifies one of the user processes or user programs running on a machine. IP address identifies uniquely the network interface. And physical address as I said this is a characteristic of the network interface card, and every network interface card has a unique network address.

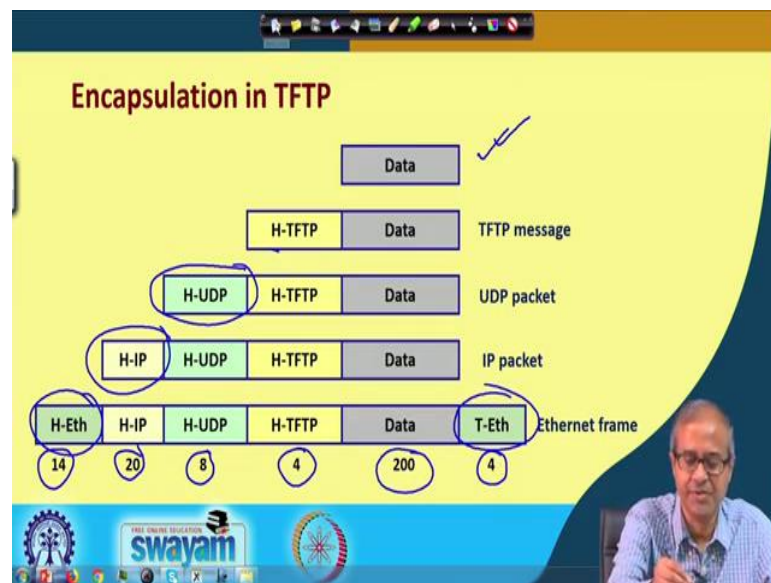
(Refer Slide Time: 25:36)



Well, there is a concept called encapsulation. Let me briefly talk about it with the help of an example. You see, you have seen the network protocol stack. So, in the network protocol stack, data flows in one direction, and the receiving end, it will flow in the reverse direction. Now, as data flows down, some headers and sometime some trailers are added to a packet. And on the other side, as the data moves up this headers are getting removed or stripped out progressively ok.

So, here I will take an example of a very simple application TFTP. TFTP is a file transfer application, it stands for Trivial File Transfer Protocol. So, as an example I am assuming that there is a TFTP client, a program which is trying to send 200 bytes of data to a TFTP server ok. Now, for TFTP protocol, the header is 4 bytes, that is defined. So, with this assumption let us see.

(Refer Slide Time: 26:57)



This is the data which you want to send, this is 200 bytes; this is 200 bytes. So, when it goes to the TFTP application. So, TFTP application appends a header. This is a header of TFTP which is 4 bytes. This is the data which TFTP application generates. It senses it down to the transport layer to UDP. TFTP uses UDP, does not use TCP that is why it comes to UDP layer.

Now, UDP header is 8 bytes, this we shall see later. So, at the UDP layer an 8-byte of header gets added. Then it comes to the IP layer, for IP layer 20 bytes of header gets added.

And at the lowest level, it is the Ethernet, at the Ethernet layer, 14 bytes of header and 4 bytes of trailer gets added, a header gets added here, a trailer gets added here.

So, ultimately in the network, in the LAN whatever data gets transmitted, it is this whole data you see how many bytes $14 + 20 + 8 + 4 + 200 + 4$, so many bytes of data will finally, get transmitted, but ultimately I was trying to transmit only 200 bytes. So, the remaining things are overhead, networking overhead, this is something you need to remember.

So, as I said as we move down from the application down to the lowest layer these headers slowly get added, but at the receiving end when you receive this whole thing, so it will again be sent up to the application and here the reverse too will happen. Step by step this headers and trailers will get removed; in the next step, this will get removed; next step, this will get removed; next step this will get removed, and finally, we get our 200 bytes of data right, this is how the thing works.

So, with this we come to the end of this lecture. So, here we had talked about some aspects of the TCP/IP protocol. We shall be we shall be continue with our discussion on TCP/IP protocol in our next lecture as well.

Thank you.