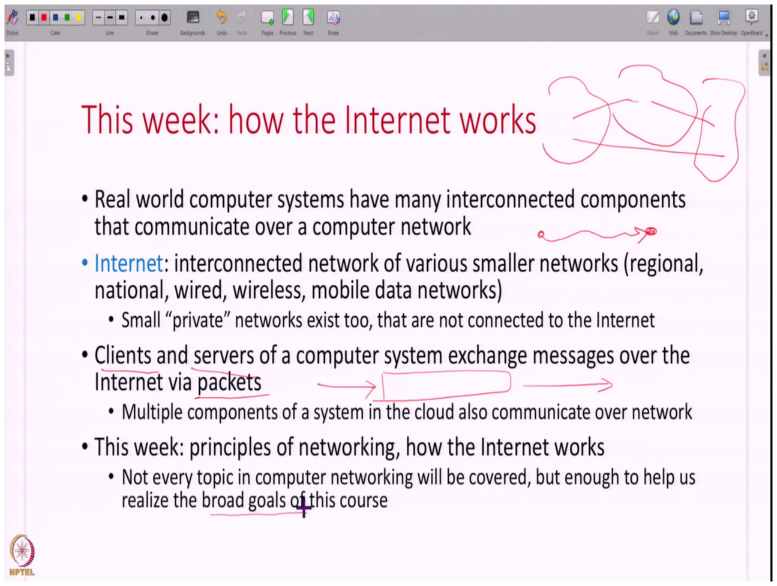
Design Engineering of Computer Systems
Professor. Mythili Vutukuru
Computer Science and Engineering
Indian Institute of Technology, Bombay
Lecture 31
Introduction to computer networking

Hello, everyone. Welcome to that twenty first lecture in the course Design and Engineering of Computer Systems. So, in this week, we are going to study a little bit more detail on how computer networks work, and specifically, how the Internet works. So, this will tie together in the broad theme of the course where we are trying to understand how real world computer systems work. So, let us get started.

In this first lecture of the series of five lectures on how the Internet works, I am going to give you a brief overview of the various components of the Internet, and then in the next four lectures we are going to get down into more details.

(Refer Slide Time: 00:56)



This week: how the Internet works

- Real world computer systems have many interconnected components that communicate over a computer network
- Internet: interconnected network of various smaller networks (regional, national, wired, wireless, mobile data networks)
  - Small "private" networks exist too, that are not connected to the Internet
- Clients and servers of a computer system exchange messages over the Internet via packets
  - Multiple components of a system in the cloud also communicate over network
- This week: principles of networking, how the Internet works
  - Not every topic in computer networking will be covered, but enough to help us realize the broad goals of this course

So, here is a brief overview of what is the Internet. So, the Internet is nothing but a collection of various networks, different organizations, regional, national networks, wired, wireless networks, all of them, there are different networks that are all connected and talking to each other, exchanging information that is the Internet. And all real world computer systems have many connected components which are distributed over this large Internet.

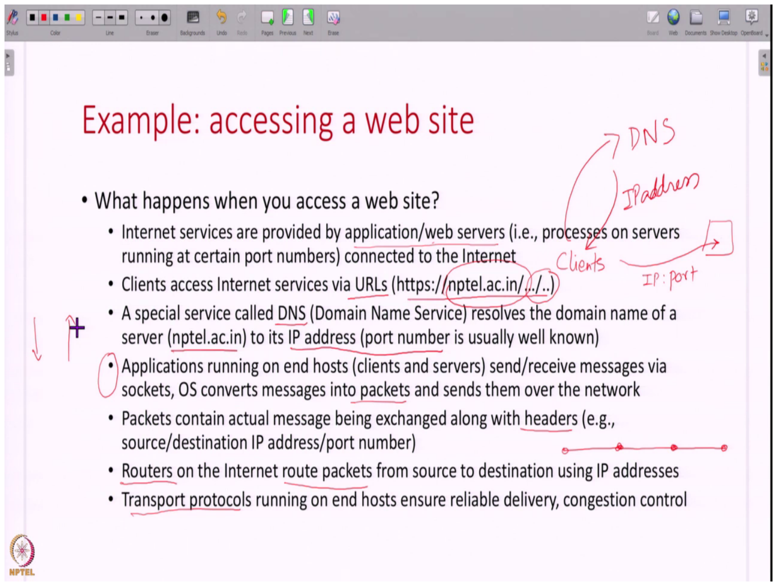
For example, if you are accessing, say, a website, there are some computer out there that has all the information, the content of the website and some client out here that connects over the network, and accesses this content. So, understanding how networking works and how the internet works is essential to understand any concept in computer systems.

So, typically, computer systems are designed as clients and servers. And these clients and servers talk over the network and exchange information. And the unit of information exchange on the network is called a packet. A packet is nothing but a stream of bytes. So, a client or server, each of which put some information in a packet, sends it to the other side, and so on.

So, in this week, as I said, we are going to cover some basic principles of how networking works and how the Internet works. Note that there are full courses dedicated to this topic. So, I am not going to be able to cover a full course worth of content here, but just enough to help us understand the broad goal of this course, which is how real world computer systems work. So, with that introduction of what this week is about, let us get into the details.

So, I will start with a simple example. So, suppose you open your browser and you visit a website, say an e-commerce website or to book your tickets for travel or any such thing, so suppose you access a website. So, let us just see through all the steps of what happens when you access a website.

(Refer Slide Time: 03:07)



Example: accessing a web site

- What happens when you access a web site?
  - Internet services are provided by application/web servers (i.e., processes on servers running at certain port numbers) connected to the Internet
  - Clients access Internet services via URLs (https://nptel.ac.in/../..)
  - A special service called DNS (Domain Name Service) resolves the domain name of a server (nptel.ac.in) to its IP address (port number is usually well known)
  - Applications running on end hosts (clients and servers) send/receive messages via sockets, OS converts messages into packets and sends them over the network
  - Packets contain actual message being exchanged along with headers (e.g., source/destination IP address/port number)
  - Routers on the Internet route packets from source to destination using IP addresses
  - Transport protocols running on end hosts ensure reliable delivery, congestion control

So, here is a rough outline of what happens. So, Internet services, websites or any other applications, ticket reservation to social media, everything, all of these services are provided by servers, application or web servers. So, what are the servers? These are nothing but processes that are running on a certain computer, listening at a certain port number. We have seen how to write a simple server like this using the socket API. So, of course, real world servers are much more complex, but the basic idea is the same.

So, you have a process or a set of processes listening on one or more machines at certain port numbers and they have the content of the website, they have information about all the products being sold at an e-commerce site or information about all reservations in a ticket booking portal, whatever it is, so the servers have all the information. And clients, users of the computer system access these servers over the internet using a URL.

So, for example, if you want to access courses on NPTEL, you will type this address in the address bar of your browser. This is called the URL. You will put that in and the clients will reach the content at the server. So, what happens when you access this URL? What happens when the client accesses this URL?

So, this URL has two parts. It has a domain name which is nptel.ac.in and some other information about what content you want to access at that domain. So, if you access this URL, the domain name is nptel.ac.in and some other information. So, what the client will do is it will take this domain name and using this domain name it look up a service on the internet called the DNS, clients will contact the DNS, and the DNS will give you back the IP address of the server.

So, this domain name is nothing but an easy to remember name, which can be used instead of an IP address to talk to a server. So, you use the domain name, DNS will give you the IP address and then use this IP address and the port number. Usually port number is well known. Web servers listen at port 80, and so on these numbers are well known. So, you get the IP address from DNS, then use the port number and together you will be able to connect to the server using the combination of IP and port. And we have seen this. The connect system call, for example, that we have studied last week, takes these IP address port number and opens a connection to the server.

And then applications that are running on end hosts like client, server like the socket programs we have seen. Once they are connected this way, they will send and receive messages. You can either use sockets, you can use other programming libraries, it does not matter. In the end, all of them work the same way where you exchange messages. And these messages when you write into a socket, the operating system converts these messages into packets and sends them over the network. So, the server has opened a socket, the client connects to the server socket, write some data into the socket, then this data goes over the network as packets.

Who makes these packets? The OS makes, converts the messages written into sockets into packets and sends them over the network. And what is this network? This network is nothing but a series of routers which are boxes that pass on the packets from one point to the other from the source to the destination. So, all of these routers on the Internet, they will look at this destination IP address. You say this packet has to go to so and so server, the IP address will be there. And all of these routers will look up that IP address and forward the packets from the source to the destination.

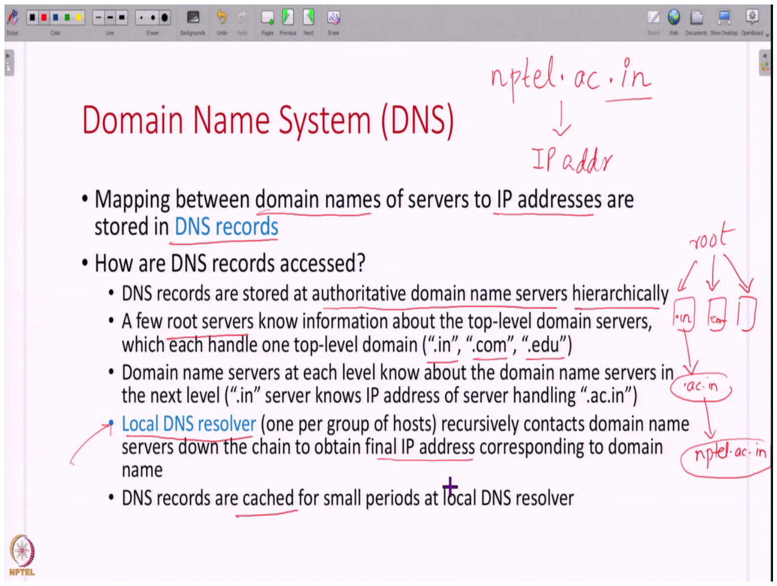
And that is why the packets in addition to the actual message, whatever you want to access a certain course or a video in addition to that message along with that every packet will also have some headers like I want this packet to go to so and so IP address to so and so port number all of that information also will be there in the packet. Along with the actual message, you will also have the headers. And the routers will route the packets from the source to the destination using the information in these headers.

And finally, you also have something called transport protocols that are some extra mechanism running at the client and server to ensure various things like reliable delivery, congestion control. For example, if the network, the path going to the server is very congested, you might want to slow down, send a little bit later, if some message was lost, you might want to resend it again, there are all of these mechanisms that are implemented by transport protocols. So, this is a rough idea of how a client accesses service, any service on the Internet.

And in the next few lectures, we are going to understand each of these different components in more detail, what are transport protocols, what are routers, how does routing happen from the source to the destination, what are the various headers that are present in packets, all of these things, what is DNS, all of these things we are going to study in more detail.

The one part that we have already seen last week is this part which is writing messages into sockets and reading from sockets. How that works, we have already seen. Everything else, all other pieces of the puzzle we are going to discover this week.

(Refer Slide Time: 9:13)



Domain Name System (DNS)

- Mapping between domain names of servers to IP addresses are stored in DNS records
- How are DNS records accessed?
  - DNS records are stored at authoritative domain name servers hierarchically
  - A few root servers know information about the top-level domain servers, which each handle one top-level domain (".in", ".com", ".edu")
  - Domain name servers at each level know about the domain name servers in the next level (".in" server knows IP address of server handling ".ac.in")
  - Local DNS resolver (one per group of hosts) recursively contacts domain name servers down the chain to obtain final IP address corresponding to domain name
  - DNS records are cached for small periods at local DNS resolver

So, first let us begin with the domain name system or DNS. What is DNS? DNS is a service that is available on the Internet using which you can find out IP addresses of various hosts. So, the mapping between domain names that is human readable names of servers to the IP addresses these mappings are stored in DNS records. And these DNS records are available for all clients to query on the Internet. So, how are these DNS records stored and how are they accessed?

So, these DNS records are stored at what are called authoritative domain name servers. So, there are some special servers which do not serve regular content like videos or web pages, but these servers actually maintain these DNS records and give them out to whoever needs this information. And of course there are like a large number of domain names and IP addresses on the Internet.

So, is everything stored in one server? No, it is stored hierarchically. What does this mean? So, if you have a domain name like nptel.ac.in, so firstly there are, what are called, root DNS servers. So, these have information about all the top level domains like dot in, which is our top level domain here or dot com which is the most popular top level domain or dot edu. There are several top level domains like this specific to various areas or countries and so on.

And there are certain servers that handle all information about that top level domain like there is a server that knows all information about domain names ending in dot in, something else about dot com, and so on. And then there are root servers that have information about all of these top level domain servers. So, if you want to resolve this name into an IP address, you will first go to the root server. The root server says, okay, I do not know about this name, but you talk to the dot in server and it will give you the address of the dot in server.

Then, you will go to the dot in server and the dot in server will give you the IP address of the dot ac dot in server. So, the domain name servers at each level will know about the name servers at the next level. Dot in will know about all the other dot in, dot ac dot in, dot or net dot in, all those other servers that information will be known by the dot in server.

And then you talk to the ac dot in server it will tell you about the nptel.ac.in server. So, every server knows about information, the IP address of the name servers at the next level. And then that way you keep traversing this tree hierarchically starting from the root to the top level domain server to other, other, other domain servers, and finally, you will reach a server that knows about the IP address of nptel.ac.in.

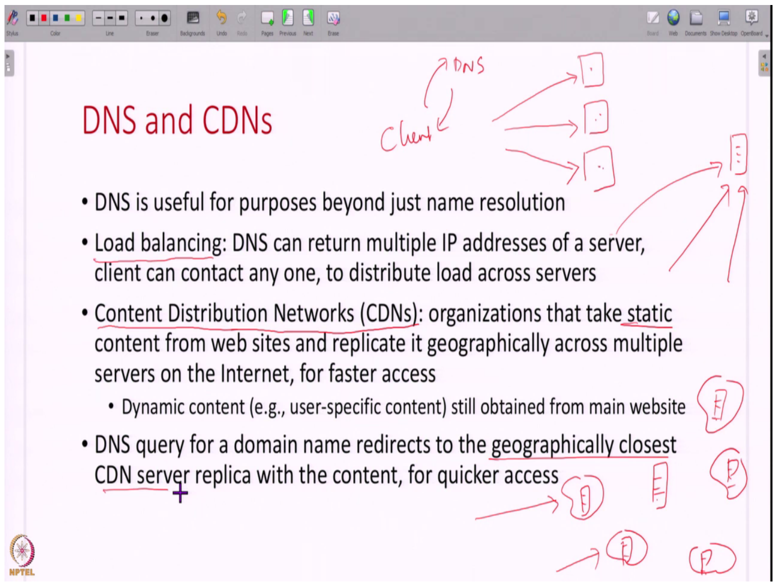
And who is doing all this traversal? There are special pieces of software called DNS resolvers. So, in every organization, every group of hosts, together, they will set up a DNS resolver. And this DNS resolver will do all this work for you. So, in your network, if you are accessing nptel.ac.in, then you will first talk to your DNS resolver. And then this DNS resolver will talk to the root server, top level domain servers, everybody else. It will talk to ac dot in server, then it will find out the IP address of nptel.ac.in and then that IP address it will return to you. So, DNS resolvers resolve a DNS name by walking down the name hierarchically.

And once you obtain the final IP address, these DNS records, this information about this mapping from this name to the IP address, all of this information is also cached for small periods of time at the local DNS resolver. Why, because if somebody has resolved this name to an IP address, and immediately somebody else will ask, then you do not have to do all of this again, go to the root, go to top level domain, all of this you do not have to do it again, you can just remember.

So, the final result, as well as the IP addresses of all of these intermediate servers that you had to talk to all of that is also cached at DNS resolvers. And of course, you cannot cache it forever because what if this IP address changes. So, therefore, these caches have some information about how long you should cache these DNS records for, and accordingly, they are cached.

So, DNS is a very important component of the Internet today. And if DNS does not work, you cannot access any service on the Internet because it is very hard for us to remember what the IP address of that service is. We always remember the domain name. And therefore, DNS needs to be up all the time, it needs to be fast. So, the performance of DNS is very crucial for various Internet applications today.

(Refer Slide Time: 14:42)



DNS and CDNs

- DNS is useful for purposes beyond just name resolution
- Load balancing: DNS can return multiple IP addresses of a server, client can contact any one, to distribute load across servers
- Content Distribution Networks (CDNs): organizations that take static content from web sites and replicate it geographically across multiple servers on the Internet, for faster access
  - Dynamic content (e.g., user-specific content) still obtained from main website
- DNS query for a domain name redirects to the geographically closest CDN server replica with the content, for quicker access

So, the next thing is what else is a DNS system useful for? Is it just to resolve names to IP addresses? Actually, people use DNS for many other purposes. For example, for load balancing, so suppose this nptel server or any server is not implemented at one IP address, but there are multiple processes which are all serving content. Why, because maybe one server can get overloaded. Therefore, you want some users to access content from the server, some from the server, some from the server, that is you want to do some load balancing across multiple servers, and each of these servers will have different IP addresses also.

So, how do you do that? The way you will do that is you will store all these different IP addresses in DNS and DNS will return either all of them or one of these, whatever. So, the client, when you talk to DNS, you will get back all of these IP addresses, then some clients will pick this server, some will pick this server, some will pick this server and the load balancing will happen. So, not that DNS does not have to return exactly one unique IP address for every domain name, it can return multiple IP addresses for the purpose of load balancing, so that clients have the choice.

Then the other important concept where DNS is useful is the concept of what are called content distribution networks or CDNs. So, what are CDNs? CDNs are basically organizations that take content from websites and replicate this content all around the Internet, so that you can access it faster. So suppose there is a website that has some web page and say this web page is located at some server in India, then clients from all over the world will be coming to this one server in India and accessing it and all the links can get blocked due to too much traffic.

So, instead of that, what you will do is this content can be given to companies which are called CDNs, and what these companies they have enough data centers, networks all around the world. So, they will take this content of a website, make copies of it, and store it all around the world. You say one in America, one in Africa, one in Europe, whatever, different places they will have their data centers and they will replicate this content.

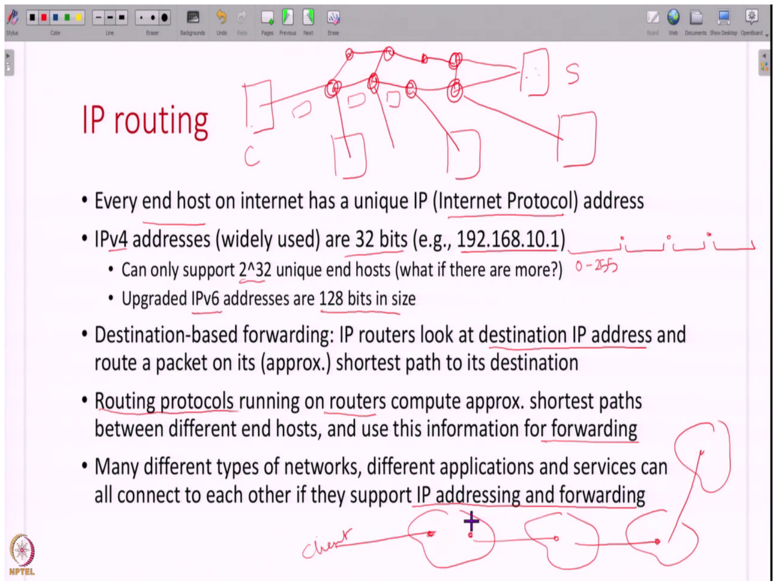
So, now if somebody from the US. wants to access this website, this content, they do not have to come all the way to India, they will just use a local copy from the CDN that is located in USA. Somebody else from Europe can simply access the copy that is there in Europe. So, in this way what CDNs do is they will replicate the content of a website or any other Internet service all over the Internet in geographically distributed locations, so that everybody can access their closest replica and get content faster.

Not that you can only do this for static content, that is content that does not change on a website. If you have content that is constantly changing, then of course, you cannot give it to a CDN. You still have to go to the main server for that. And CDNs, how do CDNs work? They work using DNS.

So, when you resolve a domain name to an IP address in DNS, then DNS will find the closest replica of the content geographically that is a CDN has replicated content, it will tell DNS, okay, if somebody from this region request for this domain name, give them this IP address. If somebody from another region asked for the same domain name, give them the IP address of this server that is geographically closer to them.

So, CDNs work along with DNS in order to redirect requests to the geographically closest CDN server replica. This ensures that content is accessed much faster on the Internet today. So, once again CDNs are also a very important part of the Internet infrastructure today and are hugely responsible for ensuring that we actually access content on the Internet much faster. So, next, we come to IP routing. So, we will study this in more detail in a later lecture. But I would like to give you a high level overview of what is IP routing.

(Refer Slide Time: 19:05)



IP routing

- Every end host on internet has a unique IP (Internet Protocol) address
- IPv4 addresses (widely used) are 32 bits (e.g., 192.168.10.1)
    - Can only support 2^32 unique end hosts (what if there are more?)
    - Upgraded IPv6 addresses are 128 bits in size
- Destination-based forwarding: IP routers look at destination IP address and route a packet on its (approx.) shortest path to its destination
- Routing protocols running on routers compute approx. shortest paths between different end hosts, and use this information for forwarding
- Many different types of networks, different applications and services can all connect to each other if they support IP addressing and forwarding

So, every computer or every end host on the Internet has a unique IP address assigned to it. Actually, it is not the end host, it is the interface, but we will come to that a little bit later. So, for now, just assume that every end host on the Internet has a IP address given to it. And what is an IP address? IP stands for Internet Protocol. And what is an IP address? IP address, there are two versions of this protocol, v4 is what is commonly used.

So, an IPv4 address is nothing but a 32 bit string. And the way you write it is something like this. You take this 32 bit, split it into four groups of 8 bits each, and these 8 bits will be a number between 0 and 255 in decimal, and then you will take this number dot this number dot this number dot this number and that is how you will represent an IP address in the dotted decimal notation.

And inside this notation, if you look at it, it is nothing but a string of 32 bits. This uniquely identifies every end host, whether you are a client, server, you are an e-commerce website, you are a browser, whoever you are, you are browsing the Internet, you are watching videos, whoever you are on the Internet, you will get a unique IP address. So,

32 bits are also not enough, because you can only address 2 to the 32 hosts. What if you have more hosts than that, and we do today. And therefore, the Internet is upgrading itself to a next version, IPv6, where the addresses are 128 bits in size. But let us leave that aside for now. So, you have some IP addresses, whether IPv4 or IPv6 that uniquely identify hosts on the Internet. And then all of these hosts are connected by routers.

So, there are many routers on the Internet today, whose job is to connect all of these millions and billions of computers that are part of the Internet, these routers, this infrastructure of routers connects all of these end hosts. And whenever one end host sends a packet to another end host that packet is forwarded to the correct destination by these routers. And how do the routers do this? They look at the destination IP address, and then route a packet on approximately, of course, you will not go all round and round and send it, you will try to send it on the shortest possible path to the destination.

So, suppose here is a client and here is a server and there are many paths to go to this server, then each router on the path will pick the shortest path from the client to the server and forward data from the client to the server, server to the client and so on. So, the client is sending many packets, and each of these packets will have the address of the destination, will have the server's address inside them and each router will look at this destination address and send the packet to the server.

So, then the next question comes up, how do these routers know what is the path? How does this router know I should send the packet here, how does this guy know I should send the packet here. So, which is why, one way to do this is you could have had sort of a centralized design where somebody knows the entire topology of the Internet. And they will tell every router, hey, you send the packet here to reach this guy and so on. This information can be told to all the routers.

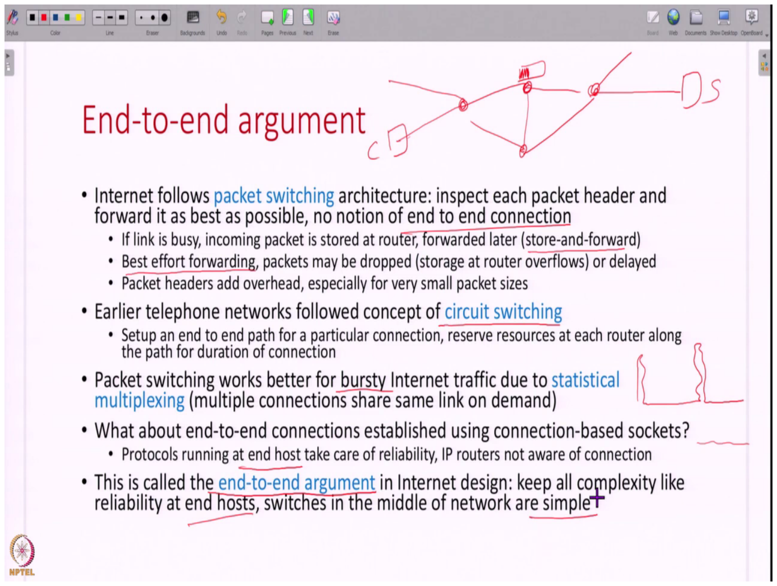
But of course the Internet is so huge, it has billions of hosts and nobody knows the entire topology. So that is hard to do. Instead, these routers run what are called routing protocols. That is they exchanged some messages between themselves to figure out all of these paths on the Internet. So, these routing protocols run at the routers. So, these routers talk to each other, exchange some information, and as a result of that, they are able to compute the shortest paths between any two hosts on the Internet. And once they know the shortest path, they will use that during forwarding.

So, every router does the routing protocol part of it, which is calculating the shortest paths, and then using this information for actually forwarding the packets sent by the end hosts. And one thing to note is that the Internet is not just one coordinated network maintained by one person, it is many different networks, running many different applications set up by many different organizations and all of these are united by this IP routing, addressing and forwarding.

So, whichever network you are, as long as you can forward IP packets, you can become part of the Internet. Your server could be running in some data center managed by somebody else. Your client could be your browser on your phone inside a mobile data network does not matter. As long as everybody understands IP addresses and can forward IP datagrams, we can all talk to each other. In some sense, IP unites all of the different networks that comprise the Internet today.

So, next what I am going to talk to you about is some sort of design principles behind how the Internet was designed, what was some of the thought process that went into designing the Internet.

(Refer Slide Time: 24:52)



End-to-end argument

- Internet follows packet switching architecture: inspect each packet header and forward it as best as possible, no notion of end to end connection
  - If link is busy, incoming packet is stored at router, forwarded later (store-and-forward)
  - Best effort forwarding, packets may be dropped (storage at router overflows) or delayed
  - Packet headers add overhead, especially for very small packet sizes
- Earlier telephone networks followed concept of circuit switching
  - Setup an end to end path for a particular connection, reserve resources at each router along the path for duration of connection
- Packet switching works better for bursty Internet traffic due to statistical multiplexing (multiple connections share same link on demand)
- What about end-to-end connections established using connection-based sockets?
  - Protocols running at end host take care of reliability, IP routers not aware of connection
- This is called the end-to-end argument in Internet design: keep all complexity like reliability at end hosts, switches in the middle of network are simple

So, the Internet today follows what is called a packet switching architecture. That is, you have end hosts say your client and server, and you have a bunch of routers or devices that switch packets along the path and each of these routers does not really know that you are talking to this server. They are not dedicated to just forwarding your traffic. So, they forward traffic from many hosts to many other hosts. So, there is no notion of an end to end connection on the Internet today.

Every router, it look at every packet, every packet has header information that says where this packet should go, where it is coming from, where it should go to and every router will look at each packet independently and forward it as best as it can. And sometimes if a router is busy when a packet comes up, it may not be able to immediately send the packet. It might store the packet. That is why these routers are also called store and forward switches. When a packet comes, you store it for some time, then when the router is free, it look at the header, decide what to do with the packet and send it along.

So, sometimes some router may not know what to do with the packet, it may not have any place to store the packet, it might be too busy, it can drop the packet anything can happen, so which is why it is called best effort forwarding on the Internet today. And of course, every packet because it needs to have a header, it adds also some overhead especially for small packets. So, this is a very different architecture from what was used before the Internet came up.

When you had telephone networks, that was called the concept of circuit switching. What is circuit switching? In olden telephone networks, when you made a call, when somebody dialled, somebody else, then a notion of a path would be established along all routers between this starting point of the call and the terminating point of the call.

And all of these routers would be aware of this connection, they would reserve some resources for this connection, because why traffic is coming in continuously, they would reserve resources, and everybody along the path knew that this connection was set up and when the call ended, this connection would be taken down. That is everybody knew of the end to end connection, maintain state reserve, resources for this end to end connection. That is called circuit switching.

In the Internet today, that does not happen. If you are talking to an NPTEL server, nobody, no router on the path knows about it or cares about it. You send a packet, they will forward it that is all. So, why is this? Why have we moved away from circuit switching to packet switching? The reason is that, Internet traffic is busty.

So, you will download some set of packets from the NPTEL website, you will send a request, get some data. Then as long as you are reading that page and until you click on the next link, nothing is happening. There is no traffic being sent. So, Internet traffic is busty. That is you will send some traffic, for a long time there will be nothing, then this connection will send some more traffic for a long time there will be nothing.

So, when you have busty traffic, it does not make sense to set up a connection, reserve resources, tell everybody about it, everybody keeps state about it, remembers this connection, it does not make sense. Why, because multiple connections can share the same resources. You do not have to reserve any resources for the connection, unlike voice traffic which is more uniform and goes on continuously, voice traffic is sort of more constant rate, whereas Internet traffic is busty. Therefore, the Internet works on the principle of packet switching.

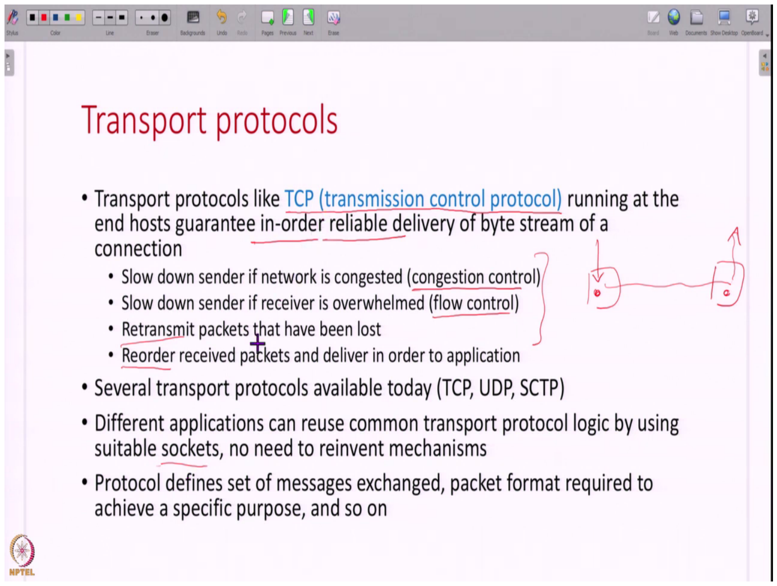
If you have something to send, I will send it, if you do not have anything well and good. But I am not going to make any special efforts for any particular connection that is the design philosophy of the Internet. But then, you might be wondering then what about all this connection based sockets, we established a connection to a server and all of that, what is happening to all of that.

So all of that information, the fact that you are talking to the server and some ways of establishing, reliability, all of that, only happens at the end hosts. So, this argument, this design philosophy is called the end to end principle or the end to end argument in Internet design. Whatever complexity you have, for reliability, you have to remember that I have sent so many packets, that packet is lost, I have to retry it again, I have to slow down, the server is too slow, all of that logic happens only at the end hosts. Nothing happens in the middle of the network.

The middle of the network the switches are simple. They will just live life one packet at a time. Look at a packet forward it that is all. They are not aware of any end to end connections, but the end hosts who are talking to each other they are aware of these end to end connections and whatever logic you have pertaining to a connection like reliability or congestion control or retransmissions, whatever you want to do specific to that connection, you will only do it at the end host.

So, this is called the end to end principle on the Internet. And this has greatly simplified the Internet architecture and this principle is responsible for the humungous growth of the Internet today, because network switches themselves are very simple. There is no complexity there. And this end to end logic is implemented in transport protocols which run at the end host.

(Refer Slide Time: 30:34)



Transport protocols

- Transport protocols like TCP (transmission control protocol) running at the end hosts guarantee in-order reliable delivery of byte stream of a connection
  - Slow down sender if network is congested (congestion control)
  - Slow down sender if receiver is overwhelmed (flow control)
  - Retransmit packets that have been lost
  - Reorder received packets and deliver in order to application
- Several transport protocols available today (TCP, UDP, SCTP)
- Different applications can reuse common transport protocol logic by using suitable sockets, no need to reinvent mechanisms
- Protocol defines set of messages exchanged, packet format required to achieve a specific purpose, and so on

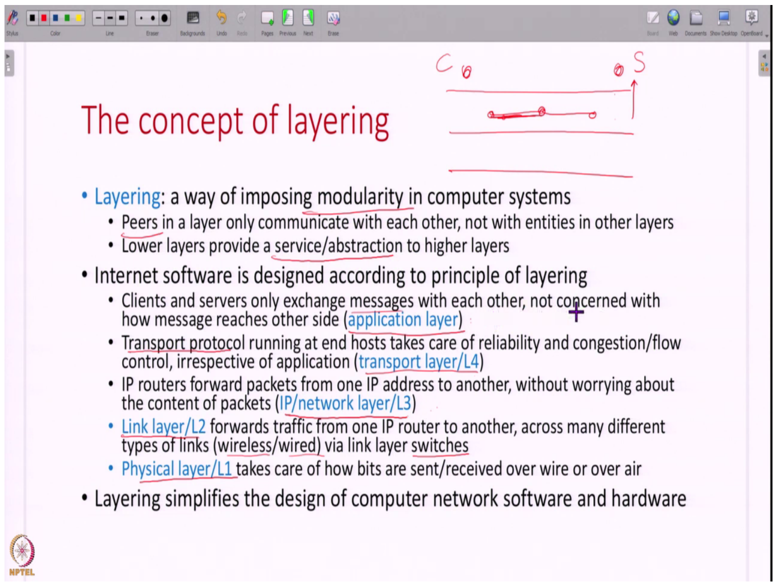
So, popular transport protocol is what is called TCP or transmission control protocol. So, this runs at the end host and does various mechanisms, has various mechanisms in order to deliver all the bytes of a connection reliably and in order. For example, the logic that is running at the end host, you will slow down the sender if you find that the network is congested and if you find or you find that the receiver is too slow you will do what is called congestion control which is slowing down the sender if the network is congested. Slowing down the sender if the receiver is congested that is called flow control.

If some packet has been lost, some router dropped it, you will retransmit that packet. If packets have been jumbled up in the network, you will reorder them, give them to the application in the correct order. When you write into a socket you will read on the other end from the socket in order, not some jumbled up garbled stream. So, all of this logic is guaranteed only by the end host, not by the routers that are in the middle.

And we have several transport protocols today, like TCP, UDP, SCTP, we will see them later. And different applications can reuse all of the functionality of a transport protocol. So, every application does not have to do congestion control, reliability on its own. If you simply use TCP sockets, you will get all of that functionality.

And what is a protocol when I say transport protocol or all of these, the word protocol is used often in networking. What does it mean? Just means a way for two entities to communicate with each other, what messages should I exchange, what packet format should I use. All of these things are specified by the protocol. And the Internet has many protocols that are in use.

(Refer Slide Time: 32:29)



The concept of layering

- Layering: a way of imposing modularity in computer systems
  - Peers in a layer only communicate with each other, not with entities in other layers
  - Lower layers provide a service/abstraction to higher layers
- Internet software is designed according to principle of layering
  - Clients and servers only exchange messages with each other, not concerned with how message reaches other side (application layer)
  - Transport protocol running at end hosts takes care of reliability and congestion/flow control, irrespective of application (transport layer/L4)
  - IP routers forward packets from one IP address to another, without worrying about the content of packets (IP/network layer/L3)
  - Link layer/L2 forwards traffic from one IP router to another, across many different types of links (wireless/wired) via link layer switches
  - Physical layer/L1 takes care of how bits are sent/received over wire or over air
- Layering simplifies the design of computer network software and hardware

And finally, the one final concept with respect to how the Internet is designed that I would like to discuss is the concept of layering. So, the Internet is built as a series of layers. Internet software and hardware is built as a series of layers and layering is nothing but a way to impose modularity in computer systems. If you have many different entities, like there are routers, there are transport protocols, there are applications, there are many different things on the Internet. And if everybody is talking to everybody else, it gets very unmanageable.

So, therefore, you divided all of these entities into layers and only the entities in one layer will talk to each other. And the entities in the other layer are not aware of the entities in one layer. So, the lower layers will talk to each other and they will provide some service to the higher layers and the higher layers will talk to each other, provide some service to the layer above them and so on. Peers in a layer talk to each other and the lower layers provide some service or some abstraction to the higher layers.

So, the Internet what are the various layers in the Internet today? So, first you have the clients and servers. You have your browser that is talking to a website all of these are clients and servers on the Internet and they exchange information, they exchange messages with each other, all of this is called the application layer. Then these messages are sent reliably by the transport layer. So, the transport layer like TCP takes care of taking this message, ensuring that it reaches reliably on this connection on the other site.

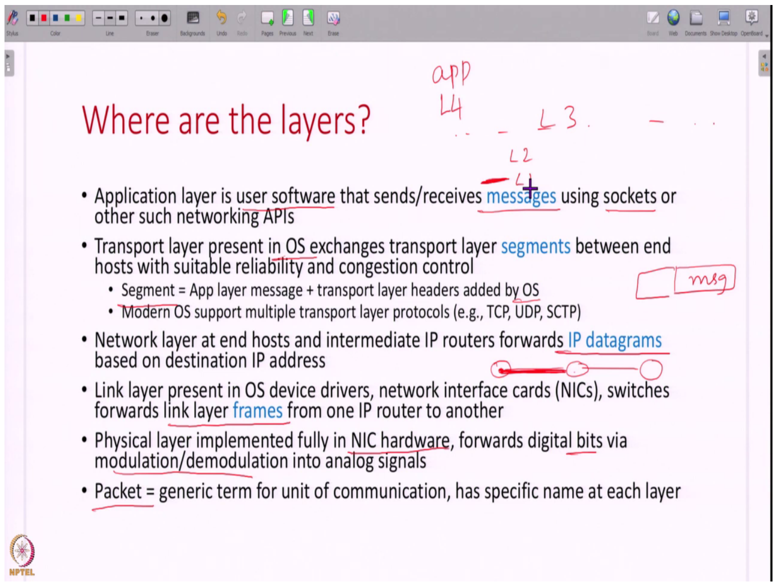
And finally, you have the next layer is you have the network layer or the IP layer. So, this is called layer 4, this is called layer 3. This IP layer is just forwarding your packets. You do not care whether it is from a browser to some server to an e-commerce website, to a ticket reservation website, you do not care. The IP router simply just forward packets from one IP address to the other. They do not know what is there inside the packets.

So, note that, can you see the concept of layering here. This guy does not care about this guy. This guy does not care about this guy. Each of them works independently to do their job. And then you have the layer two. Underneath this network layer you have what is called the link layer. That basically on each of these link how do you forward packets.

From one IP router to the other how is traffic forwarded. That is taken care of by the link layer, because you have many different kinds of links, you have wireless links, wired links, all of that is handled by the link layer. The link layer has a bunch of switches which forwards traffic from one IP address to the other.

And finally, you have the physical layer, layer one, which is actually how are bits sent over the wire or over the air. So, this concept of layering simplified the design of network software and hardware, because each layer is only worried about its functionality and not of what is happening everywhere else on the Internet.

(Refer Slide Time: 35:37)



Where are the layers?

- Application layer is user software that sends/receives messages using sockets or other such networking APIs
- Transport layer present in OS exchanges transport layer segments between end hosts with suitable reliability and congestion control
  - Segment = App layer message + transport layer headers added by OS
  - Modern OS support multiple transport layer protocols (e.g., TCP, UDP, SCTP)
- Network layer at end hosts and intermediate IP routers forwards IP datagrams based on destination IP address
- Link layer present in OS device drivers, network interface cards (NICs), switches forwards link layer frames from one IP router to another
- Physical layer implemented fully in NIC hardware, forwards digital bits via modulation/demodulation into analog signals
- Packet = generic term for unit of communication, has specific name at each layer

And where are all of these layers situated? So, the application layer, the actual layer that sends and receives messages this is usually built in the form of user software, software that sends and receives messages using sockets or any other such networking library that is your application layer. And the transport layer is located mostly inside operating systems, which exchange, so the transport layer will take the message sent by the application layer, add its own headers to it and this is called a segment.

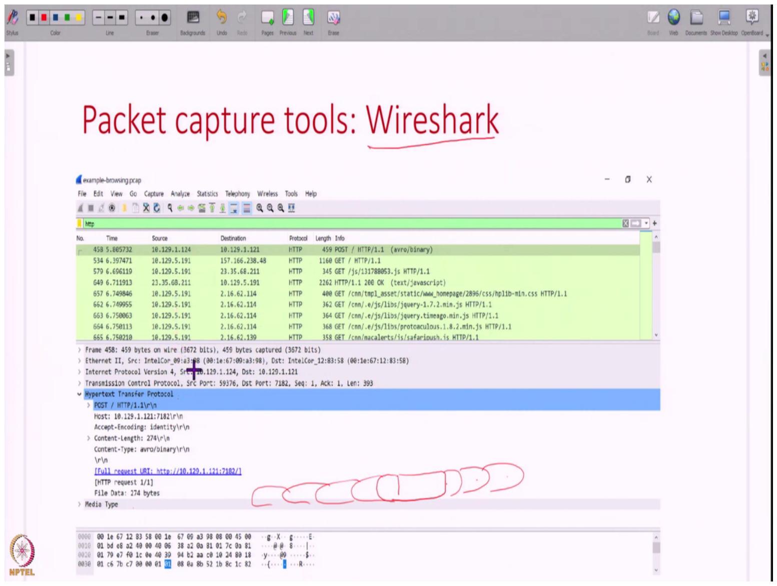
The transport layer makes a segment and all of this is done by the operating system. And then these segments are sent out on the network. And modern operating system support many transport layer protocols. And all of this transport layer logic is implemented inside the OS when you read and write into sockets.

Then you have the network layer that runs at the end host as well as all the IP routers on the Internet, all of these IP routers they run the network layer. And what they do, they forward IP datagrams. Note that the packet is a generic term, but each layer will give this packet a separate name like application layer message, a transport layer segment, an IP datagram and so on.

Then finally, the link layer, it is present in all the network cards, device drivers, switches which forward packets. And at the link layers these packets are called link layer frames between one IP router to the other. On this link whatever happens that is the link layer. And finally, you have the physical layer which runs fully in hardware inside network cards and routers and all the hardware.

It basically deals with how do you send bits and how do you convert these bits into analog signals like electromagnetic waves. There are techniques called modulation, demodulation, how you convert a bit into a wave and send it out, all of that is taken care of by the physical layer. So, this is the location of the various layers on the Internet. At the end host you have the application, you have the transport layer, then at all the routers you have the network layer, on each link you have the link layer and finally on each wire or wireless, physical medium you have the physical layer.

(Refer Slide Time: 38:05)



Packet capture tools: Wireshark

example-browsing.pcap
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
http
NPTEL

```
No.  Time      Source         Destination     Protocol Length Info
 458 5.805732  10.129.1.124   10.129.1.121    HTTP      459 POST / HTTP/1.1  (avro/binary)
 534 6.397471  10.129.5.191   157.166.238.48  HTTP     1160 GET / HTTP/1.1
 579 6.696119  10.129.5.191   23.35.68.211    HTTP      345 GET /js/131788053.js HTTP/1.1
 649 6.711913  23.35.68.211   10.129.5.191    HTTP     2262 HTTP/1.1 200 OK  (text/javascript)
 657 6.749846  10.129.5.191   2.16.62.114     HTTP      400 GET /cnn/tmpl_asset/static/www_homepage/2896/css/hplib-min.css HTTP/1.1
 662 6.749955  10.129.5.191   2.16.62.114     HTTP      362 GET /cnn/.e/js/libs/jquery-1.7.2.min.js HTTP/1.1
 663 6.750063  10.129.5.191   2.16.62.114     HTTP      364 GET /cnn/.e/js/libs/jquery.timeago.min.js HTTP/1.1
 664 6.750113  10.129.5.191   2.16.62.114     HTTP      368 GET /cnn/.e/js/libs/protoaculous.1.8.2.min.js HTTP/1.1
 665 6.750210  10.129.5.191   2.16.62.139     HTTP      358 GET /cnn/macalerts/js/safaripush.js HTTP/1.1
```

```
> Frame 458: 459 bytes on wire (3672 bits), 459 bytes captured (3672 bits)
> Ethernet II, Src: IntelCor_09:a3:98 (00:1e:67:09:a3:98), Dst: IntelCor_12:83:58 (00:1e:67:12:83:58)
> Internet Protocol Version 4, Src: 10.129.1.124, Dst: 10.129.1.121
> Transmission Control Protocol, Src Port: 59376, Dst Port: 7182, Seq: 1, Ack: 1, Len: 393
v Hypertext Transfer Protocol
  > POST / HTTP/1.1\r\n
    Host: 10.129.1.121:7182\r\n
    Accept-Encoding: identity\r\n
  > Content-Length: 274\r\n
    Content-Type: avro/binary\r\n
    \r\n
    [Full request URI: http://10.129.1.121:7182/]
    [HTTP request 1/1]
    File Data: 274 bytes
> Media Type
```

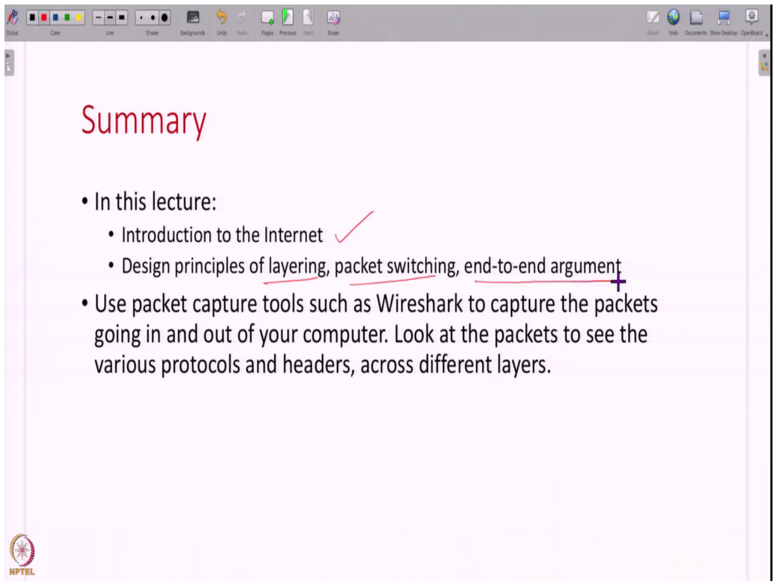
```
0000  00 1e 67 12 83 58 00 1e  67 09 a3 98 08 00 45 00   ..g..X..g.....E.
0010  01 bd e8 a2 40 00 40 06  38 a2 0a 81 01 7c 0a 81   ....@.@.8....|..
0020  01 79 e7 f0 1c 0e 40 39  94 b2 aa c0 10 24 80 18   .y....@9.....$..
0030  01 c6 7b c7 00 00 01 01  08 0a 8b 52 1b 8c 1c 82   ..{........R....
```

And what is a good way for you to visualize all of these layers is to use what a called packet capture tools. So, Wireshark is a popular example of a packet capture tool. You can install this tool on your computer and actually see all the packets flowing on your network. And if you click on each packets, if you look at, this is the interface, this is how the Wireshark's screen looks like, there are many packets.

And if you click on each packet, you can actually see all the layers. You have one payload and this payload is put inside various headers, various headers or various layers and you can also have footers, but you mainly have headers. And then how the packet is built, innermost there is some layer. There is some message that is put in this application layer headers, http is the application layer protocol, then there TCP headers.

You can click on each of these and see what are all the headers added and what are all the layers at work and each not that it adds its own headers and on the other side when you are receiving a packets you will keep removing these headers.

(Refer Slide Time: 39:15)



Summary

- In this lecture:
  - Introduction to the Internet
  - Design principles of layering, packet switching, end-to-end argument
- Use packet capture tools such as Wireshark to capture the packets going in and out of your computer. Look at the packets to see the various protocols and headers, across different layers.

So, this is a summary of what we have seen in this lecture today. We have seen how the Internet works. We have studied principles like layering, packet switching, the end to end argument. And with this lecture I hope that you have a rough idea of what happens when you access any Internet service over the network.

And to help you understand these concepts better, please install tool like Wireshark which is very easy to install, very easy to use and capture some packets maybe when you are browsing a website, look at all the packets that are going in and out of your computer, look at all the protocols, the headers, the various layers to see all of this in action in front of your eyes.

Thank you all. That is all I have for this lecture. We will continue discussion on all of these topics in this week. Thanks.