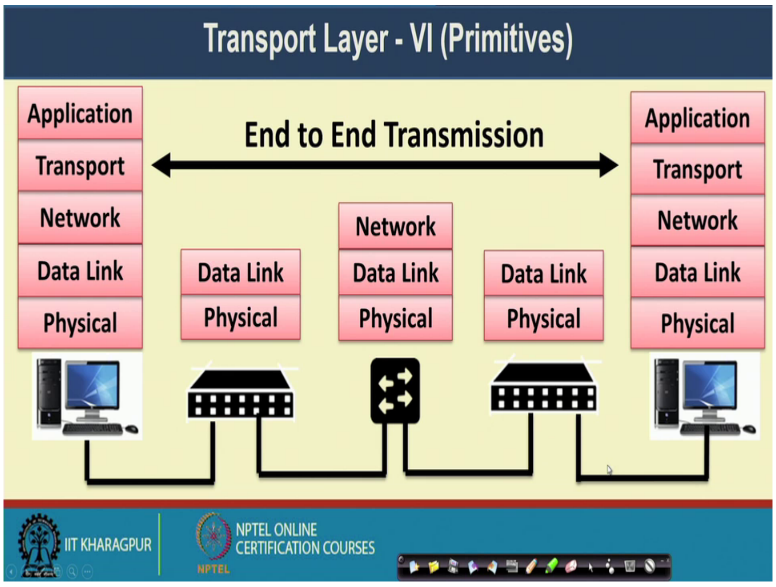
Computer Networks and Internet Protocol

Prof. Sandip Chakraborthy
Department of Computer Science and Engineering
Indian Institute of Technology, Kharagpur

Lecture - 18
Transport Layer Primitives

Welcome back to all of you in this course on Computer Network and Internet Protocols. So, till now at the transport layer we have to looked in to different kind of service primitives.

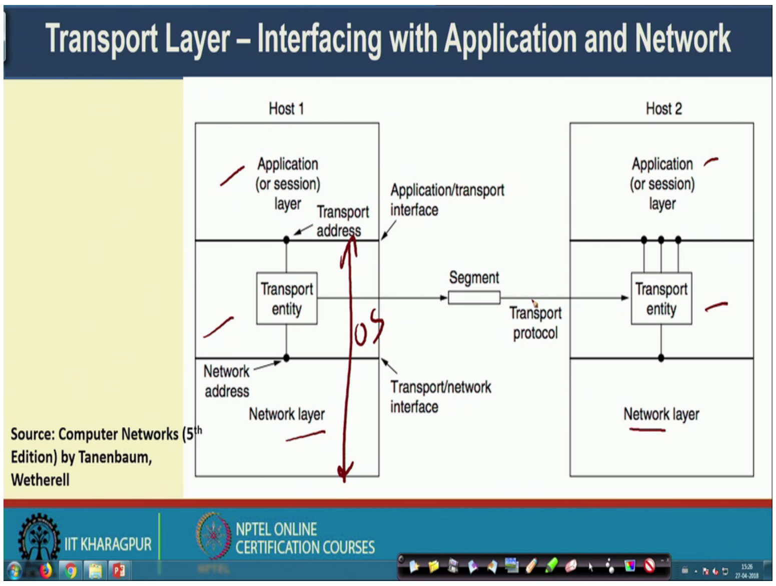
(Refer Slide Time: 00:25)



Transport Layer - VI (Primitives)

So, going from there now we will look in to that how you can combine all this service together and develop and complete n to n transport layer protocol. So, we look into this combination of multiple service together here and then will go to the details of the TCP protocol in details.

(Refer Slide Time: 00:45)

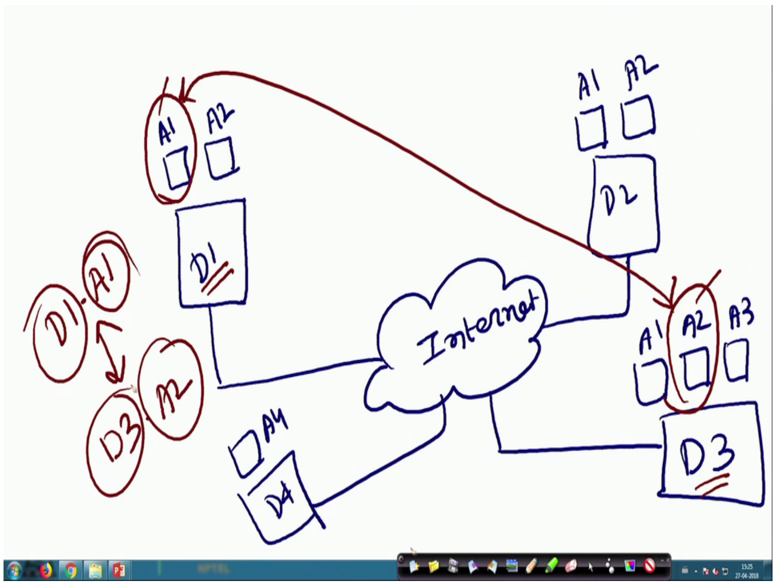

Transport Layer – Interfacing with Application and Network

Source: Computer Networks (5th Edition) by Tanenbaum, Wetherell

Host 1
Host 2
Application (or session) layer
Transport address
Application/transport interface
Transport entity
Segment
Transport protocol
Network address
Network layer
Transport/network interface
OS
IIT KHARAGPUR
NPTEL ONLINE CERTIFICATION COURSES

So, as we have discussed earlier like at the transport layer whenever you are interfacing it with the application layer with specific application service. So, the transport layer it is providing you the n to n connectivity. So now when the transport layer is providing you the n to n connectivity it may happen in a hypothetical scenario that there are say one single machine which is trying to communicate with another machine.

(Refer Slide Time: 01:14)

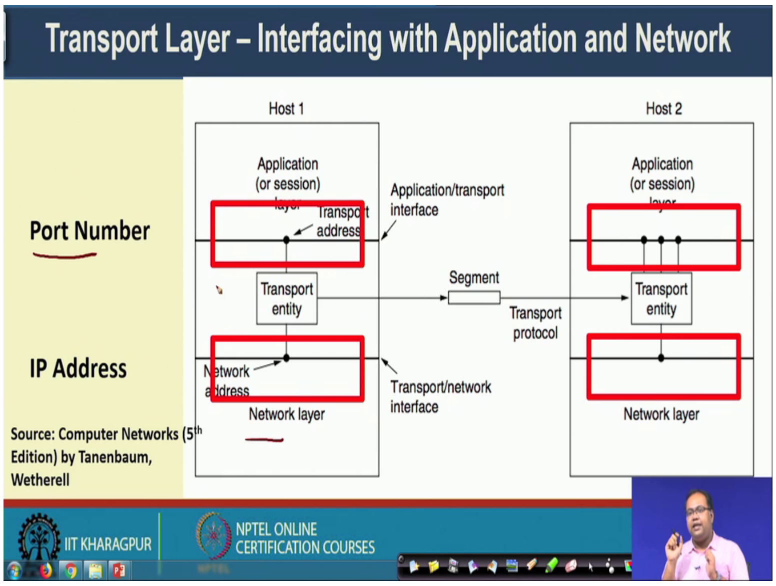


So, this is one desktop D 1 this is another desktop D 2 we have communicate over the internet. So, this is my internet cloud and there are other machines which are available there say D 3 and D 4. Now on this machine and a single machine there can be multiple application which can be running all together say this is A 1 this is A 2, here you are running again 1 application A 1 other application A 2 here you are running say 3 applications A 1 A 2 A 3 here you are running 1 applications say A 4.

So, that way on a single machine because we are utilizing this kind of malty tasking environment, they are can be multiples such applications which are running all together. Now it may happen that in transport protocol that application at D 1, so application at D 1 once to make communicate with application to at D 3. So these 2 application need to communicate with each other. So, at the network protocol stack what you have to do you have to first uniquely identify this machines that D 1 and D 3 want to communicate and that is not sufficient at the same time you need to ensured that the application 1 at D 1 once to communicate with application to over D 3.

So, A 1 over D 1 once to communicate with A 2 over D 3 so these communication part need to be established. So, the question comes that how will you uniquely identify a machine and then how will you uniquely identify an application running on top of a machine. So, to look into that we use 2 different addresses here, so we have the network layer on top of the network layer we have the transport layer and on top of the transport layer I have the application layer and as we discussed earlier that this part it is implemented as a part of your operating system. And then the transport layer they are sending these n to n segments. So, the packets are going where the network layer, but we are considering transport layer pipe a logical pipe between these 2 transport layer entity.

(Refer Slide Time: 03:45)



Transport Layer – Interfacing with Application and Network

Port Number

IP Address

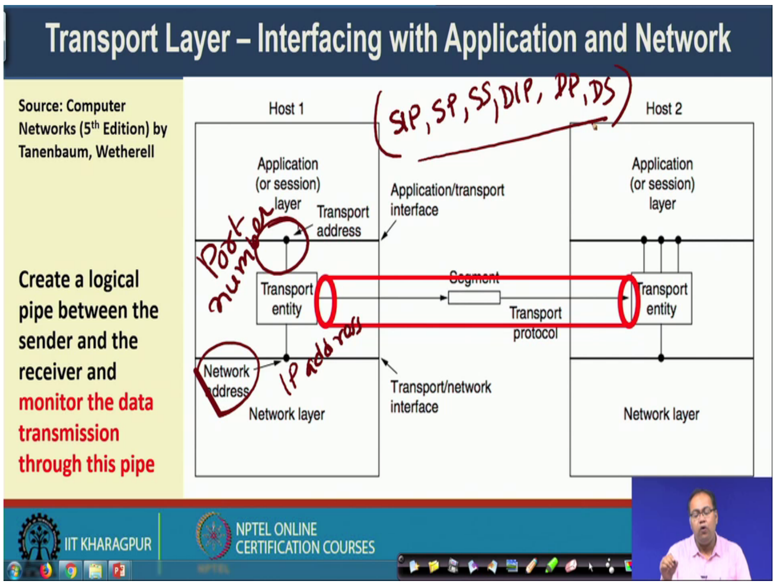
Source: Computer Networks (5th Edition) by Tanenbaum, Wetherell

Host 1
Host 2
Application (or session) layer
Transport address
Application/transport interface
Segment
Transport entity
Transport protocol
Network address
Network layer
Transport/network interface
IIT KHARAGPUR
NPTEL ONLINE CERTIFICATION COURSES

Now, to identify these transport layer entity we use this port number, so the port number uniquely maps this transport entity to a particular applications. So, the example that we are talking about that application 1 or D 1 once to talk with application 2 at D 3. So, these individual applications they are identify over the network with the help of this port number. Similarly individual machines at the network that are identified by the IP address, so we buying the IP address with this network layer and we buying the port number with the transport layer.

So, this transport layer header it uses this source port number and the destination port number that will look in to when you look in to the details of the TCP protocol and a UDP protocol that are the transport layer header you have to provide the source port number and a destination port number. So, this source port number and a destination port number will uniquely identify the application, which is trying to send the data or which is trying to receive the data. So, that is the usefulness of the port number.

(Refer Slide Time: 04:58)



Transport Layer – Interfacing with Application and Network

Source: Computer Networks (5th Edition) by Tanenbaum, Wetherell

Create a logical pipe between the sender and receiver and monitor the data transmission through this pipe

Host 1
Host 2
(SIP, SP, SS, DIP, DP, DS)
Application (or session) layer
Transport address
Application/transport interface
Port number
Transport entity
Segment
Transport protocol
Network address
IP address
Network layer
Transport/network interface
IIT KHARAGPUR
NPTEL ONLINE CERTIFICATION COURSES

Then on top of this transport layer we logically define a pipe, so this is again a logical pipe and you want to implement all the services transport layer services on top of this logical pipe. So, it is just like in the telephone call you are making a hello, so whenever you are making a hello you want make sure that the other person is correctly receiving your message and whenever you are saying something that hi how are you wait for some amount of time if you are getting the respond that hello I am well, so you are happy that the other end has saved your message.

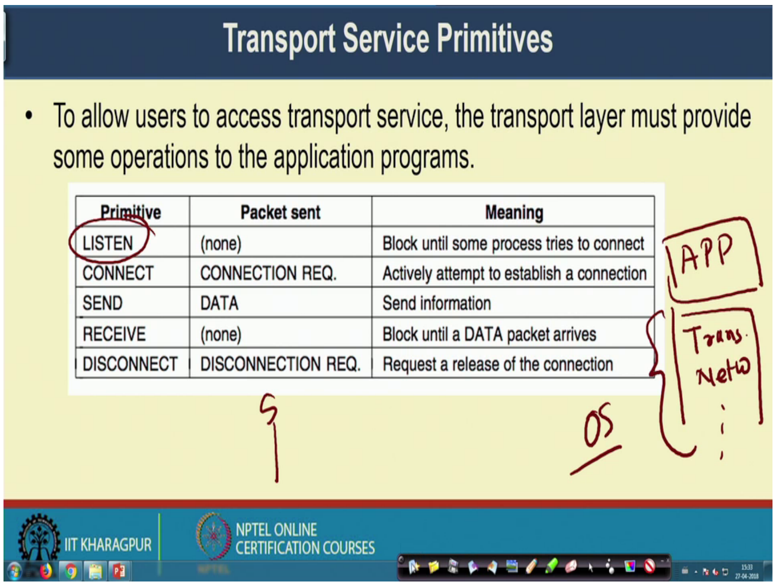
If you have said that hello how are you and you are waiting for some 2 minutes and no response is coming, then again you will say that hello are you hearing, so those are the protocols those are the kind of logical channels which you are establish establishing. Now all your messages like whenever you are saying: hello, how are you this message is embedded to a signal, and then transferred over the physical where which is to connect your telephone network.

So, the same way things happens in the data network, whenever sending data from the transport layer the transport layer is assuring n to n that you are able to send the data correctly at the other end of the system and the other end of the system is receiving the data correctly, because as we are looked into that the lower layer of the protocol stacks starting from the network layer and the below their on reliable, so packet may get drop from there. So, because packet may get drop from their at the transport layer actually since that whether the packet is getting drop from their and if packet is getting drop it is identified with the help of that sequence number, if the packet is getting drop then you retransmit the packet over the same pipe.

So, unique pipe here between 2 transport layer entity it is identified with the help of this address which is the network layer address. So, at the network layer I have this IP address and here I have the port number. So, I have this source IP I have source port, I have destination IP, I have destination port which is uniquely identified this pipe, but remember another point that we have discussed earlier like if a system is getting crashed. And it is restarting you have to also have an initial sequence number and to avoid the delay duplicate you need to ensure that that sequence number initial sequence number which you are generating, it is not using any sequence number of the forbidden region from the previous connection, which is utilizing the same source IP source port destination IP destination port pair. So, that is why this initial sequence number say source sequence number and this destination sequence number they also become part of uniquely identifying this logical pipe.

So, in TCP or in transport layer protocol TCP kind of transport layer protocol we identify this logical pipe with the help of this 6 staples, the source IP the source port the source initial sequence number the destination IP the destination port and a destination initial sequence number.

(Refer Slide Time: 08:18)



Transport Service Primitives

- To allow users to access transport service, the transport layer must provide some operations to the application programs.

Primitive	Packet sent	Meaning
LISTEN	(none)	Block until some process tries to connect
CONNECT	CONNECTION REQ.	Actively attempt to establish a connection
SEND	DATA	Send information
RECEIVE	(none)	Block until a DATA packet arrives
DISCONNECT	DISCONNECTION REQ.	Request a release of the connection

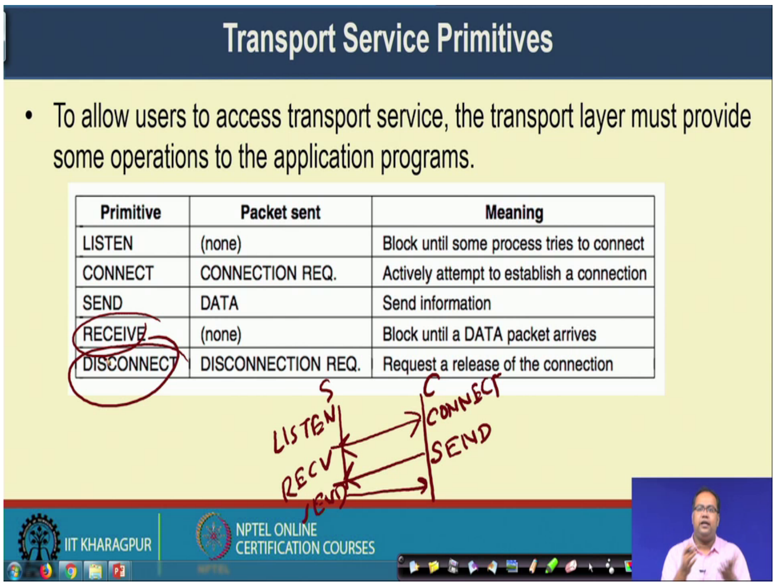
Now, let us look in to some hypothetical primitive to enable user to right a transport layer application. So, the thing is that if you again remember that at the operating system label I have the implementation of the transport layer and then the network layer and then the lower layer of the protocol stack and these part is implemented in your insight your operating system and in the users space you can write your own application. Now if your application say if you are building a chat application and in that chat application if you want to send data over the network then your application need to directly interface or directly interact with the transport layer.

Now whenever you are saying that you need to directly interact with the transport layer, your operating system should provides certain primitives through which you will be able to make your transport layer active and then send the data to your transport layer; after that everything will be taken care of by the transport layer and other lower layers of the protocol stack, but from the application layer you should ask for the specific service that you want from the transport layer. Now to get those service let us first try to design a hypothetical transport layer protocol, by utilizing the various services that we have learnt till now and after that we look in to the TCP protocol in details.

So, that way understanding TCP will be much easier for you. So, to look in to this hypothetical protocol we are thinking of a client survive is application. So, I have a server that server is ready to accept the connection, then I have a client the client can send certain messages to the server. For the time being just think of a hypothetical protocol were the client will send the server as a message like hello server and the server will listen that message and reply back to a client that I am fine. So, to do that what the server has to do the server has to fast to be in the listen state, what is this listen state?

That the server here it is waiting for an incoming connection because, see whenever you want to connect to something if the machine is not in the listen state, you will not be able to initiate a connection, you will not be able to randomly initiate a connection with any of the machine in the world, the machine need to be ready to accept the connection. So, whenever you say that is the server is in the listen state what we are ensuring we are ensuring that the server is ready to listen some message. So, initially the server is in the listen state then we have the connect states, so in the connect state from the client you are sending so the server and this is the client.

(Refer Slide Time: 11:00)



Transport Service Primitives

- To allow users to access transport service, the transport layer must provide some operations to the application programs.

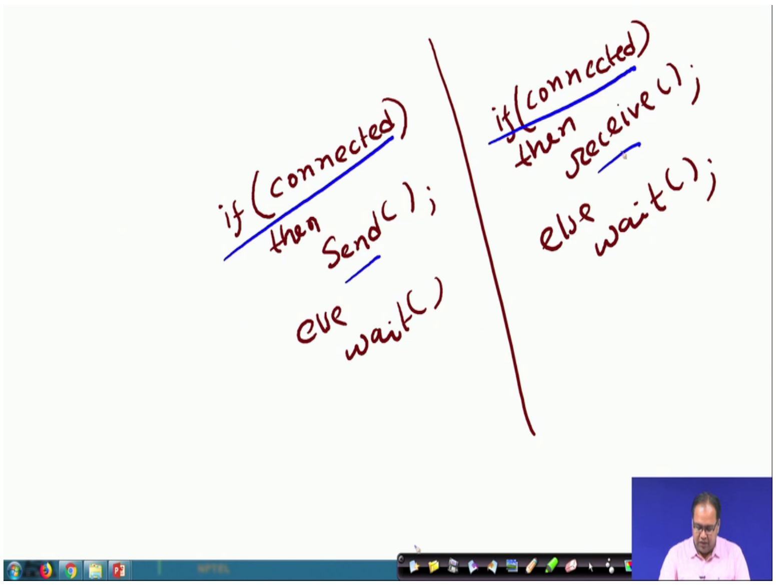
Primitive	Packet sent	Meaning
LISTEN	(none)	Block until some process tries to connect
CONNECT	CONNECTION REQ.	Actively attempt to establish a connection
SEND	DATA	Send information
RECEIVE	(none)	Block until a DATA packet arrives
DISCONNECT	DISCONNECTION REQ.	Request a release of the connection

So, the server is in the listen state, now from the client sight you are making a connect call whenever you are making a connect call, then you are actually asking the transport layer to initiate a n to n connection, so the transport layer will initiate a connection. So, if it is A 3 way hand shaking that we have learnt earlier it will used at 3 way hand shaking for initiating the connection, then once this connection is establish then you can call the send function from your application program to send the data send the data to the corresponding server.

Now once the server gets this data, so server need to accept the data from the transport layer. So, if you remember earlier the diagram like the data will come and the data will keep on waiting or the receiver buffer at the transport layer. And from the application you have to make certain function call to get the data from that transport layer buffer. So, for that we have this receive call, so the server will make a received call to receive the data from the transport layer buffer. Now that way you can send that hello message and the server can say I am fine server can again make a send call to send server can again make a send call to send for the data and that way this call can go on.

So, once this data transfer is complete, then you send the disconnect message or disconnect function call to disconnect this particular connection. Now here the interesting point is this connect and the disconnect call. So, in a transport layer if you want to get the transport layer services along with connection establish state. So, what you have to ensure that whenever you are making a send call or the received call you are there in the connect state. That means, before you have made a send call and a received call you need to ensure that well the system has already established the connection.

(Refer Slide Time: 13:17)

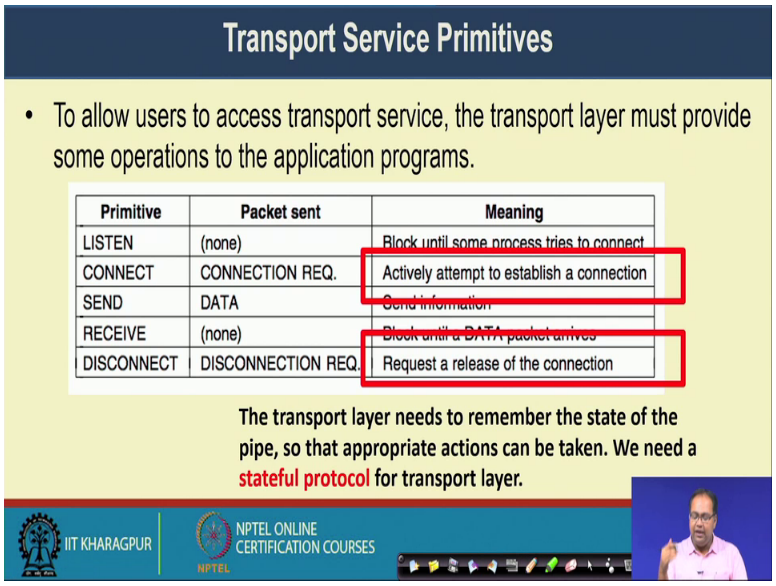


```
if (connected)
    then Send();
else
    wait()
```

```
if(connected)
    then receive();
else
    wait();
```

So that means to initiate a connection what you have to do, you have to write the code in this way that if connected I am just writing some pseudo code, then send else wait so that is at the sender side ok. Similarly at the receiver side it will be if connected then make a receive call else wait for the connection. Now in this case you can see that well every time you want to make a call to the send or receive you have to check that the system is in the connected state. So, if only the system is in the connected state then only we will be able to make a call to this send the receive function.

(Refer Slide Time: 14:25)



Transport Service Primitives

- To allow users to access transport service, the transport layer must provide some operations to the application programs.

Primitive	Packet sent	Meaning
LISTEN	(none)	Block until some process tries to connect
CONNECT	CONNECTION REQ.	Actively attempt to establish a connection
SEND	DATA	Send information
RECEIVE	(none)	Block until a DATA packet arrives
DISCONNECT	DISCONNECTION REQ.	Request a release of the connection

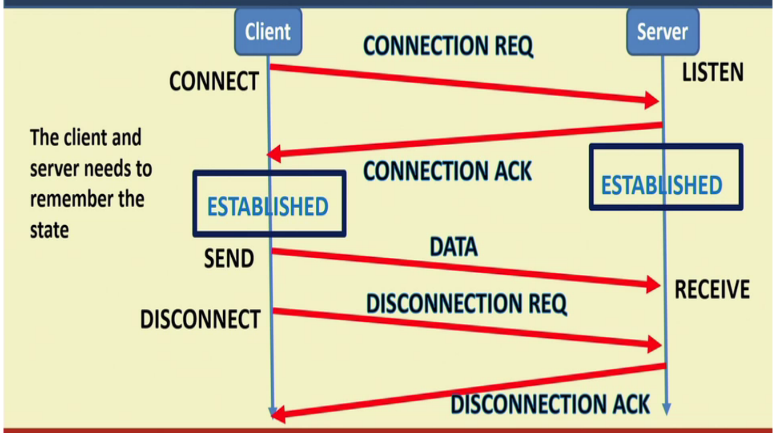
The transport layer needs to remember the state of the pipe, so that appropriate actions can be taken. We need a stateful protocol for transport layer.

IIT KHARAGPUR NPTEL ONLINE CERTIFICATION COURSES

So, that way that way this particular primitive is important because, what we say that the transport layer needs to remember the state of the pipe the pipe logical pipe that we have defined earlier, so that appropriate actions can be taken. So, if you are making a send call before initiating the connection so that call is not a valid call. So, we need a state full protocol for a transport layer, so what is mean by a state full protocol that with every connection you will remember that what was the state of that particular pipe through which you are going to send the data.

(Refer Slide Time: 14:56)

Transport Service Primitive – Connection Establishment



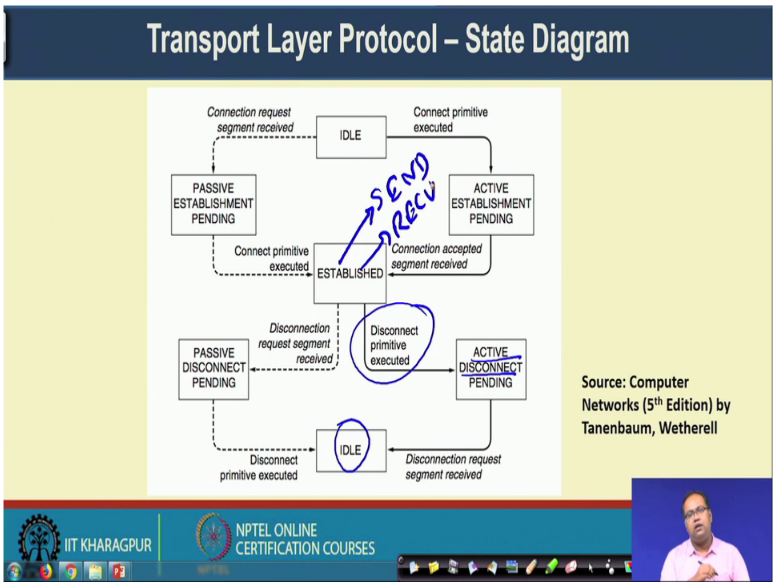
The client and server needs to remember the state

IIT KHARAGPUR NPTEL ONLINE CERTIFICATION COURSES

So, first you have to initiate the connection. So the system is in the connect state the example that I have given you that the client is in the connect state the server is in the listen state you have sent a connection request and got an connection acknowledgement both are in the establish state.

Establish state means the connection has established then you can send the data make a send call to send the data, the server can make a receive call to receive the data even if after that the some server wants to send the data server can make a send call and the client can make a receive call. And once this is done then you can make a disconnect call to disconnect this particular request. So, this established is the state that the server and the client need to remember before making the send call and the received call.

(Refer Slide Time: 15:50)



Transport Layer Protocol – State Diagram

Source: Computer Networks (5th Edition) by Tanenbaum, Wetherell

Connection request segment received
IDLE
Connect primitive executed
PASSIVE ESTABLISHMENT PENDING
ACTIVE ESTABLISHMENT PENDING
SEND
RECV
Connect primitive executed
ESTABLISHED
Connection accepted segment received
Disconnection request segment received
Disconnect primitive executed
PASSIVE DISCONNECT PENDING
ACTIVE DISCONNECT PENDING
IDLE
Disconnect primitive executed
Disconnection request segment received
IIT KHARAGPUR
NPTEL ONLINE CERTIFICATION COURSES

So, this entire thing we can represent in the form of a state transition diagram. So, this is an important concept with the on concept of this transport layer, where if you want to maintain transport layer services you need to maintain this state of your pipe logical pipe that you are defining on top of the transport layer. So, these state transition diagram will tell you that on receives and of which message, how you are moving from 1 state to another state. So, let us look in to this example in details, so initially you are in the idle state now in the idle state so you make a connect call.

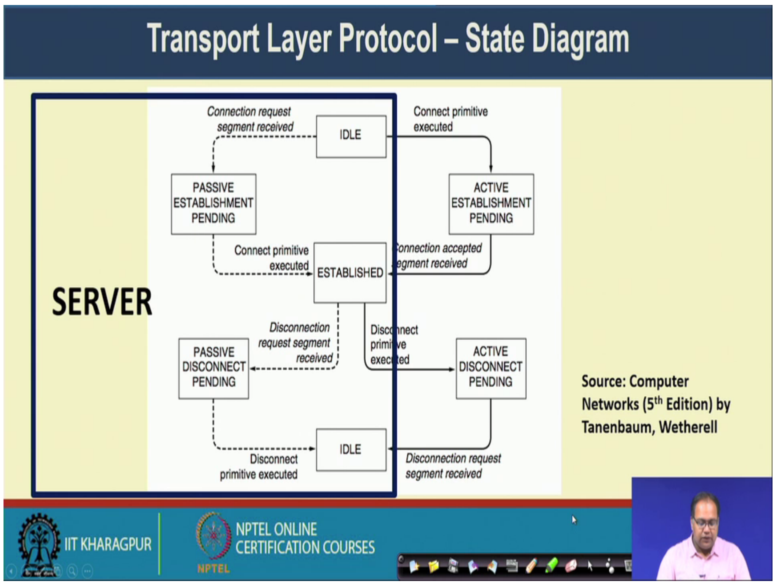
So, once you have connect call so this solid line is the client side and this doted line is the server side. So, you have make a connect call, so once you have make the connect calls did that time this is you have made the active establishment, active establishment means you have initiated the connection; similarly the server it has received the connection request segment. So, once the server has receive the connection request segment it is in the passive establishment state; that means, it has got a connection request message they need execute the connection primitive; that means, if it is A 3 way hand shaking it execute that 3 way hand setting otherwise it send the acknowledgement and it moves to the establish state.

Similarly the client when it gets the connection accepted segment it received these connection establish segment accepted segment and it moves to the established state and this established state you can start transmitting the data whenever you are in the establish state.

Now to come out of the establish state you have disconnect the things, then again if the client initiate the disconnection message, so connect that the client execute the disconnect primitive; after the client execute disconnect primitive the client has executed it. So, it is the active disconnection after that similarly at the server side the server, if it receive the disconnection request segment. So, it moves to the passive this connection then once it execute the disconnect primitives send the acknowledgement, the server moves to the idle state; similarly when the client receive this disconnection request from the server that gives an acknowledgement to it is request it moves to the idle state.

So, that way so this client by executing this connection primitive it moves to the established state, the server moves to the established state and you execute the disconnect primitive and move to the idle state back again. And here you can see that this state transitions are initiated by sending some messages or receiving some messages and whenever you are in a proper state then only you are allowed to do further task. For example, whenever you are at the established state then only you are able to execute the send and a received call, otherwise you are not allowed to do that.

(Refer Slide Time: 19:08)

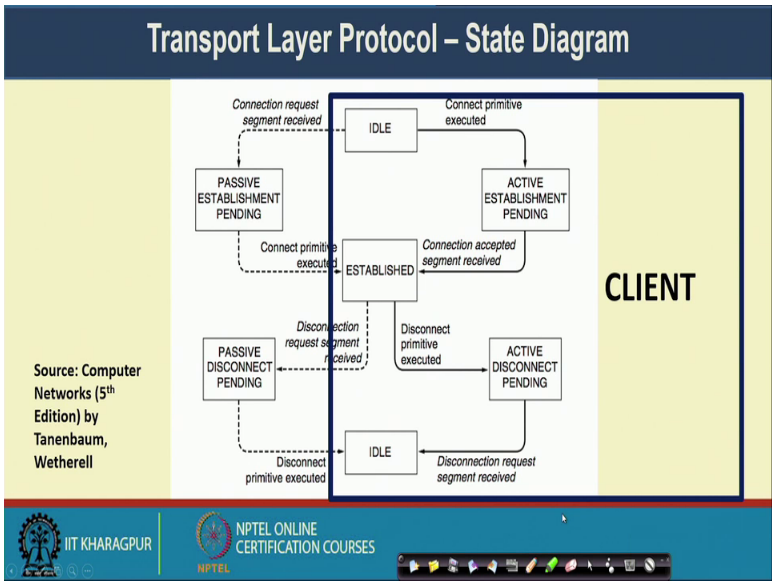

Transport Layer Protocol – State Diagram

SERVER
IDLE
PASSIVE ESTABLISHMENT PENDING
ACTIVE ESTABLISHMENT PENDING
ESTABLISHED
PASSIVE DISCONNECT PENDING
ACTIVE DISCONNECT PENDING
IDLE
Connection request segment received
Connect primitive executed
Connect primitive executed
Connection accepted segment received
Disconnection request segment received
Disconnect primitive executed
Disconnect primitive executed
Disconnection request segment received

Source: Computer Networks (5th Edition) by Tanenbaum, Wetherell

IIT KHARAGPUR
NPTEL ONLINE CERTIFICATION COURSES

Well, so this is the server side and this is the client side that we have discussed.

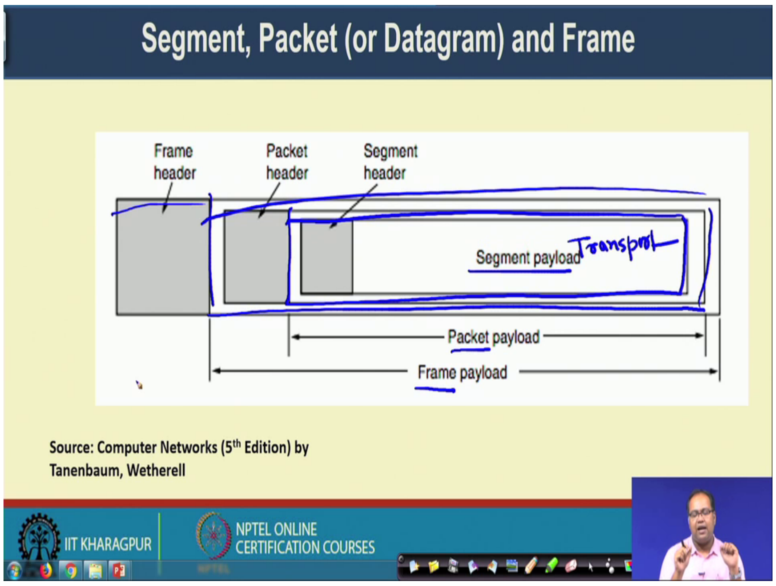

Transport Layer Protocol – State Diagram

CLIENT
IDLE
PASSIVE ESTABLISHMENT PENDING
ACTIVE ESTABLISHMENT PENDING
ESTABLISHED
PASSIVE DISCONNECT PENDING
ACTIVE DISCONNECT PENDING
IDLE
Connection request segment received
Connect primitive executed
Connect primitive executed
Connection accepted segment received
Disconnection request segment received
Disconnect primitive executed
Disconnect primitive executed
Disconnection request segment received

Source: Computer Networks (5th Edition) by Tanenbaum, Wetherell

IIT KHARAGPUR
NPTEL ONLINE CERTIFICATION COURSES

(Refer Slide Time: 19:16)



Segment, Packet (or Datagram) and Frame

Source: Computer Networks (5th Edition) by Tanenbaum, Wetherell

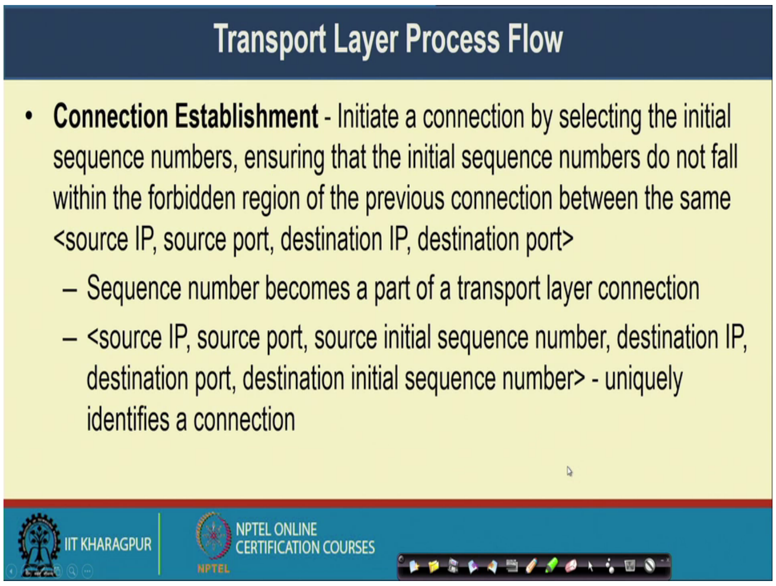
So, in the context of transport layer till now I have use this terms segment packet frame interchangeably everything to point that a network packet which is going over the link, but technically we make a differentiation between the segment the frame and a packets. So, in general at the transport layer whatever you are getting, so that is called a segment.

So, at the segment is the concept at the transport layer now after getting a segment at the transport layer, you adopt the segment header at the transport layer and pass it to the lower layer that is the network layer. So, this entire thing that you are passing to the network layer that becomes the network layer payload and in the network layer that concept we call it as a packet.

So, packet the term packet is normally used to denote the primitive at the network layer, with that you adopt the packet header and send this entire thing to the data link layer. In the data link layer this entire thing that you are receiving from the network layer that is termed as a frame, so this is the frame payload at the data link layer you adopt the frame header with the frame payload and send it to the physical layer for physical transmission.

So, the segment it is used at the transport layer, so in the transport layer the data primitive we call it as a segment at the network layer, the data primitive we term it as a packet or in the contest of UDP we call it as a data gram and then at the data link layer we call it as a frame. So, till now you have use the term interchangeably, because this terminologies has not been define to you, but now onwards will use this terminology whenever we are there at a particular layer of the protocol stack. In the context of flow control I have used the term frame and segment interchangeably because as we have seen that this concept of flow control is there at the transport layer as well as at the data link layer. So, the flow controls are executed on top of segment as well as it is executed on top of frames, so we should not have any confusion there; but for the other primitives try to utilize this proper term which is there, ok.

(Refer Slide Time: 21:33)



Transport Layer Process Flow

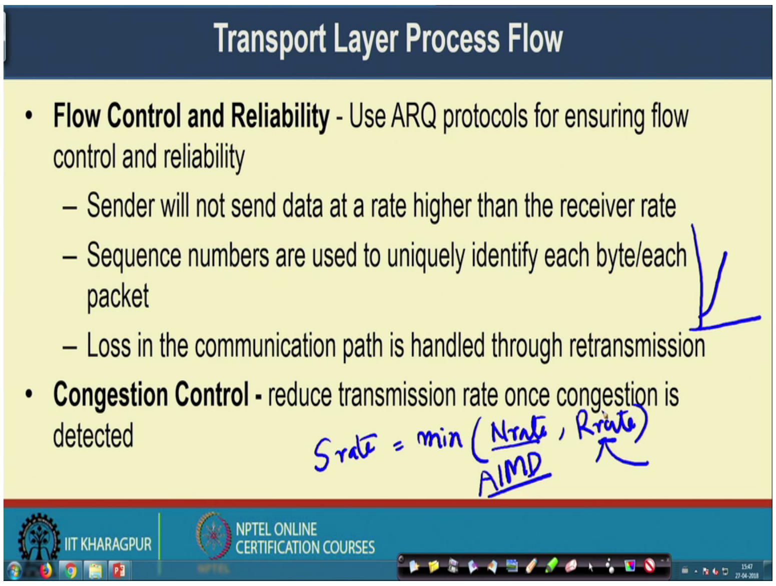
- **Connection Establishment** - Initiate a connection by selecting the initial sequence numbers, ensuring that the initial sequence numbers do not fall within the forbidden region of the previous connection between the same <source IP, source port, destination IP, destination port>
  - Sequence number becomes a part of a transport layer connection
  - <source IP, source port, source initial sequence number, destination IP, destination port, destination initial sequence number> - uniquely identifies a connection

IIT KHARAGPUR
NPTEL ONLINE CERTIFICATION COURSES

Now, let us look in to this entire transport layer process flow by combining all the service primitives that we have learned. So, initially you need to have this connection establishment. So this connection establishment it initiate a connection by selecting the initial sequence number. And whenever you are selecting the initial sequence number you need to ensure that this initial sequence number do not fall within the forbidden region of the previous connection between the same source IP source port destination IP destination port pair. And that is why we include the sequence number as a part of identifying these n to n type, which we normally call as socket in the terms of unique later on will look how we do the programming on top of a socket.

So, this same logical pipe is termed as a socket, so we defined uniquely identify a connection uniquely identify a socket with the 6 tuples, the source IP source port source initial sequence number destination IP destination port and destination initial sequence number.

(Refer Slide Time: 22:39)



Transport Layer Process Flow

- **Flow Control and Reliability** - Use ARQ protocols for ensuring flow control and reliability
  - Sender will not send data at a rate higher than the receiver rate
  - Sequence numbers are used to uniquely identify each byte/each packet
  - Loss in the communication path is handled through retransmission
- **Congestion Control** - reduce transmission rate once congestion is detected

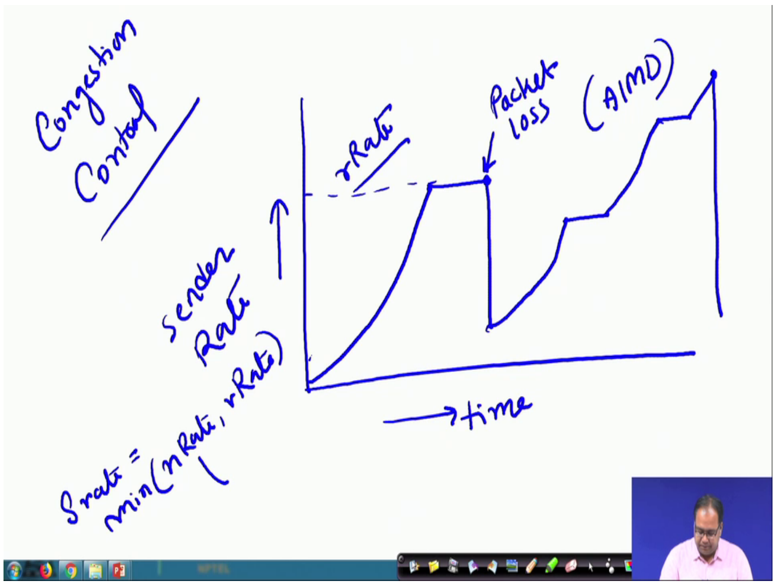
$S_{rate} = \min(\underbrace{N_{rate}}_{AIMD}, R_{rate})$

Then comes the flow control and reliability, once you have set up these initial sequence number then that initial sequence number will be used further to ensure the flow control and reliability with the help of your Q protocols. So, this ARQ protocols see it ensures the flow control and reliability, so the sender will not send data at a rate higher than the receiver rate; as well as in the congestion control prospective we have seen that the sender rate should be minimum of the network supported rate.

That means, the congestion rate and the receiver rate, then the sequence number they are used to uniquely identify each bite for a bite sequence number or if you designing a protocol with the packet sequence number with fix size packets, then this, this sequence number will denote uniquely identify a packet and loss in the communication part it is handle to this transmission in the flow based flow based control mechanism in the based flow control mechanism. So, you make a retransmission to re transmit the packet over that same connection, then we have the congestion control the congestion control algorithm it reduces the transmission rate once congestion is detected.

So, we have seen that the sender rate I am writing it has Srate is minimum of the network rate and your receiver rate. So, this receiver rate is something that is advertised by the receiver with every individual acknowledgement and this network rate the idea is that you apply this a IMD protocol additive increase multiplicative decrease protocol to ensured both efficiency and free on a simultaneously. And with this help of the a IMD protocol what you do in case of congestion control, you gradually increase the rate and you see that this rate will gets saturate when it will reach at the receiver rate. So, ideally let me try to draw a proper diagram so that the things become easier for you to understand.

(Refer Slide Time: 24:45)

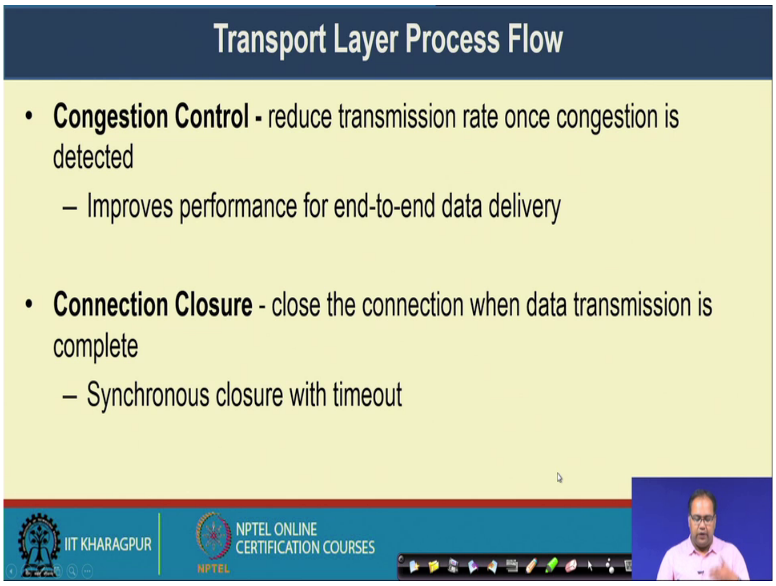


So, initially you increase that so I am I am here with respect to time I am plotting the sender rate for congestion control algorithm. So, initially which increase the sender rate, so once so my formula is that sender rate is equal to minimum of network rate and receiver rate. So, initially you start network rate with a very low rate say some 1 kbps. And gradually try to increase it, so whenever you are increasing it the minimum is becoming the network rate after that when the network rate will exceed the receiver rate it will get saturated here, so this is my receiver advertise rate.

So, after that once you are experience a packet loss you are experiencing a packet loss here, so once you are experiencing a packet loss you apply this a IMD concept additive increase multiplicative decrease concept to drop the rate again and start this procedure again gradually increase the network rate ok. At this point if the receiver advertise some different rate it will get saturated here, after that if the network again advertise that well it can support higher rate again you start increasing it based on the network rate it will get saturated here, based on the receiver rate and then it increases again and some time if there is a congestion detection with the help of a loss with the detection of a packet loss you again drop the way.

So, that way the sender rate gradually increases in a transport layer and it helps you to find out to handle the congestion as well as the flow control simultaneously. So, here we can see that this is the flow control algorithm and the congestion control algorithms are couple together.

(Refer Slide Time: 27:04)



Transport Layer Process Flow

- **Congestion Control** - reduce transmission rate once congestion is detected
  - Improves performance for end-to-end data delivery
- **Connection Closure** - close the connection when data transmission is complete
  - Synchronous closure with timeout

IIT KHARAGPUR
NPTEL ONLINE CERTIFICATION COURSES
NPTEL

So, this congestion control it reduces the transmission rate once transition is detected and as you have seen that it improves the performance for n to n data delivery. So, dynamically based on this rate you start sending the data other in to receive the data send back the acknowledgement and accordingly will pause it and once you want to close the connection the data transmission is over. Then you execute this connection closure primitive that close the connection when the data transmission is complete. And as we have seen earlier that although a synchronous closure is good, but a synchronous closure is not possible to implement in a distributed system with unreliable channel. So we go with synchronous closure with time out.

So, that is all about the basic service primitives of transport layer, in the next class onwards will start looking in to the transmission control protocol of the TCP protocol in details which is widely used in the network. So, around 80 percent of the traffic over the global internet it uses this TCP protocols; so will look in to the TCP protocol in details which is a widely accepted transport layer protocol.

Thank you all for attending this class.