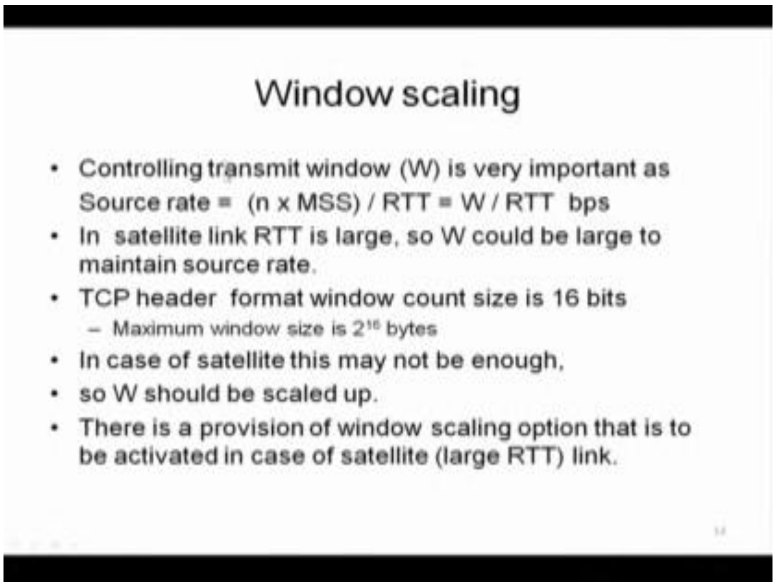
Satellite Communication Systems

Prof. Kalyan Kumar Bandyopadhyay
Department of Electronics and Electrical Communication Engineering
Indian Institute of Technology, Kharagpur

Lecture - 37
Effect on Higher Layer – II

Welcome back, we are talking about the TCP window scaling option; window size can be increased because the satellite link as a large RTT.

(Refer Slide Time: 00:31)

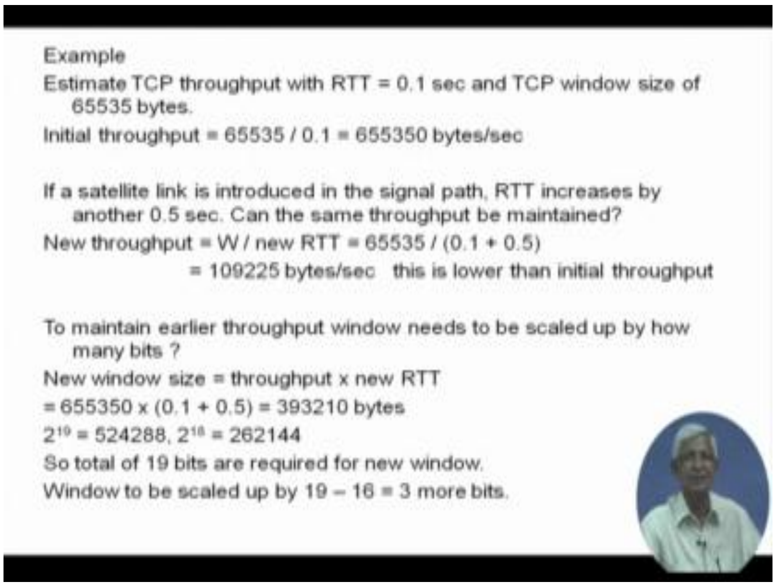


Window scaling

- Controlling transmit window (W) is very important as Source rate = (n x MSS) / RTT = W / RTT bps
- In satellite link RTT is large, so W could be large to maintain source rate.
- TCP header format window count size is 16 bits
  - Maximum window size is 2^{16} bytes
- In case of satellite this may not be enough,
- so W should be scaled up.
- There is a provision of window scaling option that is to be activated in case of satellite (large RTT) link.

This was the last slide we have seen, there is a provision for the window scaling option that is to be activated in case of satellite link that is large RTT.

(Refer Slide Time: 00:41)



Example

Estimate TCP throughput with RTT = 0.1 sec and TCP window size of 65535 bytes.

Initial throughput = 65535 / 0.1 = 655350 bytes/sec

If a satellite link is introduced in the signal path, RTT increases by another 0.5 sec. Can the same throughput be maintained?

New throughput = W / new RTT = 65535 / (0.1 + 0.5)

= 109225 bytes/sec this is lower than initial throughput

To maintain earlier throughput window needs to be scaled up by how many bits ?

New window size = throughput x new RTT

= 655350 x (0.1 + 0.5) = 393210 bytes

2^{19} = 524288, 2^{18} = 262144

So total of 19 bits are required for new window.

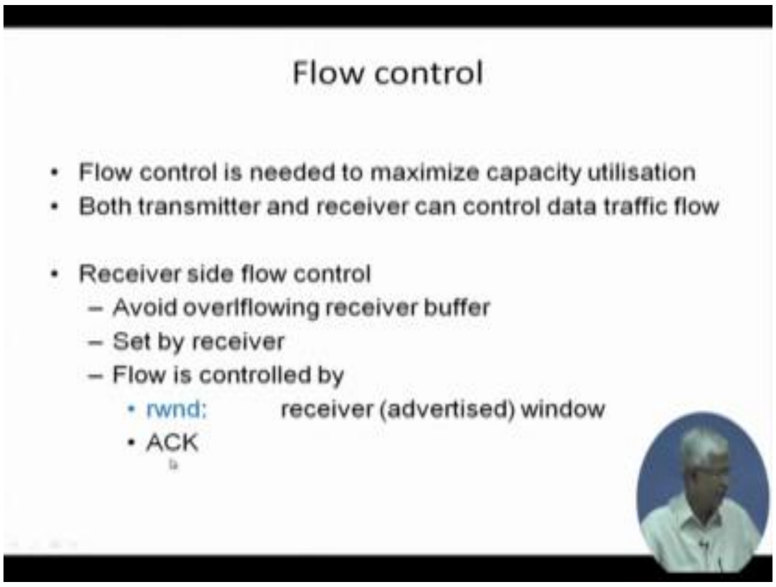
Window to be scaled up by 19 – 16 = 3 more bits.

Let us take a example and try to see how it is; estimate TCP through put with RTT of 0.1 second and TCP window size of 2 to the power 16 bytes that is 65535 bytes. Now the initial throughput is in the size divided by RTT, which is 655350, this 0.1 in the denominator, so many bytes per second that is the initial throughput.

Now, if a satellite link is introduced in the signal path; RTT increases by another 0.5 second, so it will become 0.1 plus 0.5, 0.6 second, can be same throughput be maintained? no a new throughput will be; W by new into size. Since the RTT has increased, it will decrease, so initial value was 655350 by bytes per second, now it would be 0.09225 bytes per second, this is the lower than initial through put.

So, what is to be done to maintain the earlier throughput window needs to be scaled up by how many bits? Let us calculate that; new window size throughput into new RTT and which is throughput was 655350 and new RTT is total 0.6, so the window size will be 393210 bytes. So, this is 2 to the power how many bites; it represents 2 to the power 19 is 524288, 2 to the power 18 means 262144; 2 to the power 18 is smaller than this, so 19 bites are required and normally the window is 16 bites, so 3 more bites are required.

(Refer Slide Time: 02:23)



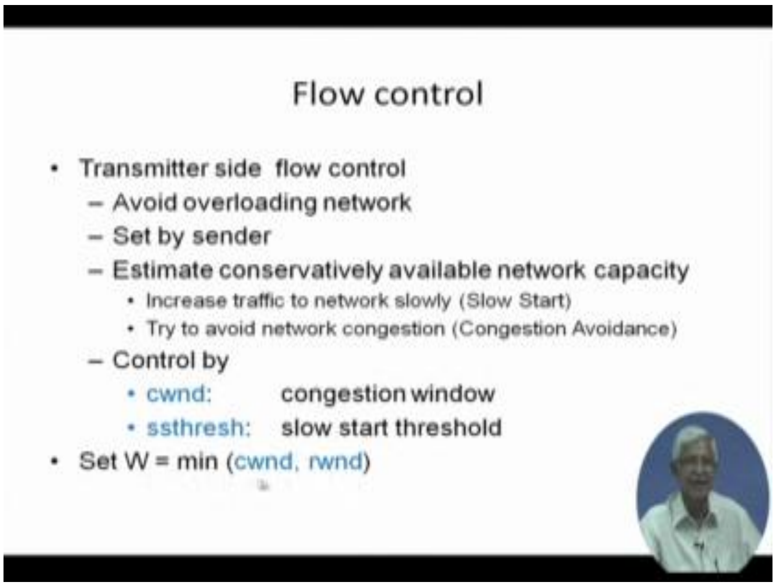
Flow control

- Flow control is needed to maximize capacity utilisation
- Both transmitter and receiver can control data traffic flow

- Receiver side flow control
  - Avoid overlflowing receiver buffer
  - Set by receiver
  - Flow is controlled by
    - rwnd: receiver (advertised) window
    - ACK

So, window scaling has to be by 3 more bites, so this is how we quickly calculate. Now let us go another important issue which is called flow control; it is very important. Flow control is needed to maximize the capacity utilization, now both transmitter and receiver can control the traffic, transmitter and receiver both can control this traffic. Receiver side flow control is receiver buffer, to avoid overflowing the received buffer; see buffer if it is full, it is not yet read properly and then another data has come it will over flow. So, receiver has to stop that and it this is set by the receiver, so that flow control is set by receiver and this is set by a parameter which called rwnd, rwnd receiver window which is a receiver advertised window. Receiver at the beginning of the collection it says, this is my rwnd and of course, he can control by ACK, he can hold the ACK, he can delay the ACK, so the ACK is another thing which is a clock of the total system. So, with rwnd advertisement as well as by ACK he can control the flow of the input traffic.

(Refer Slide Time: 03:27)

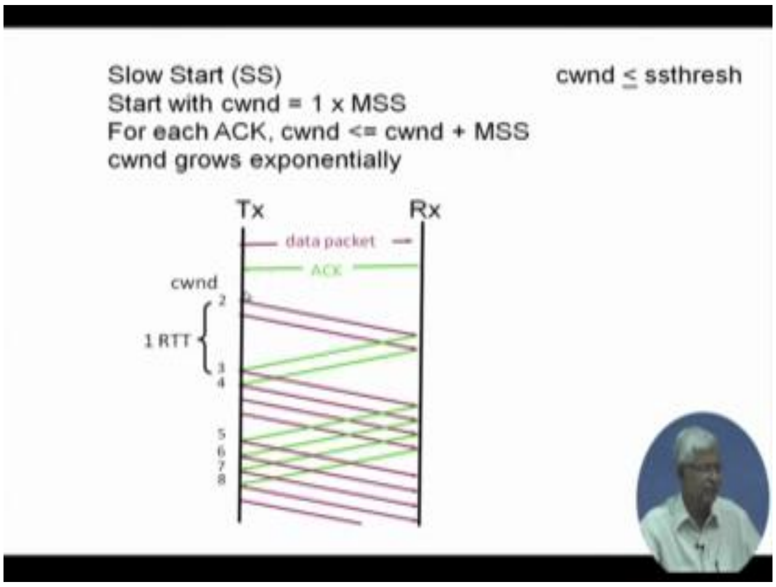


Flow control

- Transmitter side flow control
  - Avoid overloading network
  - Set by sender
  - Estimate conservatively available network capacity
    - Increase traffic to network slowly (Slow Start)
    - Try to avoid network congestion (Congestion Avoidance)
  - Control by
    - cwnd: congestion window
    - ssthresh: slow start threshold
- Set W = min (cwnd, rwnd)

And the transmitter side flow control; let us see it avoids the overloading network. The network capacity it has to understand, but it is transmitted; how you will understand, so you will go slowly. So, it is set by the sender, the transmitter will be setting and estimate; it estimates conservatively to avoid network over flow; capacity over flow that is increased traffic, slowly which is named as a slow start and then avoid congestion; avoid network congestion is called congestion avoidance, two different phases it will go slowly and then avoid, tries to avoid congestion. For that he sets two parameters which is cwnd which is congestion window and slow start threshold; ss threshold, there is a threshold value and congestion window will be variable on that, threshold value will change depending on the situation.

(Refer Slide Time: 04:31)



Slow Start (SS)

cwnd $\leq$ ssthresh

Start with cwnd = 1 x MSS

For each ACK, cwnd <= cwnd + MSS

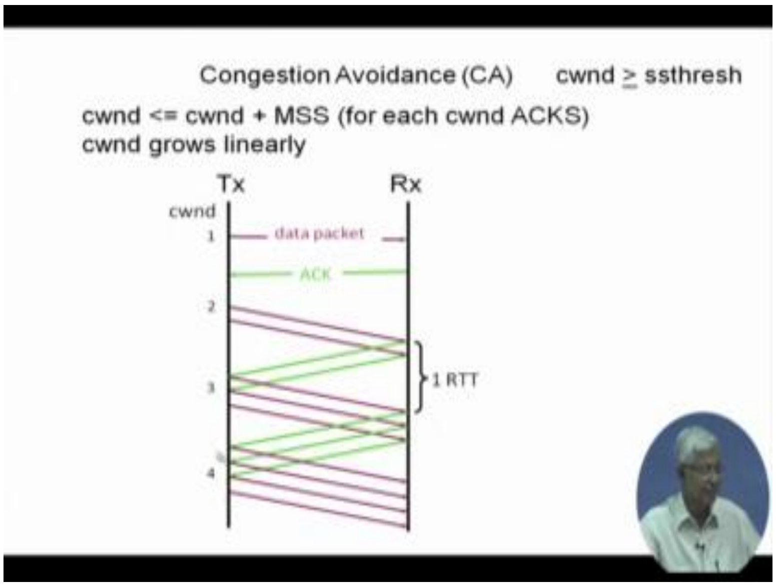
cwnd grows exponentially

So, initially it will be setting the window is equal to minimum of cwnd and rwnd, that receiver cannot accept beyond that, so rwnd and cwnd will be very; let us see. Slow start, slow start phase this is defined as when cwnd is less than threshold, slow start threshold. So, it starts with one time MSS; that is maximum segment size, now I am showing the initial days of internet; now-a-days they have done lot of research and this has become too into MSS, but let us not go into the too much of details of that, it just a philosophy we are discussing.

So, start up the; it is easier to understand also, start with cwnd 1 into MSS then for each of the ACK of these MSS or the sequence; the cwnd will be increased by cwnd whatever early was there plus 1 MSS and cwnd; obviously, grows exponentially. Let us try to understand this picture; transmitter has sent a data packet, one segment and the acknowledgment for the segment has come, so it will increase by 1 MSS, so you can window has increased cwnd has increased from 1 plus 1, 1 acknowledgment has come.

So, it will send two of them 2 segments, so for 2; 1 acknowledgment comes. So, for each acknowledgment it increases by 1, so for two it will increase by total 2; so earlier 2 plus 2 makes it 4, the 4 acknowledgments comes; make it 4 plus 4; 8, for each acknowledgment it will increase by 1 and earlier more than earlier 1 cwnd plus MSS, so therefore, is going by 2 to the power exponentially.

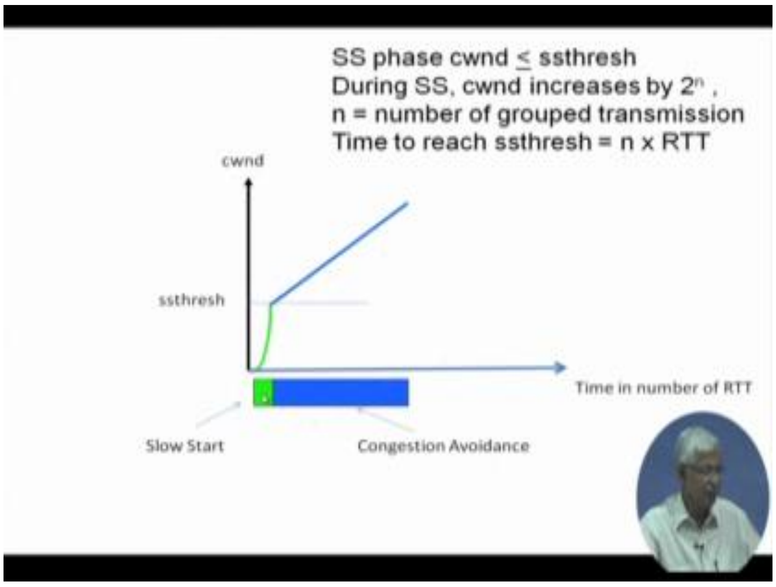
(Refer Slide Time: 05:58)


Congestion Avoidance (CA) cwnd $\geq$ ssthresh

cwnd <= cwnd + MSS (for each cwnd ACKS)
cwnd grows linearly

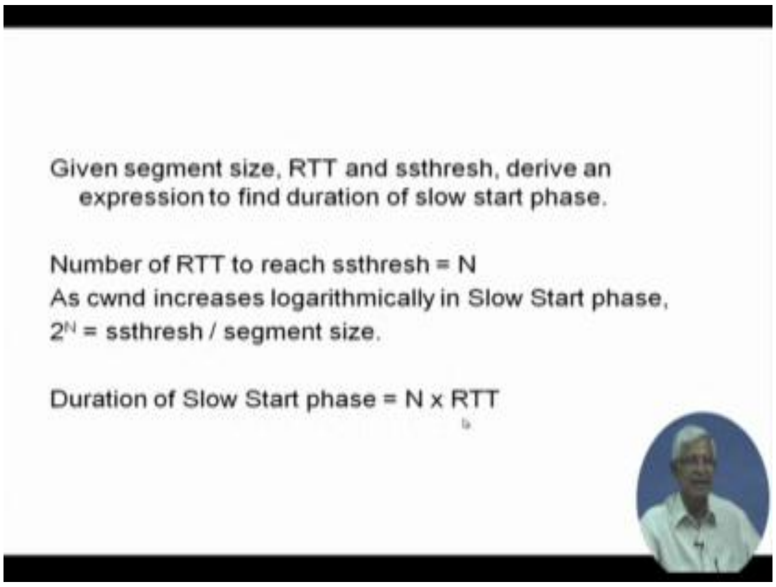
And in the congestion avoidance when the cwnd is greater than s s threshold, then for any group of acknowledgment it is increases by only by 1 MSS. So, it grows linearly; pictorially the data packet is sent, acknowledgment has come; let us assume this cwnd is that situation. So, for acknowledgment 1 increases, so now two data packets have been sent. So, when 2 ACK comes, this will group in this group it increase by 1, so it makes it 3, when 3 comes in a group; then it increase by 1, so makes it 4 like that.

(Refer Slide Time: 06:35)


SS phase cwnd $\leq$ ssthresh
During SS, cwnd increases by 2^n ,
n = number of grouped transmission
Time to reach ssthresh = n x RTT

So, it can be plotted like this that; this x axis is RTT and cwnd in terms of MSS, so there is a s s threshold set and cwnd increases slowly, but then exponentially quickly and once it reaches s s threshold it goes linearly. So, this is the slow start phase and this is the congestion avoidance phase, so slow start phase is cwnd is lower than s s threshold and congestion avoidance phase is cwnd is higher than s s threshold that you have to remember. So, during slow start cwnd increases by 2 to the power n, n is the number of grouped transmission. So, time to reach this s s threshold; if we say this will be my throughput, so time to reach that is n times RTT is 2 to the power n and n times how many RTTs is passed, so n times RTT.

(Refer Slide Time: 07:31)



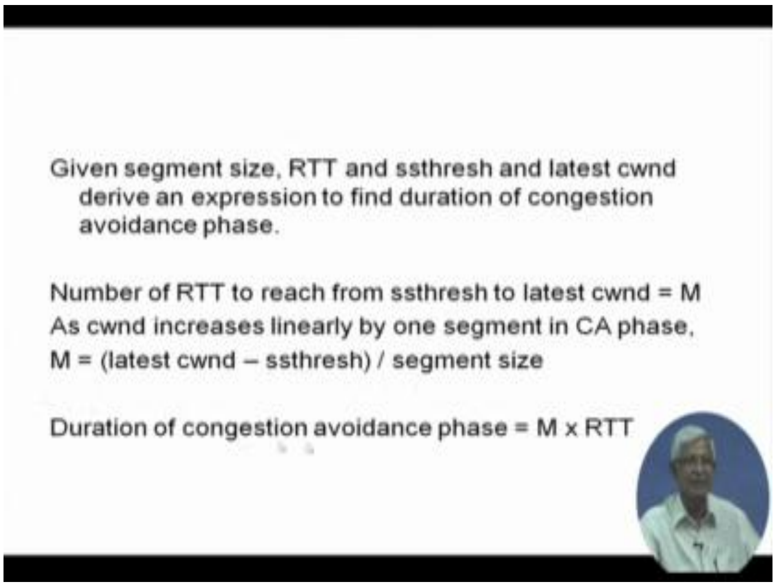
Given segment size, RTT and ssthresh, derive an expression to find duration of slow start phase.

Number of RTT to reach ssthresh = N
As cwnd increases logarithmically in Slow Start phase,
2^N = ssthresh / segment size.

Duration of Slow Start phase = N x RTT

Given the segment size, RTT and s s threshold, derive an expression to find duration of slow start phase and number of RTT to reach s s threshold is n and as cwnd increases logarithmically in slow start phase, 2 to the power n equal to s s threshold by segment size, so duration of slow start phase is n into RTT.

(Refer Slide Time: 07:54)



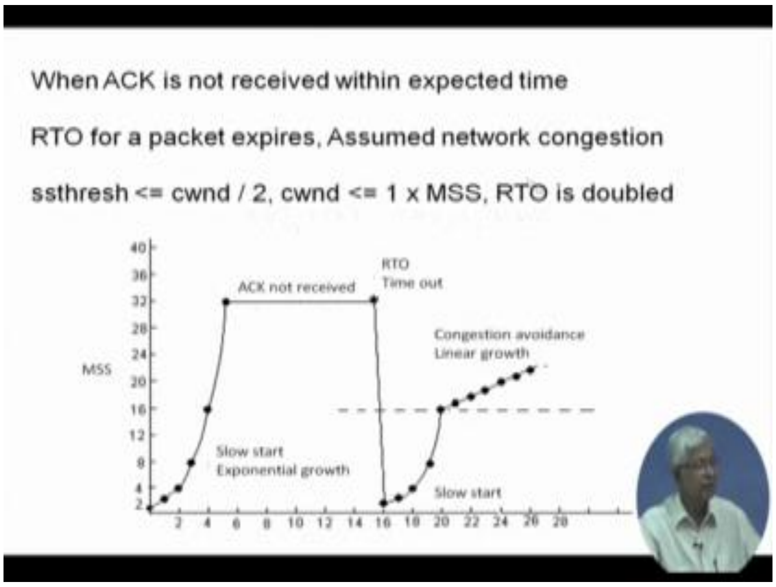
Given segment size, RTT and ssthresh and latest cwnd derive an expression to find duration of congestion avoidance phase.

Number of RTT to reach from ssthresh to latest cwnd = M
As cwnd increases linearly by one segment in CA phase,
M = (latest cwnd – ssthresh) / segment size

Duration of congestion avoidance phase = M x RTT

Given segment size, RTT and s s threshold and the latest cwnd derive an expression to find duration in the congestion avoidance phase. So, when it is going linearly that is cwnd is more than s s threshold. Number of RTT to reach from s s threshold to the latest cwnd is called M. So as cwnd increases; linearly by one segment in congestion avoidance phase, so M is latest cwnd which is higher than s s threshold minus s s threshold, so how much I have divided by the segment size; so I got m. So, M times RTT that is the duration of congestion avoidance phase.

(Refer Slide Time: 08:35)



When ACK is not received within expected time

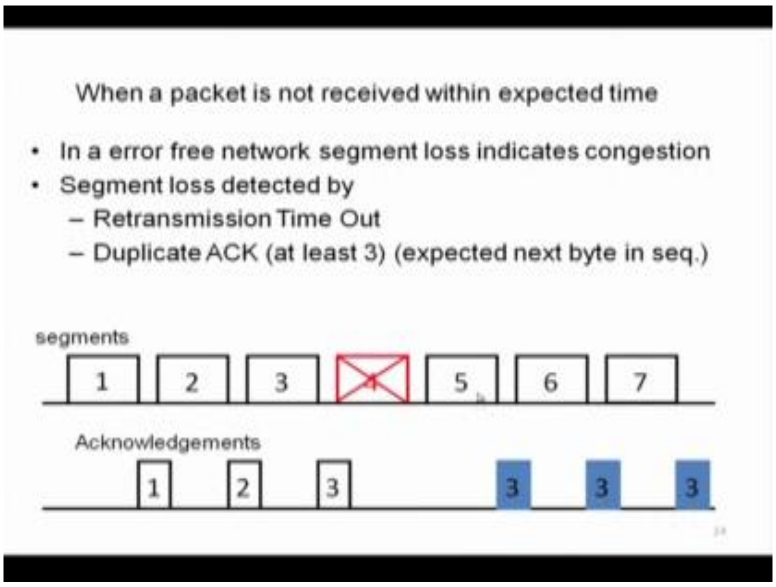
RTO for a packet expires, Assumed network congestion

ssthresh <= cwnd / 2, cwnd <= 1 x MSS, RTO is doubled

So, it goes like that, so it goes x axis is the RTT where the y axis is MSS. So, it goes like this exponentially and let us see at a time, let us say acknowledgment not received then; that means, RTO fired, once the RTO fires; TCP flow control, sender flow control says that some congestion has occurred. So, you start from the beginning, you drop and start sending only 1 byte, one segment and since and s s threshold whatever was there is also reduce, so it goes slowly and once it reaches s s threshold it grows.

So, s s threshold level is also reduced by half; RTO for a packet expires assumed network congestion, so s s threshold is reduced by cwnd by 2 whatever cwnd was there. So, cwnd is reduced to 1, the actual case todays scenario it is doing and RTO is doubled.

(Refer Slide Time: 09:41)

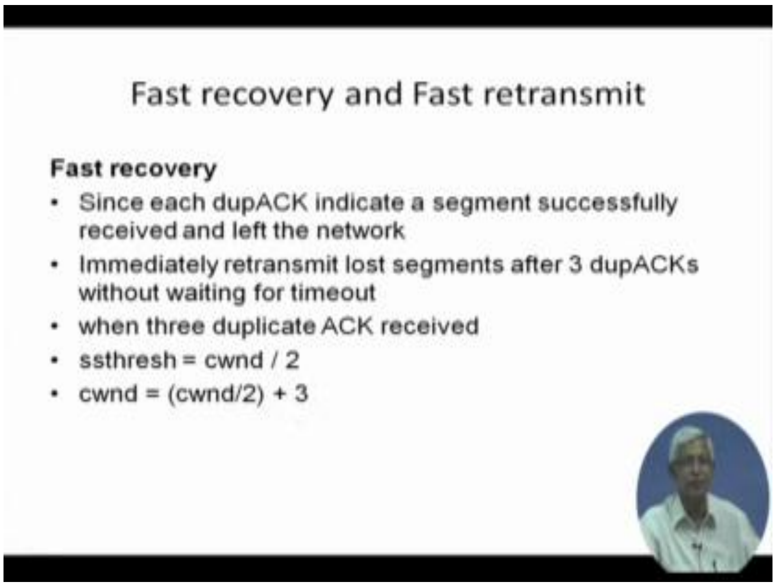

When a packet is not received within expected time

- In a error free network segment loss indicates congestion
- Segment loss detected by
  - Retransmission Time Out
  - Duplicate ACK (at least 3) (expected next byte in seq.)

segments
1
2
3
4
5
6
7
Acknowledgements
1
2
3
3
3
3

Let this details, I am going to details of that, when a packet is not received within the expected time; in a error free network segment loss indicates a congestion that is the TCP protocol, but segment loss is detected by the retransmission time out or duplicate ACK, we have seen earlier the duplicate ACK says that there is segment loss, but subsequent sequent has gone as expected next byte in the sequence, like that in a buffer segment 1, 2, 3, 4 has been not received so there is a hold.

(Refer Slide Time: 10:14)



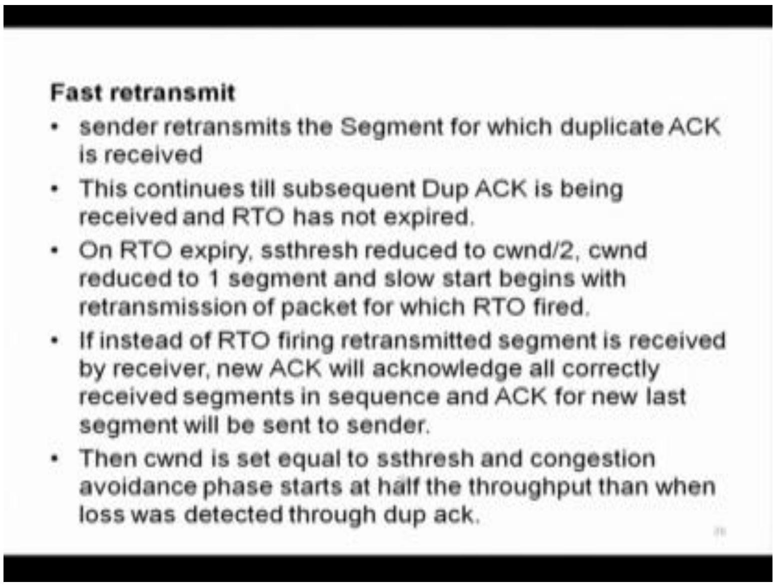
Fast recovery and Fast retransmit

Fast recovery

- Since each dupACK indicate a segment successfully received and left the network
- Immediately retransmit lost segments after 3 dupACKs without waiting for timeout
- when three duplicate ACK received
- ssthresh = cwnd / 2
- cwnd = (cwnd/2) + 3

So, we will send the acknowledgment for 3, 3, 3 like that all subsequent segment when it is received. Now there is say protocol, if RTT do not wait till RTT, a segment is loss 3 acknowledgment has been received duplicate ACKs, so you can called a fast recovery. So, since each duplicate ACK indicates that segment is successfully received and left the network, so you can fill up the network immediately retransmit the loss segments after the 3 duplicate ACKs, without waiting for its timeout; if it is time out then you have to go down to almost 0 that is one segment you have start and then when 3 duplicate ACKs received, s s threshold is reduced to 2 half, that is the fast recovery phase and cwnd; new cwnd by 2 and since 3 segments have been sent, so it will be increased by 3, so slightly inflating that, so this is the quick recovery of the system without going down to slow start.

(Refer Slide Time: 11:07)

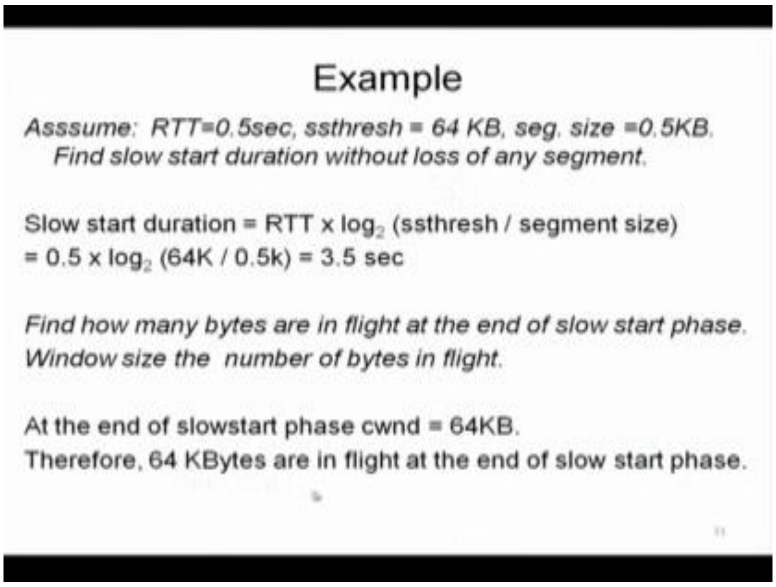


Fast retransmit

- sender retransmits the Segment for which duplicate ACK is received
- This continues till subsequent Dup ACK is being received and RTO has not expired.
- On RTO expiry, ssthresh reduced to cwnd/2, cwnd reduced to 1 segment and slow start begins with retransmission of packet for which RTO fired.
- If instead of RTO firing retransmitted segment is received by receiver, new ACK will acknowledge all correctly received segments in sequence and ACK for new last segment will be sent to sender.
- Then cwnd is set equal to ssthresh and congestion avoidance phase starts at half the throughput than when loss was detected through dup ack.

And then there is something called fast retransmit, when sender retransmit the segment for which duplicate ACK is received, this continues till subsequent duplicate ACK is being received and RTO has not expired. For each duplicate ACK, 1, 1 segment can be sent, if the RTO expires; s s threshold will be reduced to cwnd by 2; cwnd received reduced to 1 segment and slow start begins, that is a usual RTO procedure and if RTO fire has not occurred and the actual, the new acknowledgment is new segment is acknowledged all previously received segment, then you continue for the last segment sending, then cwnd is set equal to s s threshold and congestion avoidance in phase starts at half the throughput; there was a loss. So, which may indicate congestion, so you reduce the throughput and when the loss was detected and through duplicate ACK, so this is called fast retransmit without going slow start phase, a half way through it will starts.

(Refer Slide Time: 12:08)



Example

Asssume: RTT=0.5sec, ssthresh = 64 KB, seg. size =0.5KB.
Find slow start duration without loss of any segment.

Slow start duration = RTT x $\log_2$ (ssthresh / segment size)
= 0.5 x $\log_2$ (64K / 0.5k) = 3.5 sec

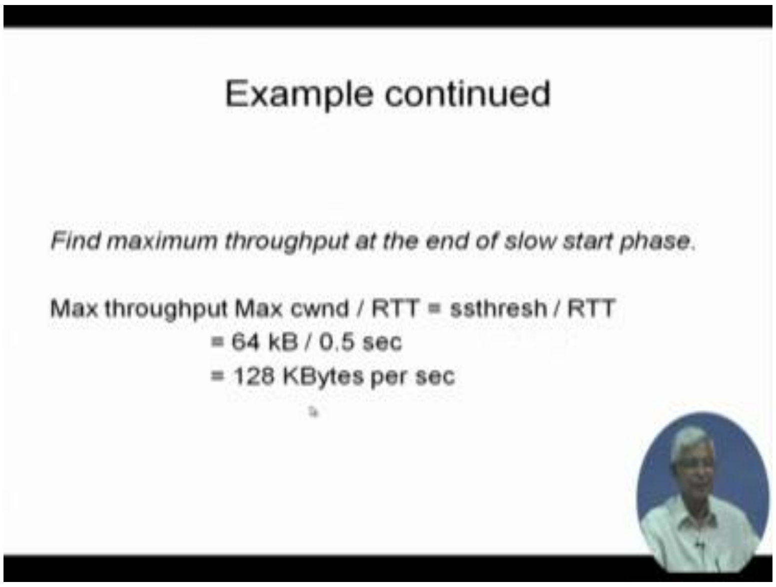
Find how many bytes are in flight at the end of slow start phase.
Window size the number of bytes in flight.

At the end of slowstart phase cwnd = 64KB.
Therefore, 64 KBytes are in flight at the end of slow start phase.

Now, we will use a quick, some example to understand with certain numbers, let us assume RTT is 0.5 second; which is satellite delay, s s threshold is 64 kilobytes and segment size is half kilobyte. Find slow start duration without loss of any segment, slow start duration will be RTT into log base 2, s s threshold and segment size we have seen this earlier. So, this is 0.5, this RTT is 0.5 second multiplied by log of 64 k by 0.5 which is 3.5 seconds, the slow start duration to reach the s s threshold. Find how many bytes are in flight at the end of slow start phase, window size, the number of window size, the number of bytes in the flight.

So, since we are reaching s s threshold level, so at the end of the slow start phase; it is 64 kilobytes which is already specified is threshold therefore, 64 kilobytes are in the flight at the end of slow start phase.

(Refer Slide Time: 13:09)



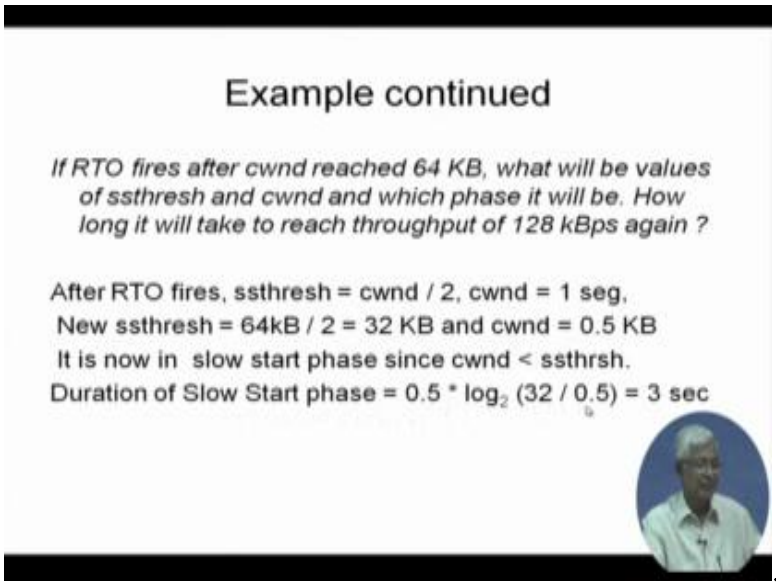
Example continued

Find maximum throughput at the end of slow start phase.

Max throughput Max cwnd / RTT = ssthresh / RTT
= 64 kB / 0.5 sec
= 128 KBytes per sec

And find the maximum throughput at the end of slow start phase; that is at when it reaches the s s threshold, so maximum throughput is the maximum cwnd by RTT, s s threshold by RTT; which is 64 kilobytes by 0.5 seconds which is 128 kilobytes per second, after that congestion level phase start.

(Refer Slide Time: 13:27)



Example continued

If RTO fires after cwnd reached 64 KB, what will be values of ssthresh and cwnd and which phase it will be. How long it will take to reach throughput of 128 kBps again ?

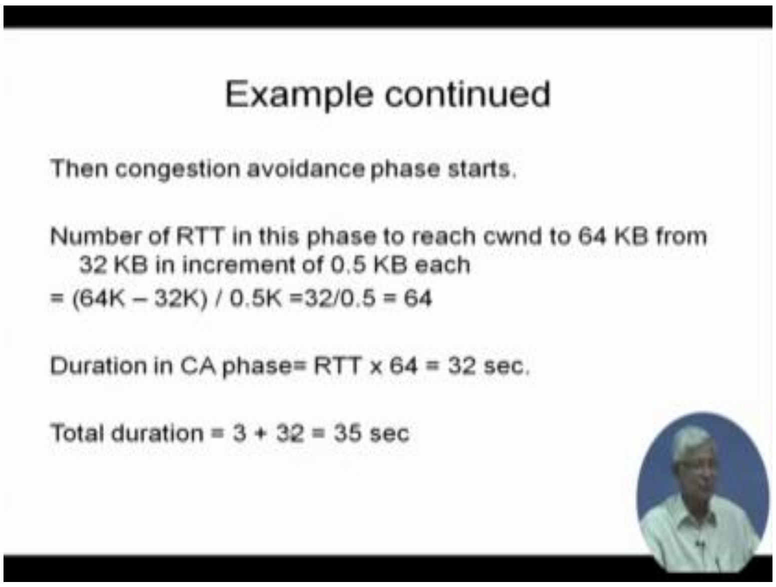
After RTO fires, ssthresh = cwnd / 2, cwnd = 1 seg,
New ssthresh = 64kB / 2 = 32 KB and cwnd = 0.5 KB
It is now in slow start phase since cwnd < ssthrsh.
Duration of Slow Start phase = 0.5 * $\log_2 (32 / 0.5)$ = 3 sec

.

So let us do some calculations on that, before that okay; if RTO fires after cwnd reached s s threshold that is 64 kilobytes, what will be the values of new s s threshold and cwnd and which phase it is, how long it will take to reach the throughput of 128 kilobytes per second again. So, after RTO fires, s s threshold will be cwnd by 2 and cwnd will be sent to 1 segment, which is; s s threshold will be half of 64 bytes which is 32 kilobytes and cwnd is 0.5 kilobyte.

It is now in the slow start phase because cwnd is lower than new s s threshold, so duration of this phase will be again 0.5 log 2, now it is 32 kilo bytes. So, 32 by 0.5 that is equal to 3 second, this time it will be 3 second, then the congestion avoidance phase starts.

(Refer Slide Time: 14:25)



Example continued

Then congestion avoidance phase starts.

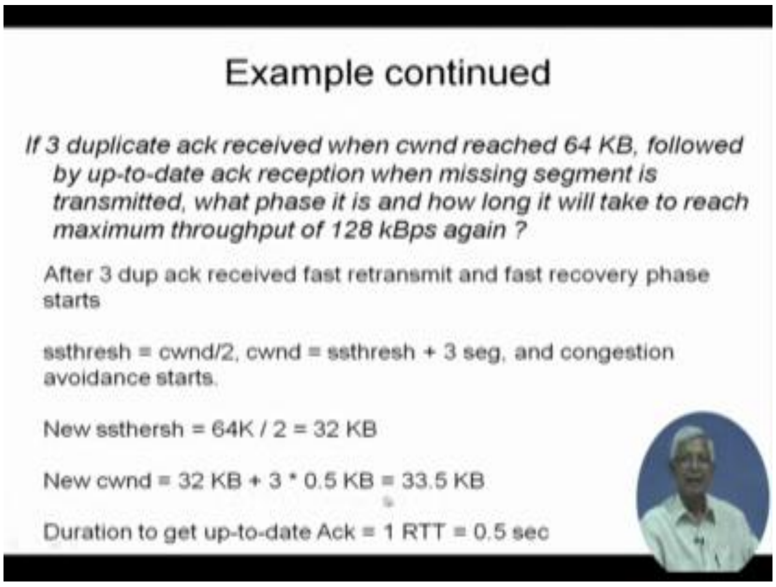
Number of RTT in this phase to reach cwnd to 64 KB from 32 KB in increment of 0.5 KB each
= (64K – 32K) / 0.5K =32/0.5 = 64

Duration in CA phase= RTT x 64 = 32 sec.

Total duration = 3 + 32 = 35 sec

So, the number of RTT in this phase to reach the cwnd 64 kilobytes from 32 kilobytes, in increment of 0.5 kilobyte each is, it is a linear growth, so 64 k minus 32 k divided by 0.5 k is 32 by 0.5, 64 steps are required, so duration of c w c a phase is RTT into 64; RTT is 0.5; so it becomes 32 seconds, linearly it is going very very slowly. So, total duration is 3 second plus 32 seconds that is 35 seconds, now you see with the example of the RTT is equal to 0.5, this two phases are RTO firing what happens all these things we have seen and then you see how slowly it is it is going, the maximum throughput that the network capacity what it is; it can support and how slowly we are trying to we are reaching to that maximum capacity.

(Refer Slide Time: 15:23)



Example continued

If 3 duplicate ack received when cwnd reached 64 KB, followed by up-to-date ack reception when missing segment is transmitted, what phase it is and how long it will take to reach maximum throughput of 128 kBps again ?

After 3 dup ack received fast retransmit and fast recovery phase starts

ssthresh = cwnd/2, cwnd = ssthresh + 3 seg, and congestion avoidance starts.

New ssthersh = 64K / 2 = 32 KB

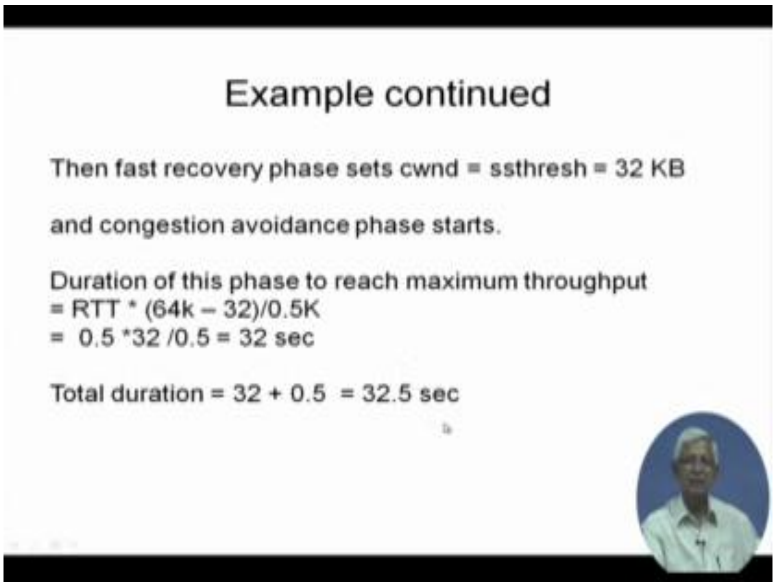
New cwnd = 32 KB + 3 * 0.5 KB = 33.5 KB

Duration to get up-to-date Ack = 1 RTT = 0.5 sec

Now if 3 duplicate ACKs received when the cwnd reaches 64 kilobytes, followed up by up-to-date ACK reception when the missing segment is transmitted, what phase it is and how long it will take to reach the maximum throughput of 128 kilo byte per second again? These have fast retransmit and fast recovery phase, so after 3 duplicate ACK received, fast retransmit and fast recovery phase starts. Now s s threshold is cwnd by 2 as per fast recovery and fast retransmit and cwnd will be s s threshold plus 3 segment because 3 duplicate ACK received, so 3 segments has increased and congestion avoidance starts. Congestion avoidance starts because cwnd is more than s s threshold this new s s threshold plus 3 segments, since it is more; so therefore it is a congestion avoidance phase.

So, new s s threshold value is 64 kilobyte by 2 is 32 kilobyte and new cwnd value is 32 kilobyte plus 3 times 0.5 kilobyte that is 33.5 kilobytes and the duration to get to up-to-date ACK is 0.5 second.

(Refer Slide Time: 16:42)



Example continued

Then fast recovery phase sets cwnd = ssthresh = 32 KB

and congestion avoidance phase starts.

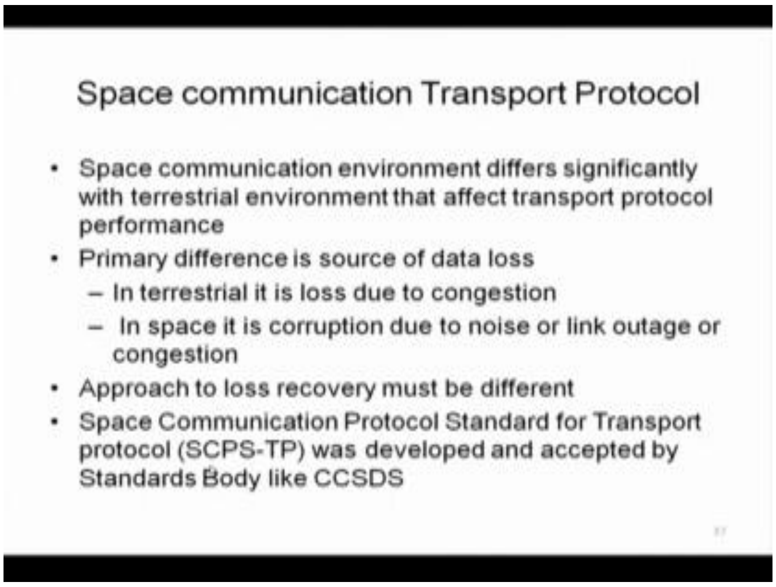
Duration of this phase to reach maximum throughput
= RTT * (64k – 32)/0.5K
= 0.5 *32 /0.5 = 32 sec

Total duration = 32 + 0.5 = 32.5 sec

Then the fast recovery phase sets the cwnd equal to s s threshold 32 kilobyte and congestion avoidance phase starts. Now because the cwnd is more than s s threshold; duration of this phase to reach the maximum throughput which is 64 kilobyte is 64 k minus 32 k divided by k is missing here, 32 k divided by 0.5 into RTT, so that makes it 32 second just like our earlier example it takes that much of time, so the total duration is 32.5 seconds.

So, this process of our with some simple type of example what we have seen is that; if there is a packet loss either by RTO firing or by duplicate ACK received, whichever way you go because RTTs are large and window is maintained as 64 kilobyte, you are reaching of that more than 30 seconds; you are reaching the maximum network capacity or maximum throughput it takes much a long time, so this is the one of the major issue that people have faced even the coming to this long delay fat network, where the packet drop may not be that is very important, the packet drop may not be due to congestion. In the normal TCP protocol, it is assume that it is due to congestion any packet drop is due to congestion not because of error.

(Refer Slide Time: 18:29)



Space communication Transport Protocol

- Space communication environment differs significantly with terrestrial environment that affect transport protocol performance
- Primary difference is source of data loss
  - In terrestrial it is loss due to congestion
  - In space it is corruption due to noise or link outage or congestion
- Approach to loss recovery must be different
- Space Communication Protocol Standard for Transport protocol (SCPS-TP) was developed and accepted by Standards Body like CCSDS

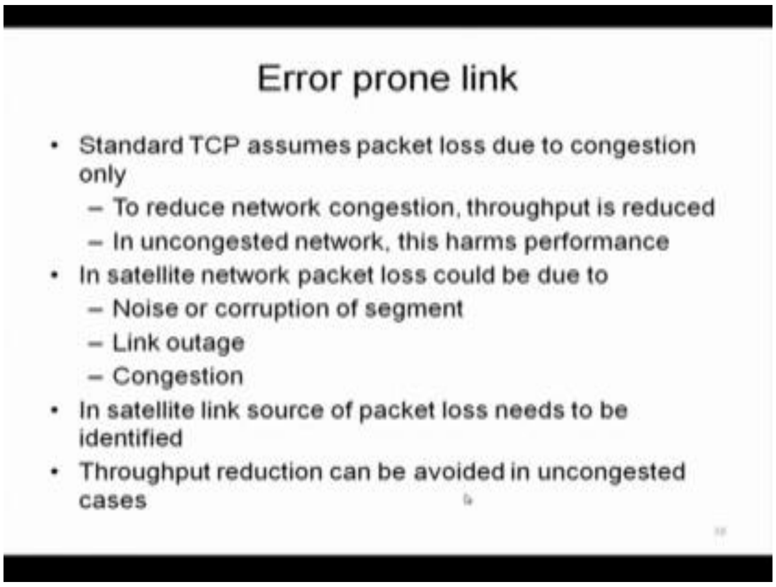
So, therefore the space communication transport protocol which is slightly modification of the standard TCP for the space related use has been proposed. The space communication environment differs significantly with the terrestrial environment, that aspect that is the affect the transport protocol as you take. So, transmit protocol performance would be different, not the expected, so the primary difference is there are many difference, so the primary difference is the what is the source of data loss because we have seen the satellite link is a error prone link and we have seen that it is intermittent connections, we have also seen that network capacity is limited because of that there may be certain data losses which may not be always congestion, there will be congestion also there is possibility of congestion.

So, in terrestrial any loss is due to congestion it is assumed that it is due to congestion. Whereas in space, it is a corruption due to noise or link outage or congestion either due to noise or the link outage or the congestion or any one of the three are possible, because of it is because of noise and link outage, it should not be treated as congestion and you should slow down the flow drastically.

So, approach to the loss recovery must be different that is one of the major issue in the modification of the TCP protocol in the space environment. So, space communication protocol is a standard for a space communication protocol standard for transport protocol the name is SCPS-TP that was developed and accepted by standard body like a CCSDS which stands for Consultative Community for Space Data System; this like ITU; International Telecommunication Union which we discussed earlier the specify, the physical satellite to Earth, Earth to satellite the link, what should be the parameter where the satellite can be placed, what should be the frequencies.

So, similarly for the space data systems ITU talks about not only space, it talks about other systems also terrestrial as well as meteorological and many many other system. Only for space there is a consultative community for space data system, they specify this that is the standard body most of the space agents, all the space agents since they follow this standard. So, they have developed or they have studied and modified the TCP protocol according space SCPS-TP then briefly touch up on SCPS-TP, what are the modifications that is done in TCP and why and how.

(Refer Slide Time: 21:29)



Error prone link

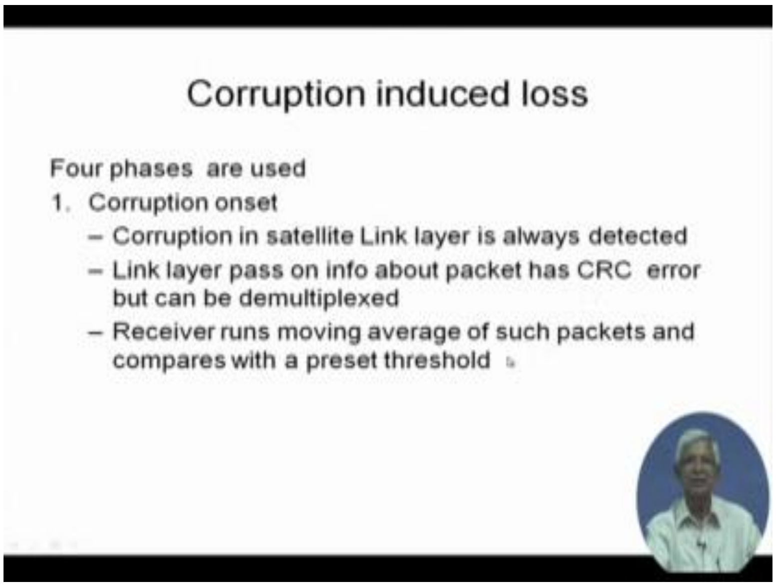
- Standard TCP assumes packet loss due to congestion only
  - To reduce network congestion, throughput is reduced
  - In uncongested network, this harms performance
- In satellite network packet loss could be due to
  - Noise or corruption of segment
  - Link outage
  - Congestion
- In satellite link source of packet loss needs to be identified
- Throughput reduction can be avoided in uncongested cases

So, standard TCP assumes the packet loss due to congestion, congestion only and to reduce the network congestion, throughput is reduced that is what we have seen that is drastically reduced and goes into slow start phase or even in fast retransmit fast recovery the network throughput is reduced to half uncongested network that harms in the performance.

So, in satellite network packet loss could be due to noise or due to corruption of a segment noise is a corruption of segment. This is a thermal noise what we are talking and it could be the link outage as we discussed that the satellite may be go behind on the shadow of another object which could be another planet or it could be link outage due to other phenomena in the propagation phenomena like heavy rain or sort of flare etcetera and it could be also due to congestion that is also another possibility.

Now in the satellite link, the source of packet loss needs to be identified like here in the case of terrestrial link the source of that error is always assumed to be congestion; here out of these 3 whether congestion link outage or corruption that has to be identified. So, accordingly throughput reduction can be avoided in the uncongested cases, this is the major thing how it is done; it is error prone link.

(Refer Slide Time: 23:10)



Corruption induced loss

Four phases are used

1. Corruption onset
   - Corruption in satellite Link layer is always detected
   - Link layer pass on info about packet has CRC error but can be demultiplexed
   - Receiver runs moving average of such packets and compares with a preset threshold

So, now will discuss in each of the three cases that how this four phases which are used for corruption that is for first is the onset of the corruption has to be recognized. Now how it can be recognized, at the TCP level we have seen the sender looks only at the receiver advertised window or it tries to see when the acknowledgement is coming or acknowledgment duplicate ACKs are coming or not from that.

In case of space link that corruption onset can be found out the from the link layer. The link layer has a protocol which detects the error, that is a there is a c r c a in case of t d m a we have seen that c r b t r Unicode missing, the probability of miss of Unicode and also there are other as the protection error checksums are there.

So, link layers somehow know that there is error; which is occurred link layer can find out. So, the corruption started that is noise has occurred; link layer finds out, so if this information is passed on to the higher layer, then higher layer can take as an understanding that it is not congestion, it is corruption that is what is happening. So, link layer pass on this information about the loss; that is the packet as a CRC error, CRC; Cyclic Redundancy Check; CRC error, but can be demultiplexed.

Receiver runs the moving average of such packets and compares with a preset threshold, so receiver does not take immediate decision; it just goes on checking how much noise is increasing. So, it goes on accumulating that this information of CRC error which is coming, how much over a period of time. So, it runs a moving average and of course, it checks with a threshold, so if it is a very spheres or spare phenomena then it ignores. It knows that there is error, but if it is exceeding certain threshold; it alerts that there is a corruption on set is happening. Anyway the time is almost up, so we will go continue the discussion in the next session.