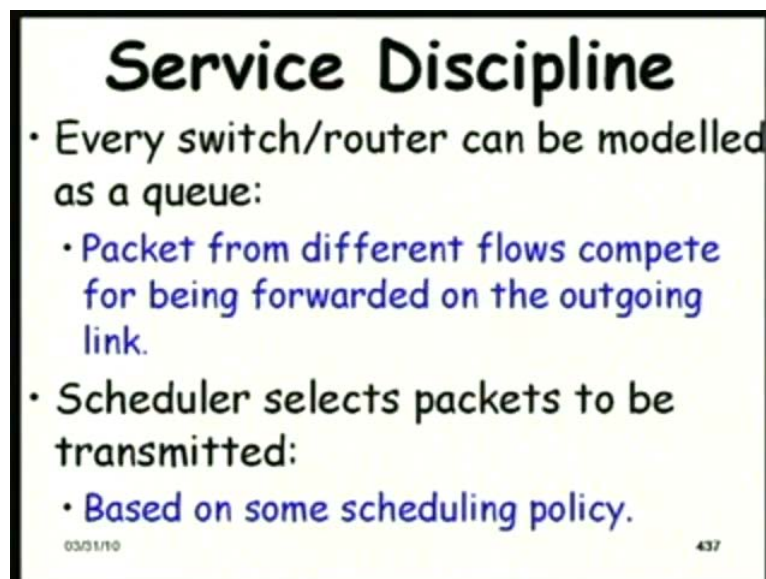


Real-Time Systems
Prof. Dr. Rajib Mall
Department of Computer Science and Engineering
Indian Institute of Technology, Kharagpur

Lecture No. # 39
Real-Time Communication Over Packet Switched Networks (Contd.)

So let us continue from where we had left last time. If you remember, we are discussing about packet switched networks and how real-time communication can be supported in these networks. So, we had looked at resource reservation and we had looked at rate control; these two things we had looked at last time. So, we were trying to discuss the service discipline and we are saying that a traffic becomes distorted as it proceeds in the network, and the service discipline's responsibility is to reset this traffic to its original characteristic.

(Refer Slide Time: 01:02)



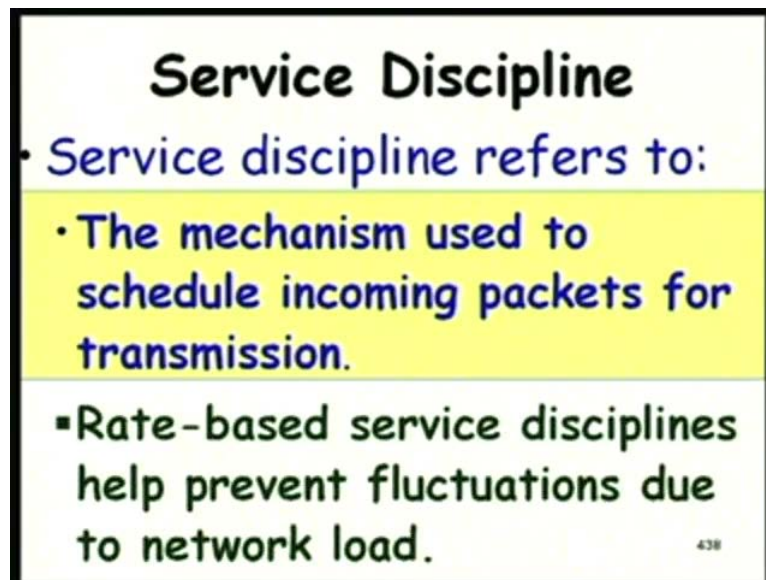
Service Discipline

- Every switch/router can be modelled as a queue:
 - Packet from different flows compete for being forwarded on the outgoing link.
- Scheduler selects packets to be transmitted:
 - Based on some scheduling policy.

03/01/10 437

So, let us proceed there from that point. So, a service discipline we had said is can be thought of as a queue. In this queue, packets from various connections reside and this typically on the outgoing link; and there is a scheduler, the service discipline includes a scheduler and this buffer. So, the buffer has the different packets from various connections. And then the scheduler selects a packet to be transmitted on the outgoing link based on the resource that has been reserved for that connection.

(Refer Slide Time: 01:52)



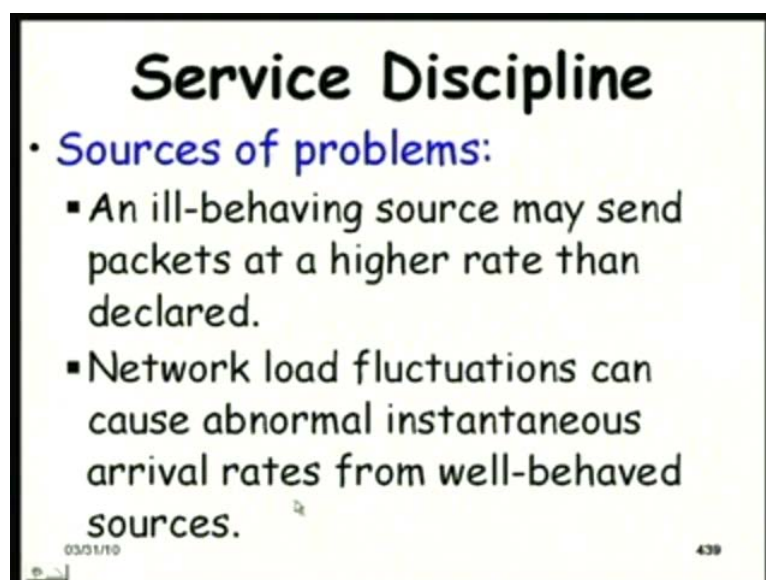
Service Discipline

- Service discipline refers to:
 - The mechanism used to schedule incoming packets for transmission.
 - Rate-based service disciplines help prevent fluctuations due to network load.

438

So, in simple words we can say that the service discipline is the mechanism used to schedule incoming packets for transmission, where we look at the resource that is reserved; the service discipline examines the resource that is reserved, and also to reset a distorted traffic. And the rate based service disciplines are used to and these help prevent fluctuations due to network load or to reset the traffic.

(Refer Slide Time: 02:33)



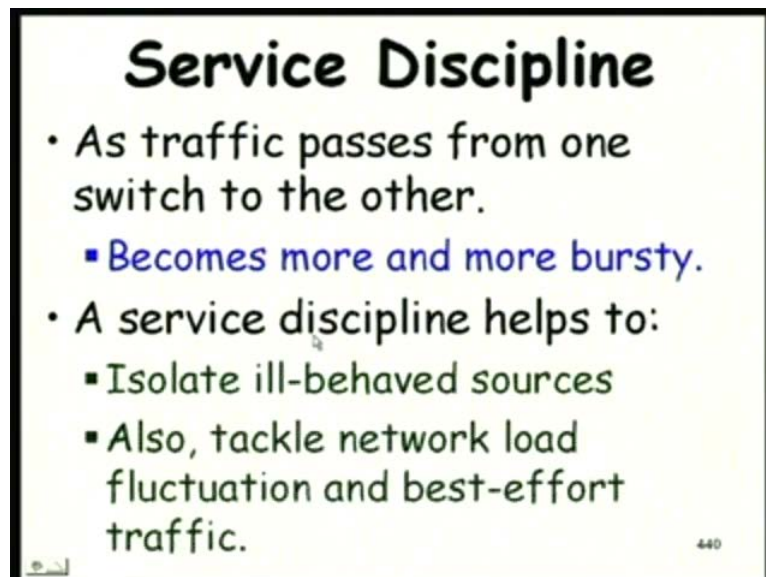
Service Discipline

- Sources of problems:
 - An ill-behaving source may send packets at a higher rate than declared.
 - Network load fluctuations can cause abnormal instantaneous arrival rates from well-behaved sources.

03/01/10 439

The sources of the problem is the load fluctuation, which cause the abnormal instantaneous rate even from well behaved sources; or in other words, in simple words, a traffic becomes bursty even when it is well behaved at the sources.

(Refer Slide Time: 03:05)



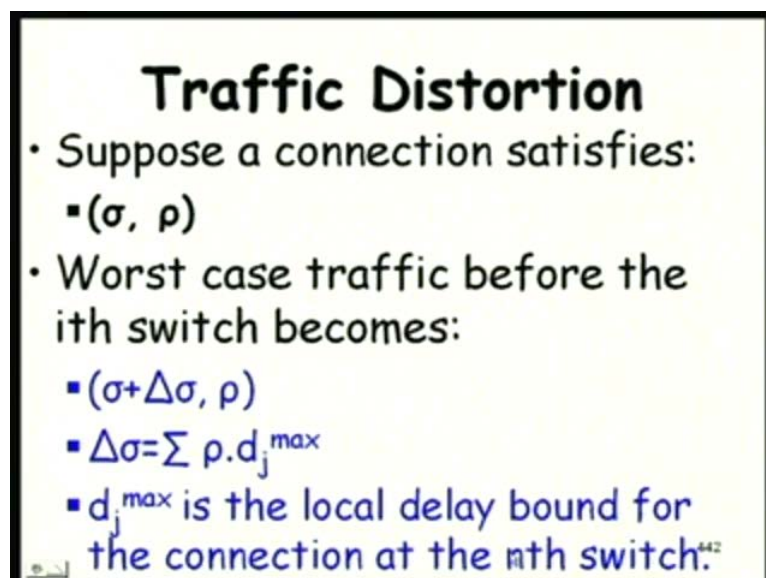
Service Discipline

- As traffic passes from one switch to the other.
 - Becomes more and more bursty.
- A service discipline helps to:
 - Isolate ill-behaved sources
 - Also, tackle network load fluctuation and best-effort traffic.

440

As the traffic passes from one switch to other, it becomes more and more bursty and the service discipline helps to tackle this.

(Refer Slide Time: 03:17)



Traffic Distortion

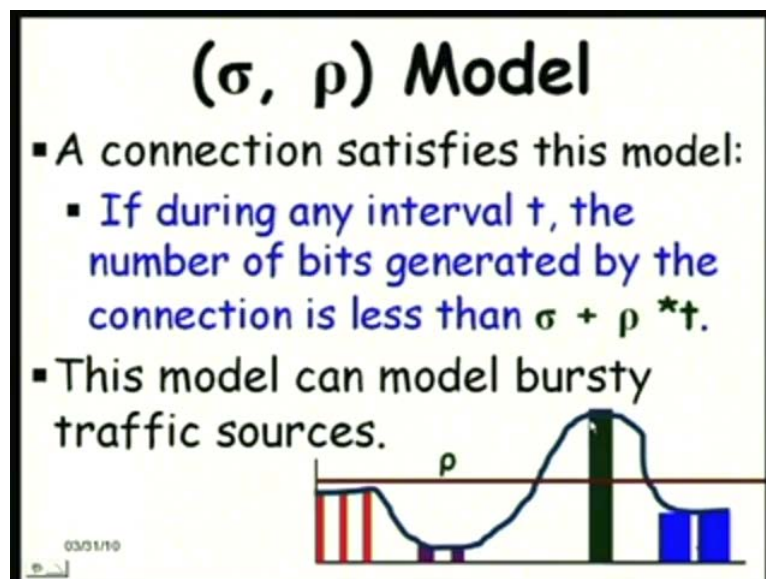
- Suppose a connection satisfies:
 - (σ, ρ)
- Worst case traffic before the i th switch becomes:
 - $(\sigma + \Delta\sigma, \rho)$
 - $\Delta\sigma = \sum \rho \cdot d_j^{\max}$
 - d_j^{\max} is the local delay bound for the connection at the n th switch.

442

So, we can think of one responsibility of the service discipline is distort some control in the traffic characteristics. So, the service discipline will reconstruct the traffic to its original

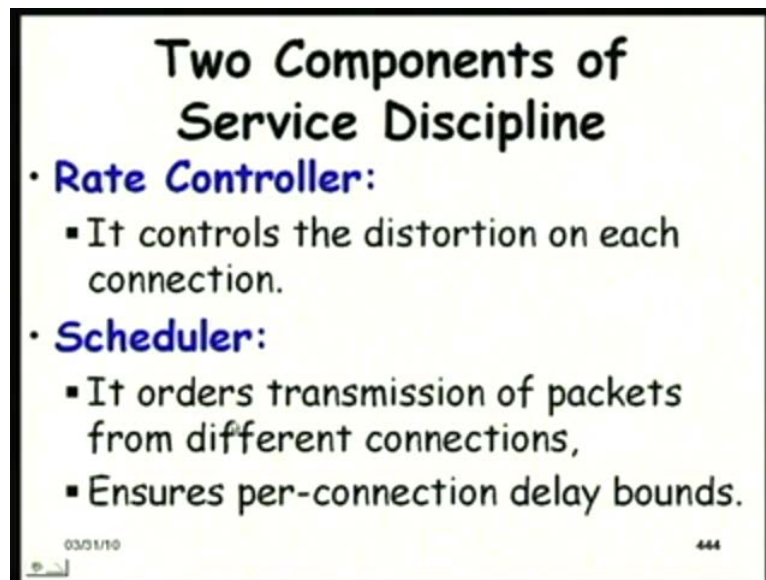
characteristic at every switch. And we are trying to examine the extent to which a traffic can become bursty after it is transmitted over a switch. The worst case traffic before the i^{th} switch will become $\sigma + \rho \sum_{j=1}^{i-1} d_j^{\text{max}}$, where σ is the burst size over the average traffic ρ and $\rho \sum_{j=1}^{i-1} d_j^{\text{max}}$ is summation of ρ into d_j^{max} , where d_j^{max} is the delay bound for the j^{th} switch, sorry j^{th} switch it should be; d_j^{max} is the delay bound for the j^{th} switch.

(Refer Slide Time: 04:46)



So, this we had already seen. Just I drawn this to help you understand what we mean by σ and ρ ; σ is the average traffic. So, the number of bit, the number of packets per t time units will be σ into t , and the burst size is given by a σ .

(Refer Slide Time: 05:15)



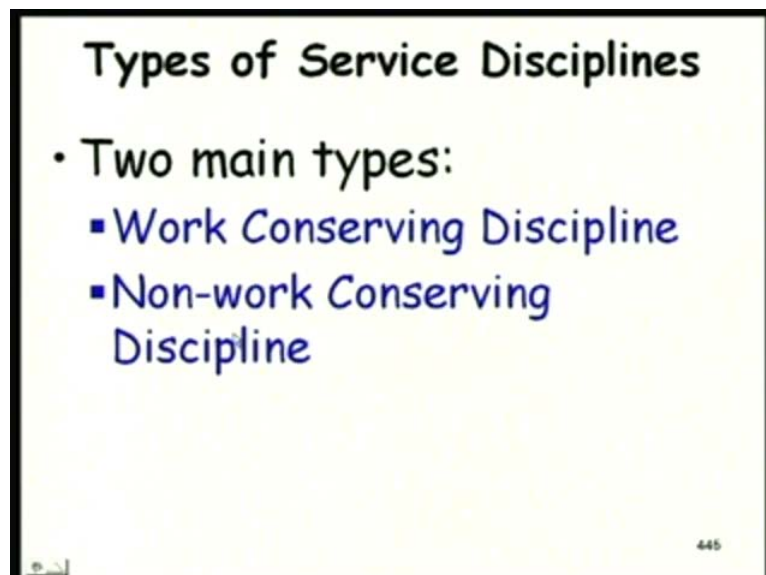
Two Components of Service Discipline

- **Rate Controller:**
 - It controls the distortion on each connection.
- **Scheduler:**
 - It orders transmission of packets from different connections,
 - Ensures per-connection delay bounds.

03/01/10 444

So, we can think of a service discipline to include a rate controller or distortion controller and a scheduler, which transmits packets from different connections based on the resource reservation. So, there are two main responsibilities of a service discipline: one is the rate controlling and other is scheduling - you have already seen that.

(Refer Slide Time: 05:43)



Types of Service Disciplines

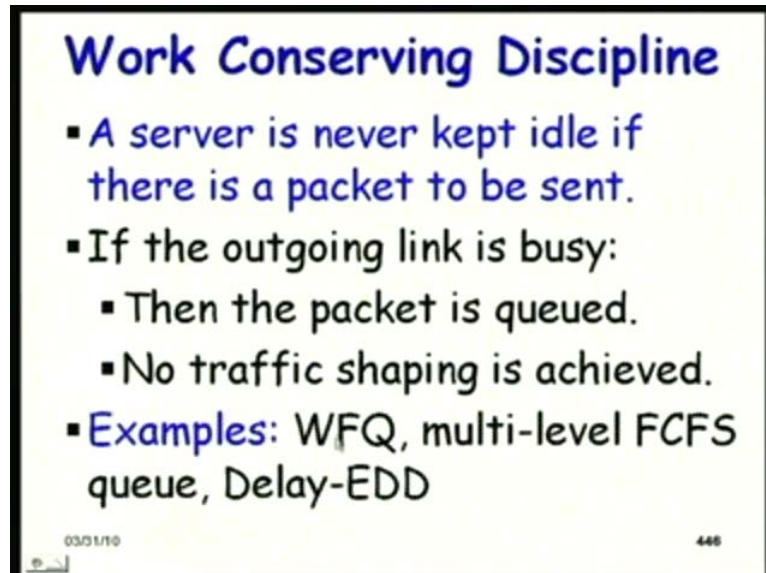
- Two main types:
 - **Work Conserving Discipline**
 - **Non-work Conserving Discipline**

445

Now let us see, what are the service disciplines that are being used popularly? We look at that, we will see that there are main two main types of service disciplines that are being used;

one is called as the work conserving discipline and the other is called as the non-work conserving discipline. Now, let us examine what are these two types of service disciplines.

(Refer Slide Time: 06:10)



First, let us look at the work conserving discipline. In the work conserving discipline, a server is never kept idle, if there is a packet to be sent. This is the service discipline we were calling it as a server. So, the server will not be idle, if there is a packet to be sent; it will send it out. It cannot be the case that the server is idling, and there is a packet is waiting. Of course, if the link is busy, then the packet will be queued, the server cannot send it out. And in work conserving discipline, no traffic shaping is achieved, because it does not delay them unnecessarily, right? If the link is free, it will transmit them.

A work conserving discipline does not reset traffic.

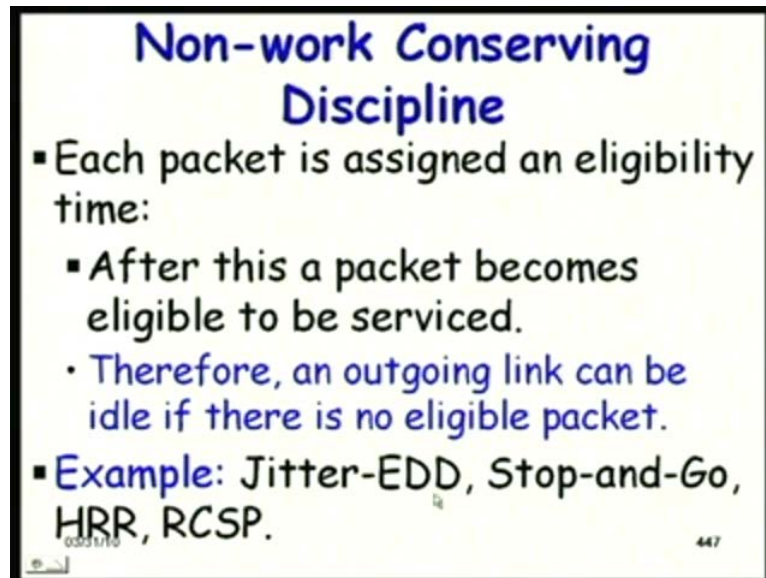
At the shaping is required, they are only at that level traffic becomes distorted.

Yeah traffic becomes distorted naturally in every packet switched network. So, if we have all cbr traffic, all are there is no burst in the sources at all, all are transmit at a cbr (constant bit rate) and then and also there is no non real-time traffic, then it will not become bursty.

This is applicable for cbr kind ((.)).

We can think of that. Wherever rate control is not a matter of concern, we can use a work conserving discipline. So, the examples of a work conserving discipline is weighted fair queue, multilevel FCFS queue, delay EDD - the names of some popular algorithms.

(Refer Slide Time: 08:12)



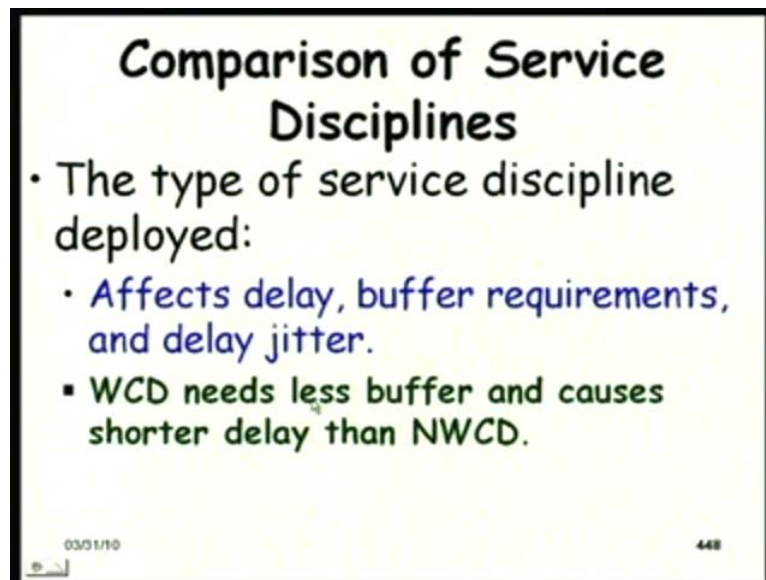
Non-work Conserving Discipline

- Each packet is assigned an eligibility time:
 - After this a packet becomes eligible to be serviced.
 - Therefore, an outgoing link can be idle if there is no eligible packet.
- **Example:** Jitter-EDD, Stop-and-Go, HRR, RCSP.

08/11/10 447

Now, on the other hand a non-work conserving discipline, here even when the packet has come and got queued, there will be an eligibility time assigned to the packet. Only after the expiry of the eligibility time, the packet will become eligible to be transmitted. So, it is possible therefore that an outgoing link can be idle and there are packets waiting, but nothing is getting transmitted **right**, because none have met their eligibility time requirement. So, here also there are many algorithms or service disciplines based on the non-work conserving discipline, for example Jitter EDD, Stop-and-Go etcetera. So, we will discuss few of the important ones.

(Refer Slide Time: 09:14)



Comparison of Service Disciplines

- The type of service discipline deployed:
 - Affects delay, buffer requirements, and delay jitter.
 - WCD needs less buffer and causes shorter delay than NWCD.

03/01/10 448

The type of the service discipline, whether you are using a work conserving or non-work conserving would depend on how much you can tolerate delay, whatever the buffer requirements at the switches, the tolerance of the delay jitter and so on. The work conserving disciplines need less buffer space and cause shorter delay than the non-work conserving discipline. Why is that? Yes, anybody would like to answer? This is the claim we are making, that the work conserving disciplines need less buffer space at the switch, and this cause shorter delay than the non-work conserving discipline.

(()) it will not buffer unnecessarily.

Yeah it does not delay them unnecessarily, that is the answer. It does not delay them unnecessarily. So, definitely the buffer requirement will be lower than the non-work conserving discipline and also since it delays them to a lower extent, the total delay will be less, is it not? Simple thing; let us proceed.

(Refer Slide Time: 10:32)

Comparison of Service Disciplines

- The type of service discipline deployed:
 - Affects delay, buffer requirements, and delay jitter.
 - WCD needs less buffer and causes shorter delay than NWCD.
 - WCD cannot bound delay jitter tightly, while NWCD can.

03/01/10 448

Another claim we are making here is that the work conserving disciplines cannot bound delay jitter to any requirement, whereas the non-work conserving discipline can. What about this? The work conserving discipline cannot bound delay jitter to any agreed quantity; so, the delay jitter will become, can become very high, while a non-work conserving discipline can bound the delay jitter. Do you agree with this? If you agree, then you will justify, why this is the case. Yes, anybody would like to tell anything about why I mean, this specific claim that a work conserving discipline cannot bound delay jitter tightly, while the non-work conserving discipline can. Can anybody try to justify this or and try to say that this is wrong?

Sir, in the in end of the series, it assigns the time delay to every packet.

Yes

And. So, that is why it is controlling the time delay (()) that ah

No, it is making it more delayed; that is all. But what about delay jitter? We are talking of delay jitter.

(()). So, it that is when it can make it more (()) or not.

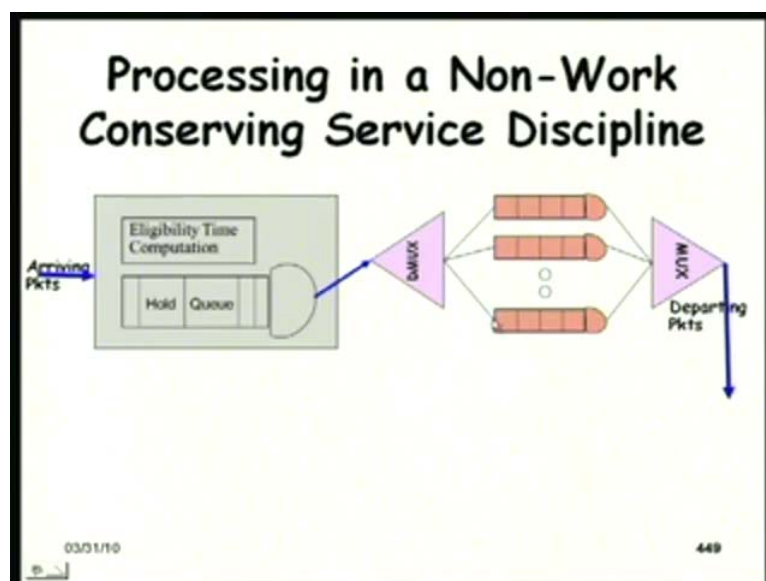
Not really. So, let us see why it is. Anybody would like to tell any other answer? Let us see why that is the case. See, if there is the traffic is bursty, the traffic has become bursty, we had

said that in the work conserving discipline, no traffic rate control is achieved; just transmits, right?

So, it is possible; depending on whether it is burst and has got queued or not, it can take different time to reach the destination. So, if it is just following a burst, it will take long time to reach the destination, longer time. But if it is common, there is not no burst; it will reach faster.

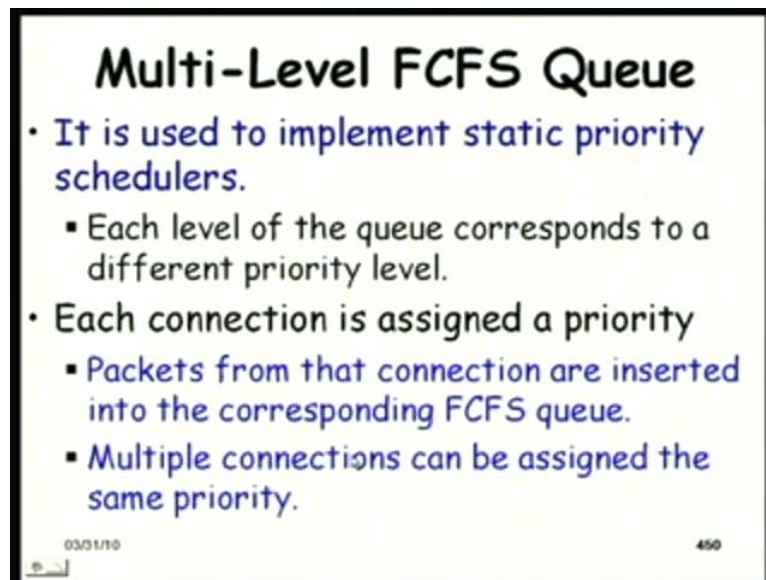
But whereas in a non-work conserving discipline, we are trying to make the traffic not bursty, right? We are controlling the burst size; so, the amount of jitter will also be less. So, the amount of time a packet will have to wait at a switch is almost constant, whereas in the other case it can either wait for a long time or can get transmitted quickly. Does that appear convincing?

(Refer Slide Time: 13:44)



So, in a non-work conserving discipline, the arriving packets they are queued and then there is a eligibility time computation. And based on that, they are sent to a demultiplexer, where they are transmitted on different queues based on the service, based on their reservation; and from there, they are transmitted on the same link. So, the different connections, they are queued here and transmitted over here.

(Refer Slide Time: 14:24)



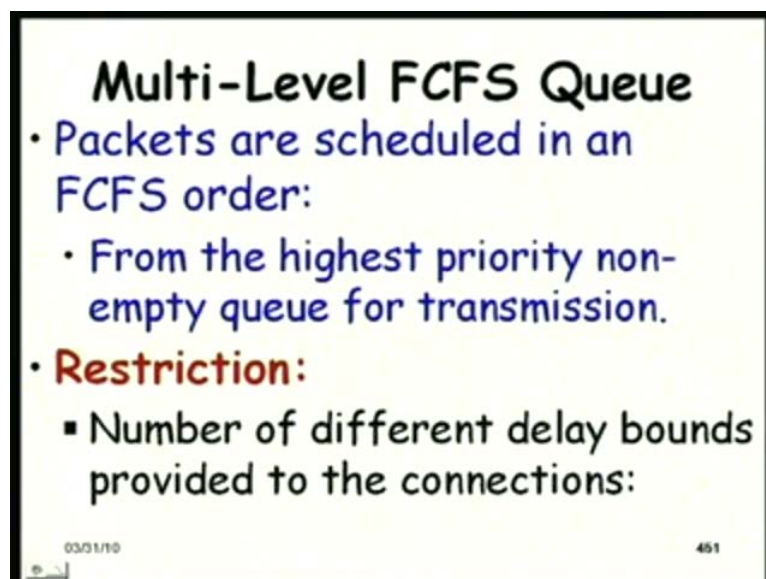
Multi-Level FCFS Queue

- It is used to implement static priority schedulers.
 - Each level of the queue corresponds to a different priority level.
- Each connection is assigned a priority
 - Packets from that connection are inserted into the corresponding FCFS queue.
 - Multiple connections can be assigned the same priority.

03/01/10 450

Now, let us look at a rate conserving, a work conserving service discipline: the name is multilevel FCFS q, a popular work conserving discipline. And it is used to implement static priority schedulers. So, here the different packets or different connections, I have given different priorities. Each level of the queue corresponds to a different priority level and naturally each connection is assigned a priority; and two connections may even have the same priority. And depending on the priority of the connection, it is inserted in the corresponding FCFS q; multiple connections even can have the same priorities.

(Refer Slide Time: 15:29)



Multi-Level FCFS Queue

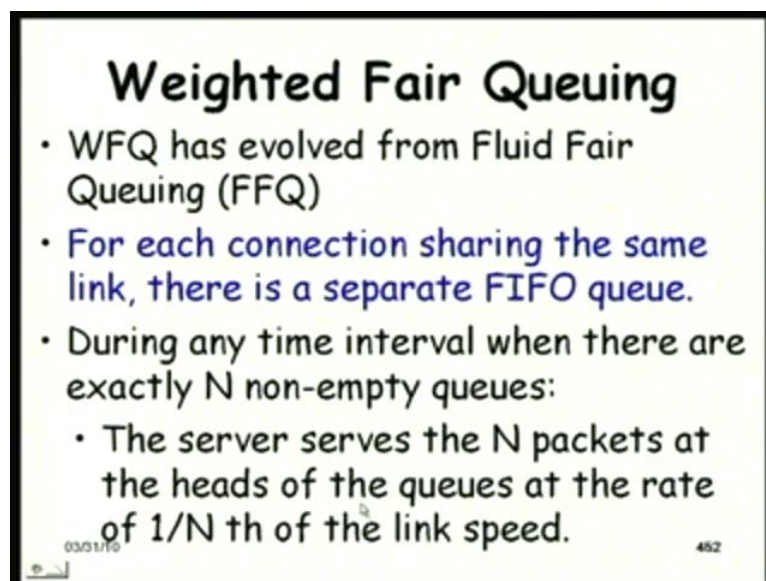
- Packets are scheduled in an FCFS order:
 - From the highest priority non-empty queue for transmission.
- **Restriction:**
 - Number of different delay bounds provided to the connections:

03/01/10 451

The packets once they are queued in the FCFS queue just follows the FCFS order. And from the highest priority queue, the transmission begins. If there is no highest priority, the queue is empty; then it will be the next that is taken for transmission. So here, how does the priorities get assigned? The priorities are get assigned by the different delay bounds that have been promised to the different connections. So, then number of delay bounds that can be promised to different connections depend on the number of priority levels, number of FCFS queues that can be supported.

So, here just see that they are just queued in different queues depending on their priority and then they are just taken up as soon as they are available, right? So, there is no waiting time involved here - unnecessary waiting time, right? As soon as the link is available, and there is no higher priority packet to transmitted, they are transmitted, right?

(Refer Slide Time: 16:54)



Weighted Fair Queuing

- WFQ has evolved from Fluid Fair Queuing (FFQ)
- For each connection sharing the same link, there is a separate FIFO queue.
- During any time interval when there are exactly N non-empty queues:
 - The server serves the N packets at the heads of the queues at the rate of $1/N$ th of the link speed.

03/01/10 452

Now, let us look at the weighted fair queuing. It is another algorithm - multilevel FCFS queue, very simple algorithm, the weighed fair queuing. Let us look at it. It has evolved from a popular algorithm called as fluid fair queuing. So, for all the connections sharing the link, there is a separate FIFO queue. So, as many outgoing links are there, that many FIFO queues will be there. And during any time when there are exactly non-empty n non-empty queues, the server serves N packets at the head of the queue at the rate of one by N of the link speed. So, what it says is that, if the FIFO queue for one link is non-empty, then it will transmit at

the rate of N packets that are at the head of the queue at the rate of 1 by Nth of the link speed.

(Refer Slide Time: 18:05)

Weighted Fair Queuing

- FFQ allows different connections to have different services shares.
 - Characterized by N positive real numbers $\phi_1, \phi_2, \dots, \phi_N$ for each queue.
- Let $B(\pi)$ be the set of non-empty queues and C be the link speed.
 - Then, the service rate for a non-empty queue i is:

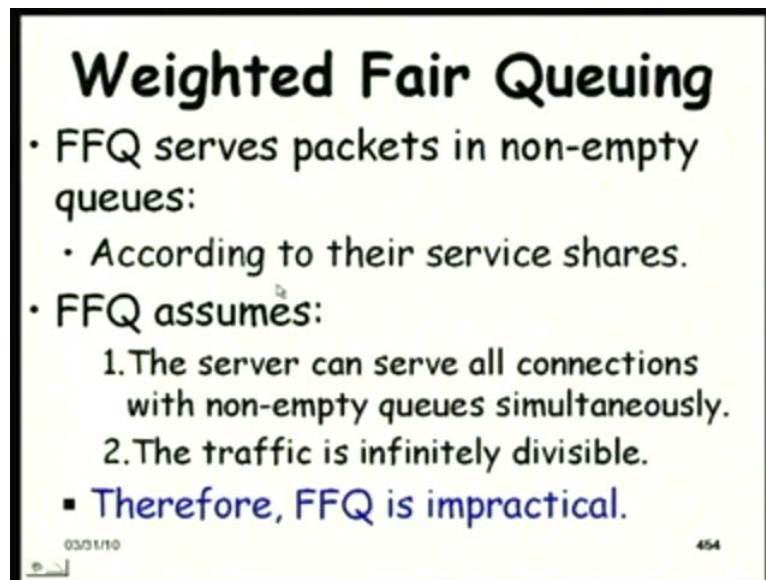
$$(\phi_i / \sum_{j \in B(\pi)} \phi_j) * C$$

03/01/10

453

But that was a simplistic thing that we said, that it just transmits n packets with the front of the queue. Actually it allows different connections to have different service shares. So, the service shares of different connections are given by $\phi_1, \phi_2, \dots, \phi_n$. Now if b is the set of non-empty queues and c is the link speed, then the service rate for a non-empty queue is ϕ_i by $\sum \phi_j$ into C . So, what it means is that for every connection on a link, its share will be C into the reservation for it divided by reservation for all others.

(Refer Slide Time: 19:09)



Weighted Fair Queuing

- FFQ serves packets in non-empty queues:
 - According to their service shares.
- FFQ assumes:
 1. The server can serve all connections with non-empty queues simultaneously.
 2. The traffic is infinitely divisible.
- Therefore, FFQ is impractical.

03/01/10 454

So, it serves packets according to their service shares; and there are two main assumptions that the server can serve all connections with non-empty queues simultaneously. So, the n packets each one with one by n , if there are equal reservation or depending on the ϕ , there will be transmitted; and another assumption is that the traffic is infinitely divisible, but we know that packet sizes are discrete.

So, the FFQ we need exact spirit is impractical, because it is based on the fluid transmission, right? Data is not; data is discrete, isn't it? It is not really a fluid. So, in the exact implementation of FFQ, it is impractical because traffic is not infinitely divisible; these are discrete, but of course, with an approximation, we can implement the FFQ.

(Refer Slide Time: 20:15)

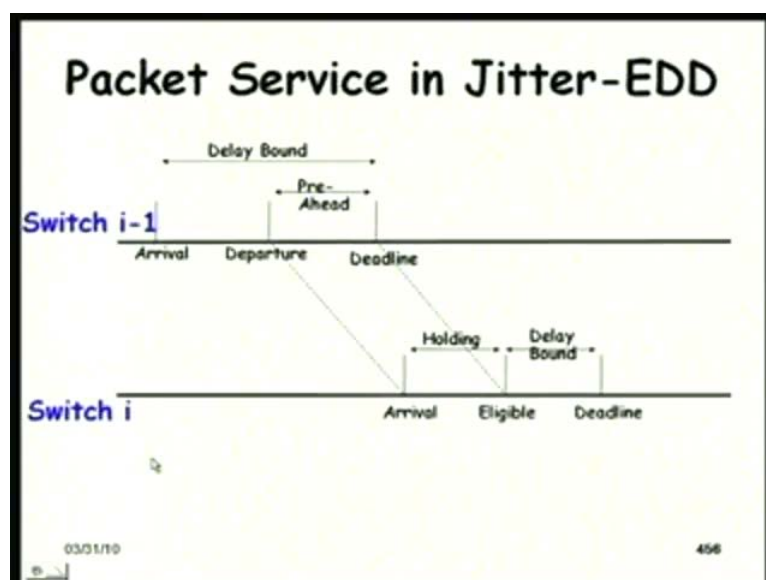
Jitter-Earliest Due Date

- When a packet is served:
 - A field in its header is stamped with the difference between its deadline and the actual finishing time.
- A regulator at the entrance of the next server holds the packet for this period:
 - Before it is eligible to be scheduled.
- It provides delay jitter bounds.

03/01/10 455

Now, let us look at the Jitter-Earliest due date. So, here when a packet is served by one link, sorry one router, a field in its header is stamped with the difference between the deadline of the packet and the actual time at which it is transmitted. And once it reaches the next router or the next server, it holds the packet for this period which is its eligibility period and it provides delay jitter bounds.

(Refer Slide Time: 20:57)



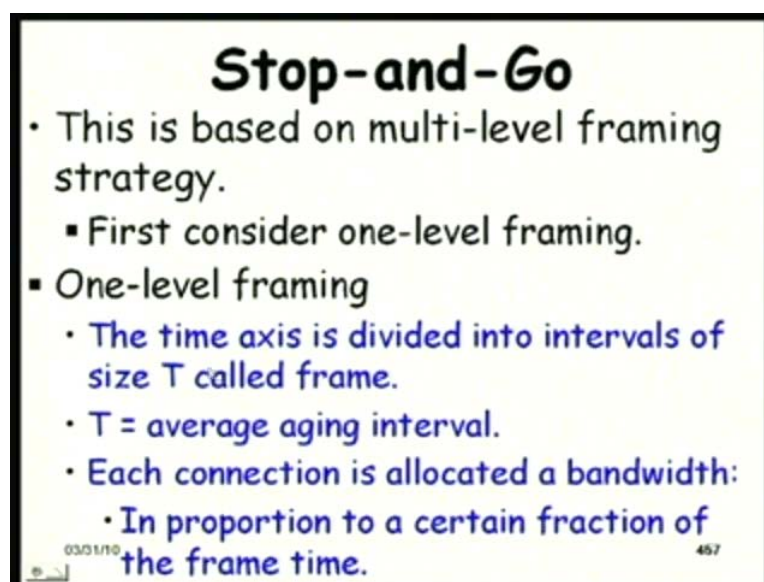
So, this is the explanation of this. See, here at every switch there is a delay bound. So, the delay bound for every switch is fixed. The total delay between, for the packet that was agreed during the connection is distributed among different switches.

Now, as it arrives at a switch, it is taken up for departure. But it should have been transmitted at this point. I mean, its deadline that was allotted for this switch, maximum deadline was this point, **right?** But it was transmitted here. So, this is called as the pre-ahead time before which it got transmitted.

So, this is stamped here; the actual time, at which it should have departed, I mean the maximum delay that should have encountered here and then the time at which it actually departed, these are stamped on the packet. And there is the packet comes to the next switch, the **Ith** switch - it is held for that time it arrives and then it becomes eligible only after the maximum time, it should have waited there.

Now, again there is a delay bounded, it might get transmitted some time here. And in the next switch, that will be taken care, right? So, it provides a good bound in the jitter.

(Refer Slide Time: 22:41)



Stop-and-Go

- This is based on multi-level framing strategy.
 - First consider one-level framing.
- One-level framing
 - The time axis is divided into intervals of size T called frame.
 - T = average aging interval.
 - Each connection is allocated a bandwidth:
 - In proportion to a certain fraction of the frame time.

03/01/10 457

Now, let us look at the stop-and-go. The stop-and-go algorithm is based on the multilevel framing strategy. But first let us consider the one level framing. And if we understand this, then we can extend it to a multilevel framing; because if we first start discussing on multilevel framing, it might appear bit confusing. So, let us look at the simplest case of a one

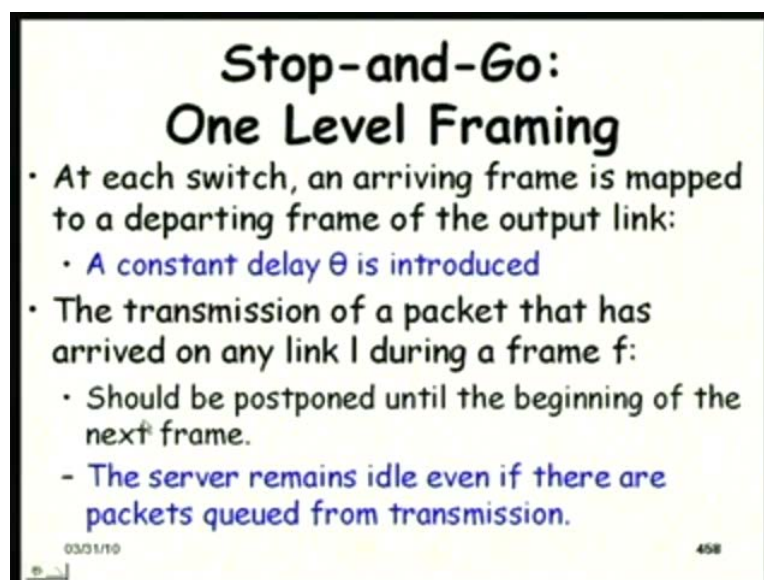
level framing. In a one level framing, we divide the time axis into frames. Each frame is of size T , T is also called as the average aging interval, also called as the frame; and T is a parameter for the one level framing.

What do you mean by aging interval?

Aging interval is the time they would have to wait, they would have to, the packet would have to age there before it can get transmitted. Now, each connection is allocated a bandwidth in proportion to a certain fraction of the frame time.

So, there are different connections for a link. Now, the link has certain bandwidth, which will be distributed among all these different flows or connections. And if you think of it, then the frame time T over frame time T , it has to be transmitted, right? Once it comes, it is it ages for t , right? It waits for T and then it gets transmitted in over the next frame. So, for the next frame, there will be a certain fraction of the time which has been alerted to it; it will be transmitted during that time, right?

(Refer Slide Time: 24:56)



**Stop-and-Go:
One Level Framing**

- At each switch, an arriving frame is mapped to a departing frame of the output link:
 - A constant delay θ is introduced
- The transmission of a packet that has arrived on any link l during a frame f :
 - Should be postponed until the beginning of the next frame.
 - The server remains idle even if there are packets queued from transmission.

03/01/10 458

So, this gets delayed by a θ which depends on the T actually, which is less than T actually; the θ is less than T . Now, a packet that arrives for a link l , during a frame f , sometime during this frame the packet arrived, it will be delayed, until the next frame. Now, even if there are packets that have got delayed and nothing as getting transmitted on the link, it just waits for the next frame to start.

So, packet that comes in one frame can get transmitted only in the next frame. And till that time, it idles. And then once the new frame starts, then the portion of the time that has been allotted for that connection, it has to wait till that time and then get transmitted.

(Refer Slide Time: 26:00)

Stop-and-Go

- Framing strategy introduces:
 - The delay of any packet at a single switch is bounded by two frame times.
 - To reduce the delay, a smaller T is desired.
- Assuming a fixed packet size P :
 - Minimum granularity of bandwidth allocation is P/T .

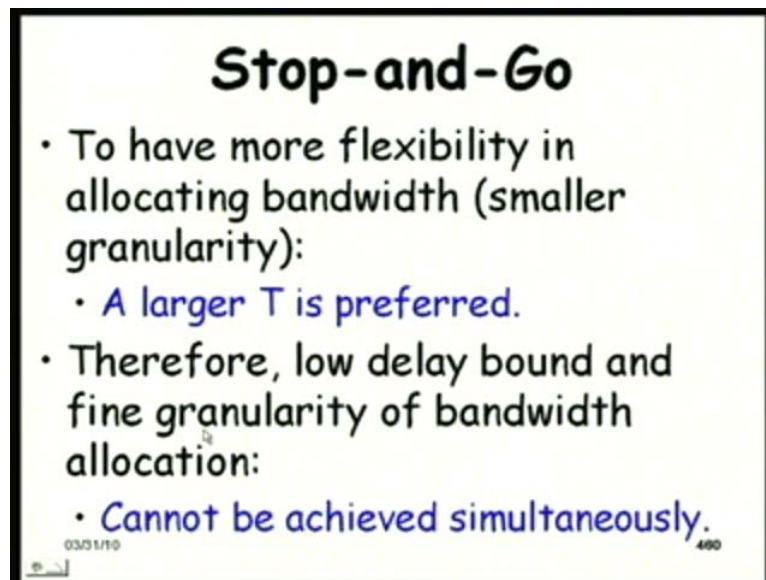
03/01/10 459

So, the one level framing strategy introduces a delay, which is bound maximum by two frames, two frame times; because, in the worst case, it can arrive at the beginning of a frame and possibly the time it could be transmitted or it was allotted to which was the end of the next frame.

The worst case, the packet, the delay of a packet at a switch is bound by two T . But see, if it is large, then the delay can be unacceptable for a connection. So, a smaller T is desired. Now, assuming that a packet as a fixed size P , then the minimum granularity of the bandwidth allocation is P by T , right?

So, see t is the time that is available to us to transmit all packets. So, one packet will take, how much time to be transmitted? P by T .

(Refer Slide Time: 26:00)



Stop-and-Go

- To have more flexibility in allocating bandwidth (smaller granularity):
 - A larger T is preferred.
- Therefore, low delay bound and fine granularity of bandwidth allocation:
 - Cannot be achieved simultaneously.

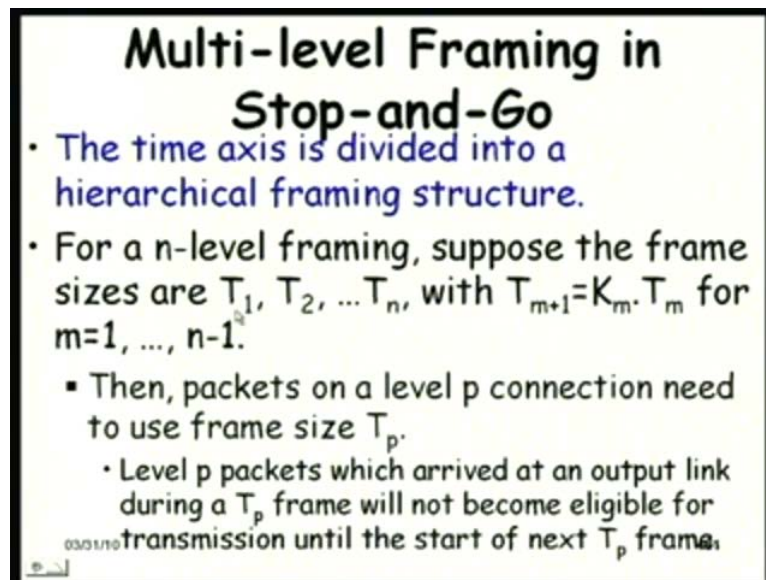
03/01/10 460

So, to have more flexibility in allocating bandwidth, a larger T is preferred. But, we saw that if the minimum granularity is P by T , so inversely proportional to T , right? On the other hand, if we have flexibility in allocating bandwidth, that is granularity, a larger t will be preferred, because if we can transmit only, let us say few packets in each frame, then we are losing flexibility, isn't it? How many packets for each frame we have can allocate? We cannot allocate one point five, is it not? We will have to allocate in integer number of packets. So, there is a conflict here.

On one hand, we want t to be increased and another hand, we want t to be reduced. So, that is what it says, a low delay bound and a fine granularity of bandwidth allocation cannot be achieved simultaneously in the stop-and-go scheme.

So, either you should agree to a larger delay on the packets, the end to end, larger end to end delay in the packets, or you have to compromise with wastage of bandwidth; because the granularity is low and bandwidth will get wasted, because you cannot give a fraction of a bandwidth to a connection.

(Refer Slide Time: 29:04)



Multi-level Framing in Stop-and-Go

- The time axis is divided into a hierarchical framing structure.
- For a n-level framing, suppose the frame sizes are T_1, T_2, \dots, T_n , with $T_{m+1} = K_m \cdot T_m$ for $m=1, \dots, n-1$.
 - Then, packets on a level p connection need to use frame size T_p .
 - Level p packets which arrived at an output link during a T_p frame will not become eligible for transmission until the start of next T_p frame.

03/01/10

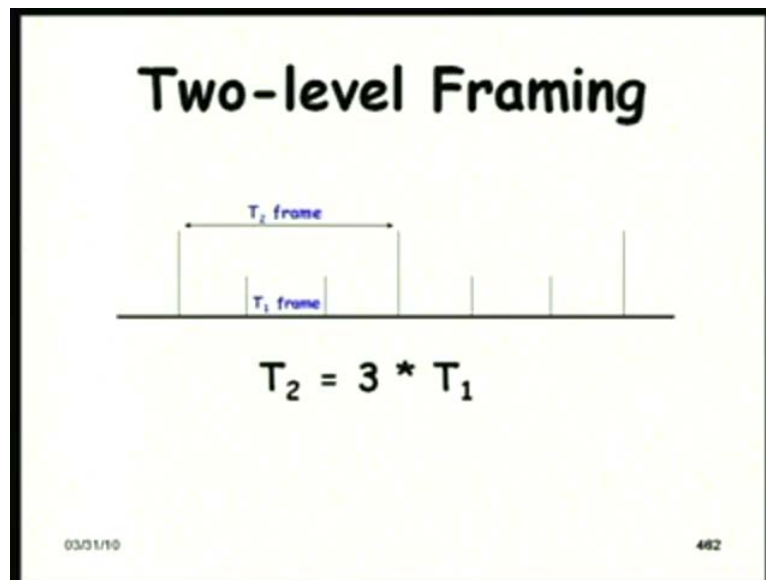
The multilevel framing is actually an extension of the single level framing. The time axis is divided into hierarchical frame structure T_1, T_2, T_n etcetera; and T_{m+1} will be some K_m into T_m . So, these are integral multiples of each other, each of the frame T_n may be 2 into T_{n-1} or 3 into T_{n-1} and so on.

Now, a packet on a level p frame, the T_p use a, they use the frame size T_p , right? So, if we assign a packet to a frame size to level p, then they will always use the frame size T_p . There may be some other packets which will use frame size t, one, some may use T_n , but some connection might use T_p .

So, the level p packets which arrive at an output link will become eligible in the next frame, next T_p frame, same thing. So, the for the T_1 frame, the packets which are using the T_1 frame, once they arrive they become eligible in the next T_1 frame. And those in the T_n frame, they will become eligible in the next T_n frame.

So, the one that are assigned on the T_n frame will undergo higher delay, isn't it? The packets that are on T_n frame or the connection that has been assigned the T_n frame will undergo higher delay than the one that have been the connection that has been assigned the T_1 frame. But if the delay is permissible, then you can assign to T_n and that will increase the network utilization, isn't it?

(Refer Slide Time: 31:06)



So, there are this just says a two level framing, where we have two frame sizes T_1 and T_2 and T_2 contains 3 T_1 . So, some packets which are assigned to some connections that are assigned the T_1 frame will undergo a maximum delay of 2 into T_1 at this switch. The one that have been assign to the T_2 frame will undergo T_2 into T_2 delay in the maximum; say, depending on the characteristic of the connections, we can assign them to the different frames.

(Refer Slide Time: 31:48)

Rate-Controlled Static Priority

- Drawbacks of previous service disciplines:
 - Sorted priority service disciplines such as WFQ are complex and difficult to implement.

03/01/10 463

Now, one thing is that the ones we so far discussed, both the work conserving and non-work conserving, is that they are rate-controlled. What about the one that we discussed just now? The single framing and multilevel framing, is that a work conserving or a non-work conserving discipline? Let me just ask at least this question, simplest question. So, we will have to think that what is the definition, we gave for a work conserving discipline versus a non-work conserving discipline and whether it satisfies that work.

I mean it satisfies which one.

Non-work

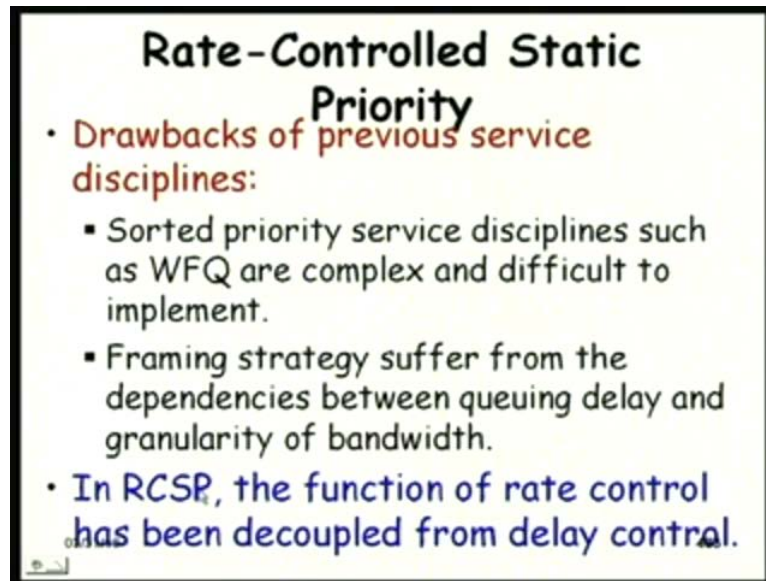
So he says non-work conserving. What about others? No answer, is it? Let us hear his answer. Why you are saying it non-work conserving?

(()).

Exactly. So, the definition of a work conserving and non-work conserving is that, in a work conserving the packet is not made to wait unnecessarily, the link is available. In a non-work conserving, the packet has certain eligibility time and till that time, even if the link is available do not be transmitted; and we had seen that a non-work conserving discipline bounds the jitter.

So, here in the framing strategy, it said that a packet as it arrives becomes eligible only in the next frame. So, it has to wait until the frame gets over, right? Even if the link is idle, it will have to wait and therefore, it is a non-work conserving discipline.

(Refer Slide Time: 33:51)



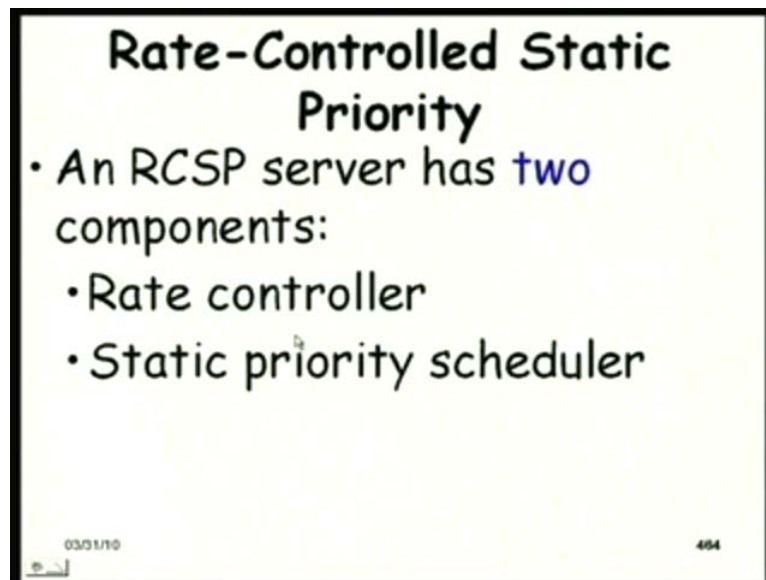
Rate-Controlled Static Priority

- Drawbacks of previous service disciplines:
 - Sorted priority service disciplines such as WFQ are complex and difficult to implement.
 - Framing strategy suffer from the dependencies between queuing delay and granularity of bandwidth.
- In RCSP, the function of rate control has been decoupled from delay control.

Now, the drawbacks of the disciplines, we discussed. All of them are rate-controlled disciplines. So, one is that they are difficult to implement. This framing strategy has the problem that there is a compromise between the queuing delay and the granularity of the bandwidth can be allocated to the connection. Because again if you think of it, it has its origin in some fluid flow, right where you can even have a small granularity flow occurring, but here packet can be one kilobyte or several kilobytes of data it need to be transmitted.

Now, we will discuss about RCSP. Here, we will see that the rate-controlled has been decoupled from the delay control. So, in the disciplines that we discussed so far, all are based on rate control but and we saw that each of them have some difficulty. Now, let us look at one algorithm called as RCSP, where the rate control and the delay control have been decoupled.

(Refer Slide Time: 35:12)



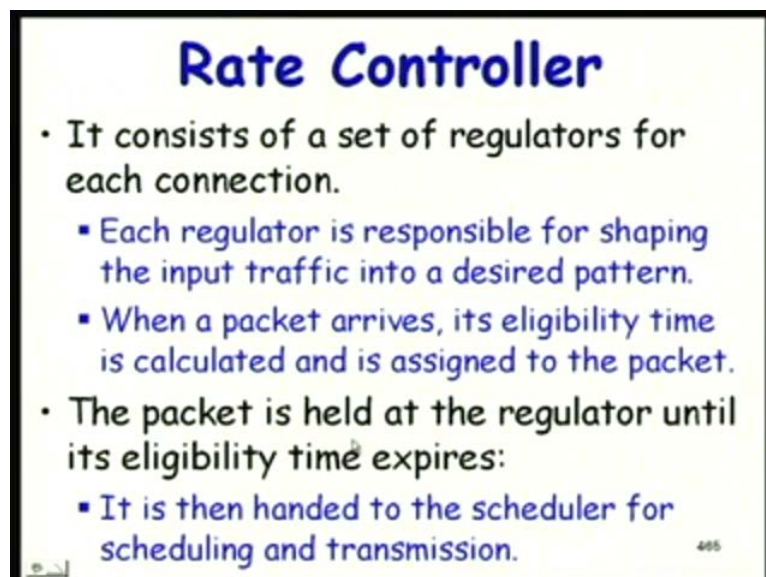
Rate-Controlled Static Priority

- An RCSP server has **two** components:
 - Rate controller
 - Static priority scheduler

03/01/10 464

As the name says, there are two components here. One is the rate controller and the other is the static priority scheduler.

(Refer Slide Time: 35:29)



Rate Controller

- It consists of a set of regulators for each connection.
 - Each regulator is responsible for shaping the input traffic into a desired pattern.
 - When a packet arrives, its eligibility time is calculated and is assigned to the packet.
- The packet is held at the regulator until its eligibility time expires:
 - It is then handed to the scheduler for scheduling and transmission.

465

The rate controller consists of a set of regulators for each connection. Now, the regulator is responsible for shaping the input traffic to the desired pattern; and when a packet arrives, its eligibility time is calculated and the eligibility time is written on the packet.

So, the rate controller is implemented through a set of regulators. The regulator will examine the packet as it arrives and it will mark it with an eligibility time. And the regulator will hold it

until the eligibility time expires. Again, see just check here that all traffic shaping are based on delaying the packet in some queue.

Now, once the eligibility time has expired, the regulator hands it to the scheduler for transmission.

(())

Not really, see there every packet just arrives and depending on the frame at which arrives, it just undergo some delay until it term comes.

(())

No, time is assigned. See the frames are fixed; see, in the multilevel framing, the frames are fixed.

Age interval you are assigning them.

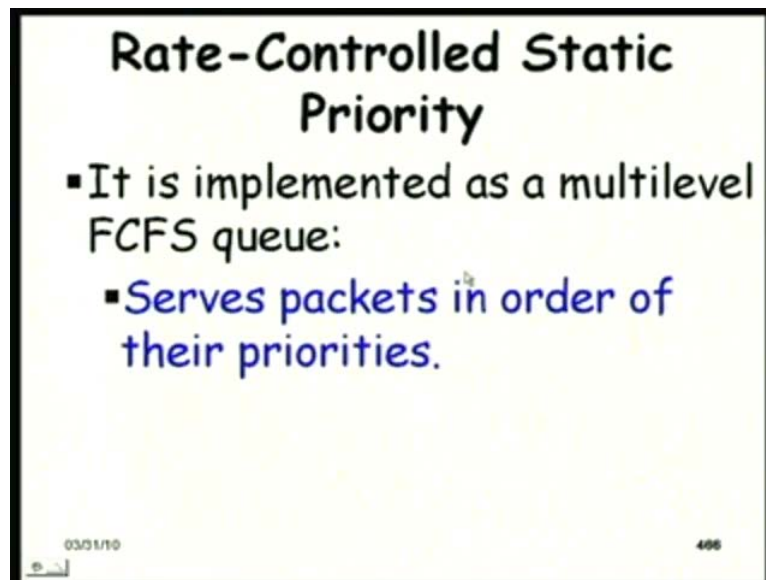
The frames, those are the using interval or the frames. And depending on wherever it appears, it just gets transmitted on the next frame; but there is no rate controller as it is, which will based on case to case delayed, right? So, here it will examine; depending on the condition for that connection, it will delay to varying extents.

So, there it just the frames where fixed and depending on where it arrived in the frame, it can get transmitted only in the next frame automatically. There is role of a rate controller there, it happens automatically.

Automatically it will be (()).

Automatically and we can think of that it is a just a rate bit transmission where a packet becomes eligible in next frame and rate or a bandwidth is assigned to it and it gets transmitted during that time, right?

(Refer Slide Time: 35:12)



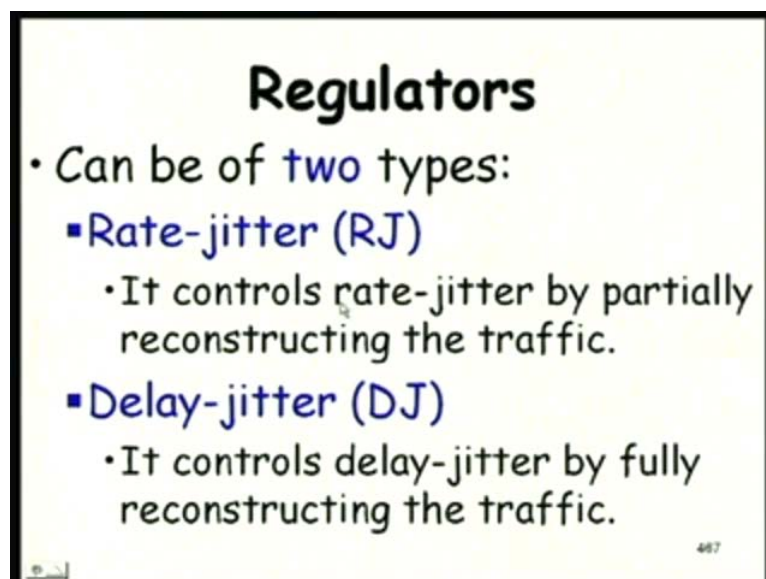
Rate-Controlled Static Priority

- It is implemented as a multilevel FCFS queue:
 - Serves packets in order of their priorities.

03/01/10 466

So, let see here. So, the scheduling algorithm here is a multilevel FCFS queue and serves packets in order of their priorities. So, as far as the scheduler is concerned nothing too much to discuss here.

(Refer Slide Time: 38:19)



Regulators

- Can be of two types:
 - Rate-jitter (RJ)
 - It controls rate-jitter by partially reconstructing the traffic.
 - Delay-jitter (DJ)
 - It controls delay-jitter by fully reconstructing the traffic.

467

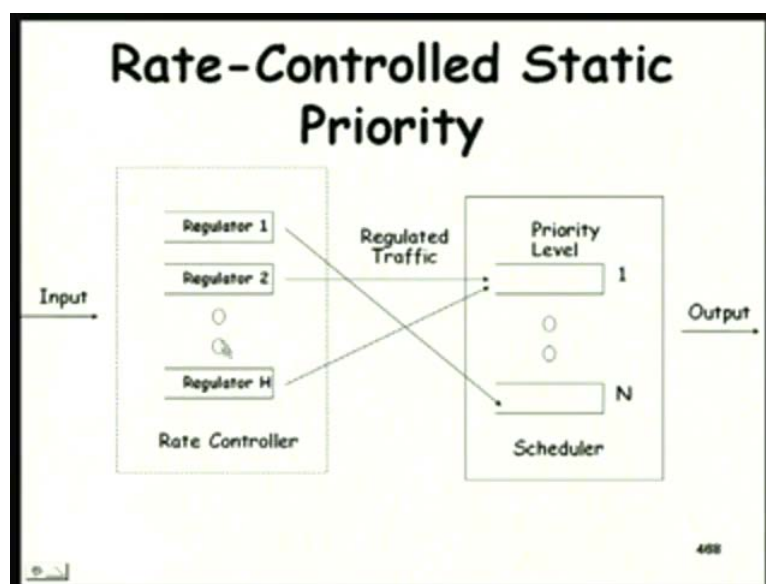
But the regulator can be two types: the rate jitter or a delay jitter type of regulator. A rate jitter type of regulator, it will examine the connection that packets that are coming from the connection and it will try to reconstruct the traffic partially. Whereas a delay jitter, it will control the delay jitter by fully reconstructing the traffic. So, we will not really go into the

details of how it does the rate jitter, how much does it delay, what do you mean by partial reconstructing etcetera. We will not go into this.

But here, what we need to understand here is that, there is a separate component called as a regulator which will look at the traffic. See, earlier in a framing it never had any analysis of the traffic; just delays them by that amount. Depending on where it arrived, it automatically gets delay till the next frame.

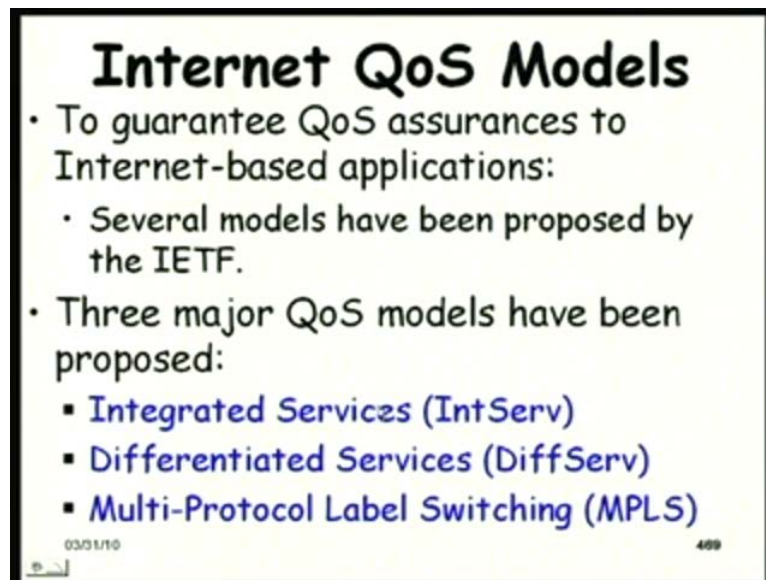
But here, it keeps track of the traffic; and depending on the current traffic rate, it will either partially reconstruct the traffic or fully reconstruct traffic, depending on what type of regulator we are trying to use. So, a more, this is the more sophisticated form of service discipline as compared to simpler service discipline, the rate based ones.

(Refer Slide Time: 40:03)



So, there are different regulators which keep track of the traffic; and then they transmit it to the scheduler after they have reset the traffic.

(Refer Slide Time: 40:14)



Internet QoS Models

- To guarantee QoS assurances to Internet-based applications:
 - Several models have been proposed by the IETF.
- Three major QoS models have been proposed:
 - Integrated Services (IntServ)
 - Differentiated Services (DiffServ)
 - Multi-Protocol Label Switching (MPLS)

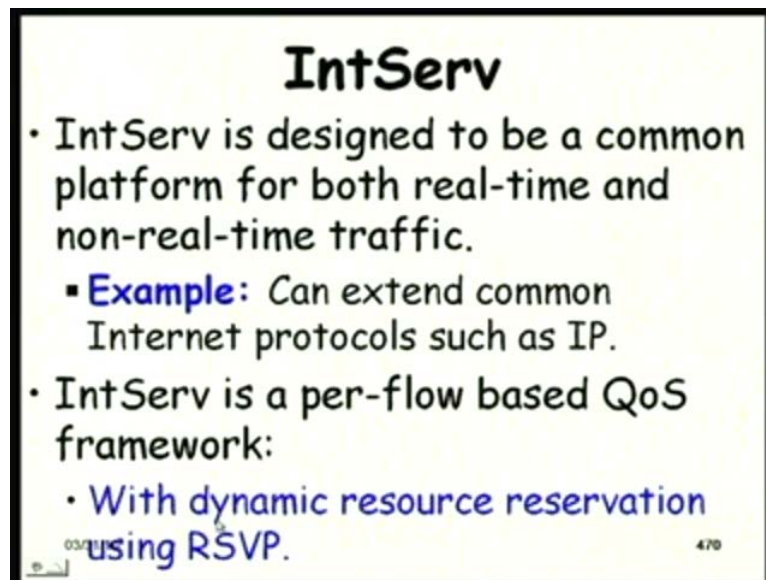
03/01/10 409

Now let us look at the quality of service models that have become popular in the internet. So far, we just looked at the components of a quality of service framer that can be used to provide quality of service guarantees. Now let see, in the Internet there are some standards that are emerging. The standard is proposed by IETF, the Internet Engineering Task Force, if I am not wrong, this is the full form - the Internet Engineering Task Force. This is the standards body for the Internet.

And this has proposed three major quality of service models. One is called as the integrated services or the intern, the differentiated services or the diffuser and the multi-protocol label switching or the MPLS.

So, these have, because this have been proposed by a standard body and are applicable to the Internet, they have drawn lot of attention. Even in a literature that you read in any other book, you might come across these terms and even on a news paper, you can come across these terms.

(Refer Slide Time: 41:46)



IntServ

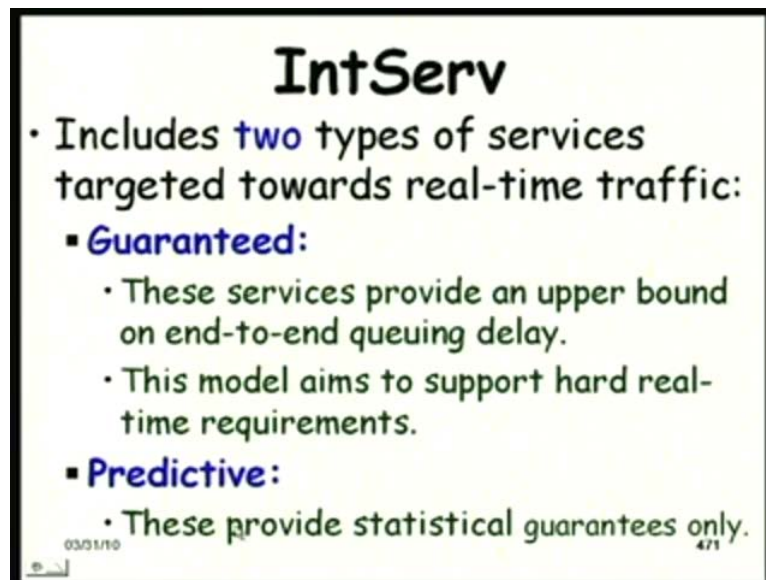
- IntServ is designed to be a common platform for both real-time and non-real-time traffic.
 - **Example:** Can extend common Internet protocols such as IP.
- IntServ is a per-flow based QoS framework:
 - **With dynamic resource reservation using RSVP.**

03/11/11 470

Let us look at the basic idea behind IntServ. IntServ is a protocol that has been designed to support both real-time and non-real-time traffic in the internet. Naturally it extends the internet protocol such as I p and it is a per-flow based quality of service framework. What it means is that for every connection, it keeps track of several information. It is a per-flow based quality of service network. So, that is for every connection, it stores several information all through the internet, all through the routers that are concerned.

And it uses a the RSVP protocol for dynamic resource reservation. We know what is RSVP. And for the real-time traffic, see the non-real-time traffic they operate has the internet is operating right now.

(Refer Slide Time: 41:46)



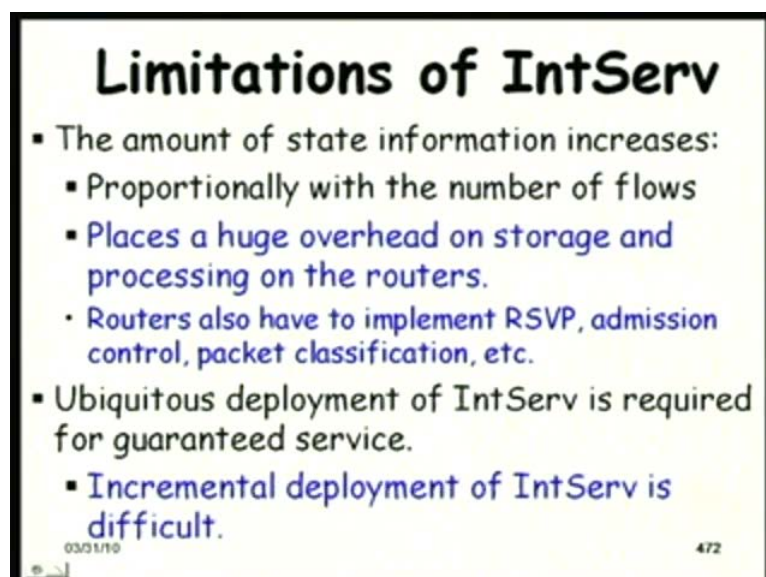
IntServ

- Includes **two** types of services targeted towards real-time traffic:
 - **Guaranteed:**
 - These services provide an upper bound on end-to-end queuing delay.
 - This model aims to support hard real-time requirements.
 - **Predictive:**
 - These provide statistical guarantees only.

03/01/10 471

But for the real-time traffic, there are two service disciplines: one is called as a guaranteed service discipline. Here, the service discipline would provide an upper bound and the end to end queuing delay. And these are the guaranteed service discipline would be used for real-time communication, hard real-time communication; whereas there is another service discipline which provides statistical guarantee only, that is what will be the average delay and so on, maximum delay, average delay etcetera.

(Refer Slide Time: 43:47)



Limitations of IntServ

- The amount of state information increases:
 - Proportionally with the number of flows
 - Places a huge overhead on storage and processing on the routers.
 - Routers also have to implement RSVP, admission control, packet classification, etc.
- Ubiquitous deployment of IntServ is required for guaranteed service.
 - Incremental deployment of IntServ is difficult.

03/01/10 472

See, I am not talking of the IntServ has been implemented in the Internet that you are using right now, but it is a standard which can be used in future. We will see. I mean, first let us look at its pros and cons and then see whether it will be used.

(())

Right now, right now the internet that we are using, it does not support hard real-time traffic. But if we have IntServ implemented on the internet, we will see the complexity of implementation. And if we, if IntServ is implemented, then it would be possible to do hard real-time applications on the internet.

Sir people are talking about remote medical (()).

Exactly, remote medical monitoring of a patient on the internet or let say you have some embedded devices which are giving remote commands controlling that, it would become possible; but let see the complexity of implementing the IntServ.

So, one complexity is that the amount of state information increases if you have large number of real-time connections; because we had seen that it is a per-flow connection and the information has to be maintained every router it uses the RSVP; and RSVP we know that the state information need to be maintained, isn't it? The reservation information and the path information need to be maintained at every router for every connection.

So, this is one complexity that the information that need to be maintained at the router can become enormous depending on a number of real-time connections that need to be supported, which can place a huge overhead on the storage and processing on the routers. And then the present routers that are there throughout, we will have to implement RSVP, admission control, packet classification etcetera.

So, it would be difficult to implement it, isn't it? Because not only that, there are large number of information that need to be stored depending on how many real-time connections have to be supported, but also we will have to change all the routers now. We have to, they have to implement RSVP, admission control, packet classification etcetera.

And universal deployment or ubiquitous deployment of IntServ would be required for a guaranteed service all through the millions or billions of routers are there; they have to implement all these things - RSVP, admission control, packet classification.

And then the amount of information they would have to store is huge and certainly a nontrivial task to; in any time sooner, you will get all these things working on the internet is looks difficult and not only that, incremental deployment of intserv is also difficult. Either you go for it fully or you do not go for it, right?

So, the intserv has its limitations. And we have understood the limitations, isn't it? Why it would be difficult to implement intern in the current internet and get it to support real-time communications.

(Refer Slide Time: 47:16)

DiffServ

- DiffServ overcomes the limitations of IntServ
 - It is simpler and is more scalable.
- It redefines the Type of Service (TOS) field in the header of IPv4 or IPv6 traffic class byte:
 - As a Differentiated Service (DS) field.
 - The first six bits are called DS Code Point (DSCP).

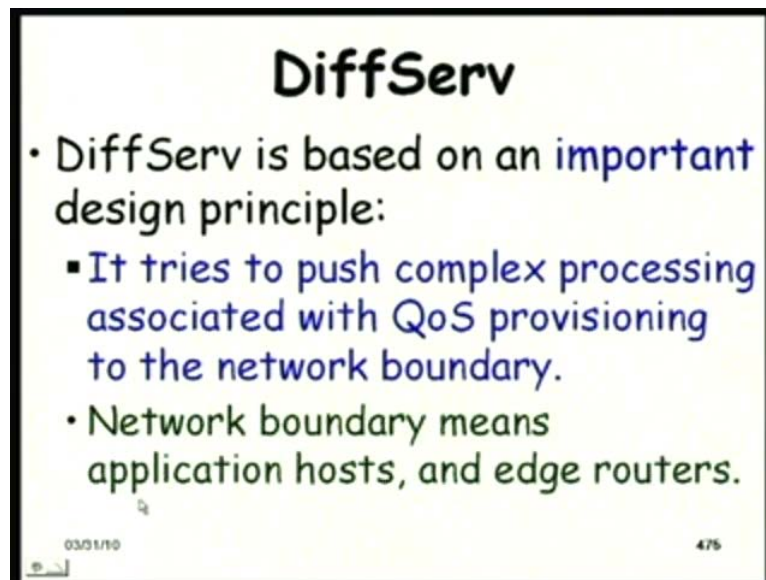
4

03/01/10 4/3

Now, let us look at the Diffserv. The diffserv does overcome some of the limitations of the intserv. One is that it is simpler less information needs to be maintained, more scalable.

So, the way it would work is that it would redefine the type of service field in the header of the IPV 4 or IP 6 packet. In the type of service, it will have a differentiated service field and first six bits in that, as it has been proposed by the IETF, the first six bit are called as the d s code point.

(Refer Slide Time: 47:16)



DiffServ

- DiffServ is based on an important design principle:
 - It tries to push complex processing associated with QoS provisioning to the network boundary.
- Network boundary means application hosts, and edge routers.

475

And there some information will be stored on the packets. So, the router does not have to maintain any information; very important, I mean the intserv and diffserv. The scalability of the diffserv arrives, because it just stores some information on header of the packets, rather than storing large amount of information on the routers or the switches.

So, here the required information is carried out, the buffer managements scheduling, all these are carried out in the packet, the information is stored in the packet and there is no special protocols and storage information at the routers.

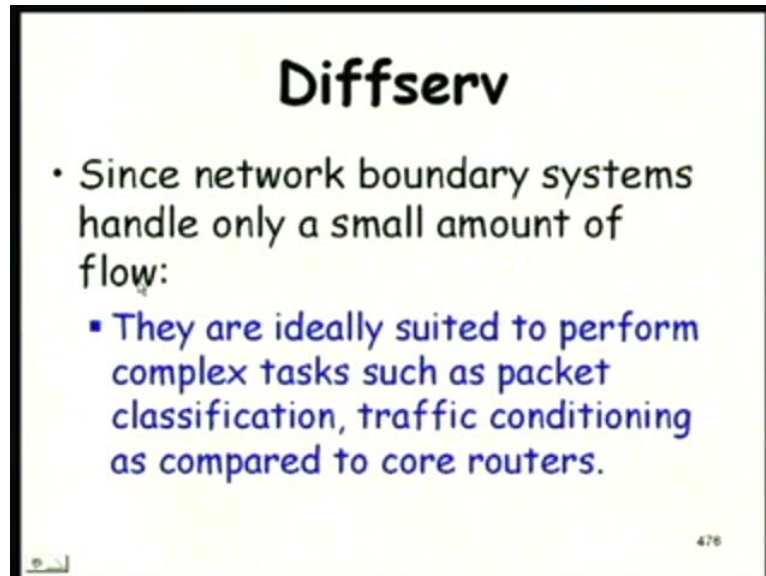
And the amount of information that only the switches have to store now is depends on how many service classes we use to support. For each service classes, the switch or router will simply have the information is to have to handle that packet. A packet is based on the class ten or class five, how does the switch or router handle that.

So, just see here, that rather than storing information compared to the number of application flows which can be hundred or thousand, here it just need to be store information of how to handle that particular packet belonging to the class, which can be six, ten or twenty maximum.

And another important principle; see, one we saw is that the information is not stored on the switches or routers, it is distributed on the packets; the other important principle behind the

design of the diffserv is that the kind of processing that is required is only done at the network boundaries, only the edge routers are responsible here.

(Refer Slide Time: 50:44)



The core routers, a packet has it gets transmitted might undergo transmission through several large number of core routers. So, if every packet from every connection needs to be processed by the core router, it has to examine the kind of class that it belongs and do differently for it; that will be a very large overhead.

But here it is trying to restrict the processing to only the network boundary; only at the host and the edge routers. And this is scalable because the edge routers handle only small amounts of flow. The core routers they carry flow from all possible applications that can pass through it.

But edge router is only for those hosts that are connected to it. These are finite in number. And therefore, the edge routers can easily perform tasks like packet classification, traffic conditioning as compared to core routers.

(Refer Slide Time: 51:22)

DiffServ

- Provides **two** service models:
 - **Premium service**
 - It is a guaranteed peak rate service
 - Optimized for regular traffic patterns
 - Incurs almost no queuing delays.
 - **Example:** virtual leased line.
 - **Assured service**
 - It provides statistical guarantees to applications.

03/01/10 477

So, there are two service models in the diffserv, just like in the intern we had seen two service model, the guaranteed and the best effort service. I think the statistical guarantee and the guaranteed service here. There are two service models again. One is the premium service, which is a guaranteed service optimized for regular or constant bit rate traffics; and this traffic which are in the premium class, they undergo almost no queuing delay at the router.

So, it becomes almost like a virtual leased line. The internet behaves as if it is a virtual leased line and the assured service class; it provides statistical guarantees to application, just like intserv.

(Refer Slide Time: 52:15)



**Soft Real-Time
Communication in a LAN**

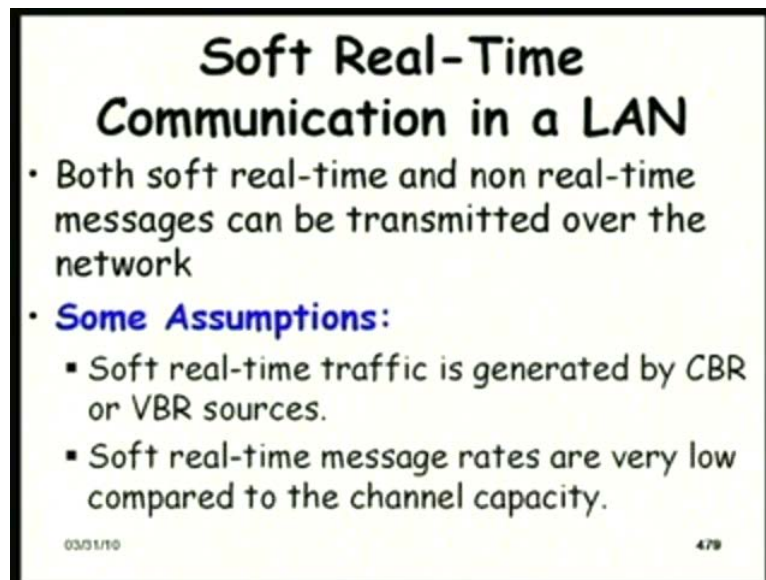
- Characteristics of Soft Real-Time Communication Networks:
 - No absolute QoS guarantees are provided to the applications.
 - Only ensures prioritized treatment for real-time messages
 - Helps to keep the message-deadline miss ratio to a minimum.
 - Provides statistical guarantees on delay bounds.

03/01/10 478

So, we have just looked at the summary of intserv and diffserv. Those who are interested can look details, just discussed only the very essential features. Now, let us see how soft real-time communication can be provided in a LAN environment. So far, we have been looking at the hard real-time applications and then we had looked at hard real-time applications in the internet.

Now, let us look at the soft real-time applications. Here, no absolute quality of service guarantees are needed by the applications and only we need prioritized treatment based on what class they belong to it, what class they belong to. We need to keep the message deadline miss ratio to a minimum for this kind of classes.

(Refer Slide Time: 52:15)



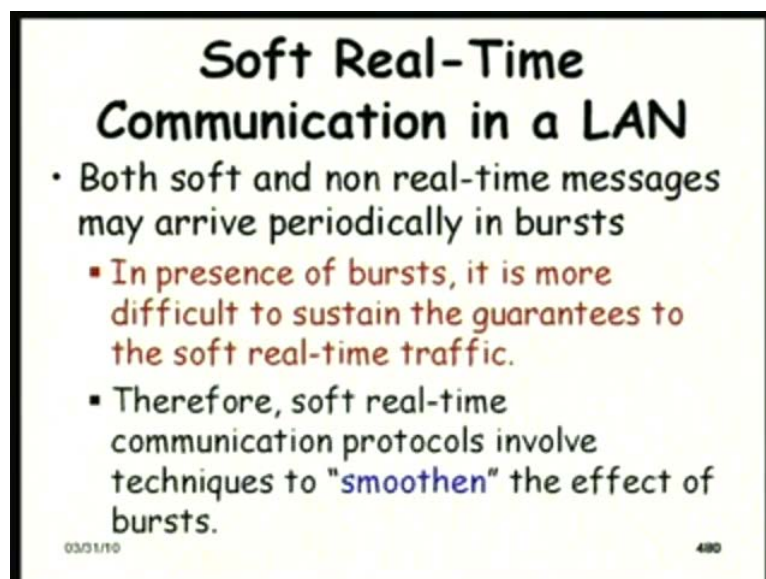
Soft Real-Time Communication in a LAN

- Both soft real-time and non real-time messages can be transmitted over the network
- **Some Assumptions:**
 - Soft real-time traffic is generated by CBR or VBR sources.
 - Soft real-time message rates are very low compared to the channel capacity.

03/01/10 479

So, some of the assumptions that we will make here is that the soft real-time traffic or either CBR or VBR and the soft real-time message rates are very low compared to the channel capacity.

(Refer Slide Time: 52:15)



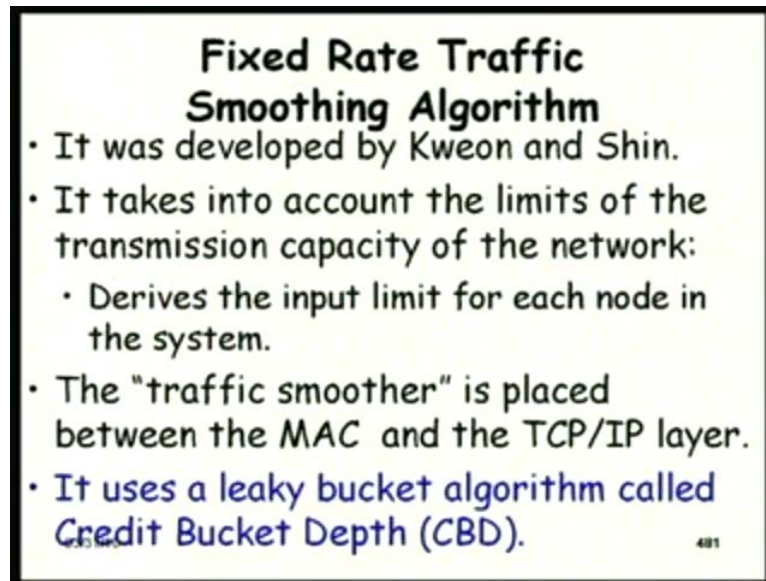
Soft Real-Time Communication in a LAN

- Both soft and non real-time messages may arrive periodically in bursts
 - In presence of bursts, it is more difficult to sustain the guarantees to the soft real-time traffic.
 - Therefore, soft real-time communication protocols involve techniques to "smoothen" the effect of bursts.

03/01/10 480

And these can arrive periodically or in bursts and in the presence of burst, it becomes difficult to sustain the guarantees. So, there can be misses for soft real-time traffic. And the way it works is by smoothing the effect of the burst, just like we had seen the leaky bucket. Similar things happen here.

(Refer Slide Time: 54:07)



Fixed Rate Traffic Smoothing Algorithm

- It was developed by Kweon and Shin.
- It takes into account the limits of the transmission capacity of the network:
 - Derives the input limit for each node in the system.
- The "traffic smoother" is placed between the MAC and the TCP/IP layer.
- It uses a leaky bucket algorithm called Credit Bucket Depth (CBD).

481

So, here one of the popular technique is by seen at all. It takes into account the limits of the transmission capacity of the network and it uses a traffics smoother in the MAC protocol of a LAN; and the traffic smoother actually uses a leaky bucket algorithm, a variation of the leaky bucket algorithm called as a credit bucket depth; small variation, we will just look at it. But since running out of time and we need little bit of time to discuss this, we will just stop here. We will continue in the next class. Thank you.