# Quality of Service

MSc – II Semester

Department of Computer Science

# Flow - Characteristics

There are mainly four types of Flow-characteristics:

- Reliability/Loss
- Delay
- Jitter
- Bandwidth

# Flow - Characteristics

Delay: time taken by a packet to go from source to destination

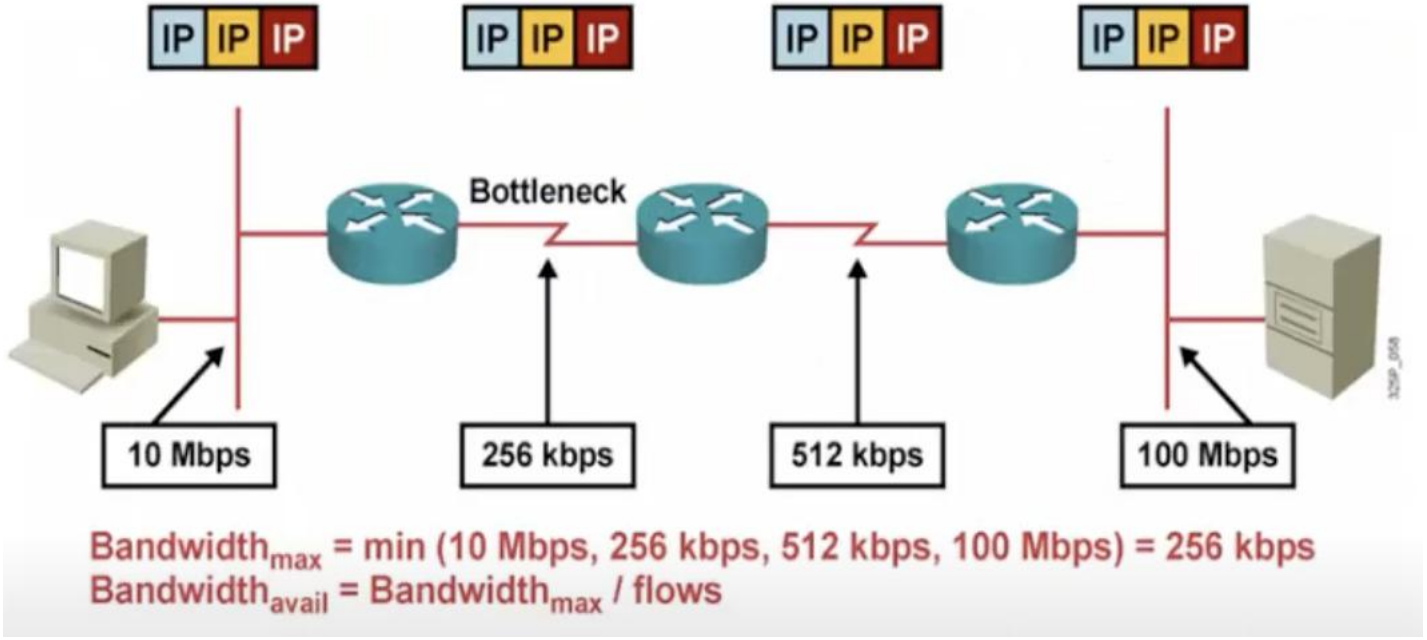
Three components :

- Transmission delay
- Propagation delay
- Queuing delay
- Different applications can tolerate delay in different degrees.

# Bandwidth:

- Amount of data that can be transmitted over a link within a fixed amount of time.
- “ *When a drain chronically runs slow even though it is not clogged, it is time to get a bigger pipe”* - QoS vs Bandwidth by Time Greene
- Some applications in the network are bandwidth-hungry e.g. video applications

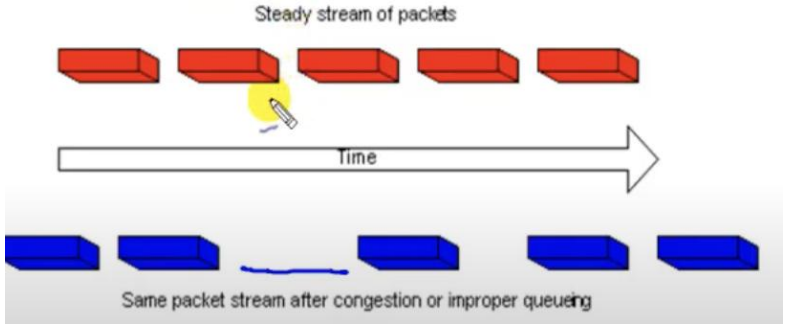
# Measuring Available Bandwidth




- Maximum available bandwidth equals the bandwidth of the slowest link.
- Multiple flows are competing for the same bandwidth, resulting in much less bandwidth being available for one single application
- A lack of bandwidth can have performance impacts on network application

# Jitter

- Variation in end-to-end delay
- Packets from the source will reach the destination with different delays.
- Jitter is generally caused by congestion in the IP network.




- Email, file sharing, web access – not sensitive to jitter
- Remote login: somewhat sensitive, since updates on the screen will appear in little burst
- Video and audio - extremely sensitive

# Packet Loss/ Reliability

- A relative measure of the number of packets (or segments or bits) that were not received compared to the total number of packets transmitted
- Loss is a function of availability – if the network is available (capacity is more than the demand) then the loss would generally be zero (this assumption is not true for wireless networks)
- Congestion increases data loss from the intermediate n/w devices
- Email, file sharing, web access, and remote login - have more stringent requirements for loss.
- Audio and Video transmissions can tolerate lost packets

# Impact of Packet Loss

- Telephone Call: “ *I can not understand you. Your voice is breaking up*”
- Teleconferencing: "*The picture is jerky. Voice is not synchronized*.”
- Publishing Company: "T*his file is corrupted*”
- Call Centre: “ *Please hold while my screen refreshes*”

# Application QoS

| | Loss | Delay (one-way) | Jitter | Bandwidth |
|---|---|---|---|---|
| Voice | <=1% | <=150 ms | < 30 ms | 21 Kbps – 320 Kbps |
| Interactive Video | <=1% | <=150 ms | <30ms | On demand |
| Streaming Video | <=5% | < Buffer time | On buffer time | On demand |
| Data | | | | Best Effort |

*a streaming video like the YouTube video

Source: https://www.ciscopress.com/articles/article.asp?p=471096&seqNum=6

# Application QoS

| Application | Bandwidth | Delay | Jitter | Loss |
|---|---|---|---|---|
| Email | Low | Low | Low | Medium |
| File sharing | High | Low | Low | Medium |
| Web access | Medium | Medium | Low | Medium |
| Remote login | Low | Medium | Medium | Medium |
| Audio on demand | Low | Low | High | Low |
| Video on demand | High | Low | High | Low |
| Telephony | Low | High | High | Low |
| Videoconferencing | High | High | High | Low |

# Formal definition of QoS

- "*QoS refers to the capability of a network to provide better service to the selected network traffic over various technologies including Asynchronous Transfer Mode(ATM), ethernet, and 802.11 networks*". – given by Cisco
- The first hop may be wireless, the second hop – ethernet and the third hop – optical network (SONET).
- So, need to provide QoS over multiple technologies
- **Goal:**

*to provide priority, including dedicated bandwidth, controlled jitter and latency and improved loss characteristics*.

- And making sure that providing priority for one or more flows does not make other flows fail.
- Fundamentally, QoS enables us to provide better service to certain flows.

# Flow:

- A stream of packets from source to destination is called Flow.
- Source to destination: (multiple definitions of source to destination)
  - Machine to machine
  - Process to process
  - Application to application
  - Socket to socket

# Issues that have to be addressed to ensure QoS

- What applications need from the network (type of QoS)
- How to regulate the traffic that enters the network
- How to reserve resources at routers to guarantee performance
- Whether the router can safely accept more traffic without violating the QoS of the existing traffic

# Why QoS is considered at the network layer

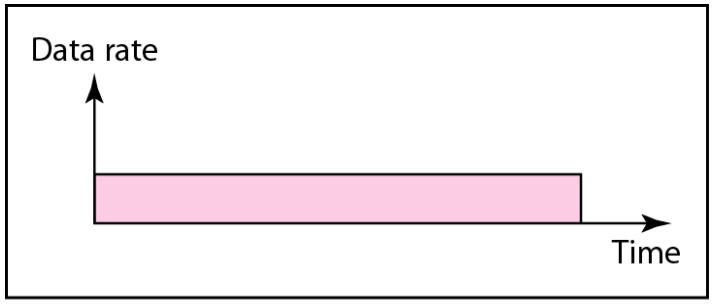
- Maintaining QoS requires both per-hop and end-to-end behavior.
- Initially, we need applications for specify end-to-end delay, end-to-end bandwidth, end-to-end jitter, and end-to-end loss.
- To provide end-to-end performance, reserve resources at every hop of the network; otherwise, end-to-end requirements can not be guaranteed.
- The network layer sits between the transport layer and the Data link layer.
- Get feedback from the transport layer and applies the things to the data link layer

# Application Classes based on QoS

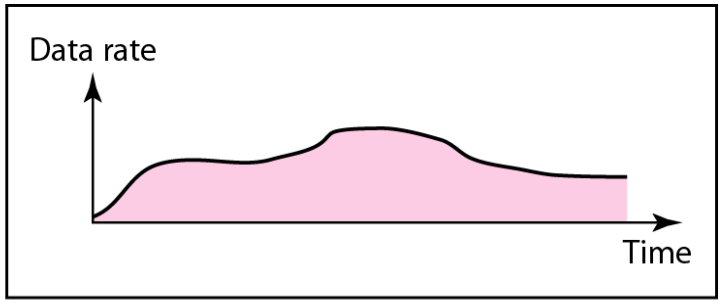
- Constant bit rate (e.g. telephone applications – VoIP)
- Real time variable rate (e.g. videoconferencing)
- Non-real-time variable bit rate (e.g. on demand video streaming)
- Available bit rate or Best effort(e.g. File transfer) – also known as lack of QoS

# Application Classes based on QoS

- Constant Bit Rate: uniform bandwidth and uniform delay
- Variable Bit rate: occurs when the video is compressed, with some frames compressing more than others.
  - E.g. sending a frame with a lot of details may require sending many bits - a shot of white requires less.



a. Constant bit rate



b. Variable bit rate

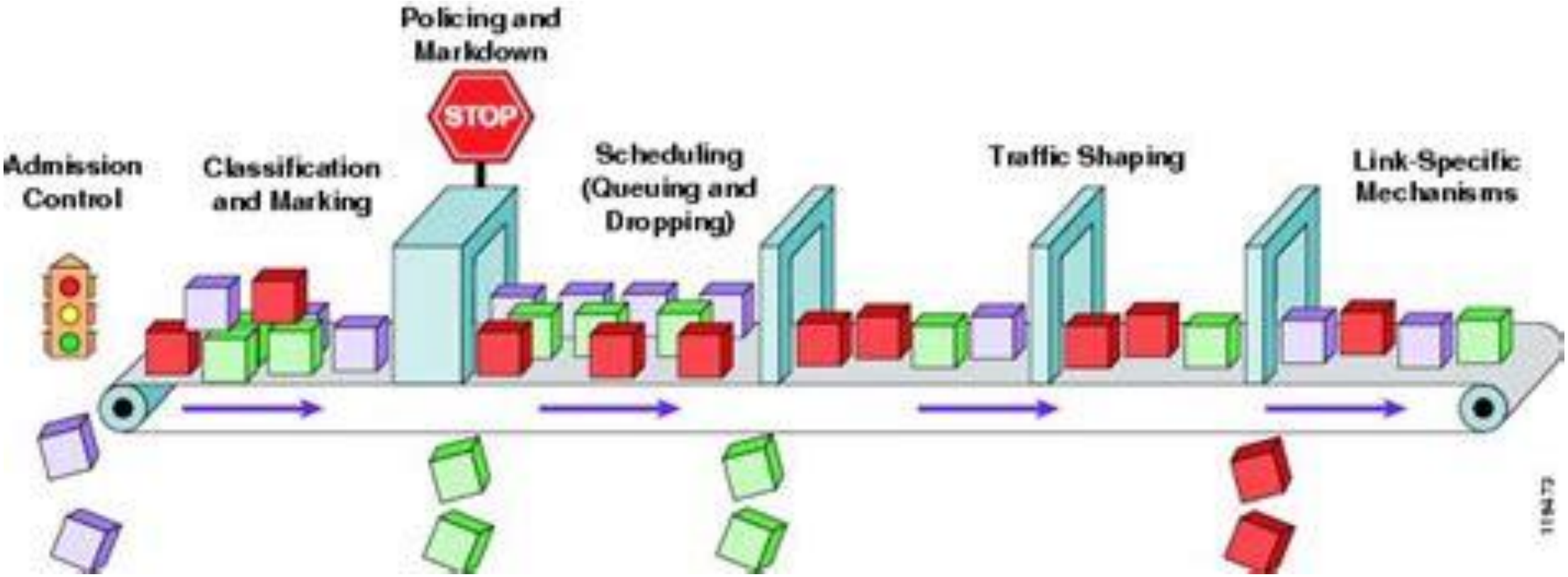
# Application Classes based on QoS

- Real-time – maximum delay by which you need to send the data. If the condition is not met, the transfer is considered to fail. (e.g. Videoconferencing/ live streaming)

- Non-Real time – do not have strict constraints on delay but have some loose bound on the amount of packet loss that can be tolerated
  - for eg. Watching a movie on demand (a few seconds of the video can be buffered before playback starts)

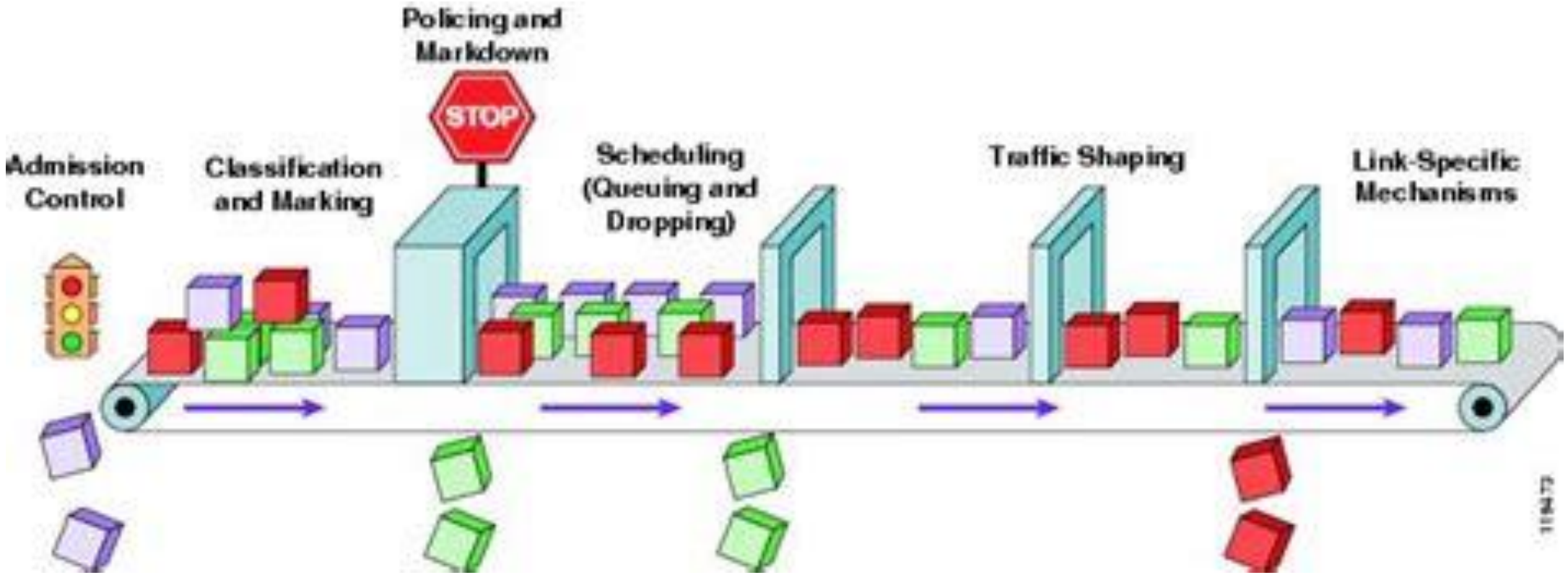
- Best effort: whatever available bandwidth in the network
- No strict requirements on available bit transfer
- For eg: email – not sensitive to delay or jitter and will take whatever bandwidth they get.

# Basic QoS Architecture




Applying certain filters at every stage to get QoS

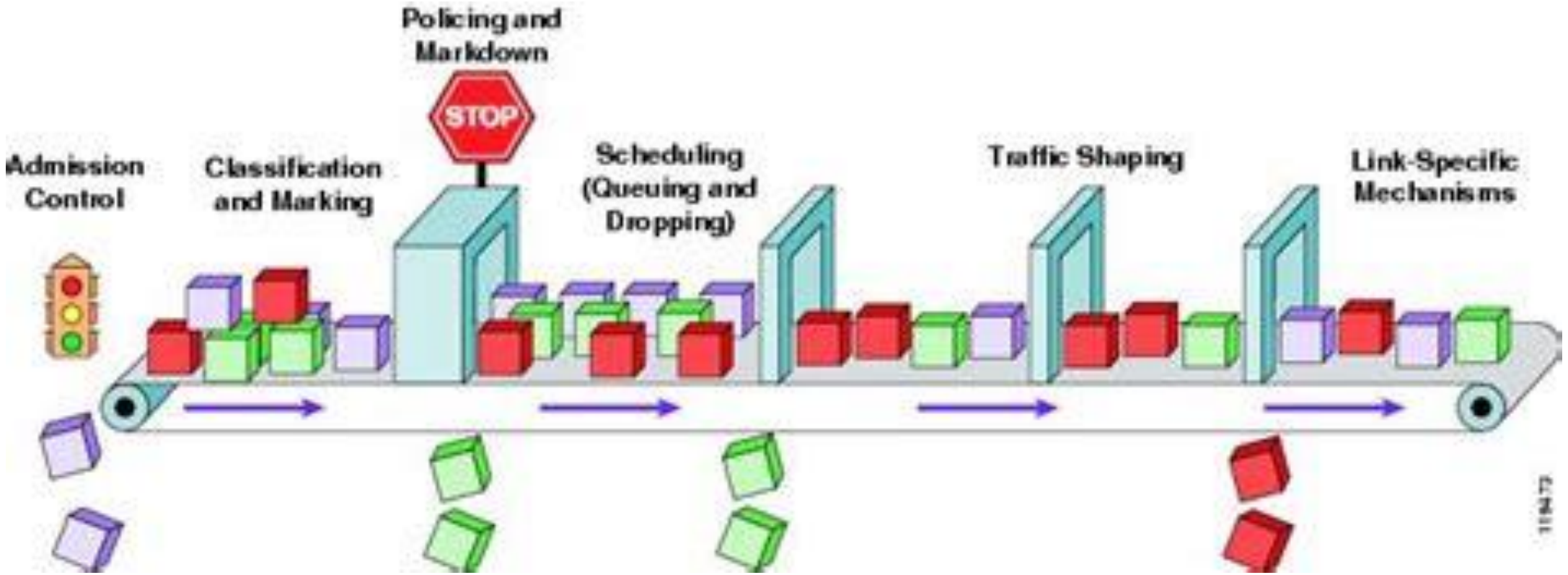
# Basic QoS Architecture




Admission Control:

new flows are entered into the network only if the QoS of the existing flows, including the new flow, can be satisfied.

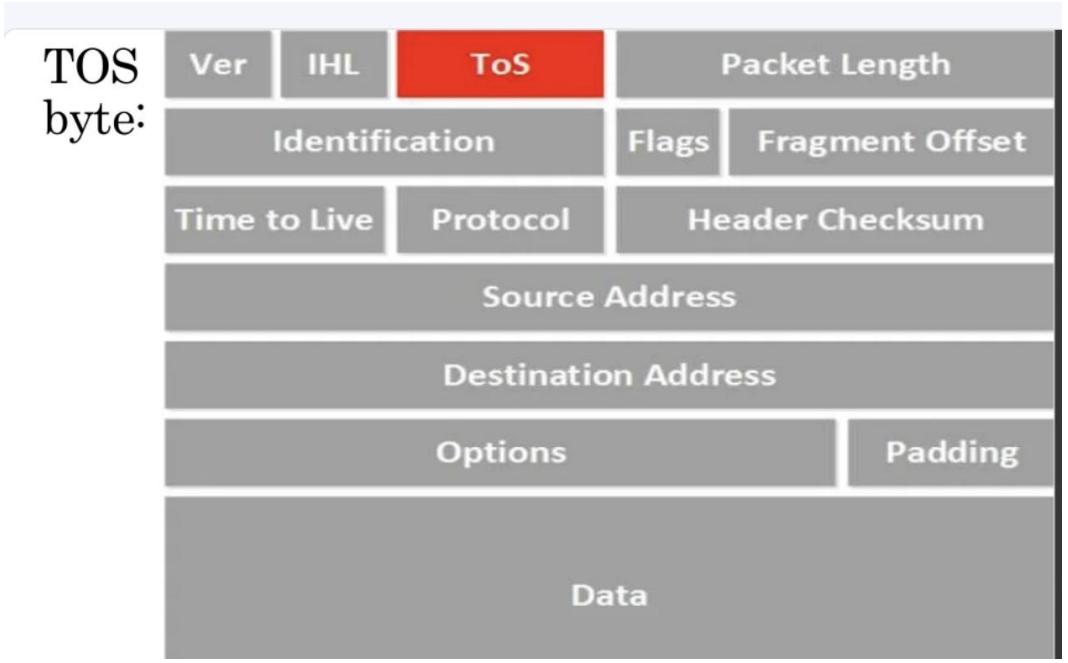
# Basic QoS Architecture




**Classification and Marking:** Classify the packets based on their application QoS requirements and mark the packets accordingly

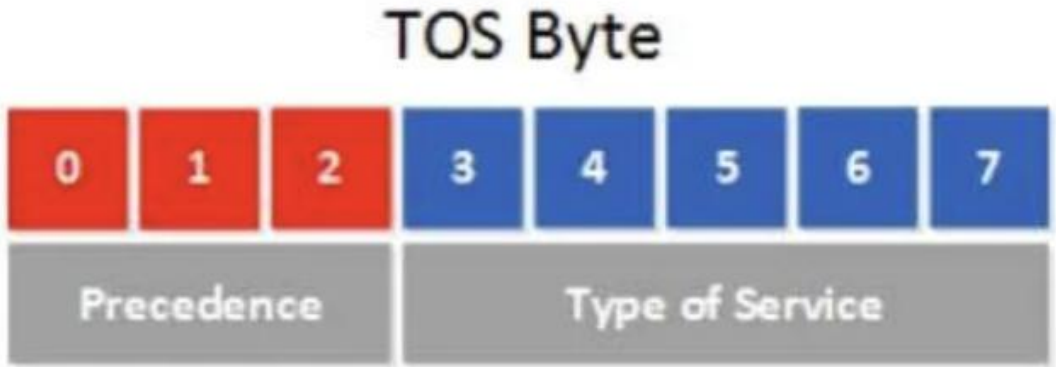
# IP type of Service Field

- IP packets have the Type of Service field (the TOS byte).
- The original idea was to specify a priority and request a route for high throughput, low delay and highly reliable service.



# IP Precedence

- In the beginning the 8 bits of the TOS byte were defined like this:




- The first 3 bits are used to define a precedence. The higher the value, the more important the IP packet is, in case of congestion the router would drop the low priority packets first. The type of service bits are used to assign what kind of delay, throughput and reliability we want.
- *It's a somehow confusing that we have a type of service "byte" and that bit 3-7 are called the type of service "bits". Don't mix them up, these are two different things.*

# Precedence:

- list of the bits and the possible combinations:

| 000 | Routine |
|---|---|
| 001 | Priority |
| 010 | Immediate |
| 011 | Flash |
| 100 | Flash Override |
| 101 | Critic/Critical |
| 110 | Internetwork Control |
| 111 | Network control |

- These eight different classes of traffic are based on the IP precedence.

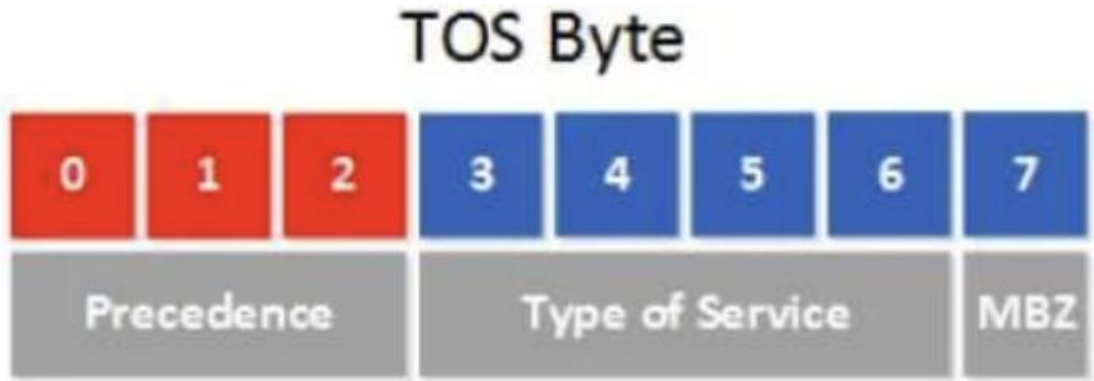
# Type of Service:

| Bit 3 | 0 = normal delay | 1 = low delay |
|---|---|---|
| Bit 4 | 0 = normal throughput | 1 = high throughput |
| Bit 5 | 0 = normal reliability | 1 = high reliability |
| Bit 6, 7 | Reserved for future use | |

This is what they came up with in 1981 but the funny thing is that the "type of service" bits that specify delay, throughput and reliability have never really been used. Only the precedence bits are used to assign a priority to the IP packets.

- The next 3 bits define the priority inside the classes.
- For example, if you want to send voice and streaming video simultaneously, you can use it under this critical class, and under the critical class, you can again relatively prioritise a voice
- give more priority to voice over the video traffic.

About 10 years later, in 1992 RFC 1349 was created that changes the definition of the TOS byte to look like this:



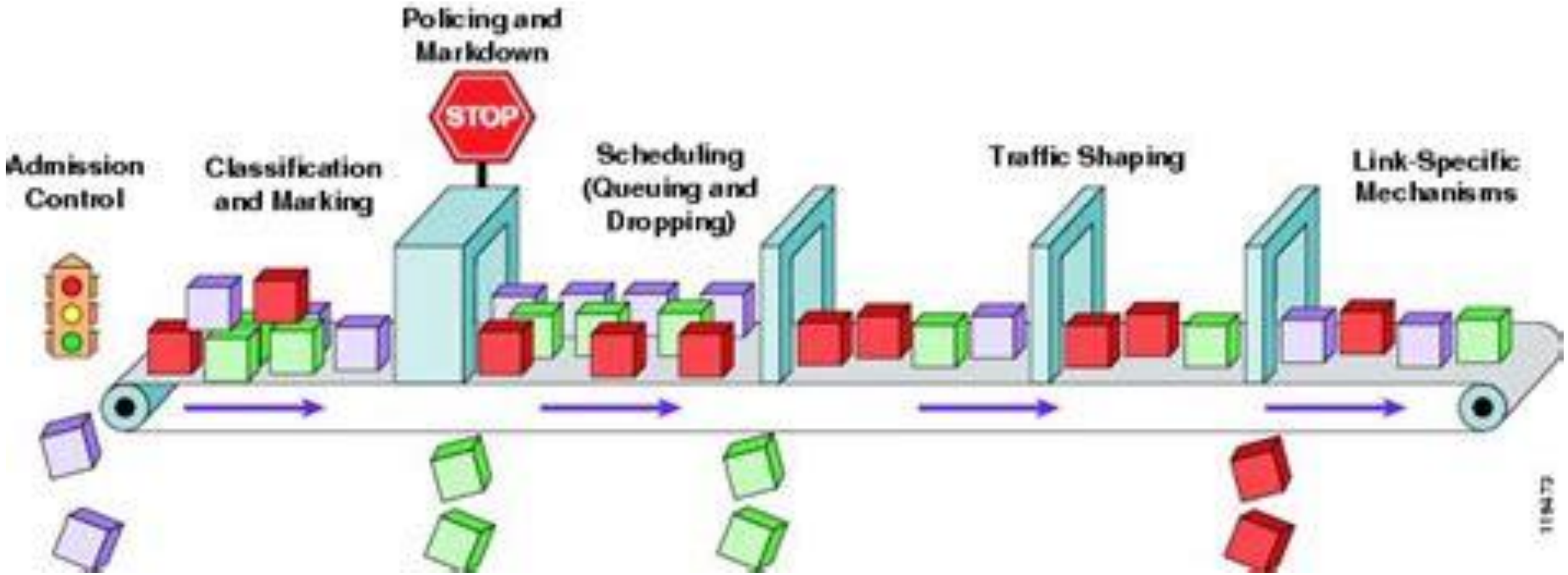

The first 3 precedence bits remain unchanged but the type of service bits have changed. Instead of 5 bits, we now only use 4 bits to assign the type of service and the final bit is called **MBZ (Must Be Zero).** This bit isn't used, the RFC says it's only been used for experiments and routers will ignore this bit.

# Type of service bits now look like this:

| 1000 | Minimize delay |
|---|---|
| 0100 | Maximize throughput |
| 0010 | Maximize reliability |
| 0001 | Minimize monetary cost |
| 0000 | Normal service |

With the old 5-bit type of service bits you could flip some switches and have an IP packet that requested low delay and high throughput. With the "newer" 4-bit type of service bits you have to choose one of the 5 options. But the type of service bits have never been really used.

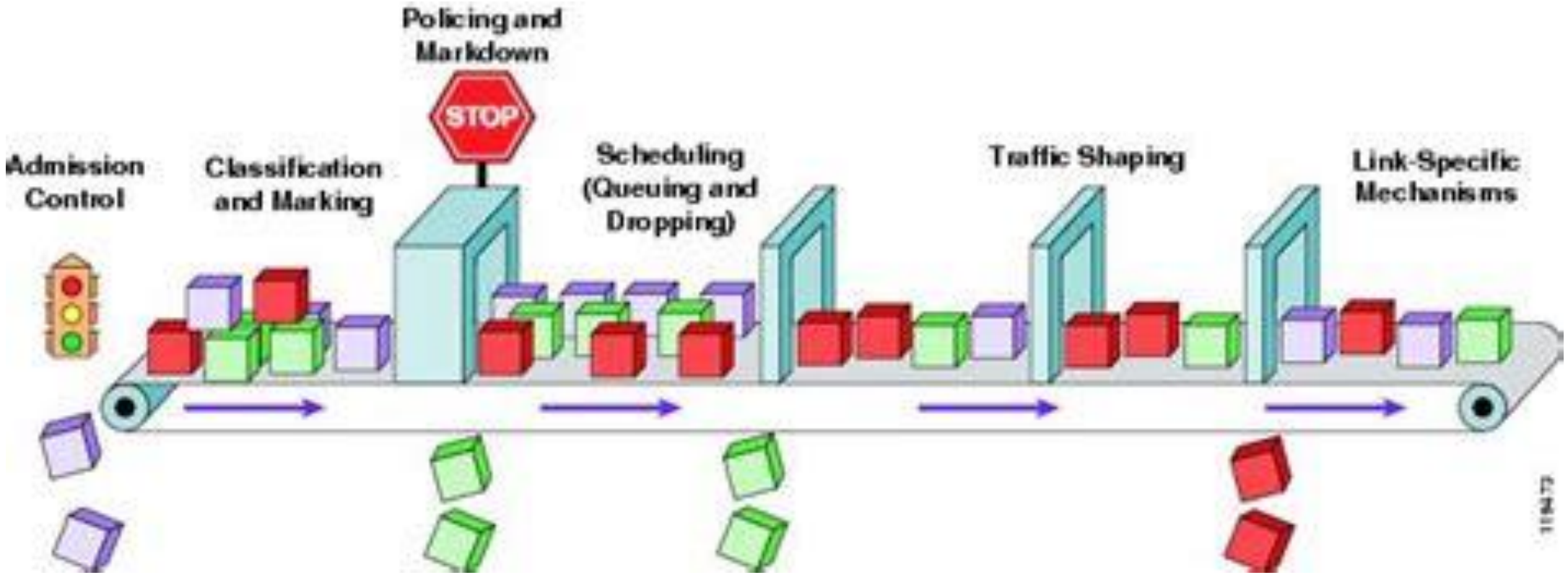
# Basic QoS Architecture




## Policing and Markdown:

Monitor the flow characteristics and check whether certain kinds of packet/flow are violating the QoS requirement or not

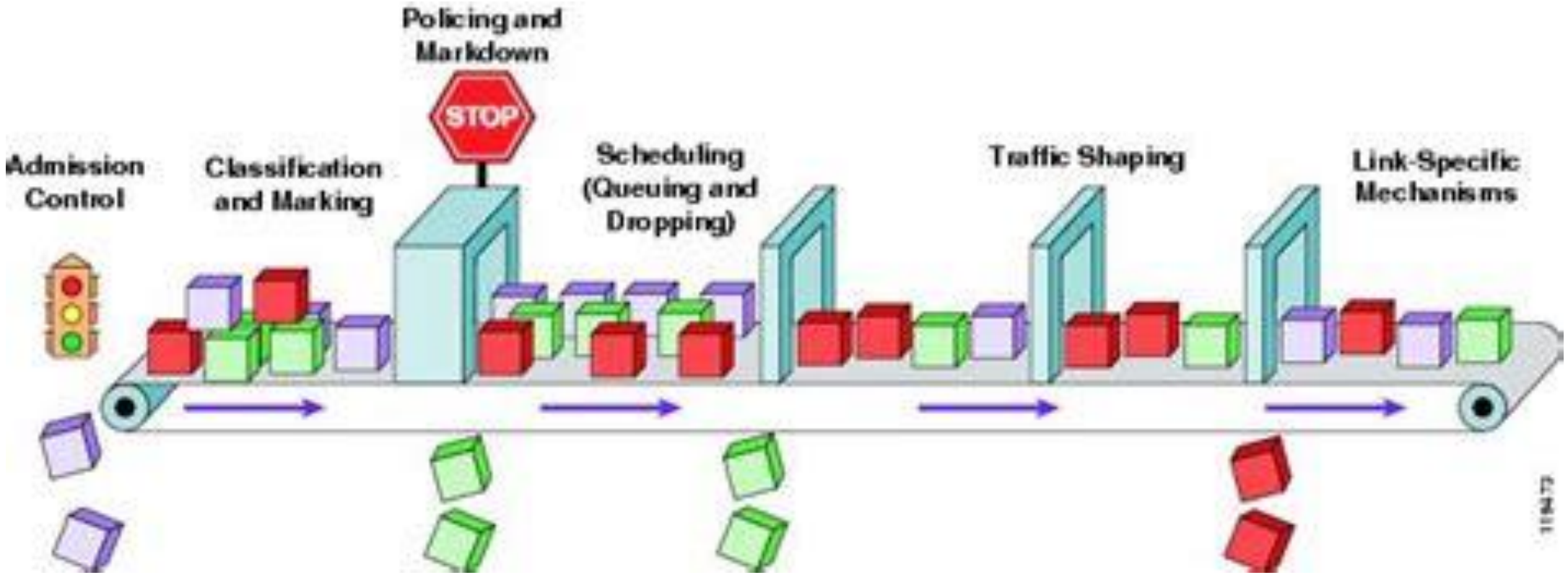
# Basic QoS Architecture




Scheduling:

Based on the markdown by traffic policing, schedule the traffic into output buffers of an interface

# Basic QoS Architecture




Traffic Shaping:

- Ensures smooth jitter in the network
- Controls the outgoing traffic rate irrespective of the income traffic rate (e.g. constant bit rate output from the interface buffer)

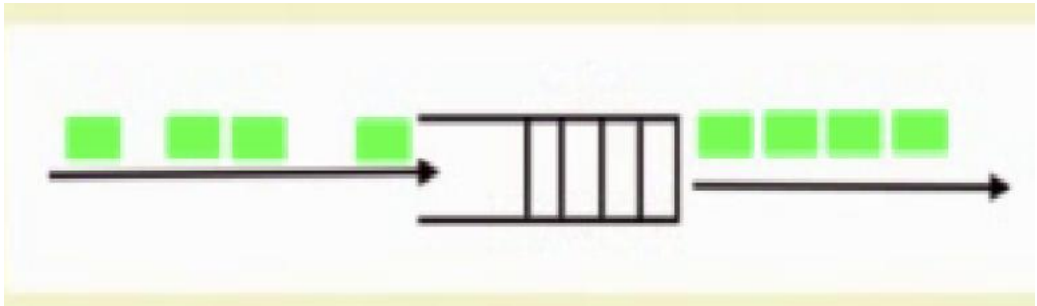
# Traffic Shaping

- A network needs to know what QoS is being guaranteed.
- In the telephone network, characterisation is simple (audio traffic)
- However, traffic in the data network is bursty
  - It typically arrives at non-uniform rates as the traffic rate varies
  - E.g. Videoconferencing with compression
  - User interaction with applications (e.g. browsing a new web page)
  - Computers switching between tasks

Burst of traffic is more difficult to handle than constant rate traffic because it can fill buffers and cause packet loss.

# Traffic Shaping

- A technique for regulating the average rate and burstiness of a flow that enters the network



- Input traffic is bursty. Output traffic has a constant packet rate – reduces the jitter

# Traffic Shaping:

- **Goal:** to allow applications to transmit a wide variety of traffic that suits their needs.

- When a flow is set up, the customer and the network have an agreement for that flow (SLA)

- SLA (Service Level Agreement): a contract between a service provider and a customer that defines the expected quality and availability of the service, as well as the penalties or remedies for any breaches.
- SLA will determine how your packets will be treated when the packets are going over the network.
- E.g. an SLA may specify that a network service should have 99.9% uptime, less than 50 ms latency, and less than 1% packet loss.
- Traffic Shaping reduces congestion and thus helps the network live up to its promise.

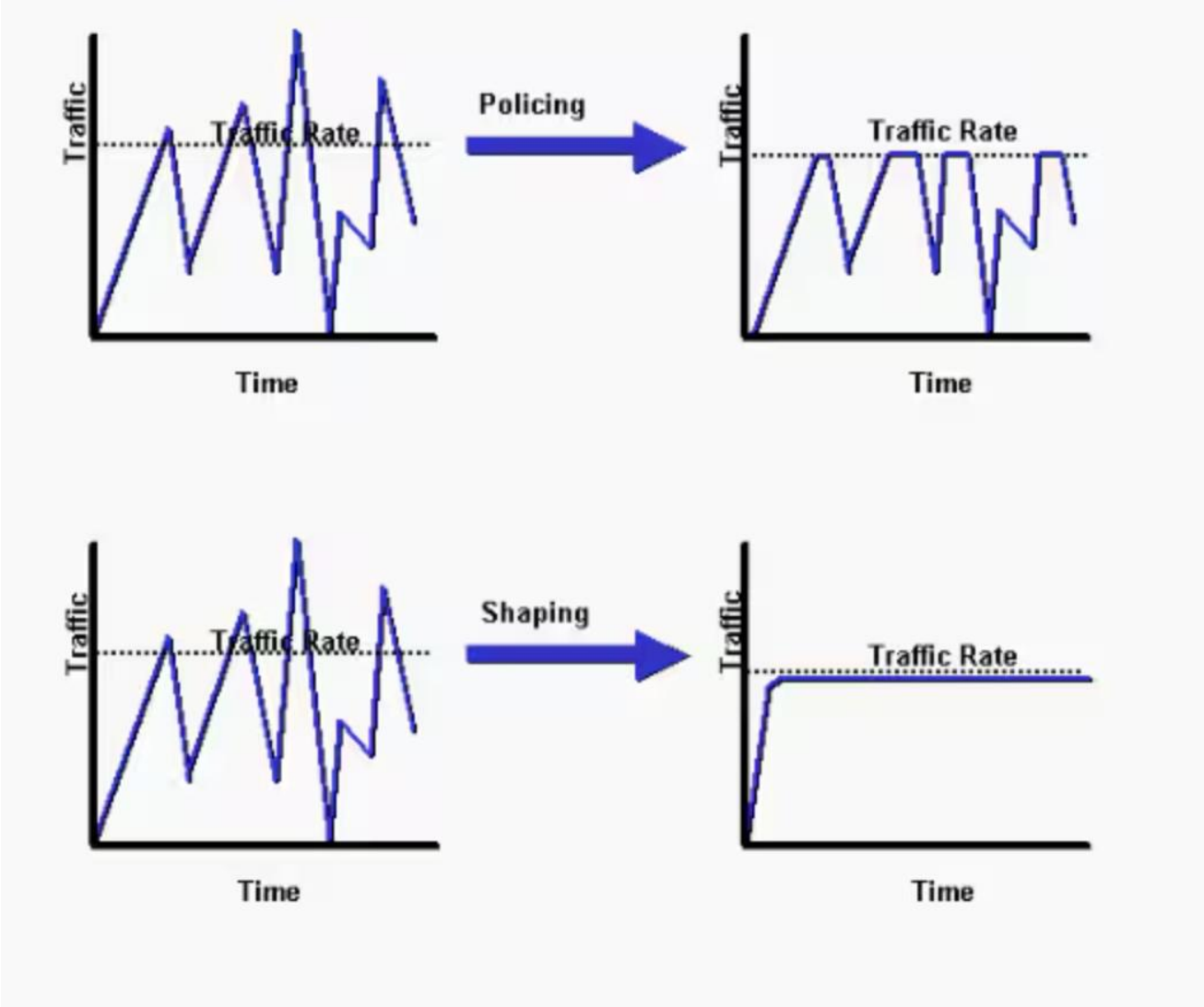
# Traffic Shaping:

- Packets violating the agreement will be dropped by the network or they might be marked as having lower priority.  This is called as Traffic Policing.

## QoS vs SLA:

- QoS and SLA are related but not the same concepts.
- QoS is a technical term that describes how a network performs, while SLA is a legal term that describes how a service is delivered.
- QoS is a means to achieve SLA, but not the only one.
- SLA may also depend on other factors, such as customer satisfaction, service availability, security, and compliance.
- QoS is usually measured and controlled by the network administrator, while SLA is usually negotiated and monitored by the business manager.

# Traffic Shaping vs Traffic Policing:



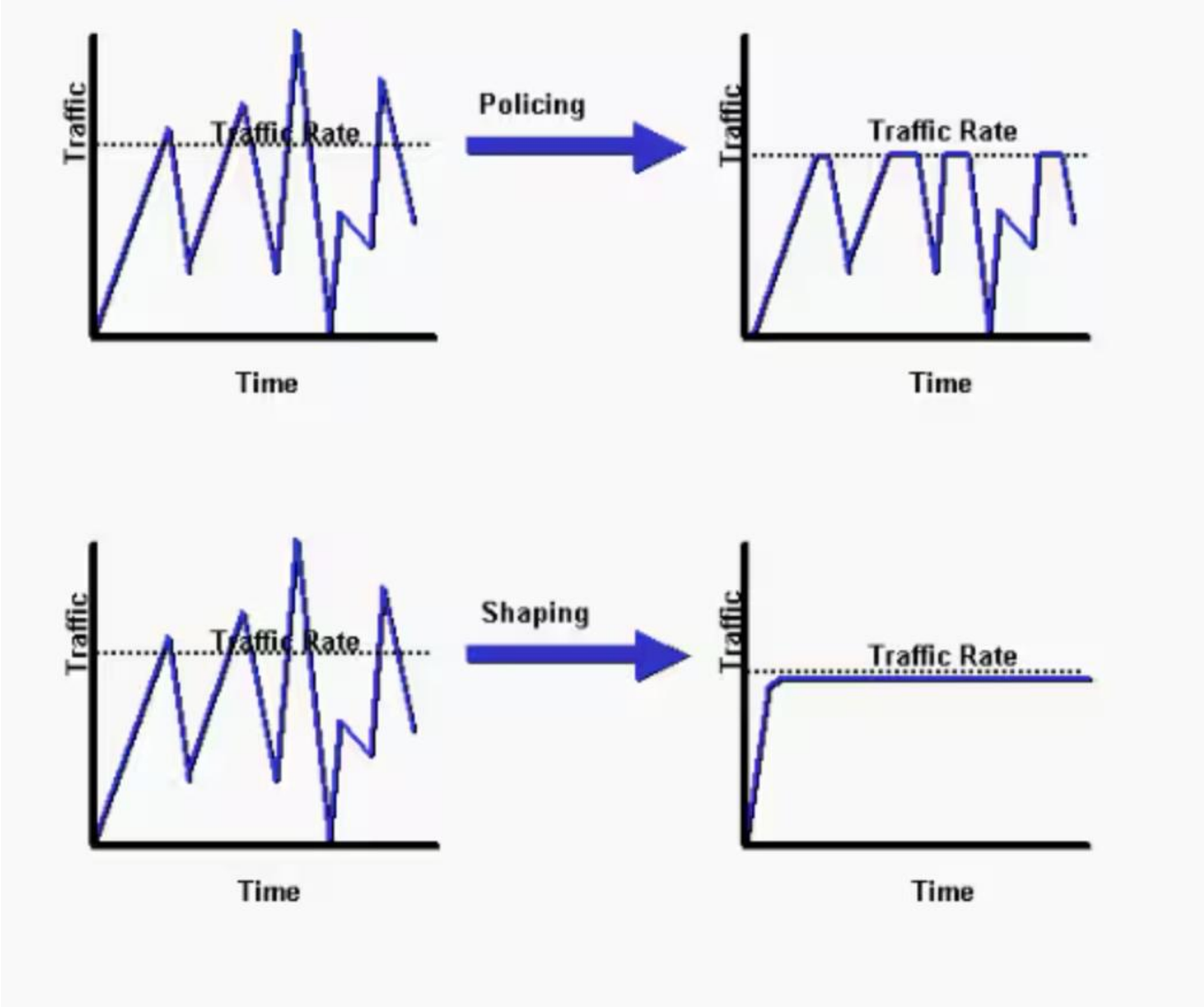

## Traffic Policing:

- Traffic policing propagates bursts.
- When the traffic rate reaches the configured maximum rate, excess traffic is dropped (or remarked).
- The result is an output rate that appears as a saw-tooth with crests and troughs.

## Traffic Shaping:

- traffic shaping retains excess packets in a queue and then schedules the excess for later transmission over increments of time.
- The result of traffic shaping is a smoothed packet output rate.

# Traffic Shaping vs Traffic Policing:




Shaping implies the existence of a queue and of sufficient memory to buffer delayed packets, while policing does not.

# Packet Scheduling Algorithm

- Algorithms to allocate router resources among the packets of a flow and between competing flows
- Three different resources are reserved:
  - Bandwidth
  - Buffer Space
  - CPU cycles

**Bandwidth:** Suppose a flow requires 1 Mbps, and the outgoing line has a capacity of 2 Mbps. If a router tries to direct three flows through the line, then it might not work.

So, reserving bandwidth means not oversubscribing to any output line.

# Packet Scheduling Algorithm

## Buffer Space (Memory):

- When a packet arrives, it has to be buffered inside the router until it can be transmitted on the chosen outgoing line.
- If no buffer is available, discard the packet.
- For good QoS, some buffers might be reserved for a specific flow so that the flow does not have to compete for buffers with other flows.

## CPU Cycles:

Also, a scarce resource

# Packet Scheduling Algorithm

- Packet scheduling also allocates bandwidth and other router resources by determining which of the buffered packets to send on the output line next.

FCFS/FIFO

- Straightforward
- Packets are sent in the same order in which they arrive.
- The amount of buffer space at each router is finite. – So drop the newly arriving packets- when the queue is full.
- Tail drop: If a packet arrives and the queue is full, the router discards that packet. This behaviour is called tail drop. – intuitive behaviour – helps to avoid congestion
- One famous algo for dropping packets before the situation becomes hopeless - Random Early Detection (RED)

# Packet Scheduling Algorithm

- FIFO – simple to implement but not suited to provide good QoS.

- The queue will be filled if a flow is aggressive and sends a large burst of packets.
- If we process packets in the order they arrive, the aggressive sender will hog most of the capacity of the routers.- Other flows will be starved, and hence, their QoS will be reduced.

- To add insult to injury, packets of the other flows that do get through will likely be delayed because they had to sit in the queue behind many packets from the aggressive sender.

(Ref: https://book.systemsapproach.org/congestion/queuing.html

# Priority Queueing Scheduling Algorithm

- A simple variation on basic FIFO queuing is priority queuing
  - Each packet marked with a priority
- The routers then implement multiple FIFO queues, one for each priority class
- Router always transmits packets out of the ***highest-priority queue*** if that queue is nonempty before moving on to the next priority queue.
- Within each priority, packets are still managed in a FIFO manner.

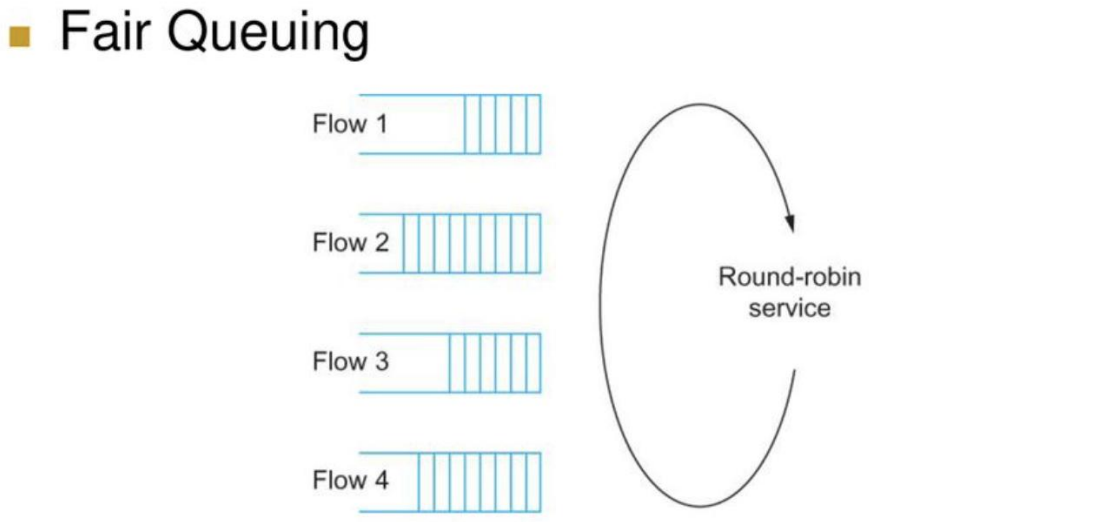
# Priority Queueing Scheduling Algorithm

- The problem with priority queuing, of course, is that the high-priority queue can starve out all the other queues.
- That is, as long as there is at least one high-priority packet in the high-priority queue, lower-priority queues do not get served
- For this to be viable, there need to be hard limits on how much high-priority traffic is inserted in the queue

# Fair Queueing Scheduling Algorithm

The main problem with FIFO queuing is that it does not discriminate between different traffic sources , or it does not separate packets according to the flow to which they belong.

Fair queueing (FQ) maintains a separate queue for each flow currently being handled by the router. The router then services these queue s on the round robin algorithm.




Round-robin service of four flows at a router

# Fair Queueing Scheduling Algorithm

- The main complication with Fair Queuing is that the packets being processed at a router are not necessarily the same length.
- To truly allocate the bandwidth of the outgoing link in a fair manner, it is necessary to take packet length into consideration.
  - For example, if a router is managing two flows, one with 1000-byte packets and the other with 500-byte packets (perhaps because of fragmentation upstream from this router), then a simple round-robin servicing of packets from each flow's queue will give the first flow two thirds of the link's bandwidth and the second flow only one-third of its bandwidth.

# Fair Queueing Scheduling Algorithm

- What we really want is ***bit-by-bit*** round-robin; that is, the router transmits a bit from flow 1, then a bit from flow 2, and so on.
  - However, it is not feasible to interleave the bits from different packets.
- Simulates this behavior instead
  - Determine when a given packet would finish being transmitted if it were being sent using bit-by-bit round-robin
  - Use this finishing time to sequence the packets for transmission.

# Fair Queueing Scheduling Algorithm

- Fair Queuing
  - To understand the algorithm for approximating bit-by-bit round robin, consider the behavior of a single flow
  - For this flow, let
    - $P_i$ : denote the length of packet $i$
    - $S_i$: time when the router starts to transmit packet $i$
    - $F_i$: time when router finishes transmitting packet $i$
    - $F_i = S_i + P_i$

# Fair Queueing Scheduling Algorithm

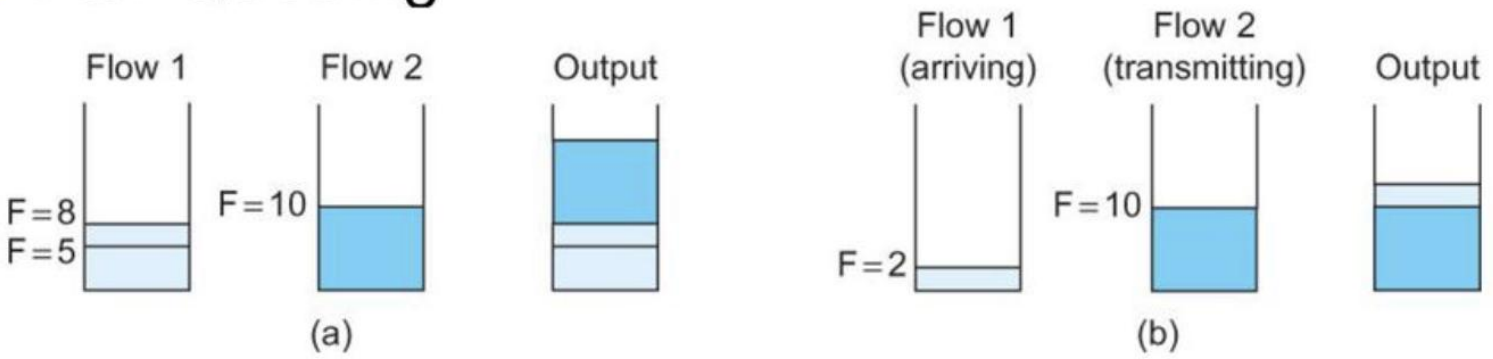
- Fair Queuing
  - When do we start transmitting packet $i$?
    - Depends on whether packet i arrived before or after the router finishes transmitting packet $i$-1 for the flow
  - Let $A_i$ denote the time that packet $i$ arrives at the router
  - Then $S_i = \max(F_{i-1}, A_i)$
  - $F_i = \max(F_{i-1}, A_i) + P_i$

# Fair Queueing Scheduling Algorithm

- Fair Queuing
  - Now for every flow, we calculate $F_i$ for each packet that arrives using our formula
  - We then treat all the $F_i$ as timestamps
  - Next packet to transmit is always the packet that has the lowest timestamp
    - The packet that should finish transmission before all others

# Fair Queueing Scheduling Algorithm

- Fair Queuing




Example of fair queuing in action:
(a) packets with earlier finishing times are sent first;
(b) sending of a packet already in progress is completed

- Part (a) shows the queues for two flows; the algorithm selects both packets from flow 1 to be transmitted before the packet in the flow 2 queue, because of their earlier finishing times.
- In (b), the router has already begun to send a packet from flow 2 when the packet from flow 1 arrives.
- Though the packet arriving on flow 1 would have finished before flow 2 if we had been using perfect bit-by-bit fair queuing, the implementation does not pre-empt the flow 2 packet.

# Fair Queueing Scheduling Algorithm

- Because FQ is work-conserving, any bandwidth that is not used by one flow is automatically available to other flows.
- Thus we can think of FQ as providing a guaranteed minimum share of bandwidth to each flow
- For example:
- if we have 4 flows passing through a router, and all of them are sending packets,
  - →then each one will receive 1/4 of the bandwidth.
- if one of them is idle long enough that all its packets drain out of the router's queue
  - →then the available bandwidth will be shared among the remaining 3 flows, which will each now receive 1/3 of the bandwidth

- the link is never left idle as long as at least one packet is in the queue. Any queuing scheme with this characteristic is said to be *work conserving*

# Weighted Fair Queueing (WFQ)

- Allows a weight to be assigned to each flow (queue)
- Specifies how many bits to send (BW) each time the router services that queue.

Example:

- A router has three flows (queues), one queue has a weight of 2, the second queue has a weight of 3, and third queue weights 1. Assuming that each flow always contains a packet to be sent, what is the percentage of bandwidth assigned to each flow ?

# Weighted Fair Queueing (WFQ)

- Solution
  - The first flow will get 1/3 of the available BW.
  - The second flow will get ½ of the available BW.
  - The third flow will get 1/6 of the available BW.
- Simple FQ gives each queue a weight of 1, which means that logically only 1 bit is transmitted from each queue each time around. This results in each flow getting $1/n^{th}$ of the bandwidth when there are n flows.

# Problem

- Suppose a router has 3 input flows and one output. It receives the packets listed in Table 1 all at about the same time, in the order listed, during a period in which the output port is busy but all queues are otherwise empty. Give the order in which the packets are transmitted, assuming:

(a) Fair queuing.

(b) Weighted fair queuing with:

  (a) flow 1 having a weight of 2, →

  (b) flow 2 having twice as much share as flow 1,→ 4

  (c) and flow 3 having 1.5 times as much share as flow 1.→ 3

| Packet | Size | Flow |
|---|---|---|
| 1 | 200 | 1 |
| 2 | 200 | 1 |
| 3 | 160 | 2 |
| 4 | 120 | 2 |
| 5 | 160 | 2 |
| 6 | 210 | 3 |
| 7 | 150 | 3 |
| 8 | 90 | 3 |

Table 1

# Fair Queueing (FQ)

(a) $F_i$ is the cumulative per-flow size.

$$F_i = \max(F_{i-1}, A_i) + P_i$$

Consider $A_i = 0$ as all packets are received at about the same time so there is no waiting.

$$F_i = F_{i-1} + P_i$$

| Packet | Size | Flow | $F_i$ |
|---|---|---|---|
| 1 | 200 | 1 | 200 |
| 2 | 200 | 1 | 400 |
| 3 | 160 | 2 | 160 |
| 4 | 120 | 2 | |
| 5 | 160 | 2 | |
| 6 | 210 | 3 | |
| 7 | 150 | 3 | |
| 8 | 90 | 3 | |

# Fair Queueing (FQ)

| Packet | Size | Flow | $F_i$ |
|---|---|---|---|
| 1 | 200 | 1 | 200 |
| 2 | 200 | 1 | 400 |
| 3 | 160 | 2 | 160 |
| 4 | 120 | 2 | 280 |
| 5 | 160 | 2 | 440 |
| 6 | 210 | 3 | 210 |
| 7 | 150 | 3 | 360 |
| 8 | 90 | 3 | 450 |

# Fair Queueing (FQ)

- So, packets are sent in increasing order of $F_i$: Packet 3, Packet 1, Packet 6, Packet 4, Packet 7, Packet 2, Packet 5, Packet 8.

| Packet | Size | Flow | $F_i$ |
|---|---|---|---|
| 1 | 200 | 1 | 200 |
| 2 | 200 | 1 | 400 |
| 3 | 160 | 2 | 160 |
| 4 | 120 | 2 | 280 |
| 5 | 160 | 2 | 440 |
| 6 | 210 | 3 | 210 |
| 7 | 150 | 3 | 360 |
| 8 | 90 | 3 | 450 |

# Weighted Fair Queueing (WFQ)

(b) Flow 1 has a weight of 2, so

$$F_i = F_{i-1} + P_i / 2$$

Flow 2 has a weight of 4, so

$$F_i = F_{i-1} + P_i / 4$$

Flow 3 has a weight of 3, so

$$F_i = F_{i-1} + P_i / 3$$

| Packet | Size | Flow | *Weighted* $F_i$ |
|---|---|---|---|
| 1 | 200 | 1 | 100 |
| 2 | 200 | 1 | 200 |
| 3 | 160 | 2 | 40 |
| 4 | 120 | 2 | |
| 5 | 160 | 2 | |
| 6 | 210 | 3 | |
| 7 | 150 | 3 | |
| 8 | 90 | 3 | |

# Weighted Fair Queueing (WFQ)

| Packet | Size | Flow | *Weighted* $F_i$ |
|---|---|---|---|
| 1 | 200 | 1 | 100 |
| 2 | 200 | 1 | 200 |
| 3 | 160 | 2 | 40 |
| 4 | 120 | 2 | 70 |
| 5 | 160 | 2 | 110 |
| 6 | 210 | 3 | 70 |
| 7 | 150 | 3 | 120 |
| 8 | 90 | 3 | 150 |

# Weighted Fair Queueing (WFQ)

- So, packets are sent in increasing order of weighted $F_i$:
- Packet 3, Packet 4, Packet 6, Packet 1, Packet 5, Packet 7, Packet 8, Packet 2.

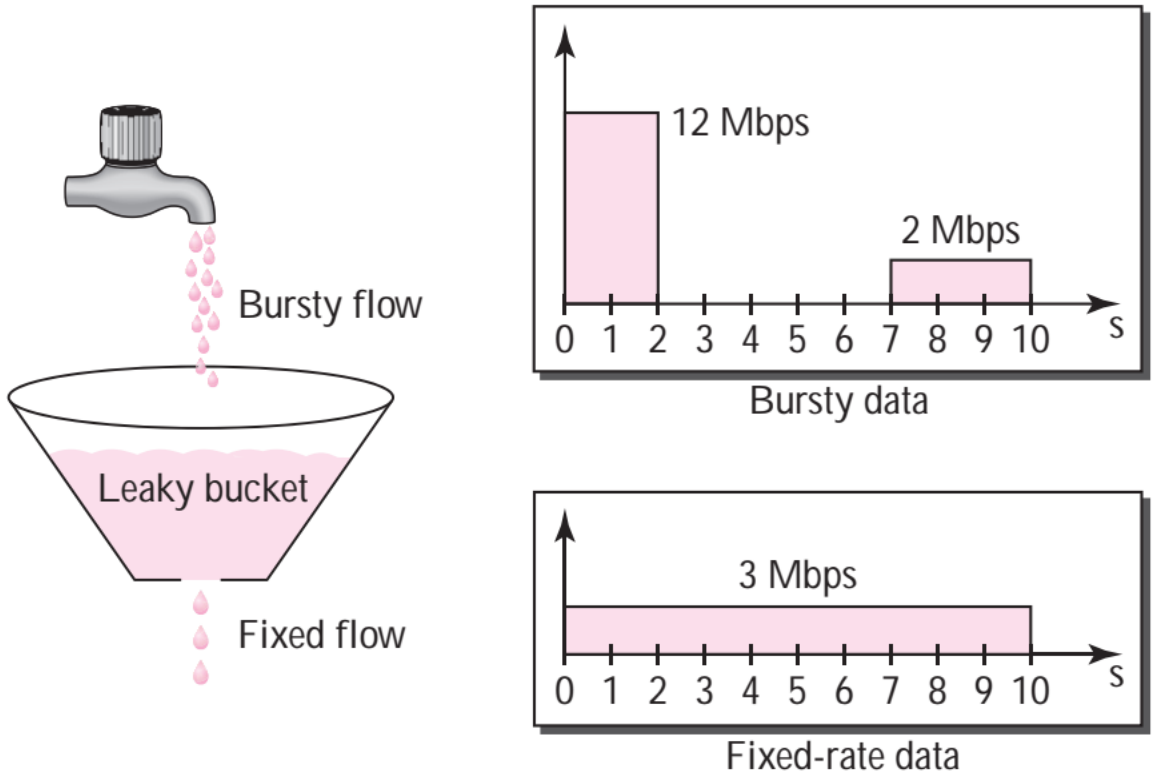
| Packet | Size | Flow | *Weighted* $F_i$ |
|---|---|---|---|
| 1 | 200 | 1 | 100 |
| 2 | 200 | 1 | 200 |
| 3 | 160 | 2 | 40 |
| 4 | 120 | 2 | 70 |
| 5 | 160 | 2 | 110 |
| 6 | 210 | 3 | 70 |
| 7 | 150 | 3 | 120 |
| 8 | 90 | 3 | 150 |

# Traffic Shaping:

- Shaping is not important for file transfers or email, which will consume any or all bandwidth.
- However, they are needed for audio and video connections as they have stringent QoS.
- Mechanism for Traffic Shaping:
  - Leaky Bucket
  - Token Bucket

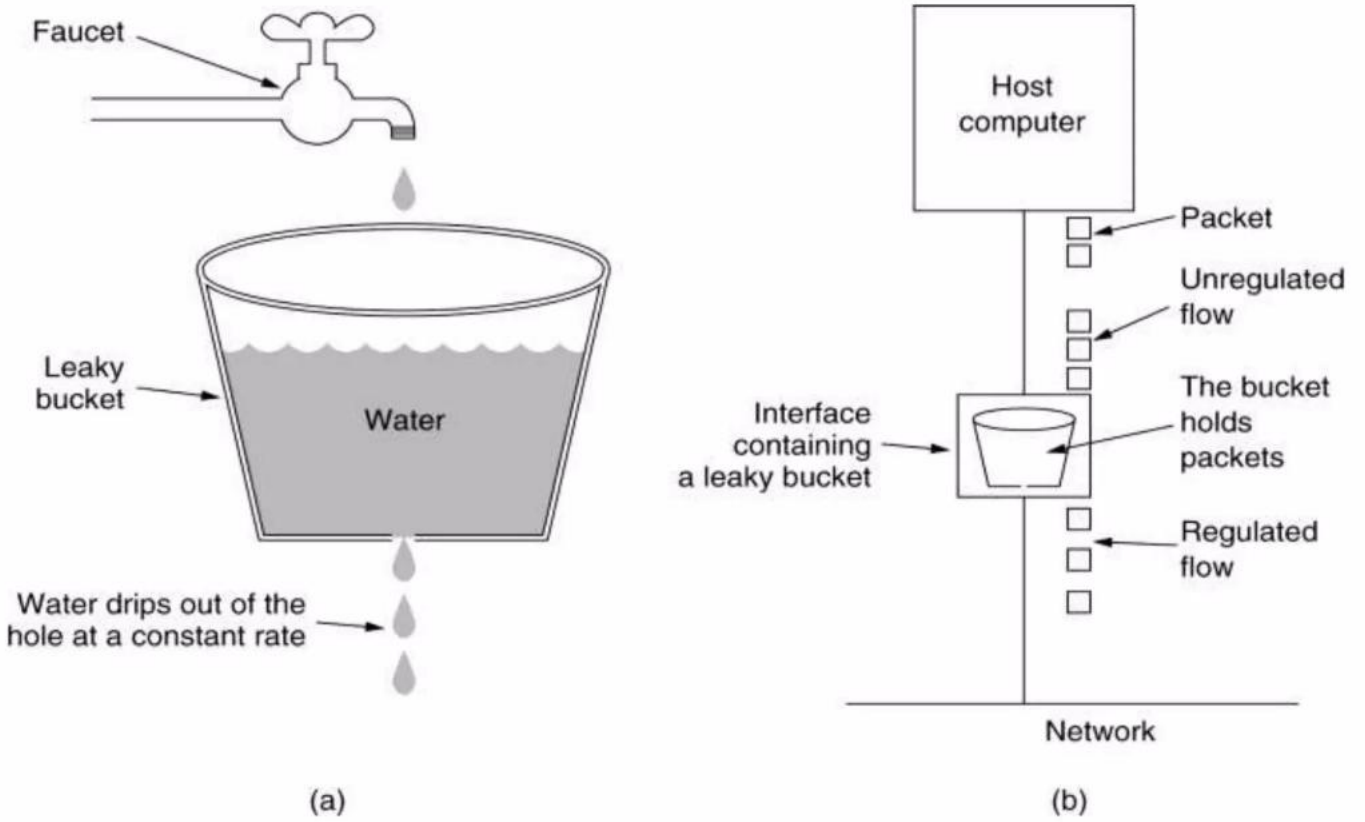
Rate Limiter: **https://www.youtube.com/watch?v=mQCJJqUfn9Y**

# Leaky Bucket:




Input rate may vary but output rate remain constant
Bursty chunks are stored in bucket and send out at average rate.

# Leaky Bucket:




(a) A leaky bucket with water.

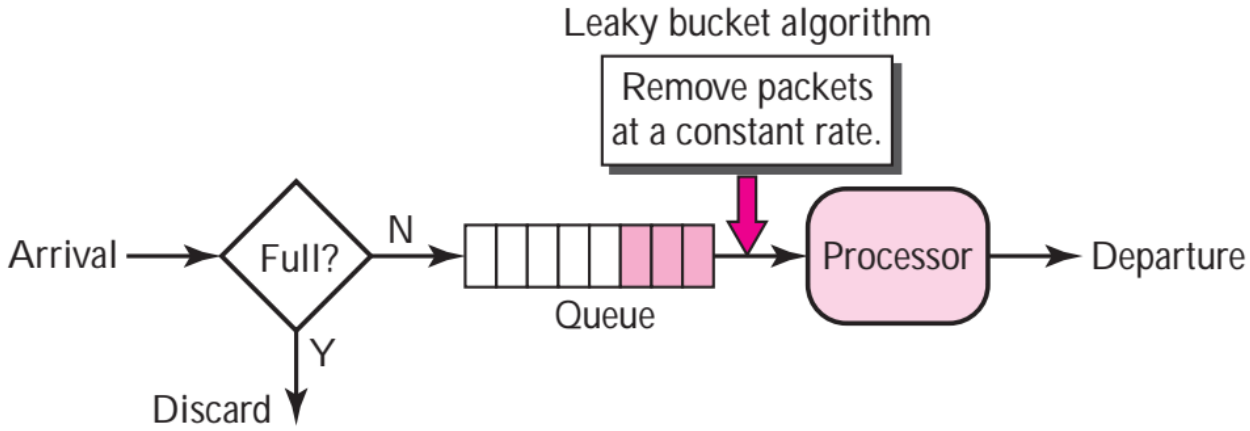
(b) a leaky bucket with packets.

Input rate may vary but output rate remain constant

Bursty chunks are stored in bucket and send out at average rate.

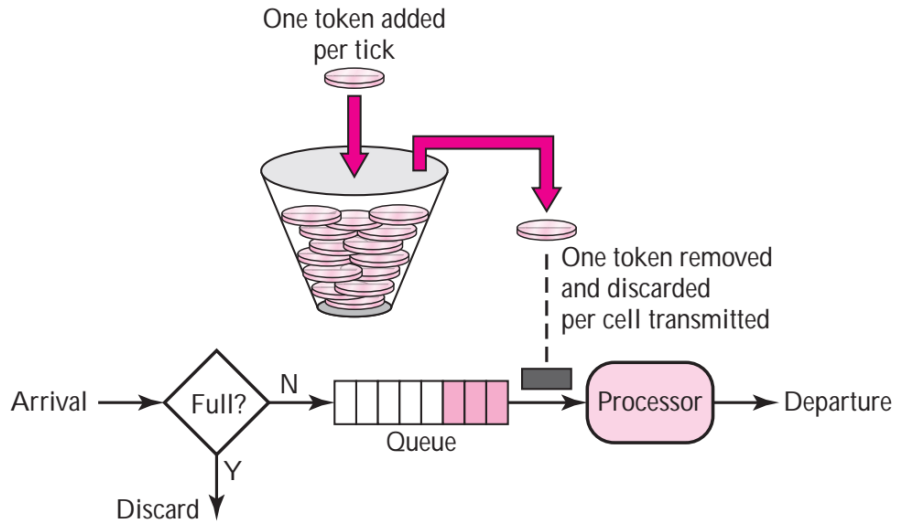
# Leaky Bucket:

Input rate may vary but output rate remain constant
Bursty chunks are stored in bucket and send out at average rate.



# Leaky Bucket:

The leaky bucket is very restrictive. It does not credit an idle host. For example, if a host is not sending for a while, its bucket becomes empty.



# Admission Control

- The user offers a flow with the **accompanying QoS requirements** to the network.
- The network then decides whether to accept or reject the flow based on its capacity and its commitments to other flows.
- If it accepts, the network reserves the capacity in advance at routers to guarantee QoS when traffic is sent on the new flow.
- Reservations must be made at all the routers along the route the packet takes through the network.

# Admission Control

- Any router on the path without reservations might become congested, and a single congested route can break the QoS guarantee.
- Many routing algorithm find the single best path between each source and each destination and send all the traffic over the best path.
- This may cause some flows to be rejected if there is insufficient capacity along the path.
- It is also possible to split the traffic for each destination over multiple paths to easily find excess capacity.

# Admission Control

- Some applications are more tolerant of an occasional missed line than others.
- Applications must choose from the type of guarantee that the network can make, whether a hard guarantee or behavior that will hold most of the time.
- Everyone would like hard guarantees, but the difficulty is that they are expensive because they constrain worst-case behavior.
- Some applications are willing to adjust the flow parameters, and some may not.

# Admission Control

**Flow Specification:**

- Since many parties are involved in the negotiation (sender, receiver, and all the routers along the path) - flows must be described accurately regarding specific parameters that can be negotiated. A set of such parameters is called a flow specification.
- As the specification propagates through the route, each router examines it and modifies the parameter as needed.
- Modifications can only reduce flow specifications (e.g. a lower data rate, not a higher one)
- When it gets to the other end, parameters can be established.

| Parameter | Unit |
|---|---|
| Token bucket rate | Bytes/sec |
| Token bucket size | Bytes |
| Peak data rate | Bytes/sec |
| Minimum packet size | Bytes |
| Maximum packet size | Bytes |

**Figure 5-32.** An example flow specification.

- This is based on RFCs 2210 and 2211 for Integrated Services, a QoS design.
- *the token bucket rate and token bucket size use a token bucket to give the maximum sustained rate the sender may transmit, averaged over a long time interval, and the largest burst it can send over a short time interval.*

# Admission Control

*Why specifying a minimum packet size is essential?*

- Processing each packet takes some fixed time, no matter how short.
- A router may be prepared to handle 10,000 packets/sec of 1 KB each but not be prepared to handle 100,000 packets/sec of 50 bytes each, even though this represents a lower data rate.

# How a router turns flow specifications into a set of specific resource reservations?

- Even with a load slightly below the theoretical capacity, queues can build up and delays can occur
- For ex. A router has a 1Gbps link and the average packet is 1000 bits, it means that it can process 1 million packets/sec.
- But, *in reality, things do not work this smoothly*.
    1. Statistical fluctuations: not all packets arrive evenly spaced apart. Sometimes, there are idle periods, and sometimes, bursts of traffic arrive.
    2. Queueing delays: if too many packets arrive, they form a queue
    3. Backlog: Even if the link is fully utilized, even a tiny bit of delay can create a queue that never clears up.

# How a router turns flow specifications into a set of specific resource reservations?

Even with a load slightly below the theoretical capacity, queues can build up and delays can occur. Consider a situation in which packets arrive at random with a mean arrival rate of $\lambda$ packets/sec. The packets have random lengths and can be sent on the link with a mean service rate of $\mu$ packets/sec. Under the assumption that both the arrival and service distributions are Poisson distributions (what is called an M/M/1 queueing system, where "M" stands for Markov, i.e., Poisson), it can be proven using queueing theory that the mean delay experienced by a packet, $T$, is

$$T = \frac{1}{\mu} \times \frac{1}{1 - \lambda/\mu} = \frac{1}{\mu} \times \frac{1}{1 - \rho}$$

where $\rho = \lambda/\mu$ is the CPU utilization. The first factor, $1/\mu$, is what the service time would be in the absence of competition. The second factor is the slowdown due to competition with other flows. For example, if $\lambda = 950{,}000$ packets/sec and $\mu = 1{,}000{,}000$ packets/sec, then $\rho = 0.95$ and the mean delay experienced by each packet will be 20 μsec instead of 1 μsec. This time accounts for both the queueing time and the service time, as can be seen when the load is very low ($\lambda/\mu \approx 0$). If there are, say, 30 routers along the flow's route, queueing delay alone will account for 600 μsec of delay.

# How a router turns flow specifications into a set of specific resource reservations?

- One method of relating flow specifications to router resources that correspond to bandwidth and delay performance guarantees - given by Parekh and Gallagher (1993, 1994).
- It is based on traffic sources shaped by (R, B) token buckets and WFQ at routers

**Token Bucket (R, B):**

• **R (Rate):** The rate at which tokens are generated, representing the average rate at which a flow can send data over time.

• **B (Burst size):** The maximum burst size that the flow can send at once, allowing short-term deviations from the rate limit, but still adhering to an overall average rate over time.

*The token bucket model helps ensure that the flow adheres to a specified rate (R) while allowing flexibility for burst traffic (up to B tokens).*

# How a router turns flow specifications into a set of specific resource reservations?

**Weighted Fair Queuing (WFQ):**

• A scheduling algorithm - allocates bandwidth to different flows based on their **weights (W)**.

• Each flow is assigned a weight that determines how much of the total router capacity it will receive.

• The weight (W) ensures that each flow receives a fair share of bandwidth relative to its importance or requirements.

# How a router turns flow specifications into a set of specific resource reservations?

Each flow is given a WFQ weight W large enough to drain its token bucket rate R

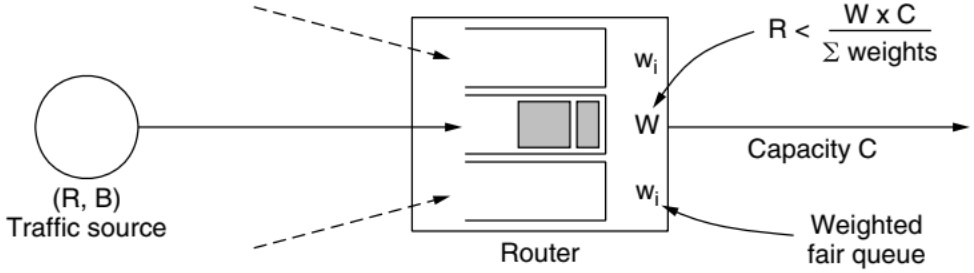



**Figure 5-33.** Bandwidth and delay guarantees with token buckets and WFQ.

Example: Suppose a flow requires a rate of **1 Mbps**, and the router and the output link together have a total capacity of **1 Gbps**.

- The weight for this flow **W** must be large enough to be allocated at least **1/1000th** of the total capacity. In this case, the weight would be **1/1000th** of the sum of all weights assigned to all flows on that router's output link.
- This guarantees that the flow receives a **minimum bandwidth** of **1 Mbps**.
- *If the flow cannot be assigned a sufficient weight (i.e., the weight cannot meet the required rate **R**), the flow cannot be admitted to the network. This ensures that the network does not overcommit resources and that the minimum bandwidth guarantee for each flow is honoured*.

# How a router turns flow specifications into a set of specific resource reservations?

- The largest queueing delay the flow will see is a function of the burst size of the token bucket.
- If the traffic is smooth, without any bursts, packets will be drained from the router just as quickly as they arrive.

  There will be no queueing delay (ignoring other delays).
- if the traffic is saved up in bursts, then a maximum-size burst, B, may arrive at the router all at once.
- the maximum queueing delay, D, will be the time taken to drain this burst at the guaranteed bandwidth, or B/R (again, ignoring other delays.
- If this delay is too large, the flow must request more bandwidth from the network.

# How a router turns flow specifications into a set of specific resource reservations?

- ***The token buckets bound the burstiness of the source, and fair queueing isolates the bandwidth given to different flows.***
- ***The flow will meet its bandwidth and delay guarantees regardless of how the other competing flows behave at the router.***
- These other flows cannot break the guarantee even by saving up traffic and all sending at once.
- Moreover, the result holds for a path through multiple routers in any network topology.
- Each flow gets a minimum bandwidth because that bandwidth is guaranteed at each router.
- In worst case,  a burst of traffic hits the first router and competes with the traffic of other flows, it will be delayed up to the maximum delay of D.
- However, this delay will also smooth the burst.
- The burst will incur no further queueing delays at later routers.
- **The overall queueing delay will be at most D**.

# Service level of QoS:

- Best effort service:
  - no service or delivery guarantee
- Differentiated service:
  - also known as soft QoS
  - traffic is grouped into classes based on service requirements
- Guaranteed service:
  - Network meets a traffic flow’s specific service requirements
  - Needs prior network resource reservation over the path

# Integrated Service

- If the SLA says that all packets belonging to a particular flow should not get delayed by 10ms, then the network ensure that all of the packet s which are coming from this flow will not be delayed by 10 msec.
- Every individual router has to take care of SLA.
- So, routers need to coordinate with each other. (that's why the name integrated architecture)

# Differentiated Service

- Does not give you guaranteed QoS.
- Tries its best to meet the service requirements.
- For internet scale implementation, DiffServ is more suitable (only try its best but no guarantee that it will meet the desired QoS)

# Design Intentions: Integrated Service

- Resources(e.g. bandwidth) must be explicitly managed
  - In order to meet application requirements for packet delay

  This implies that resource reservation and admission control are key building blocks of IntServ, through resource reservation and Admission control.

**Basic idea:** The client reserves network resources on every router "upstream" to the server using a signaling protocol

- Resource reservation Protocol (RSVP), RFC 2205
- Routing decisions are based on QoS parameters- if a particular router is loaded, the packet is not routed to that router.
- Queueing/scheduling – takes account of different flow requirements.
- Discard Policy: Avoid congestion to meet QoS

### 1.2.1 Components of an RSVP-Capable Router

Each router in the new Internet model must contain several components, as illustrated in Figure 1.2. These components interact through explicit interfaces to improve the

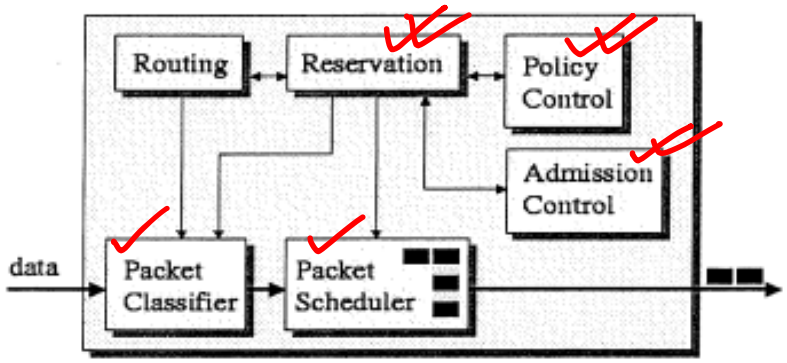



**Figure 1.2** RSVP-capable routers

modularity and independence of the scheme. In addition to the routing mechanism and the flow QoS specification scheme, the router must contain an admission control process, to determine if sufficient resources are available to make the reservation, and a policy control process, to determine if the user has permission to make the reservation. If the RSVP process gets an acceptance indication from both the admission control and policy control processes, it sends the appropriate parameter values to the packet classifier and packet scheduler. The packet classifier determines the QoS class of packets according to the requirements, and the packet scheduler manages various queues to guarantee the required QoS. To guarantee the bandwidth and delay characteristics reserved by RSVP, a fair packet-scheduling scheme, such as weighted fair queuing, can be employed. Fair scheduling isolates data streams and gives each stream a percentage of the bandwidth on a link. This percentage can be varied by applying weights derived from RSVP's reservations. The admission control process, packet classifier, and packet scheduler are collectively called traffic control.

# IntServ Components

Generally IntServ consist of **three traffic control** and one reservation mechanism

Packet Scheduler:

- Actually manages the forwarding of different packet streams
- Additional traffic shaping is done at the sender.

Packet Classifier:

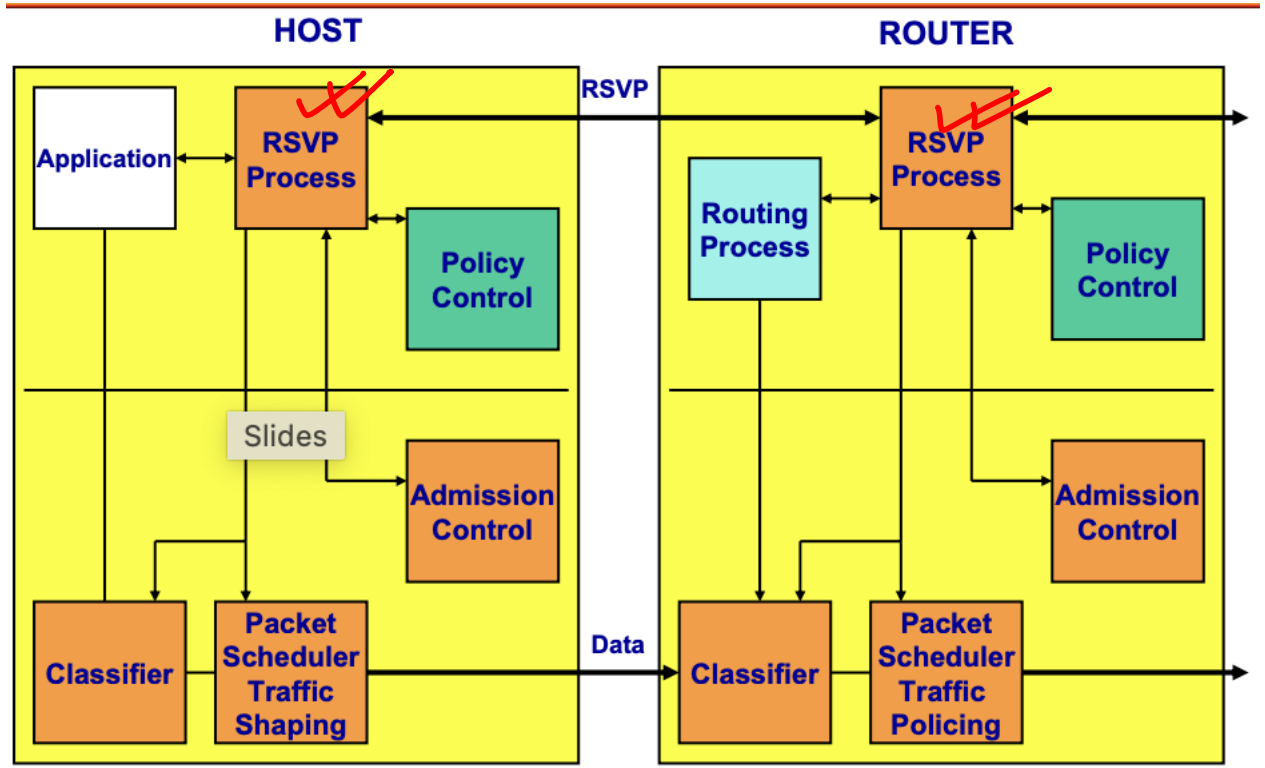
- Each incoming packet must be mapped into some class to allow traffic control; all packets in the same class get the same treatment from the packet scheduler.
- A class might correspond to a broad category of flows e.g. all video flows or all flows attributable to a particular organization
- On the other hand, a class might hold only a single flow.

# IntServ Components

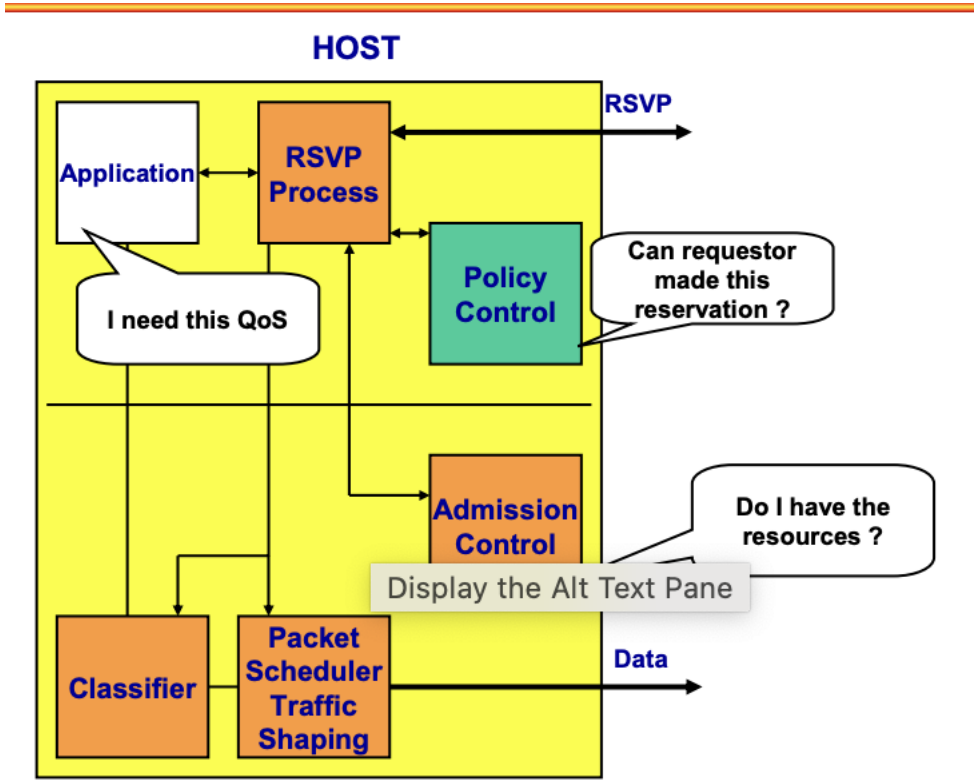
Generally IntServ consist of **three traffic control** and one reservation mechanism

- **Admission control:**
  - Determines whether the node has sufficient available resources to supply the requested QoS
  - That means whether a new flow can be granted the requested QoS without impacting earlier guarntees.
- **Reservation Setup protocol**
  - is necessary to create and maintain flow specific state in the end point hosts and in router along the path of a flow.
  - Protocol is called as RSVP(for ReSerVation Protocol)
- **Policy Control**
  - Additionally needed but outside the scope of IntServ
  - Determines whether the user has the administrative permission to make a reservation including authentication of request.

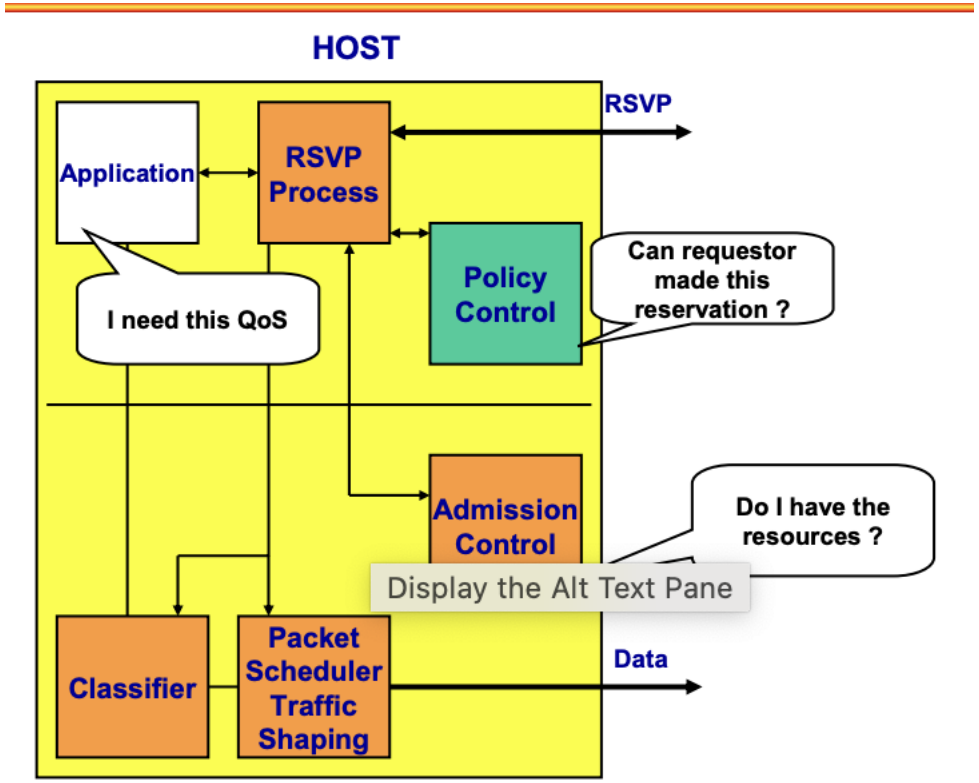
# IntServ Components



# IntServ Components



# IntServ Components



# RSVP

- RSVP **is an internet control protocol**
- RSVP messages are encapsulated within raw IP.
- RSVP uses **IP Protocol Number 46**.
- For any particular flow, the receiver can reserve resources along its path to the sender.
- **Note:** RSVP does not
  - QoS routing
  - Admission control and packet scheduling
  - Forwarding or routing of the data packets.

# RSVP

- RSVP makes resource reservations for both unicast and multicast applications
- Adapting dynamically to changing group membership as well as to the changing routes.
- RSVP is simplex - makes the reservations for unidirectional data flows
- RSVP is receiver-oriented
- The receiver of a data flow initiates and maintains the resource reservation used for that flow.

# RSVP

- RSVP maintain "soft" state in routers and hosts.
- RSVP provides several reservation models or styles to fit a variety of applications
- Router provide transparent operation through routers that do not support it.
- RSVP support both IPv4 and I6v6

# RSVP Messages

## Reservation-Request Messages

- Sent by each **receiver** host toward the senders
- Follows in **reverse** the routes that the data packets use, all the way to the sender hosts
- Delivered to the sender hosts so that the hosts can set up appropriate **traffic-control parameters** for the first hop
- RSVP does not send any positive acknowledgment messages

## Path Messages

- Sent by each **sender** along the unicast or multicast routes provided by the routing protocol(s)
- A path message is used to store the **path state** in each node
- The path state is used to route reservation-request messages in the reverse direction

# RSVP Messages

## Error and Confirmation Messages

**Path-error messages**

•Result from path messages and travel **toward senders**

•Routed hop by hop using the path state

•At each hop, the IP destination address is the unicast address of the previous hop

**Reservation-request error messages**

•Result from reservation-request messages and travel toward the **receiver**

•Routed hop by hop using the reservation state

•At each hop, the IP destination address is the unicast address of the next-hop node

•Information carried in error messages can include the following:

- Admission failure
- Bandwidth unavailable
- Service not supported
- Bad flow specification
- Ambiguous path

# RSVP Messages

**Reservation-request acknowledgment messages**

- Sent as the result of the appearance of a reservation-confirmation object in a reservation-request message
- This **acknowledgment** message contains a **copy** of the reservation confirmation
- An acknowledgment message is sent to the unicast address of a receiver host, and the address is obtained from the reservation - confirmation object
- Forwarded to the receiver hop by hop

# RSVP Messages

## Error and Confirmation Messages

**Teardown Messages**

•**Remove** the path and reservation state without waiting for the cleanup timeout period

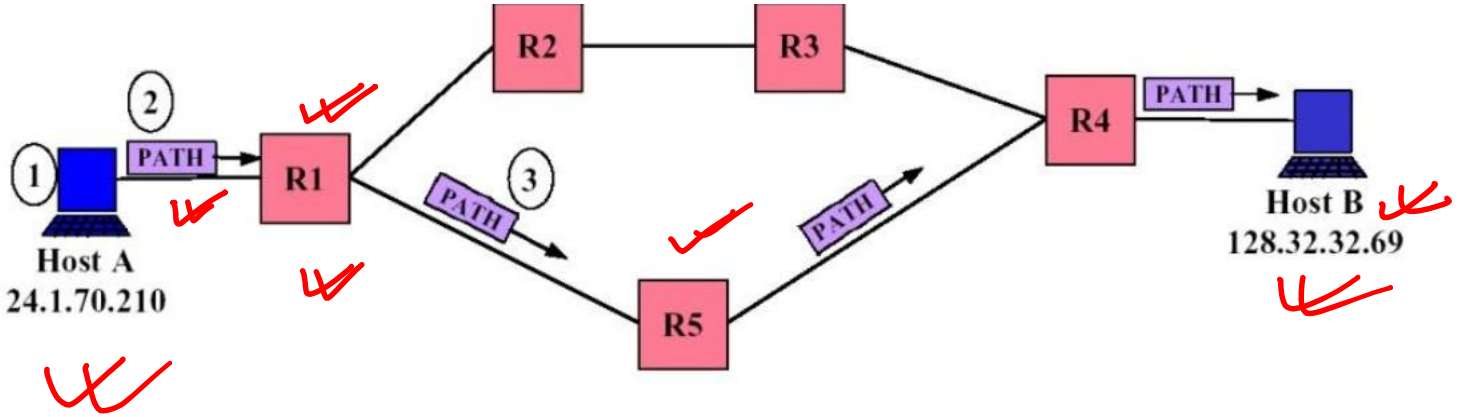
•Can be initiated by an application in an end system (sender or receiver) or a router as the result of state timeout

Two types of messages:

•**Path-teardown messages** delete the path state (which deletes the reservation state), travel toward all receivers downstream from the point of initiation, and are routed like path messages

•**Reservation-request teardown messages** delete the reservation state, travel toward all matching senders upstream from the point of teardown initiation, and are routed like corresponding reservation-request messages

# How RSVP works



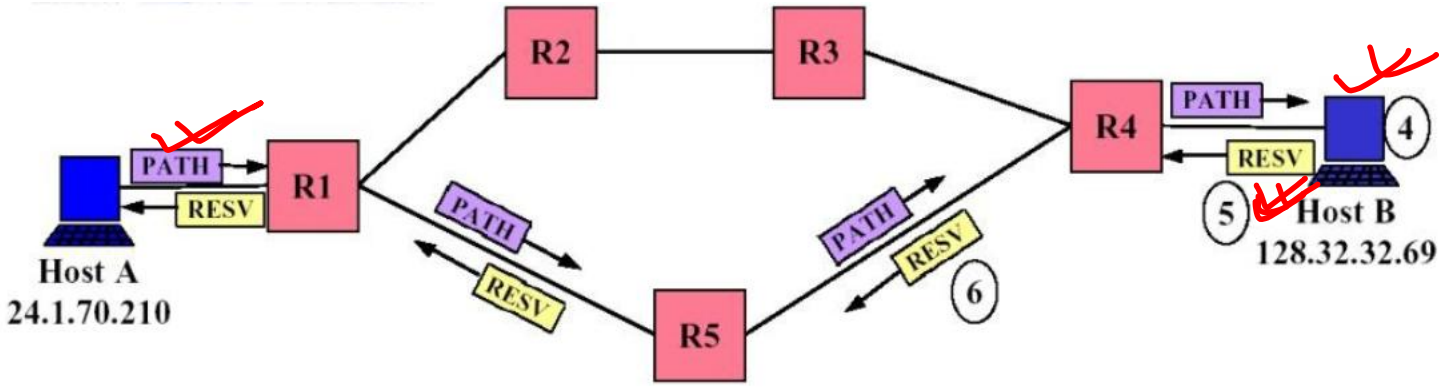

**Senders advertise PATH message.**

1. An application on Host A creates a session, 128.32.32.69/4078, by communicating with the RSVP daemon on Host A.

2. The Host A RSVP daemon generates a PATH message that is sent to the next hop RSVP router R1 in the direction of the session address 128.32.32.69

3. The PATH message follows the next hop path through R5 and R4 until it gets to host B. Each router on the path creates a soft session state with the reservation parameters.

# How RSVP works




Receivers reserve using RESV messages (Flowspec, filter spec, policy data)

4. An application on Host B communicates with the local RSVP daemon and asks for a reservation in session 128.32.32.69/4078. The daemon checks for and finds the existing session state.

5. The host B RSVP daemon generates a RESV message that is sent to the next hop RSVP router, R4, in the direction of the source address 24.1.70.210

6. The RESV message continues to follow the next path until it gets to the Host A. Each router on the path makes a resource reservation.

# RSVP in Action

When a router receives a RESV:

- Passes the request to its admission control function - to find out whether there are sufficient resources to implement the reservation request
- Also, passes the request to its policy control functions- to determine whether policy rules allow the user to make the reservation request.
- If both checks succeed, it sets parameters in the packet classifier and scheduler functions to implement the requested reservation.
- It forwards the RESV message to its upstream neighbour.
- If a router can not reserve some resources according to RESV, it will refuse the reservations and inform the receiver.
- If they can, they merge the reservation request being received and request a reservation from the previous hop router.

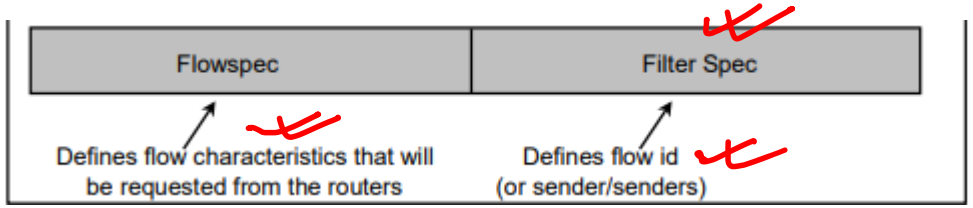
# RSVP Soft States

- RSVP is a state protocol, which means that each router in the reservation process has to maintain a resource state for each RSVP session and update it regularly.
- However, IP/UDP is an insecure protocol, so it can easily be that a message that requests to tear down the QoS connection gets lost.
- Consequently, the connection will stay in the network until the router's memory for storing states is full.
- To avoid this situation a so-called SOFT state is introduced
- In a soft state, both sender and receiver periodically send their PATH/RESV messages
- If a message does not arrive several times, one after another, the states along the routes will be deleted.
- If the loss of a data packet is the reason of the packet non appearance then the reservation will be again initialised again with the RESV messages
- *The soft state feature of RSVP increases the robustness of the protocol.*

# RSVP Reservation Request message

- Consists of *FlowSpec* + *FilterSpec*. This pair is called as flow descriptor.
- FlowSpec
  - specifies a desired QoS- used to set parameters in the packet scheduler
  - It generally include a service class and two sets of numeric parameters.
  - Rspec: defines the desired QoS by receivers
  - TSpec: describes the traffic characteristics of sender (e.g. token bucket with rate r and buffer size b)
  - FilterSpec (Sender address, TCP/UDP, Port #)
    - Used to set parameter in the packet classifier.

*Note: The FlowSpec is used to set parameters in the packet scheduler, while the FilterSpec is used in the packet classifier*



# IntServ Cons

- State and signalling overhead for large networks
- Constant refresh messages
- Per-flow Classification, Policing, Queuing, and Scheduling are significant overheads with many flows.

For every microflow router needs to:

- Identify and categorise it
- Maintain one queue
- An entry in a database

These cause:

- High CPU load
- High memory consumption
- Scalability problem

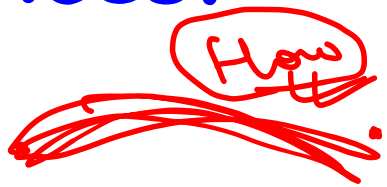
Also:

- All routers must have RSVP (Resource Reservation Protocol)
- There is no policy for the reservation control
- Stations must support signalisation => suitable primarily for small networks.

# Why Differentiated Services?

- Using Resource Reservation, we raise the network's QE (quality/efficiency) product.
- Higher quality guarantees and more efficient use of network resources.
- The more sophisticated a QoS mechanism, the more QE.
- QoS mechanisms come at a cost of increased overhead, which is associated with supporting the QoS mechanism itself.
- Any QoS mechanism should be evaluated regarding the benefit it brings in terms of increases QE product versus the cost it brings.

# Why Differentiated Services?

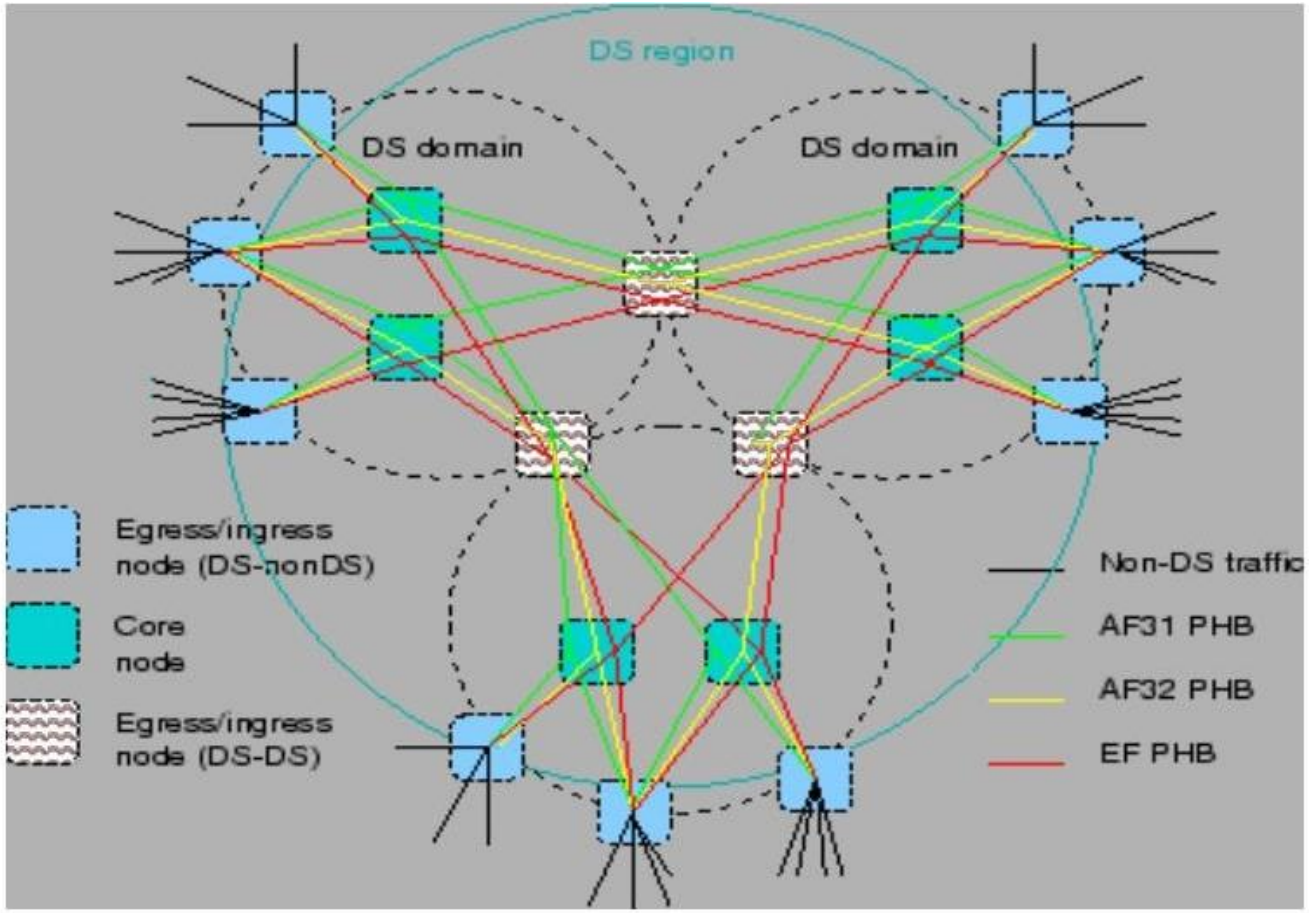



- IntServ is a resource reservation technology.
- The number of simultaneous flows in central network nodes may exceed one hundred thousand.
- If each flow lasts 10 seconds, then more than 10,000 flows per second pass.
- A large telephone exchange can handle up to a few hundred calls per second.
- Maintaining per-flow state information becomes infeasible in core routers.
- The time needed to look up database entries for five tuples in each packet is considerably more overhead than a normal destination address lookup.
- Solution: use a per packet stateless information.

# Differentiated Service, DiffServ

- Traffic is classified at the edges of the network.
- At the edge, each datagram is assigned to one of the behaviour aggregates which DS Codepoints identify
- A 6-bit pattern (called the differentiated Service Code Point, DSCP) in the IPv4 ToS Octet or the IPv6 Traffic Class Octet is used to divide packets of the service classes.
- Complex mechanisms are implemented only on boundary nodes
- DiffServ scalability comes from the aggregation of traffic.
- At the core, packets are forwarded according to PHB, per-hop behaviour associated with the DS point.

# DS, Differentiated Service Architecture




•A **DiffServ Domain** is a logical IP network where PHB definitions are applied consistently across router hops

# Edge Functions in DiffServe

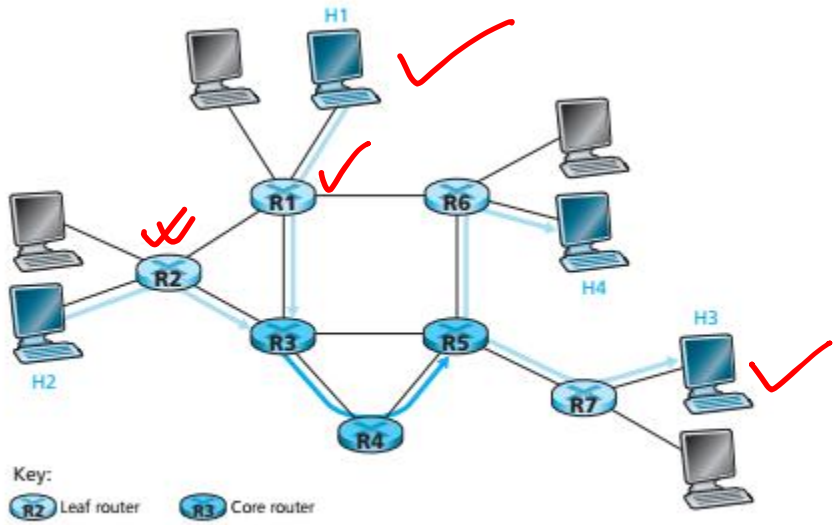



Figure 9.16 ♦ A simple Diffserv network example

- Packet Classification and Traffic Conditioning
- At the edge of the network, arriving packets are marked – the DS field in IPv4 or IPv6 header is set to some value (RFC 3260)
- A packet travelling from H1 to H3 might be marked at R1 while packet from H2 to H4 might be marked at R2.
- Different classes of service receive different service within the core network.

DSCP

# Edge Functions in DiffServe

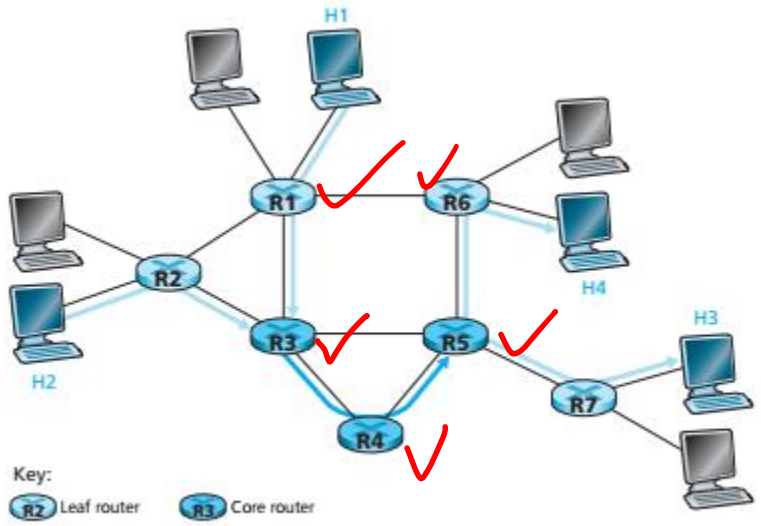



Figure 9.16 ♦ A simple Diffserv network example

- Core Functions:
- When a DS marked packet arrive at a DiffServ capable routers, the packet is forwarded onto its next hop acc to so called *Per-Hop behaviour (PHB)* associated with the packet class.
- *Crucial characteristic of the DiffServ architecture is that a router's per hop behaviour will only be based on packet markings (class of traffic to which a packet belong)*
- *So, DiffServ architecture obliviates the need to keep the router state for individual source-destination pair.*

# Per-Hop Behavior.

- The PHB groups are the mechanism for implementing service differentiation in core networks.
- PHB is defined as "*a description of the externally observable forwarding behaviour of a DiffServ node applied to DiffServ behaviour aggregate"*. (RFC 2475)
- There are several imp. Considerations embedded within:
  - A PHB can result in different classes of traffic receiving different performance (i.e. different externally observable forwarding behaviour)
  - While a PHB defines difference in performance(behaviour) among classes, it does not mandate any particular mechanism for achieving this behaviour.

*As long as externally observable performance criteria is met, any implementation mechanism and any buffer/bandwidth allocation policy can be used.*

***Difference in performance must be observable and hence measurable.***

# Per-Hop Behaviour.

- The PHB groups are the mechanism for implementing service differentiation in core networks.
- PHB is defined as "*a description of the externally observable forwarding behaviour of a DiffServ node applied to DiffServ behaviour aggregate"*. (RFC 2475)
- There are several imp. Considerations embedded within:
  - A PHB can result in different classes of traffic receiving different performance (i.e. different externally observable forwarding behaviour)
  - While a PHB defines difference in performance(behaviour) among classes, it does not mandate any particular mechanism for achieving this behaviour.
- There should not be too many PHB groups, as this complicates the efficient router design.
- DiffServ standard describes, how to mark IP packets so that they receive a particular PHB by using what is called the DSCP, Differentiated Services Code Point- a six bit field in the IP packet header that allows for

# EF, Expedited Forwarding / Premium service (RFC-2598)

- Standard PHB
- Expedited packets should be able to transit the network as if n other packets are present
- EF operates in some respects like a virtual leased or express mail, with overnight delivery.
- Low-latency, low-loss, low-jitter and assured bandwidth service class.
- Used for VoIP, video, traffic
- Provides a strict priority over other service classes.
- Can be implemented using priority queuing, along with rate limiting on the class.
- Should be specifically targeted toward the most critical applications, since if congestion exists, it is not possible to treat all or most traffic as high priority.
- Recommended DSCP value for EF is '101110'.

# Assured Forwarding Service (RFC-2597)

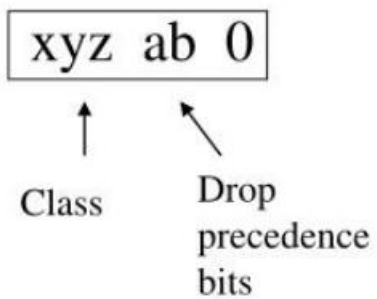
- AF means that forwarding through the metering process should be assured.
- A packet should only be dropped if congestion occurs at the output.
- A packet above the CIR (Committed Information Rate) should simply receive a lower grade PHB (Per-Hop Behavior).
- Traffic in excess of the PIR (Peak Information Rate), labeled as Red traffic, receives the highest drop probability.

# Assured Forwarding PHB (RFC-2597)

- Provides four priority traffic classes each class having its own resources. AFx classes: AF1, AF2, AF3, and AF4.
- In addition, it defines three discard classes for packets that are experiencing congestion: low, medium, high.
- So, there are total 12 classes. Each class is assigned a certain amount of buffer space and interface bandwidth, dependent on the SLA.
- Compared to registered mail—very safe, and assured.
- Flows within an AF that exceed the assigned bandwidth can be penalized.
- Or packets could be re-marked by a policer to a higher drop precedence.
- Offers low packet loss for data applications that demand something better than Best Effort but do not need EF treatment.
- Bursty service, no QoS guaranteed but low loss probability

# Assured Forwarding PHB (RFC-2597)

| DROP Precedence | Class #1 | Class #2 | Class #3 | Class #4 |
|---|---|---|---|---|
| Low Drop Prec | (AF11)<br>001010 | (AF21)<br>010010 | (AF31)<br>011010 | (AF41)<br>100010 |
| Medium Drop Prec | (AF12)<br>001100 | (AF22)<br>010100 | (AF32)<br>011100 | (AF42)<br>100100 |
| High Drop Prec | (AF13)<br>001110 | (AF23)<br>010110 | (AF33)<br>011110 | (AF43)<br>100110 |



# Does DiffServ work?

**Does DiffServ work?**

•DiffServ operates on a hop-by-hop basis.

•How can it guarantee end to end QoS?

**When:**

- Each router in the network is setup with a consistent set of PHBs.
- Network capacity planning is done well.
- Ability across multiple networks, to map DSCPs to and from other service provider or customer networks.
- This will permit custom service classes on one network to be mapped into the closest service class on another network.

**Then DiffServ proves to be very effective in delivering end to end QoS.**

# Differentiated Service, DS Summary

- A powerful and practical means of applying QoS to distinct IP applications.
- A very simple and scalable approach.
- Does not attempt, however, to address the complexities of setting up end-to-end bandwidth guarantees for a given flow across a large meshed network.
- Rapidly gaining widespread acceptance.
- Most leading routing vendors and service providers are implementing at least a few distinct forwarding behaviors (typically two to four) in order to provide some form of premium services.
- The most advanced IP service router vendors are offering a full implementation of DiffServ with extensive customisation capabilities.
- Up to 64 distinct forwarding behaviors, easy custom service class definition using sets of one, two, or three PHBs per service, and highly granular two-rate, three-color metering are all available.