Quality of Service in IP networks

Enterprise Networks

rev 1.0

Andrea Detti, Marco Bonola

University of Roma “Tor Vergata”

Electronic Engineering dept.

E-mail: marco.bonola@uniroma.2it

Ringraziamenti: devo un ringraziamento al Prof. Nicola Blefari-Melazzi, al Prof. Stefano Salsano, all’ing. Roberto Mameli, autori di presentazioni da cui sono state tratte parte delle seguenti slides.

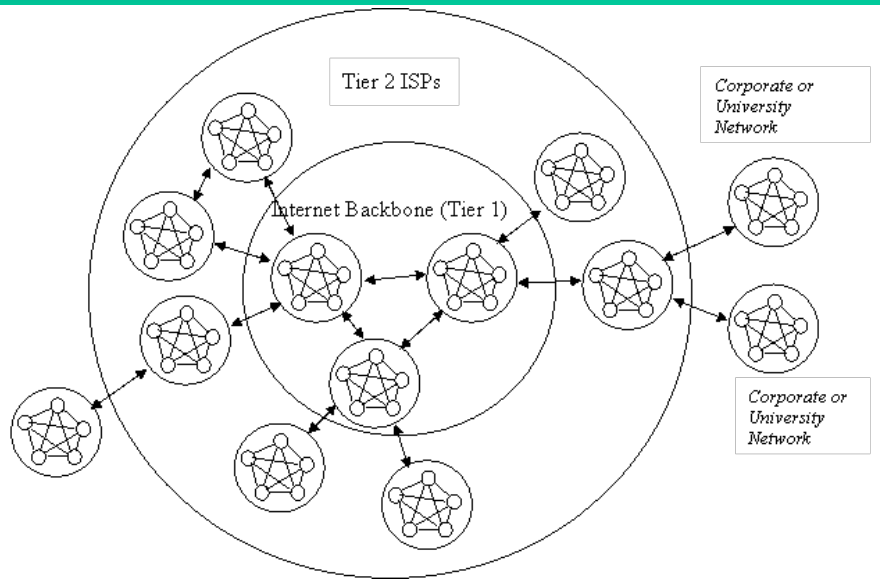
Intro

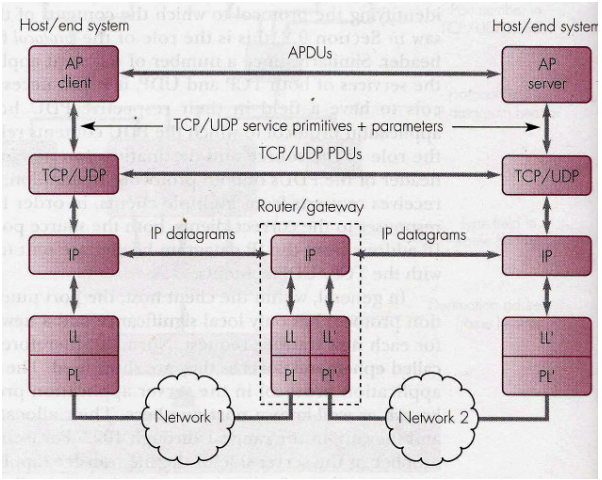
+ A network supports QoS if it provides information delivery with some kind of performance assurance

+ Typical performance parameters
  - Average delay, max delay,
  - Jitter (RFC 3550)
    - Jitter –> D(i,j) = (Rj - Ri) - (Sj - Si) = (Rj - Sj) - (Ri - Si)
    - Estimated Jitter → J(i) = J(i-1) + (|D(i-1,i)| - J(i-1))/16
  - Throughput end to end
  - Packet Loss Rate: lost_packets / tx_packets
  - Required performance parameters depends on the particular application
    - Example: VoIP vs FTP

Intro

+ QoS and internet, not good friends..
  - IP provides a connectionless, unreliable, “best effort” service
  - No access control, on demand delivery
  - Strongly heterogeneous network interconnection
  - TCP provides reliability but no delay, jitter, throughput





Key elements of a QoS framework

+ Scheduling

- Algorithms and queue mechanisms used to differentiate the use of an output interface for different communication session
- Linux: qdisc

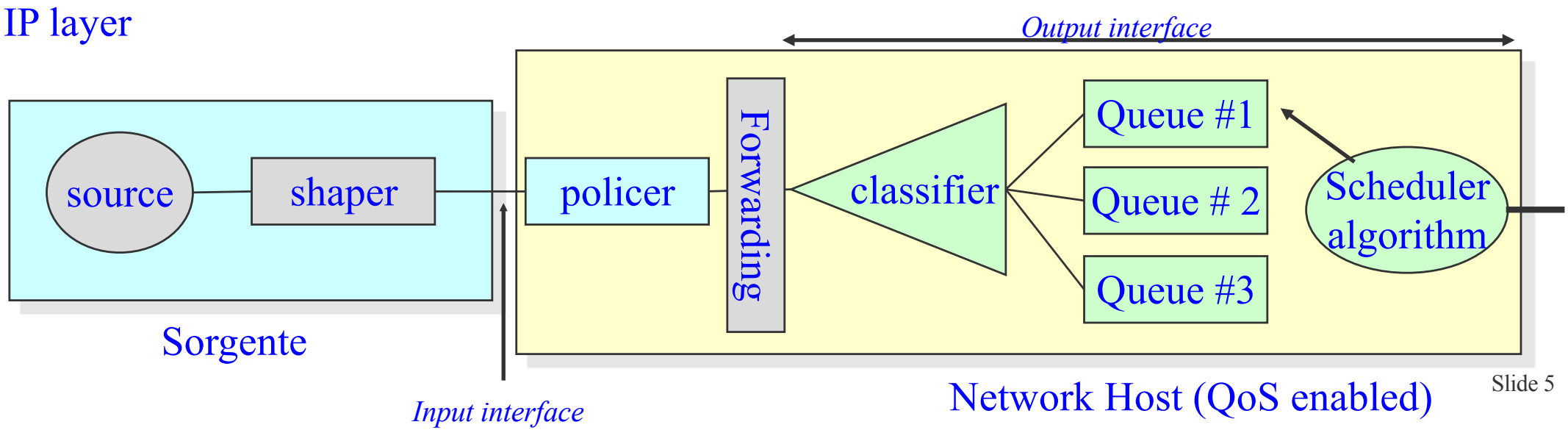
+ Classifier

- Provide packet classification in different service classes
- Each different service class is served by a specific queue
- Different service class can be transported by different communication sessions characterized by the a common parameter (eg ip.proto==UDP)
- Linux : filter

Key elements of a QoS framework

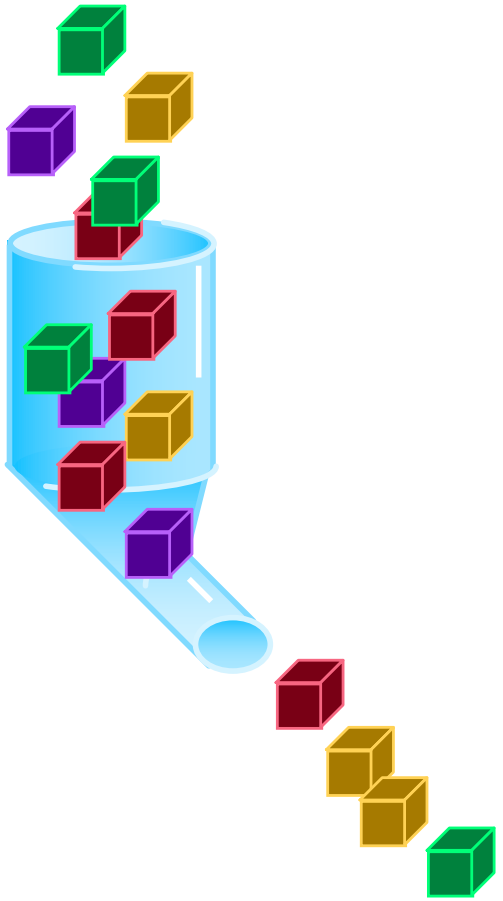
+ **Policer / Shaper**
  - **Policer**: makes incoming traffic compliant with the negotiated QoS parameters. It eliminates or classify as best efforts, all non compliant packets
  - **Shaper**: like the policer but it buffers non compliant packets in order to dealy them
  - Linux : ingress qdisc (policer), qdisc (shaper)

+ **QoS architecture**
  - It defines what elements and where they are placed in the network



Scheduling

+ Offered traffic can exceed the output capacity
+ Without queing it would be lost
+ A queue permits to buffer the exceeded traffic and re-transimit it in a second moment
+ The scheduling strategy defines: (1) how many queues per output interface and (2) from which queue retrieve the next packets
+ Scheduler examples
  - FIFO
  - Priority Queue (PQ)
  - Weighted Round Robin (WRR)
  - Deficit Round Robin (DRR)
  - Generalized Process Sharing (GPS)
  - Weighted Fair Queueing (WFQ)
  - Hierarchy Token Bucket (HTB)



FIFO

+ Simplest queuing strategy e usually set as the default option
+ Packets are transmitted in the same order they are received (FIFO = First In First Out)
+ When the input load is high, interactive communications might be affected
+ Often FIFO implementations handle the memory on a per-packet basis
  - The available memory is usually divided in Maximum Packet Size blocks
  - Possible unused memory

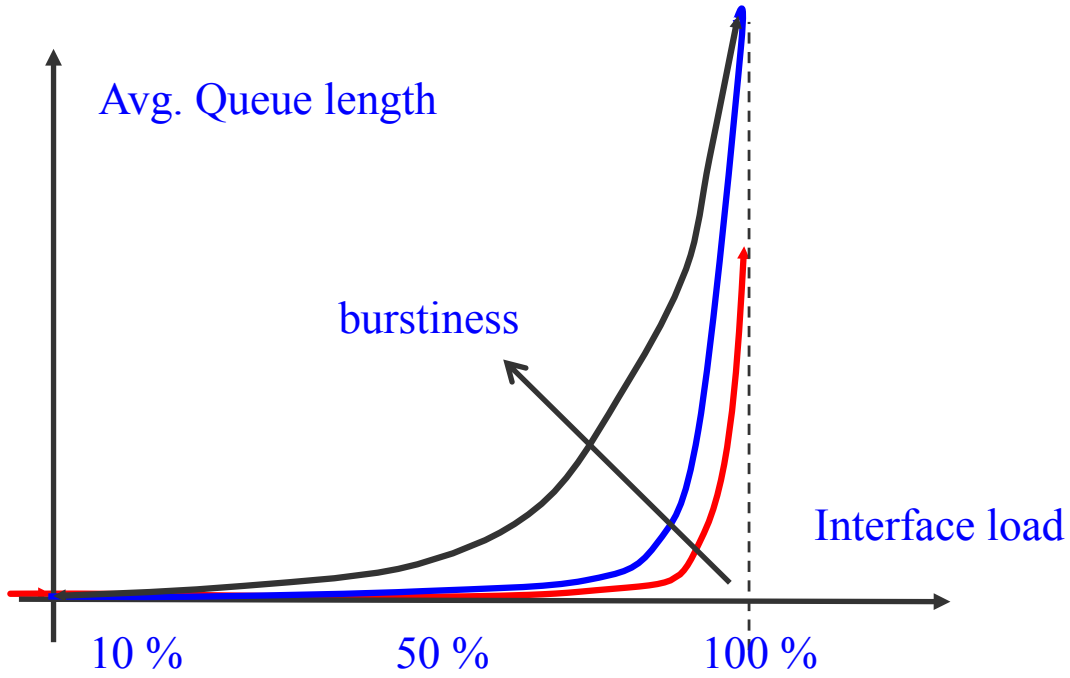
FIFO performance

+ FIFO performance worsen when the ratio between offered bit rate and output capacity exceedss a given threshold

+ This threshold depends on input traffic “burstiness”
  - burstiness s an indication of ”how much” packets arrive togheter in input
  - Burstiness = 0 → CBR
  - High burstiness: non homogeneous distribution. Packets often arrives together and then silence…
  - High burstiness in case od input traffic temporarily correlated (long range dependence)
  - Internet traffic is self-similar and long range dependent

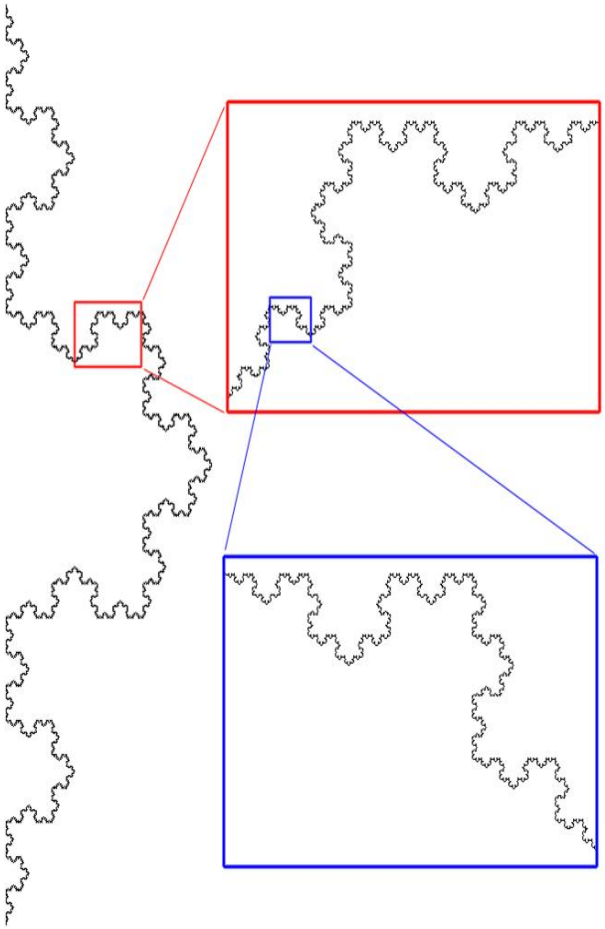
+ Higher the traffic burstines, lower the threshold (worse the performance)

+ OVERPROVISIONING
  - To guarantee a low delay, the input load must be heavy limited (eg. 50%)

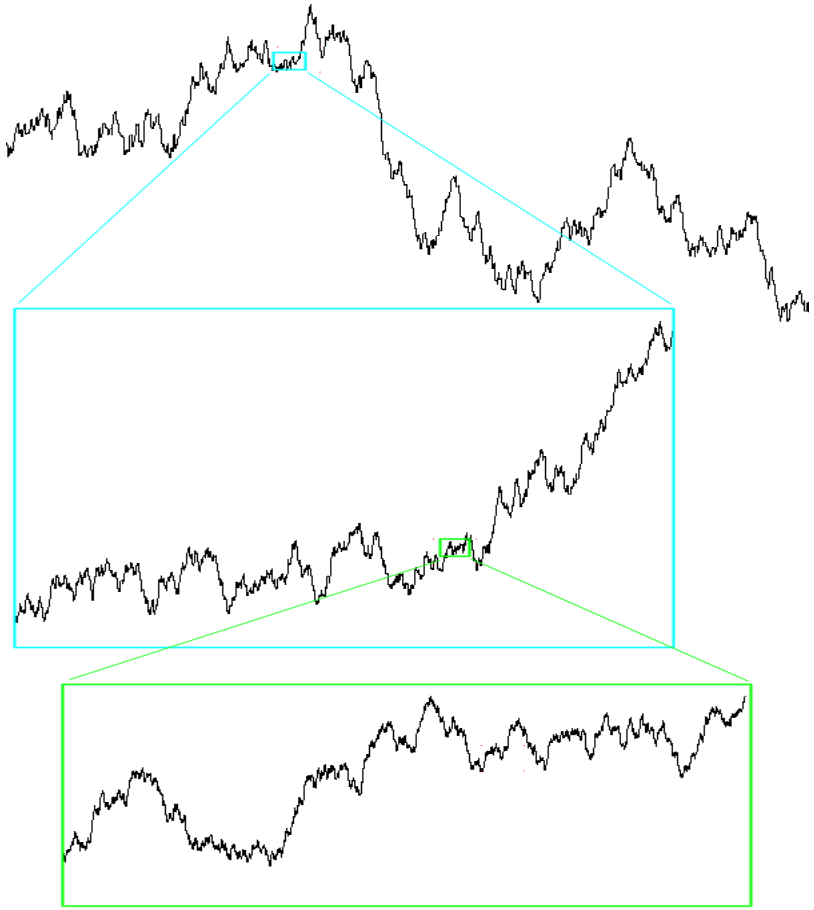


Self-Similar

Le caratteristiche statistiche sono invariati (a meno di un fattore moltiplicativo) alla scala di osservazione



Fractal



Traffic

Self-Similar vs Poisson

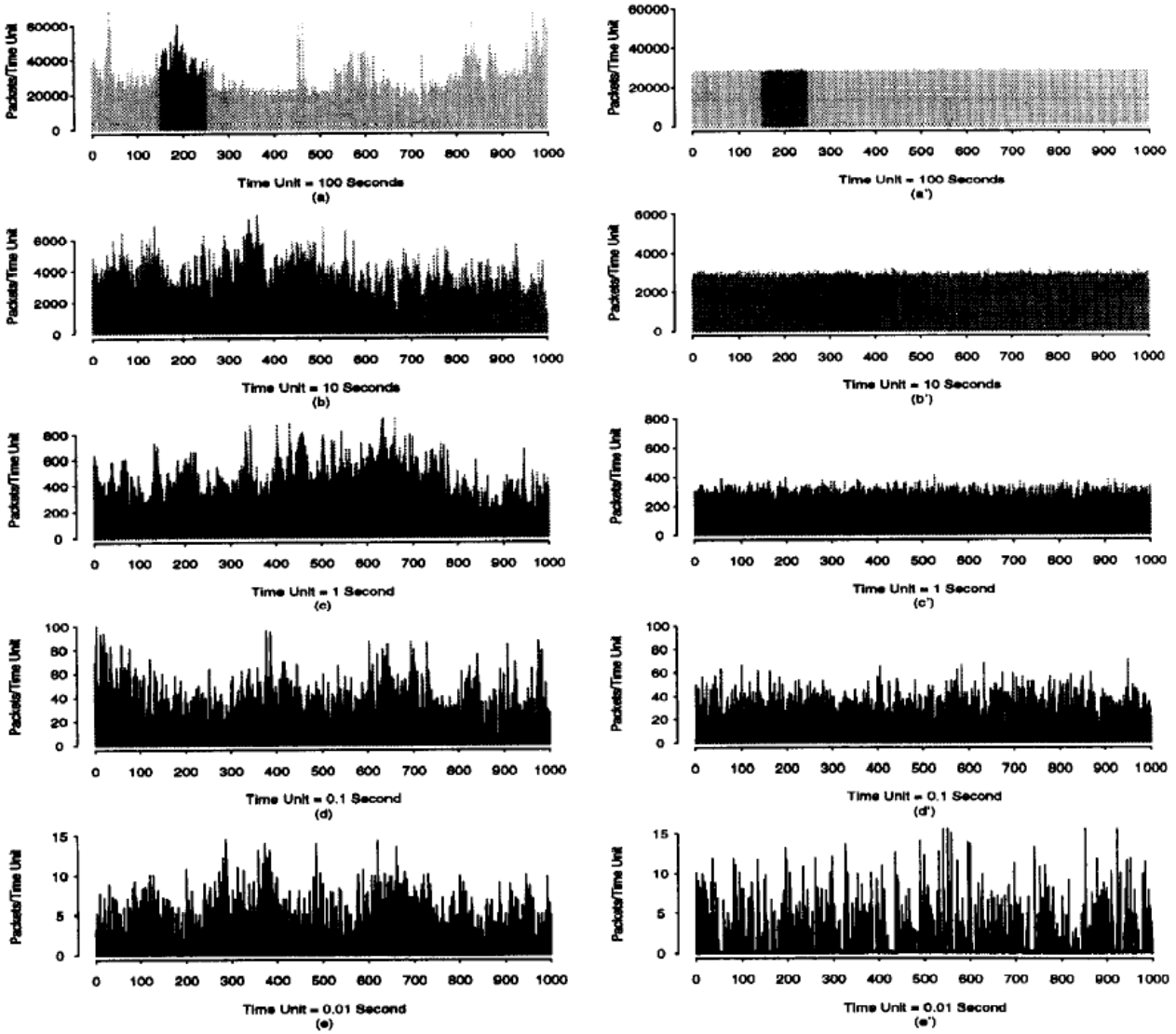
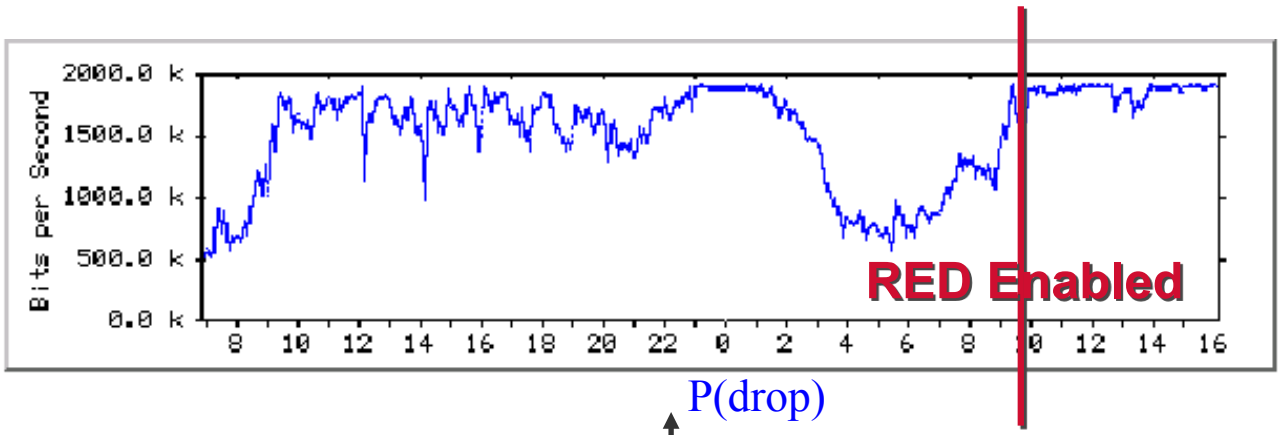
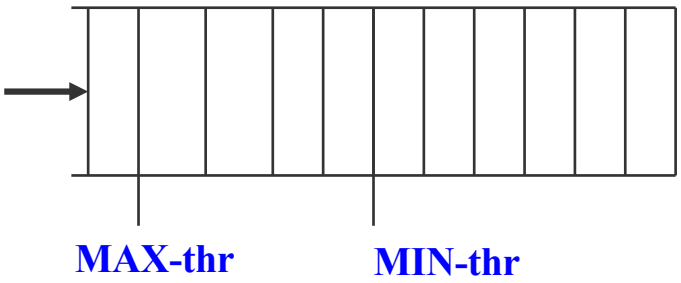


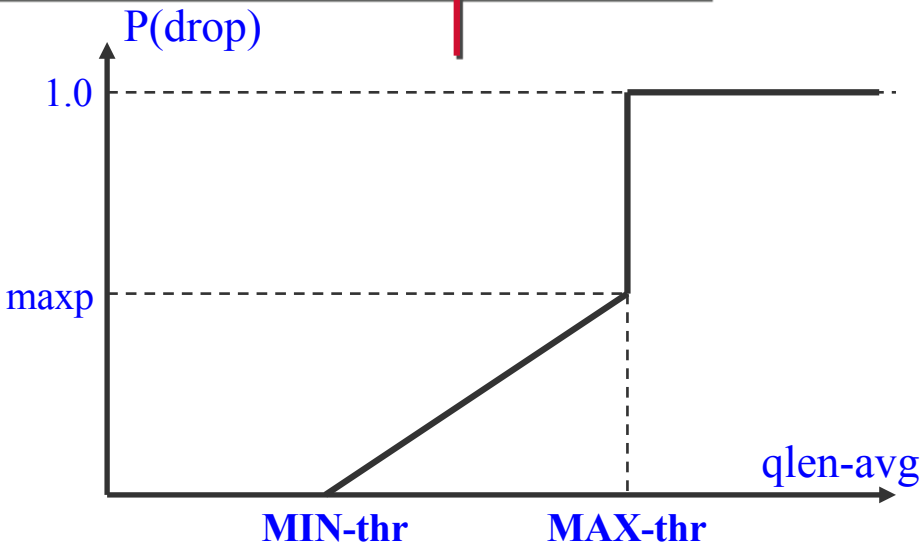
Fig. 4. Pictorial "proof" of self-similarity: Ethernet traffic (packets per time unit) on five different time scales (a)–(e). For comparison, synthetic traffic from an appropriately chosen compound Poisson model on the same five different time scales (a')–(e').

FIFO optimization for TCP traffic: Random Early Detection (RED)

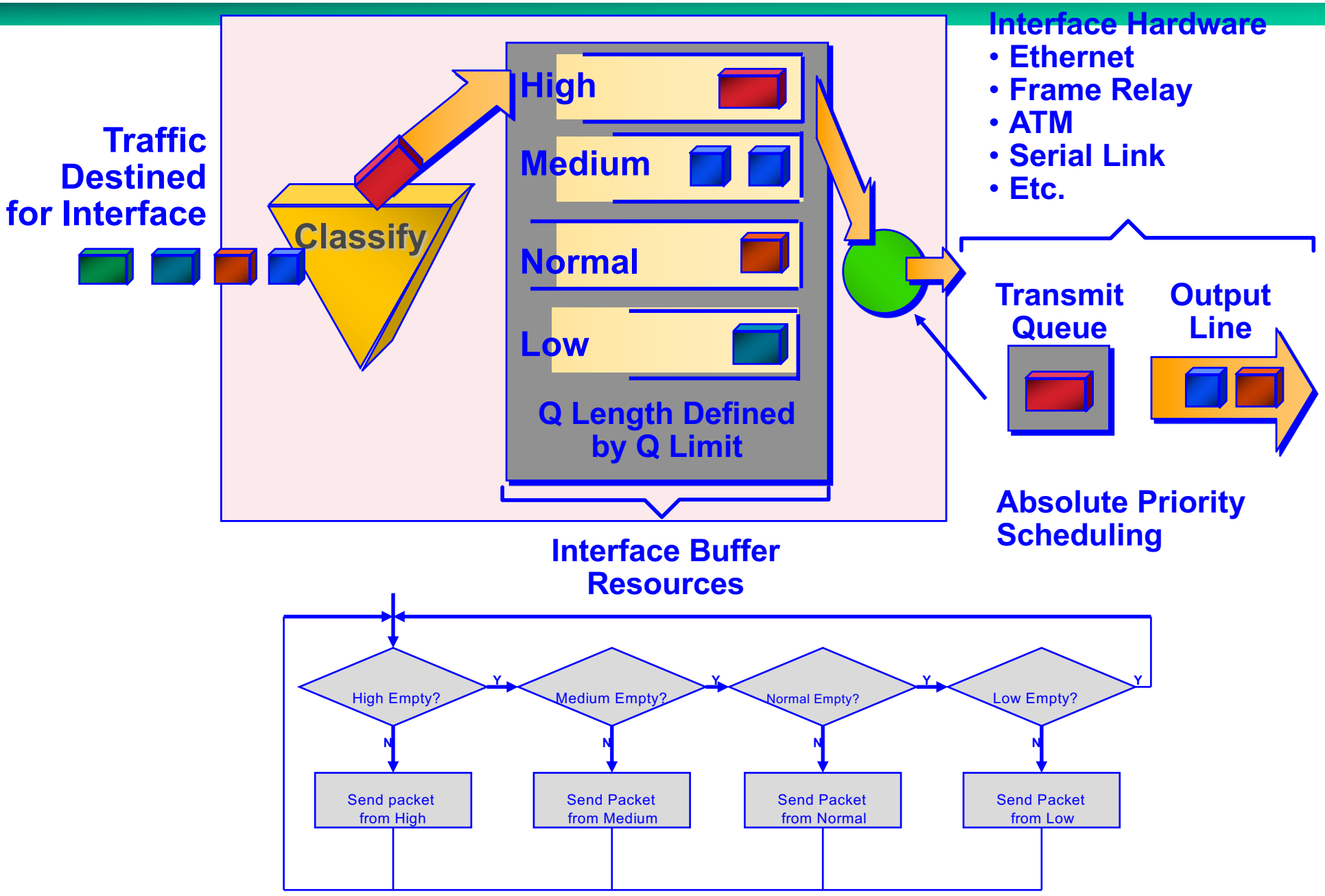
+ Random packet drop with probability proportional to the queue length
+ Eliminates TCP connections sync





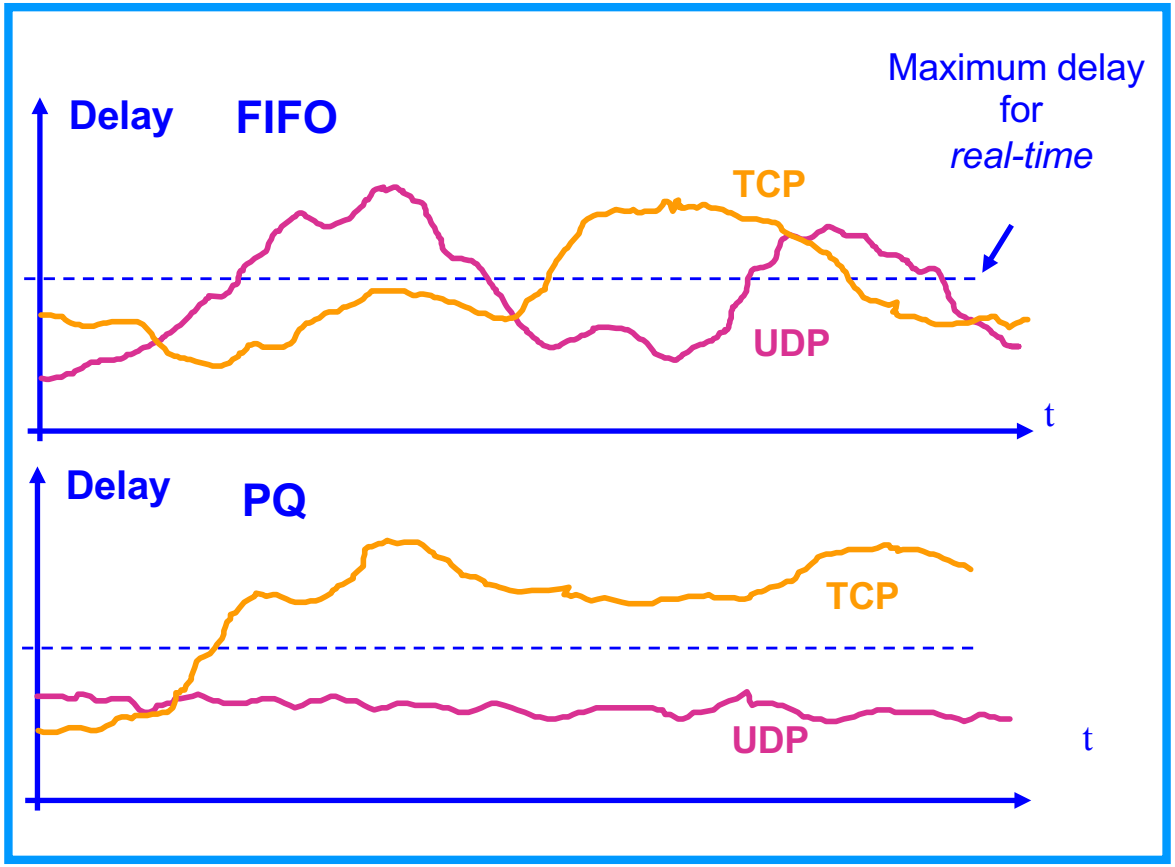


Priority Queuing (PQ)



Priority Queuing (PQ) - Esempio

2 queues: UDP traffic (low delay requirements), TCP traffic

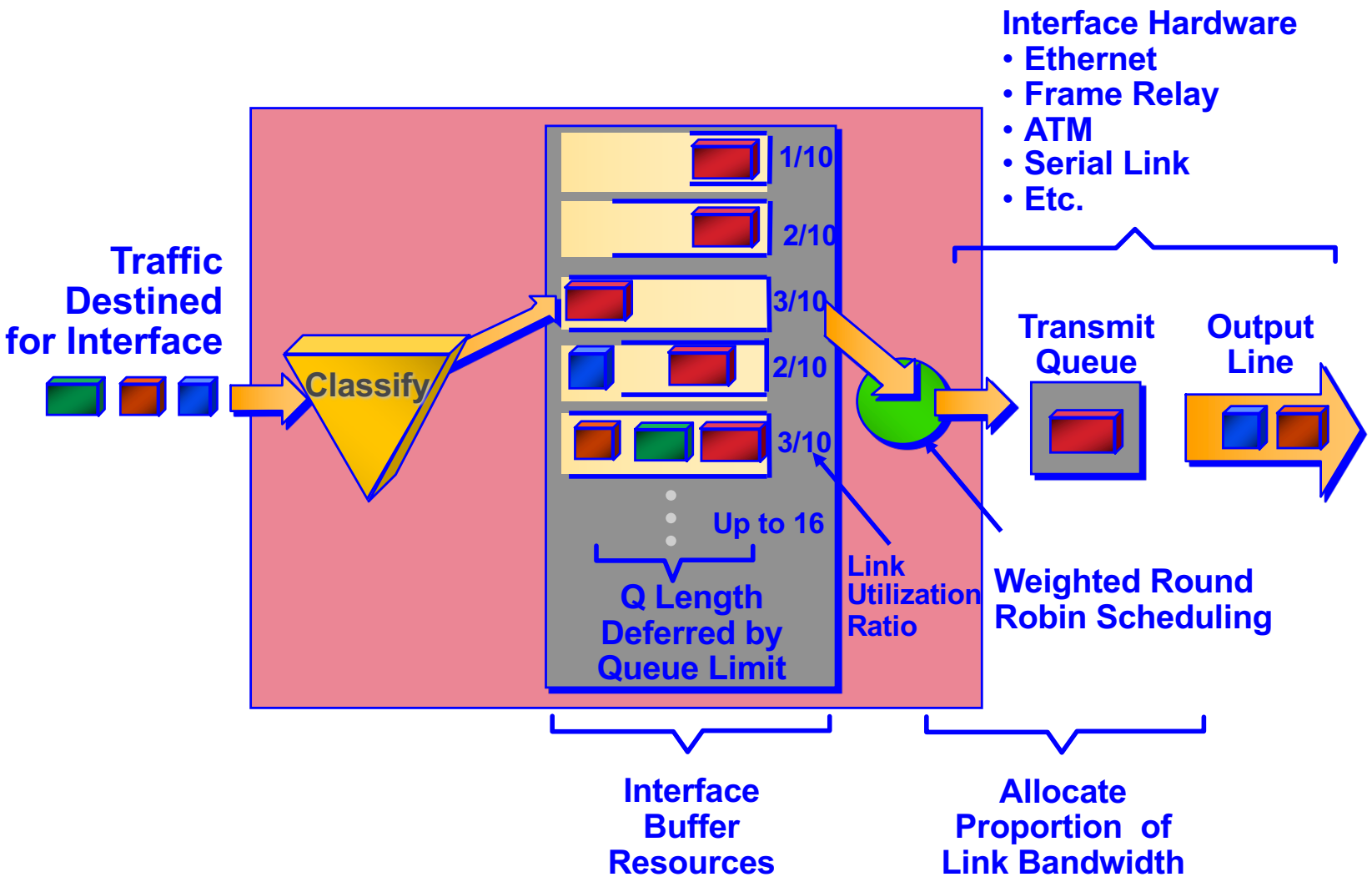


Priority Queuing (PQ) : problems

+ No upper limit for higher class priority

+ Starvation for lower classes

Weighted Round Robin (WRR)



Weighted Round Robin (WRR)

+ Weighted round-robin

- Diffeerent weights w_i per queue
- The j_{th} queue can transmit j packets per cycle
- Cycle length = $\Sigma\ w_j$

+ Problems

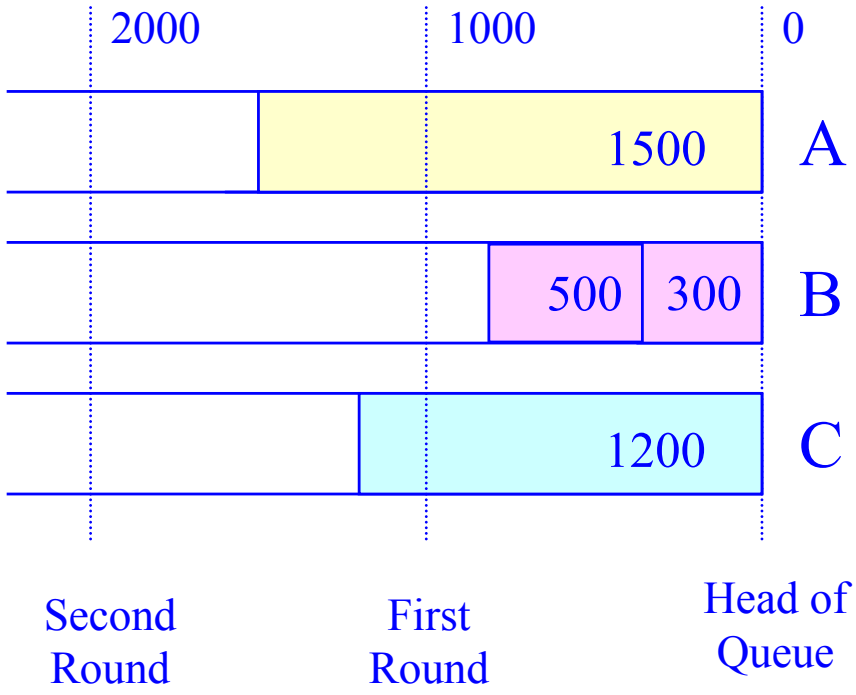
- **Variable dimension packets**
  - Unpredictable bandwidth usage
  - Unpredictable delays

Deficit Round Robin (DRR)

+ Each queue has a deficit counter that stores the number of credits (in bytes) and has a initial value zero
+ We define a “quantum” (bytes) that can be retrieved from the queue before passing to the next one
+ Each time we select a queue the deficit counter is incremented by a quantum value
+ For each packet in the queue head
  - If dimension L <= deficit counter
    - Packet transmitted
    - Deficit counter decremented
    - Next packet
  - Else
    - Next queue
+ If no packet in the queue → deficit counter = 0 (fairness)
+ Easy implementation
+ Different service class <-> different quantum values per queue (WDDR)
  - The bigger the quantum, the bigger is the bandwidth percentage associated to a class

Deficit Round Robin (DRR)

Quantum size : 1000 byte



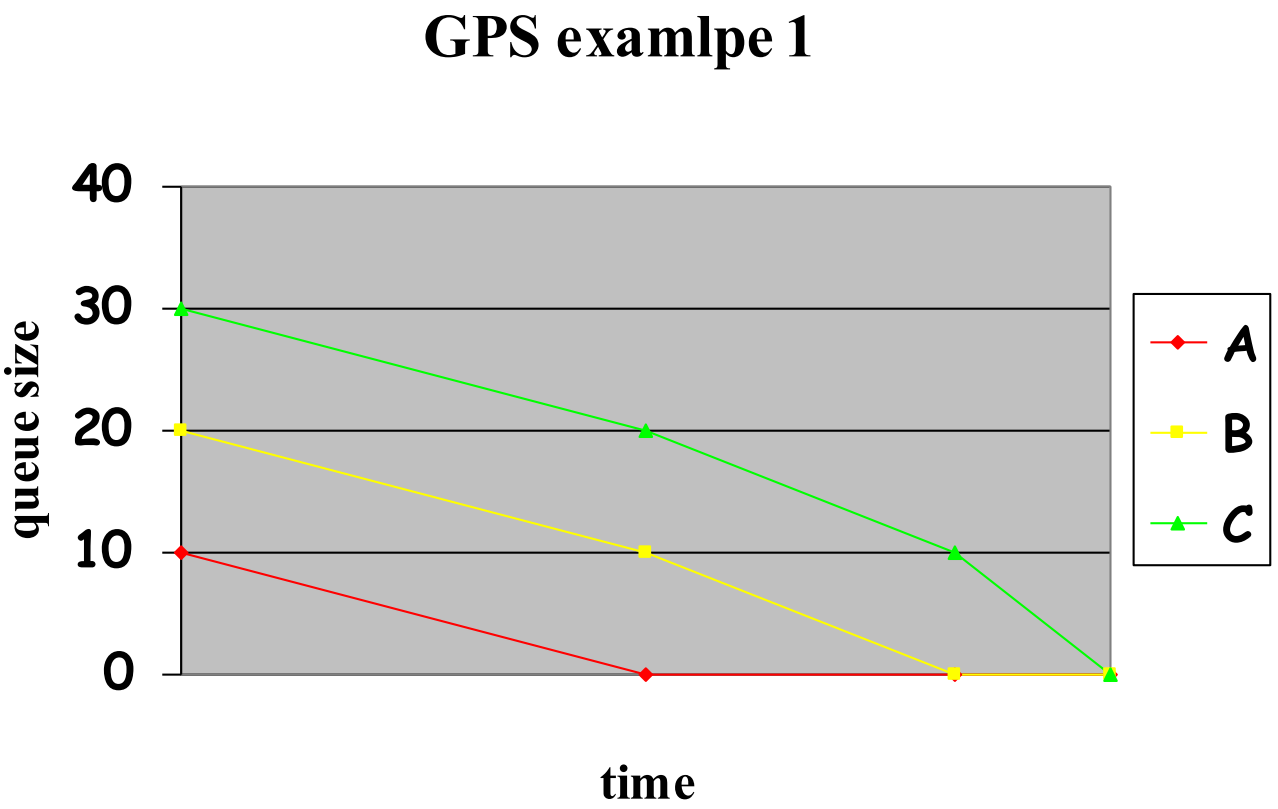
+ **1st Round**
  - **A's count : 1000**
  - **B's count : 200 (served twice)**
  - **C's count : 1000**
+ **2nd Round**
  - **A's count : 500 (served)**
  - **B's count : 0**
  - **C's count : 800 (served)**

Generalized Process Sharing (GPS)

+ **Ideal methodology**
  - **Assume we can send 1 single bit per packet**
  - **Round robin between queues (on a per bit basis)**
+ **Fair bandwidth allocation**
+ **GPS can't be implemented (tipically used as a benchmark)**
+ **By varying the number of bits transmitted per cycle we obtain different service classes**

$$Capacità_j(t) = Capacita\_linea * \frac{w_j}{\sum_{k \in ACTIVE(t)} w_k}$$

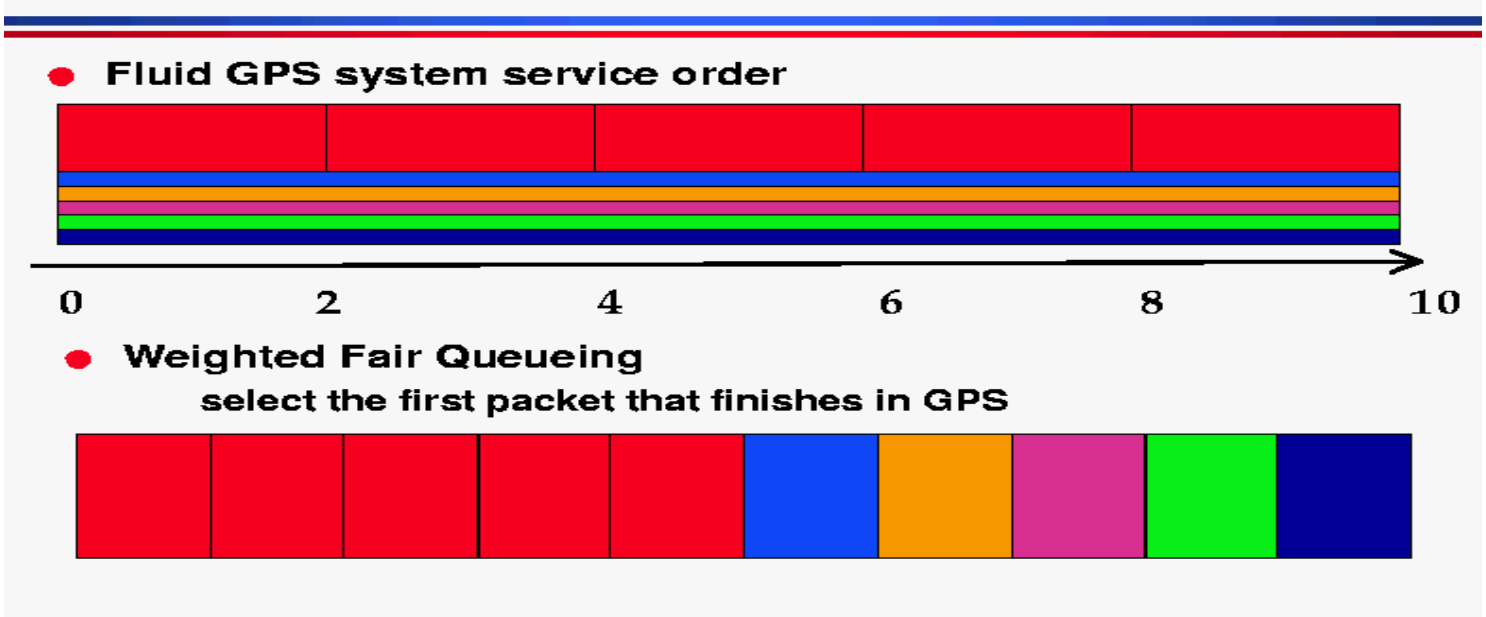
Generalized Process Sharing (GPS)



3 classes. Packets (dimension 10, 20, 30) arrive at time 0. All queues have 1 bit weight

Weighted Fair Queueing

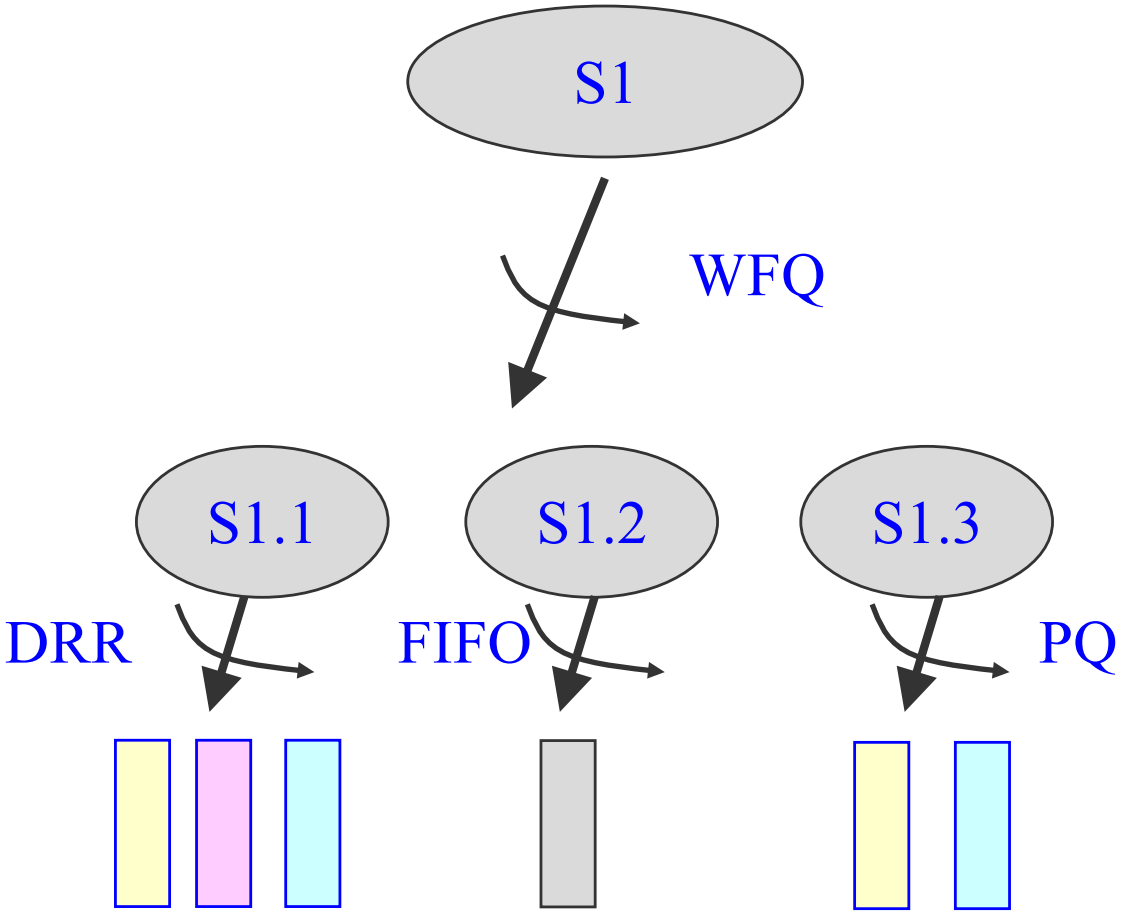
- GPS not implementable… let's try to emulate it
- WFQ simply sort the packets in the different queues by the expected virtual departure time of the last bit in the case of GPS
- High computation load (re-sort for every new packet)



Hierarchical scheduling

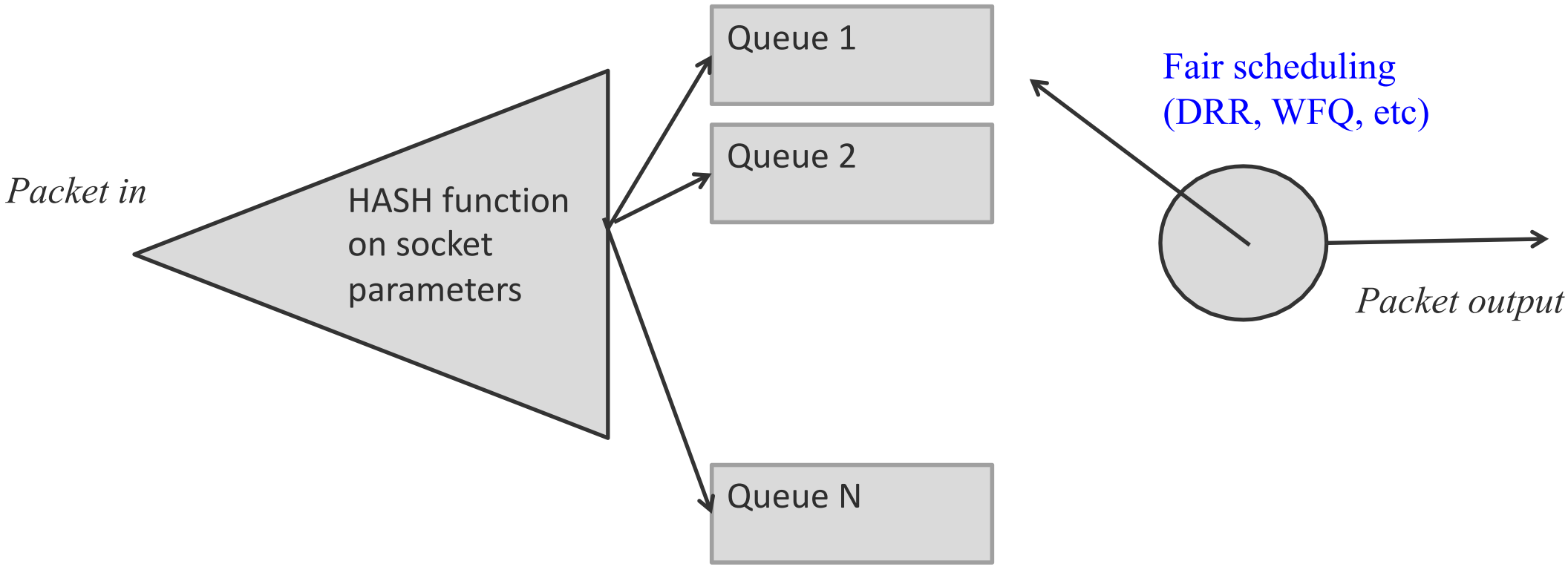
+ **Schedulers can be arranged in a hierarchical way**

+ **The low level schedulers get a packet from the queue when the higher level schedule takes a packet from them**



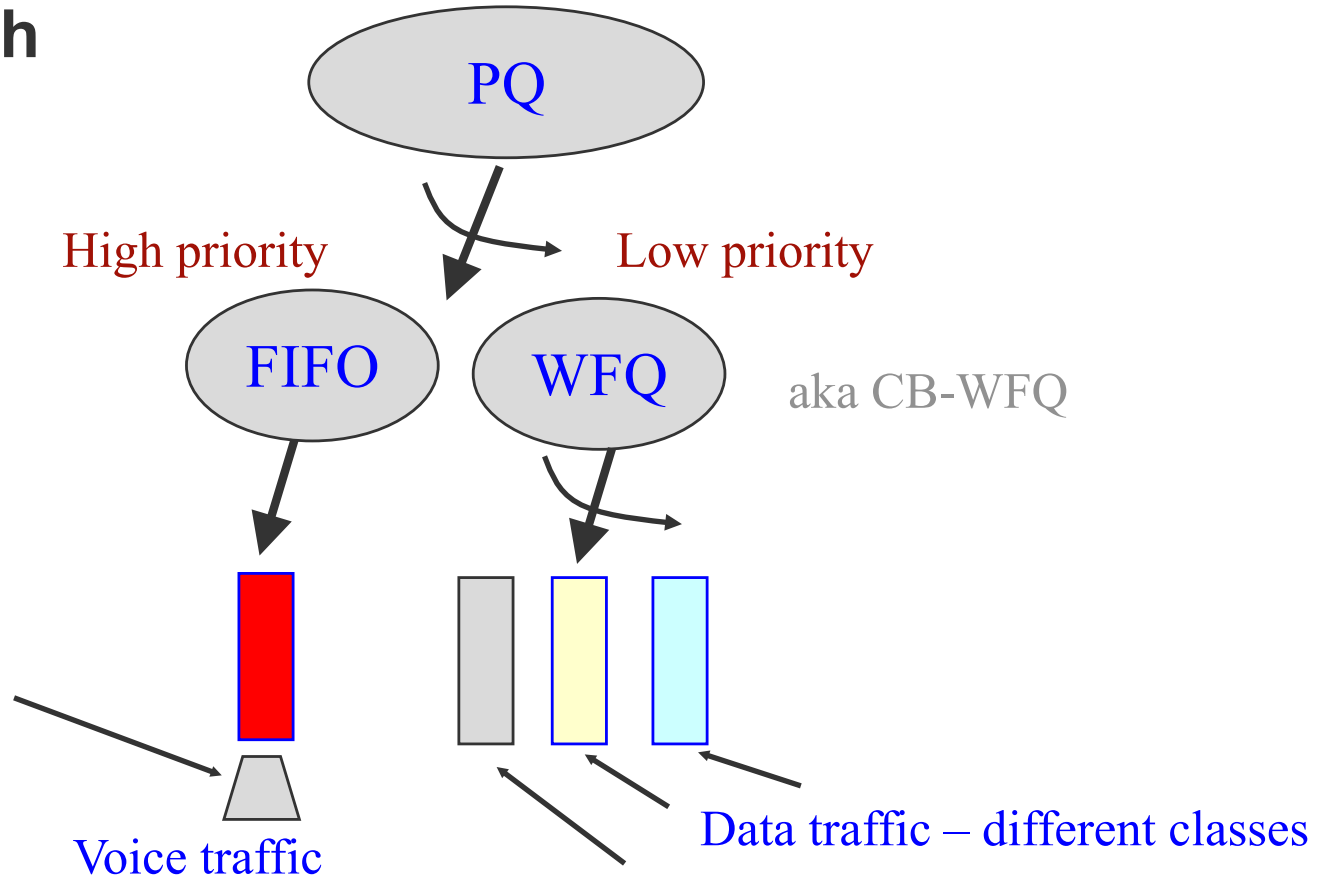
Flow fair queuing

+ **Fairness at a flow basis**
+ **Linux SFQ, Cisco fair-queue**



Cisco - Low Latency Queuing (LLQ)

+ **Scheduler implemented in different Cisco routers**
+ **High priority to real time flows**
+ **Class based queing for the remaining of the bandwidth**



Classifier

+ Packet classification

- **Inspects the packet fields and decides which queue according to specific rules**
- **Usually done with packet header fields**
  - Multi Field (MF) Classification:
  - Behavior Aggregate (BA) Classification
  - Interface based Classification

Policer and Shaper

+ **Traffic control mechanisms**

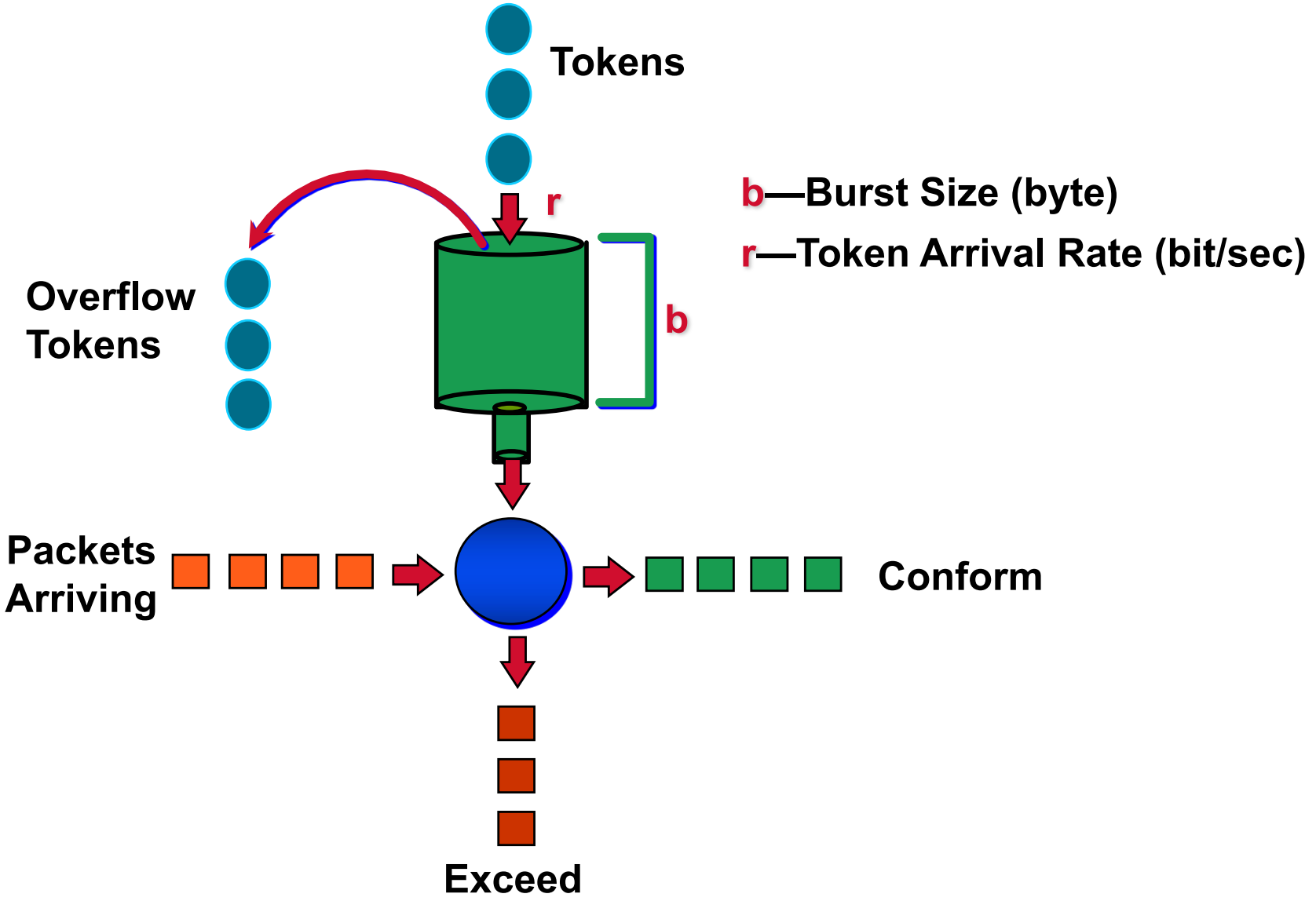
+ **Limit the amount of traffic from a given source according to specific traffic profiles (that depend on the negotiated QoS parameters)**

+ **Traffic profiles are often defined in terms of parameters of a rate-limiting tool named Token Bucket, that can be used as a policer or a shaper**

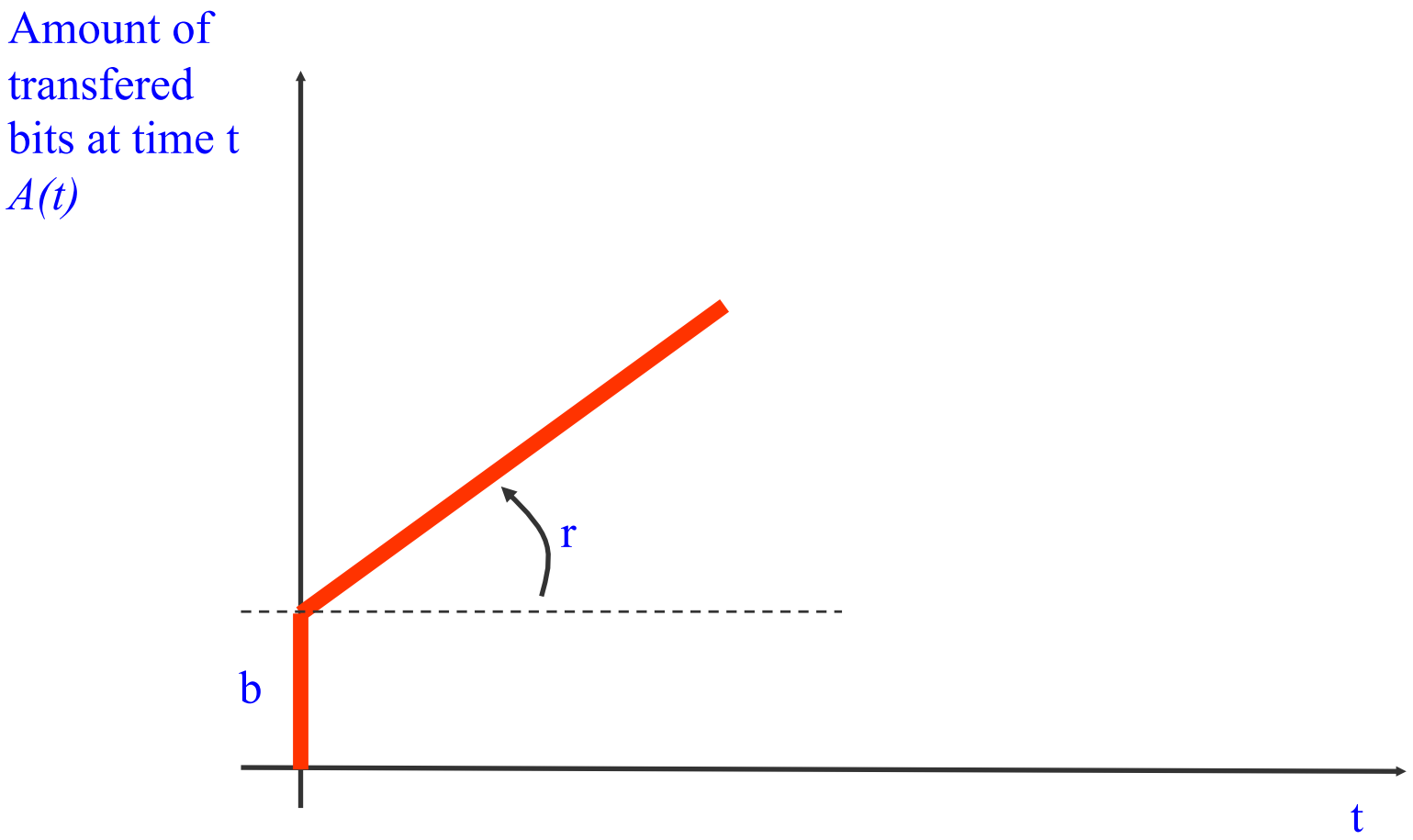
+ **Shaper:**
  - **Delay packets that are not conform to the traffic profile**
  - **Ideal shaper: Token bucket with infinite buffer**

+ **Policer:**
  - **Drop or mark non conform packet**
  - **Can be implemented as a token bucket with no queue**

Token Bucket



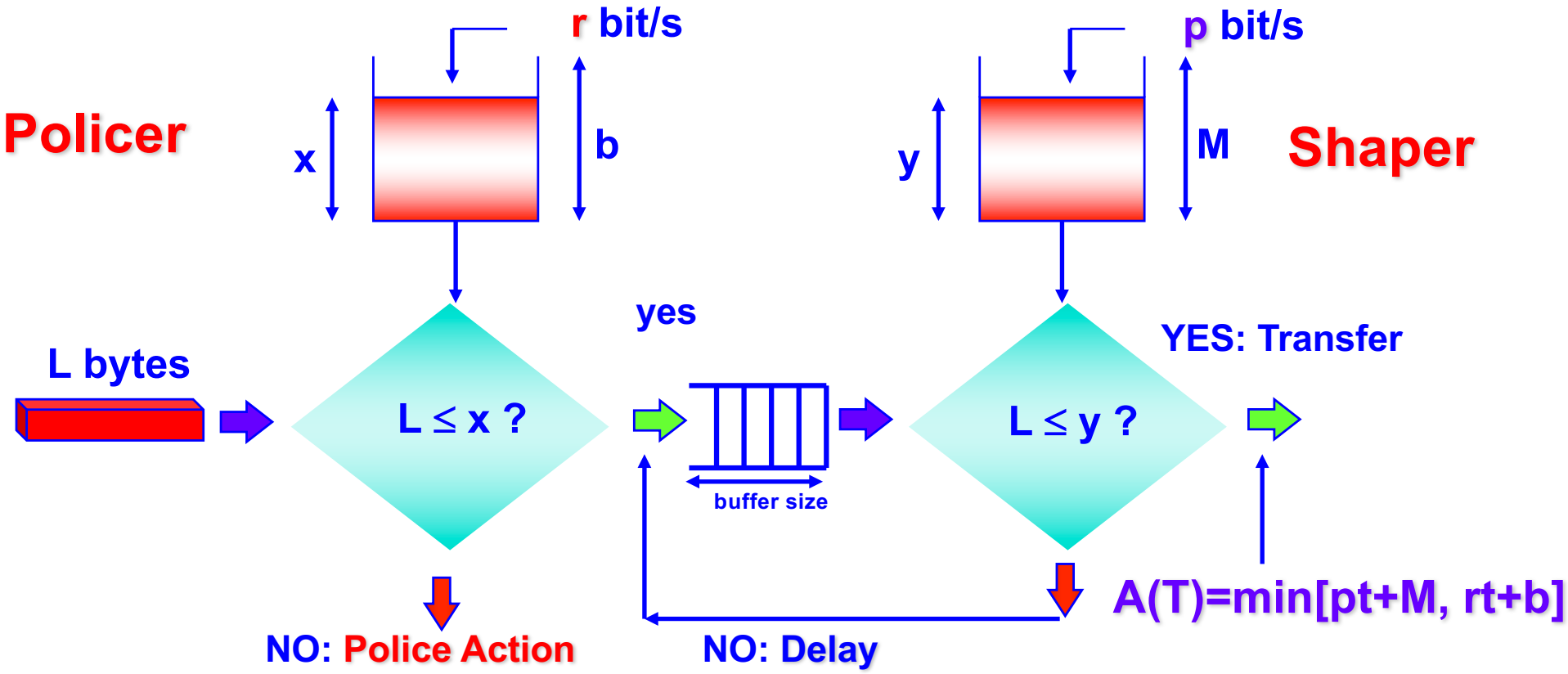
Traffic mask post Token Bucket



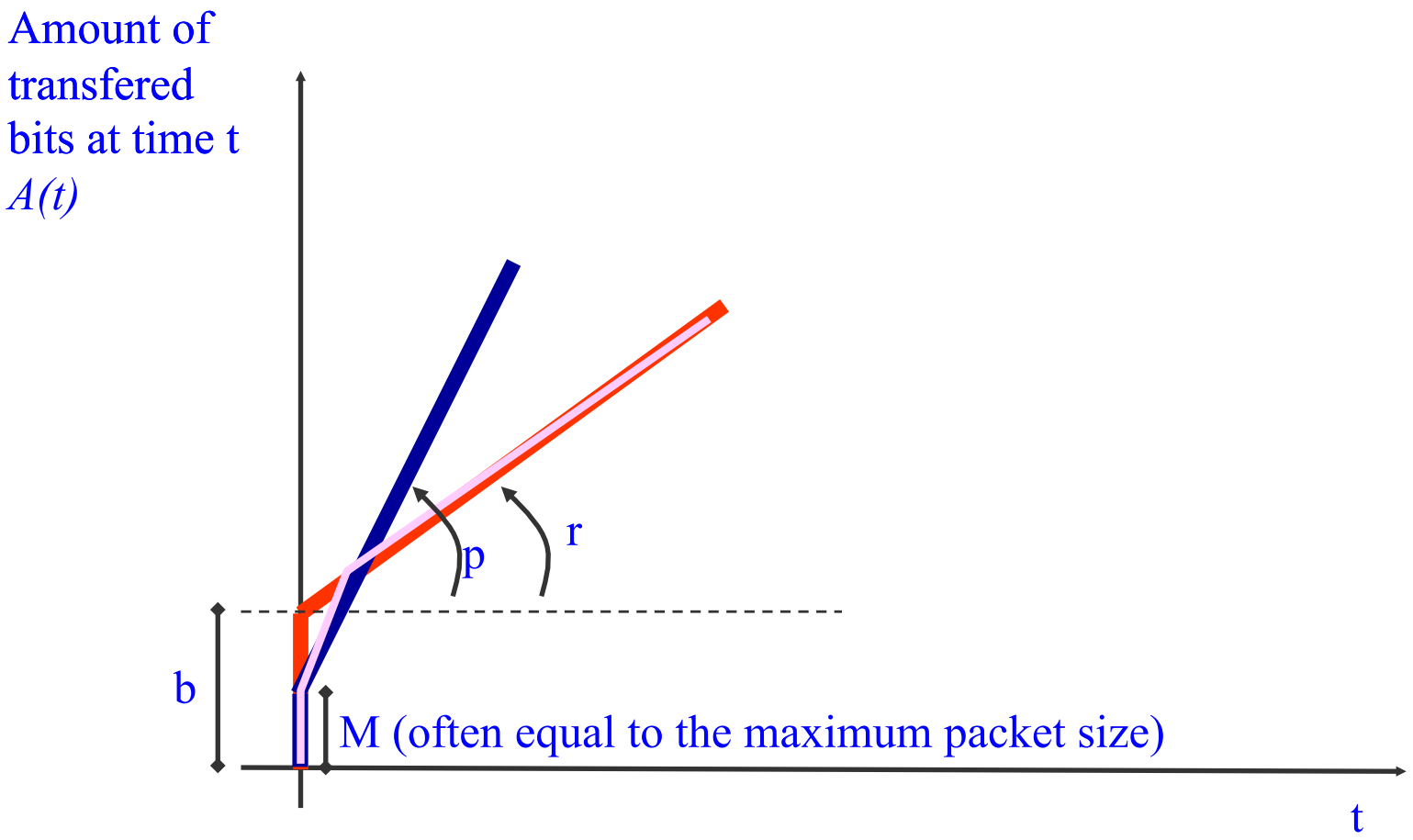
Dual Token Bucket

+ First Token bucket control the average

+ Second Token bucket control the maximum duration of transmission at peak rate



Traffic mask post Dual Token Bucket



Cisco QoS tools

Cisco QoS tools

- Access control list: traffic classification based on layer 3/4 matching criteria and access control action
- Class-map: defines traffic-class using layers 3/4/7 matching criteria
- Policy-map: defines the QoS policy action for a traffic class
- Service-policy: enforces a policy on an input/output interface

Access Control List (ACL)

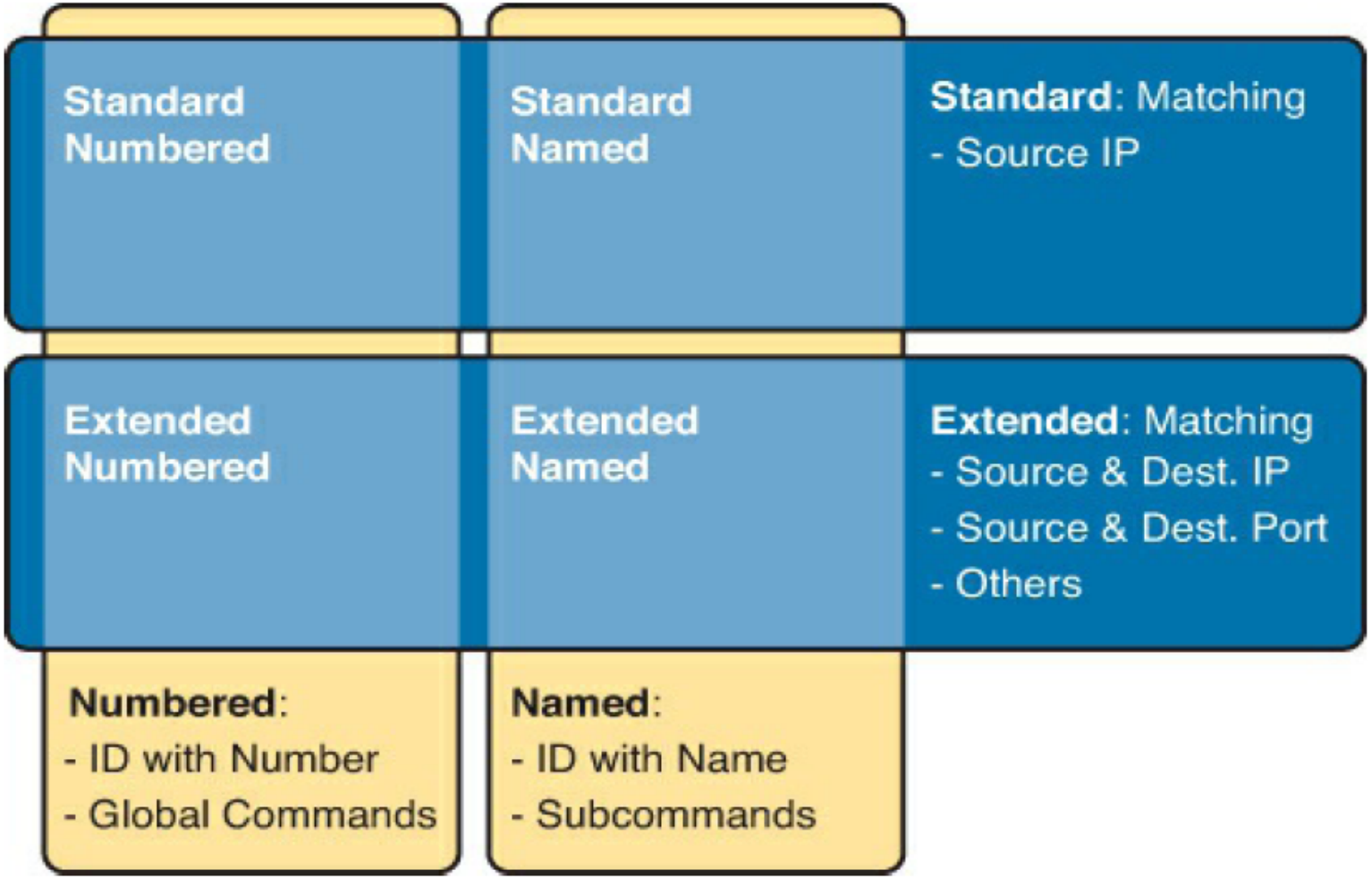
- Stack of match-action
- Match = conditions on layer 3/4 header fields
- Action = permit or deny
- E.g. FTP traffic from 172.16.3.10 can not access any FTP server in subnet 172.16.1.0/24. Other traffic is allowed
  - R1(config)# access-list 101 deny tcp host 172.16.3.10 172.16.1.0 0.0.0.255 eq ftp
  - R1(config)# access-list 101 permit ip any any
- First match-action inserted, first controlled
- Exit at first match
- None matching = deny
- ACL could be used also for rate-limiting

Access Control List (ACL)

http://www.cisco.com/c/en/us/support/docs/security/ios-firewall/23602-confaccesslists.html#acl

1-99

100-199



Standard ACL

- **Standard ACLs are the oldest type of ACL. They date back to as early as Cisco IOS Software Release 8.3**
- **Standard ACLs control traffic by the comparison of the source address of the IP packets to the addresses configured in the ACL**
- **Syntax:** `access-list access-list-number {permit|deny} {host|source source-wildcard|any}`
- **The access-list-number can be anything from 1 to 99**
- **Example**

  access-list 1 permit 10.1.1.0 0.0.0.255

  access-list 2 deny 10.1.1.125

Extended ACL

- Extended ACLs were introduced in Cisco IOS Software Release 8.3.
- Extended ACLs control traffic by the comparison of the source and destination addresses of the IP packets to the addresses configured in the ACL, as well as other per protocol specific fields (ports, ICMP codes, etc..)

IP

```
access-list access-list-number
     [dynamic dynamic-name [timeout minutes]]
     {deny|permit} protocol source source-wildcard destination destination-wildcard [precedence
     [tos tos] [log|log-input] [time-range time-range-name]
```

ICMP

```
access-list access-list-number
     [dynamic dynamic-name [timeout minutes]]
     {deny|permit} icmp source source-wildcard destination destination-wildcard
     [icmp-type [icmp-code] |icmp-message]
     [precedence precedence] [tos tos] [log|log-input]
     [time-range time-range-name]
```

Extended ACL

TCP

```
access-list access-list-number
      [dynamic dynamic-name [timeout minutes]]
      {deny|permit} tcp source source-wildcard [operator [port]]
      destination destination-wildcard [operator [port]]
      [established] [precedence precedence] [tos tos]
      [log|log-input] [time-range time-range-name]
```

UDP

```
access-list access-list-number
      [dynamic dynamic-name [timeout minutes]]
      {deny|permit} udp source source-wildcard [operator [port]]
      destination destination-wildcard [operator [port]]
      [precedence precedence] [tos tos] [log|log-input]
      [time-range time-range-name]
```

Access Control List (ACL)

+ ACL can be applied to the interface (in or out), for firewalling purposes
  - R1(config)#interface f0/0
  - R1(config-if)#ip access-group 101 out
+ If not applied to an interface, access-list can be used within a class-map for classification purposes
  - R1(config)#class-map myclass
  - R1(config-cmap)#match access-group 101
+ WARNING: By default, there is an implicit deny all clause at the end of every ACL. Anything that is not explicitly permitted is denied.
+ Commonly configured ACL
  - http://www.cisco.com/c/en/us/support/docs/ip/access-lists/26448-ACLsamples.html

Class-map

+ **Define a traffic class and layer 3/4/7 match criteria (e.g. IP dscp)**
  - **R1(config)#class-map myclass**
  - **R1(config-cmap)#match ip dscp 20**

+ **Several match criteria including Network-Based Application Recognition (Deep Packet Inspection - DPI)**

access-group	Access group
any	Any packets
class-map	Class map
cos	IEEE 802.1Q/ISL class of service/user priority values
destination-address	Destination address
discard-class	Discard behavior identifier
dscp	Match DSCP in IP(v4) and IPv6 packets
flow	based QoS parameters
input-interface	Select an input interface to match
ip	IP specific values
mpls	Multi Protocol Label Switching specific values
not	Negate this match result
packet	Layer 3 Packet length
precedence	Match Precedence in IP(v4) and IPv6 packets
protocol	Protocol
…….	

+ **May import ACL and other Class-map**

– Class-default map any packet

http://www.cisco.com/c/en/us/td/docs/app_ntwk_services/data_center_app_services/ace_appliances/vA1_7_/configuration/administration/guide/admgd/mapolcy.html#wp1318826

Class-map - Examples

class-map match-all L4_SOURCE_IP_CLASS

match source-address 192.168.10.1 255.255.255.0

match port tcp eq 23

class-map type management match-any MGMT-ACCESS_CLASS

match protocol icmp source-address 172.16.10.0 255.255.255.0

match protocol ssh source-address 192.168.10.1 255.255.255.0

Policy-map

+ Define the action on a set of traffic classes
+ Possible actions are Shaping, Policing, Scheduling (LLQ, CBWFQ)
+ E.g. shape traffic myclass at 2Mbps
  - R1(config)#policy-map myshaper
  - R1(config-pmap)#class myclass
  - R1(config-pmap-c)#shape average 2000000

Policy-map: policing

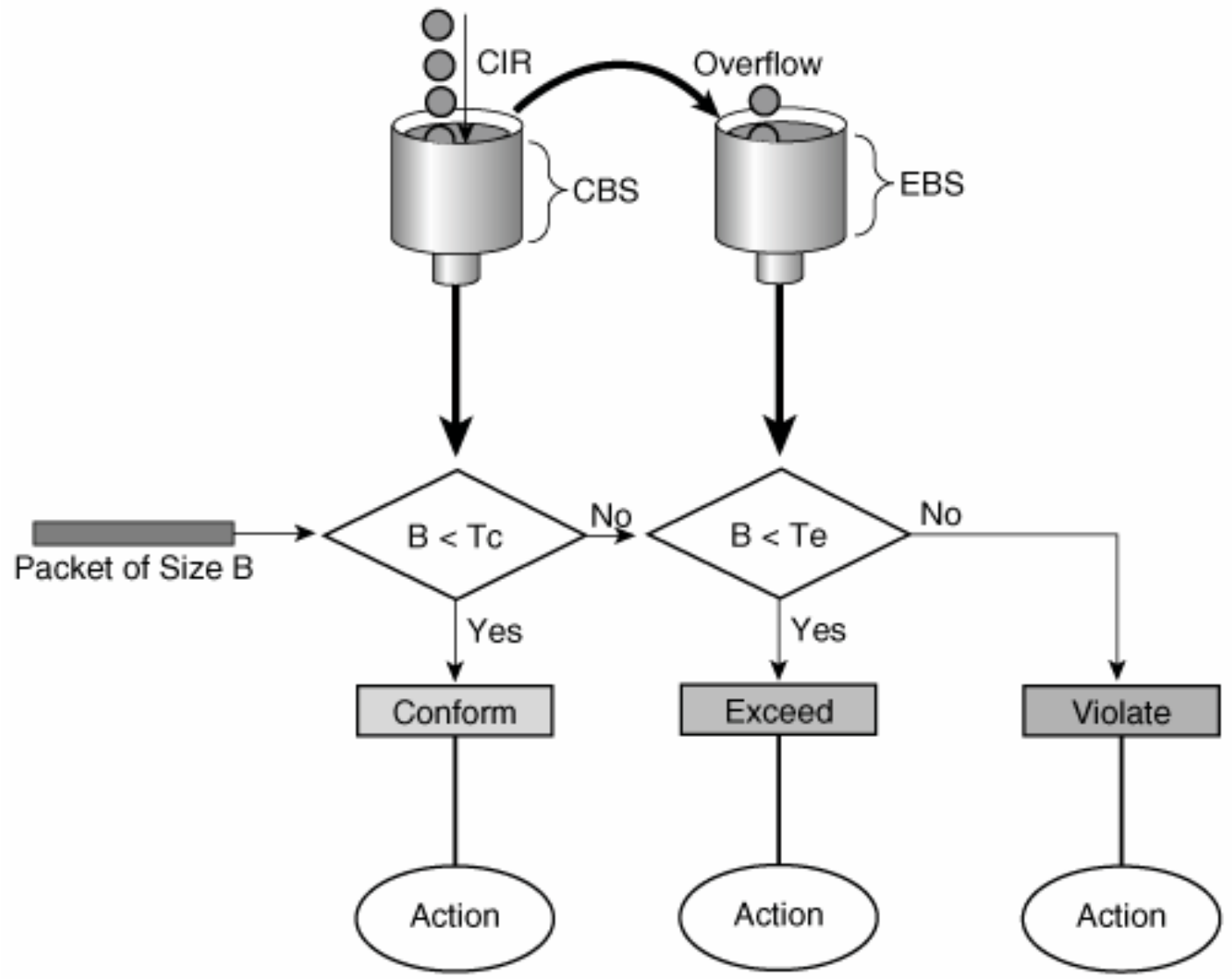
+ **Applicable on traffic class at input or output interfaces**

+ **Bandwidth management through rate limiting –**
  - **control the maximum rate of traffic sent or received on an interface.**

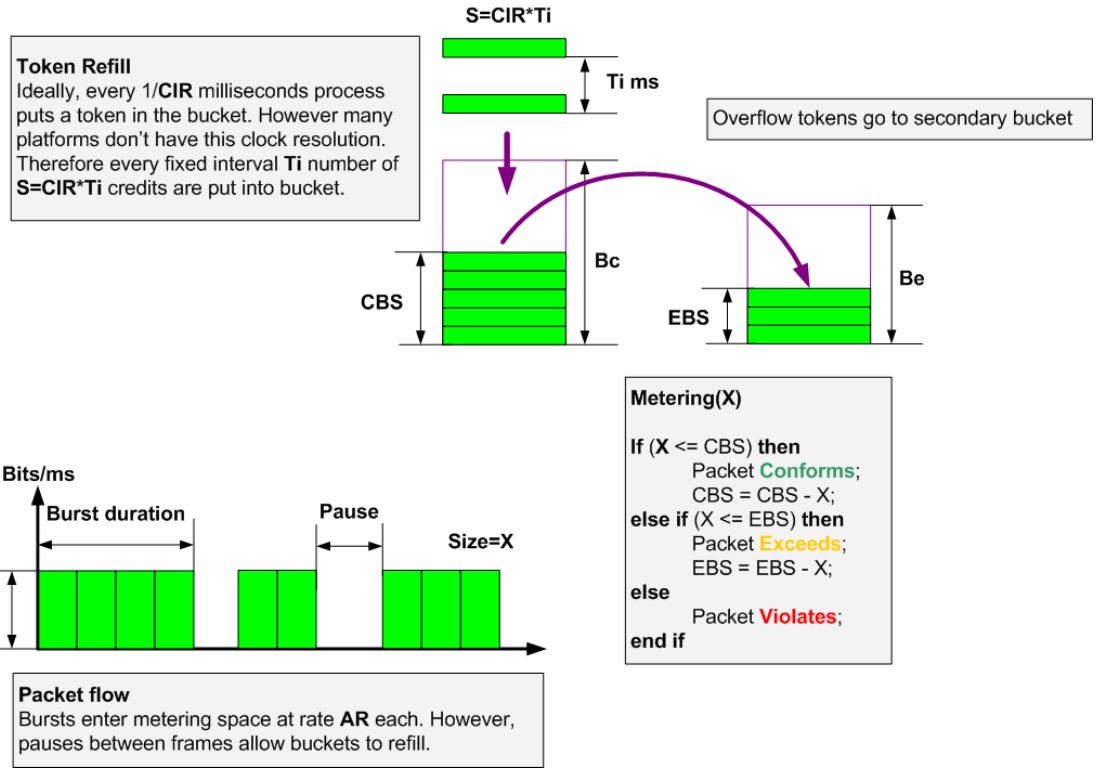
+ **Often configured on interfaces at the edge of a network to limit traffic into or out of the network.**

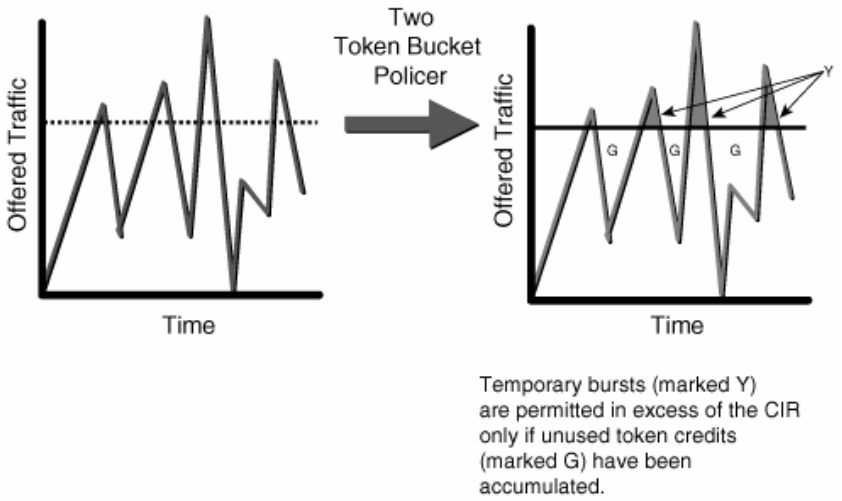
+ **Traffic that falls within the rate parameters is sent, whereas traffic that exceeds the parameters is dropped, or sent with a different priority.**
  - **Packet marking through IP precedence, QoS group, or DSCP value setting**
  - **QoS group = packet tag only used within the router**

Policing: two token buckets, single rate

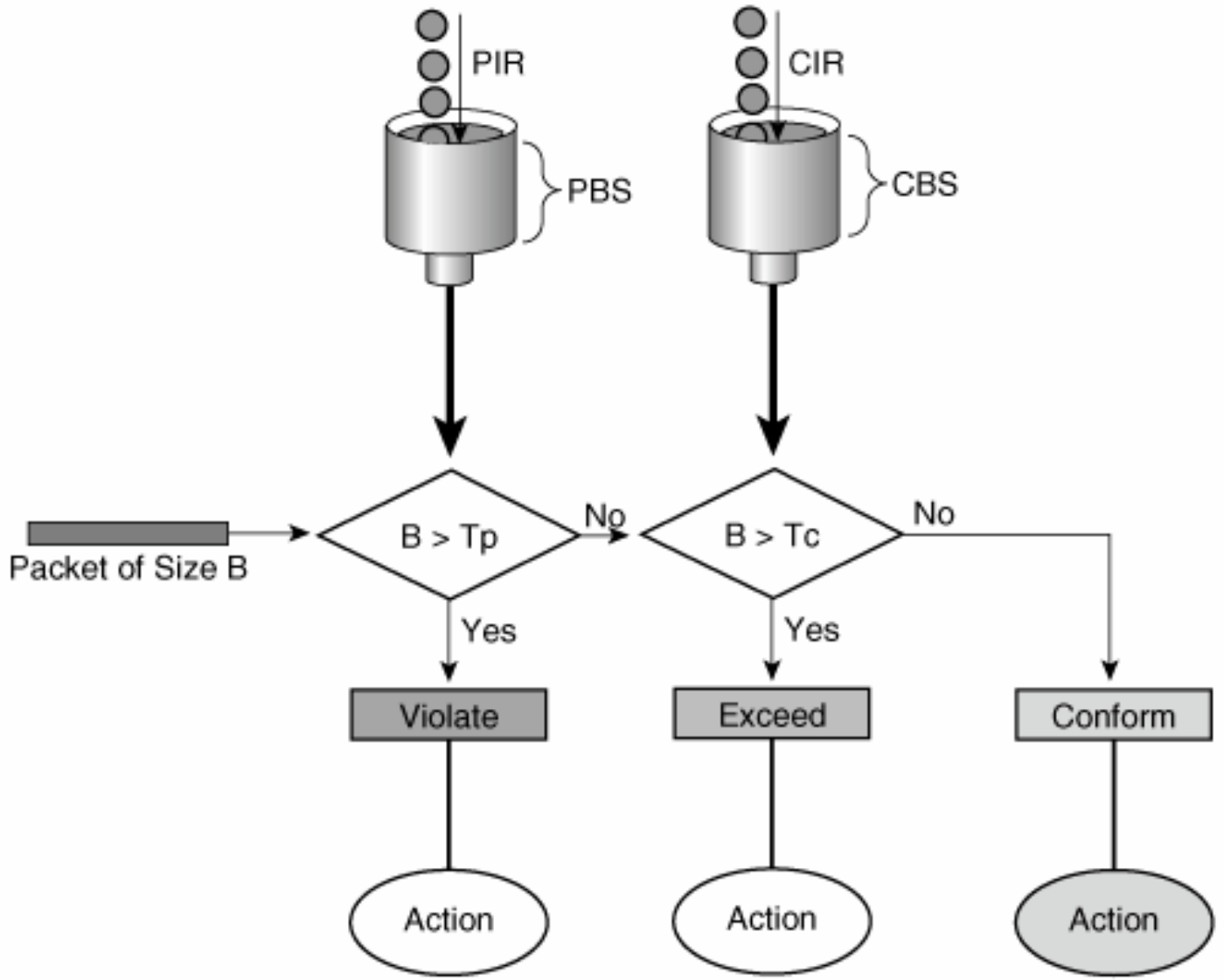


Policing: two token buckets, single rate

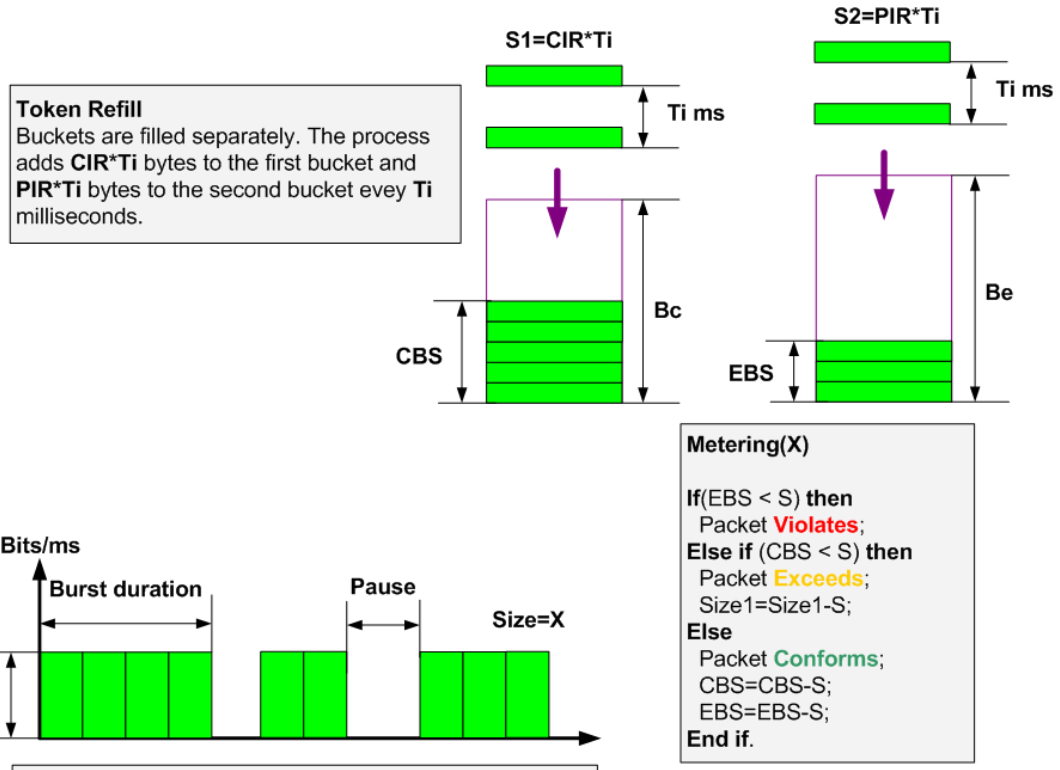


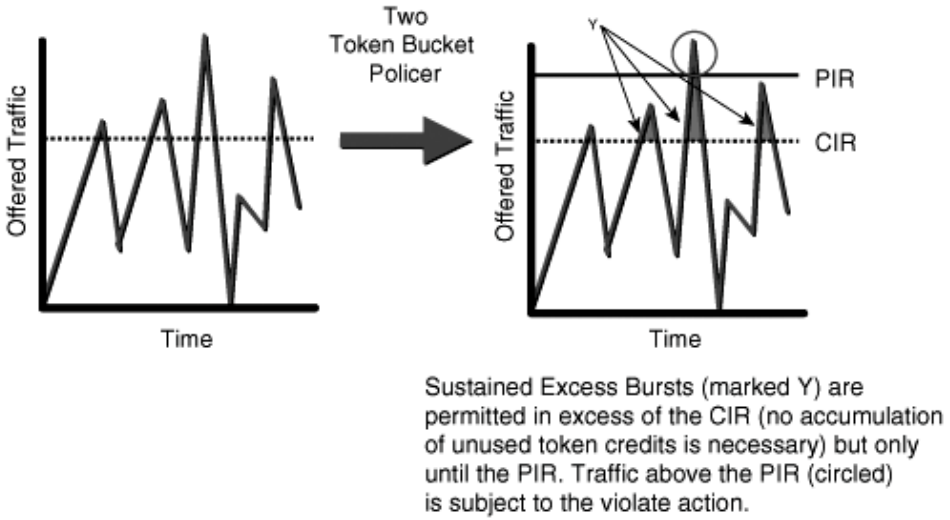


Policing: two token buckets, dual rate



Policing: two token buckets, dual rate





Sustained Excess Bursts (marked Y) are permitted in excess of the CIR (no accumulation of unused token credits is necessary) but only until the PIR. Traffic above the PIR (circled) is subject to the violate action.

Policy-map: policing

+ **E.g. Police traffic of class myclass at 8kbps, normal burst size (Bc) at 2000 bytes, and the excess burst size (Be) at 4000 bytes. Packets that conform are transmitted, packets that exceed are assigned a QoS group value of 4 and are transmitted, and packets that violate are dropped.**
  - **R1(config)#policy-map mypolicer**
  - **R1(config-pmap)#class myclass**
  - **R1(config-pmap)#police 8000 2000 4000 conform-action transmit exceed-action set-qos-transmit 4 violate-action drop**

Policy-map: shaping

+ Like policing but queue packets that would be dropped and then schedules their transmission
+ Queuing is an outbound concept; packets going out an interface get queued and can be shaped
+ Shape is only applied on output interface
+ E.g. shape traffic of class myclass to 2Mbps
  - R1(config)#policy-map myshaper
  - R1(config-pmap)#class myclass
  - R1(config-pmap-c)#shape average 2000000

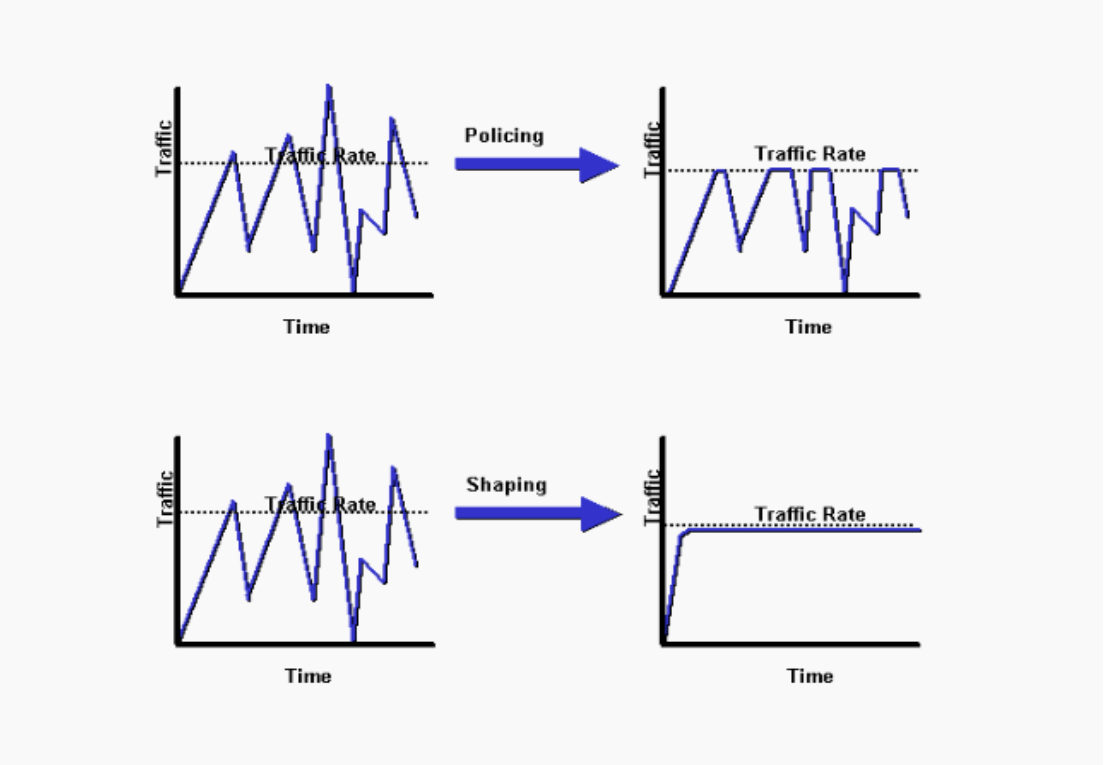
Policy-map: policing/shaping

Shaping

- Pro
  - Buffers excess packets, therefore, less likely to drop excess packets
  - Buffers packets up to the length of the queue. Drops may occur if excess traffic is sustained at a high rate
  - Typically avoids retransmissions due to dropped packets
- Cons
  - Can introduce delay resulting from queuing (especially when deep queues are used)

Policing

- Pro
  - Controls the output rate through packet drops
  - Avoids delays resulting from queuing
- Cons
  - Drops excess packets (when configured), throttles TCP window sizes, and reduces the overall output rate of affected traffic streams.
  - Overly aggressive burst sizes can lead to excess packet drops and throttle the overall output rate (particularly with TCP-based flows).



http://www.cisco.com/c/en/us/support/docs/quality-of-service-qos/qos-policing/19645-policevsshape.html

Policy-map: policing/shaping

- Both shaping and policing use the token bucket metaphor
- Besides the absence/presence of a packet queue, a key difference between shaping and policing is the rate at which tokens are replenished
- Shaping increments the token bucket at timed intervals (Tc) using a bits per second (bps) value. A shaper uses the following formula:
  - Tc = Bc/CIR (in seconds)
- Policing adds tokens continuously to the bucket. Specifically, the token arrival rate is calculated as follows:
  - (time between packets* policer rate)/8 bits per byte
  - if the previous arrival of the packet was at t1 and the current time is t, the bucket is updated with t-t1 worth of bytes based on the token arrival rate

Policy-map: marking

+ Marking network traffic allows to set or modify the attributes for traffic (that is, packets) belonging to a specific class or category

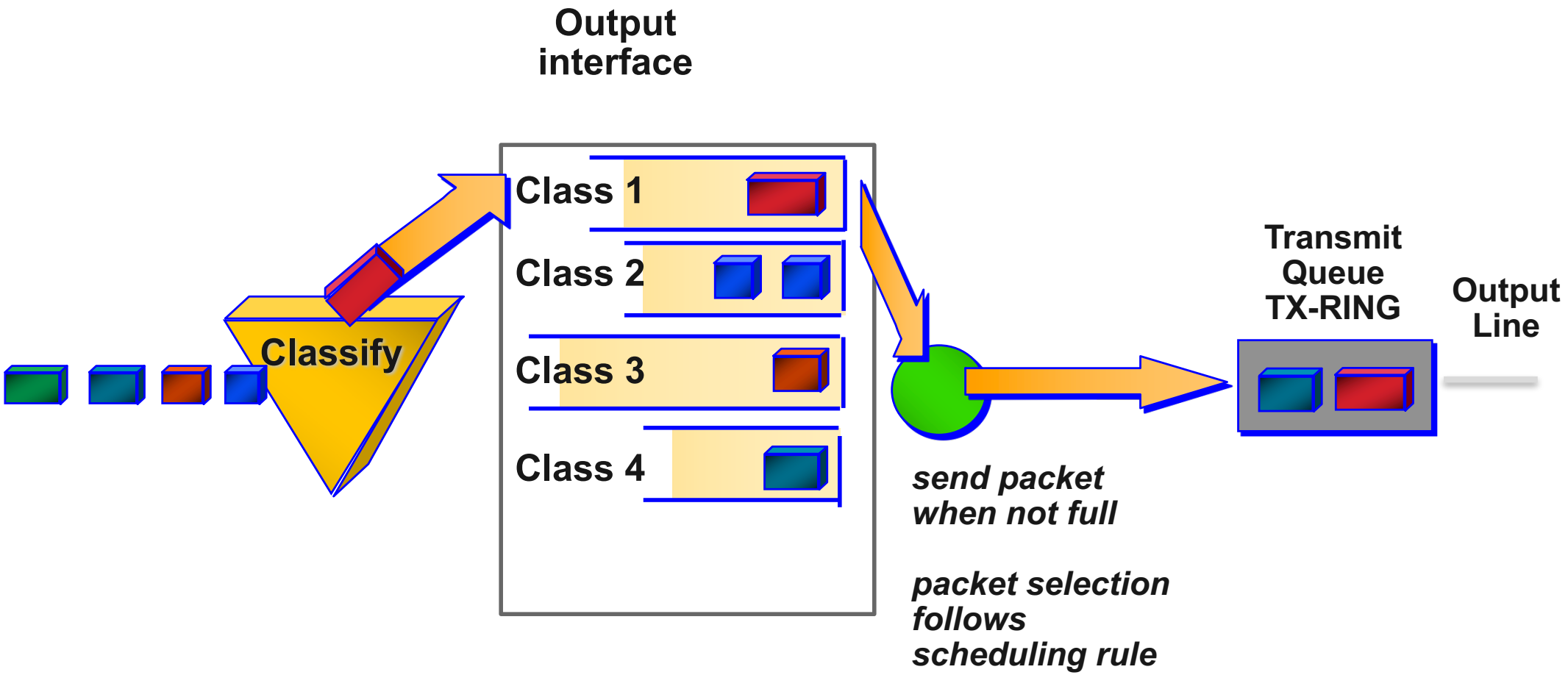
+ Often used to set the IP precedence or IP DSCP values for traffic entering a network.

+ Networking devices within the network can then use the newly marked IP precedence values to classify packets in a coarser way

+ E.g., mark traffic of classmap myclass with dscp 20
  - R1(config)#policy-map marker
  - R1(config-pmap)#class myclass
  - R1(config-pmap-c)#set dscp 20

+ Can be used also to mark MPLS EXP filed, or the QoS group

Policy-map: scheduling



Policy-map: scheduling

- After the packet is processed by a scheduler it is placed in the final FIFO queue, the TX-Ring.
- The TX-Ring provides a final queue for the physical FIFO prior to transmit onto the wire.
- Scheduler sends packet to TX-Ring until TX-Ring is full
- When congestion occurs at the egress interface i.e. the TX-Ring is full, the scheduler queues begin to buffer traffic based on their per class queue-limit allocations

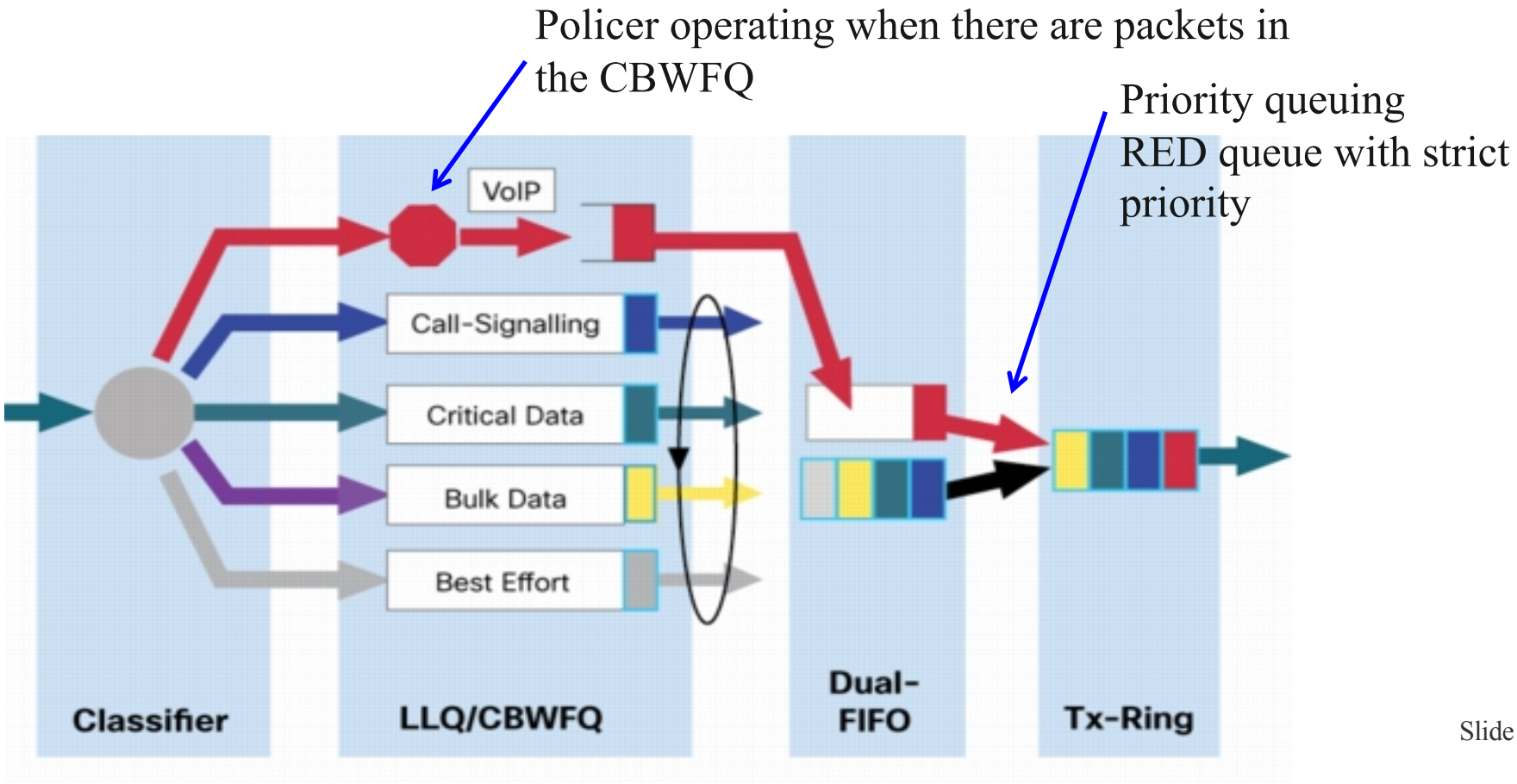
Policy-map: Scheduling CBWFQ

+ **Class Based WFQ**
  - When possible, send to TX-ring the queued packet with the lowest expected virtual departure time
+ **E.g. 2Mbps for myclass, 500kbps for remaining traffic**
  - R1(config)#policy-map cbwfq
  - R1(config-pmap)#class myclass
  - R1(config-pmap-c)#bandwidth 2000
  - R1(config-pmap-c)#exit
  - R1(config-pmap)#class class-default
  - R1(config-pmap-c)#bandwidth 500
  - R1(config-pmap-c)#exit
+ **Scheduling choice at the first match**

Policy-map: Scheduling LLQ

+ **Give strict priority to one (or more) traffic class so as limiting delay and jitter**

+ **Strict priority class used by interactive services, VoIP and video-conferencing**



Policy-map: Scheduling LLQ

+ **E.g. 3Mbps for voip class (priority), 2Mbps for myclass, 500kbps for remaining traffic**
  - **R1(config)#policy-map myllq**
  - **R1(config-pmap)#class voip**
  - **R1(config-pmap-c)#priority 3000**
  - **R1(config-pmap-c)#exit**
  - **R1(config-pmap)#class myclass**
  - **R1(config-pmap-c)#bandwidth 2000**
  - **R1(config-pmap-c)#exit**
  - **R1(config-pmap)#class class-default**
  - **R1(config-pmap-c)#bandwidth 500**
  - **R1(config-pmap-c)#exit**

Policy-map: Scheduling LLQ

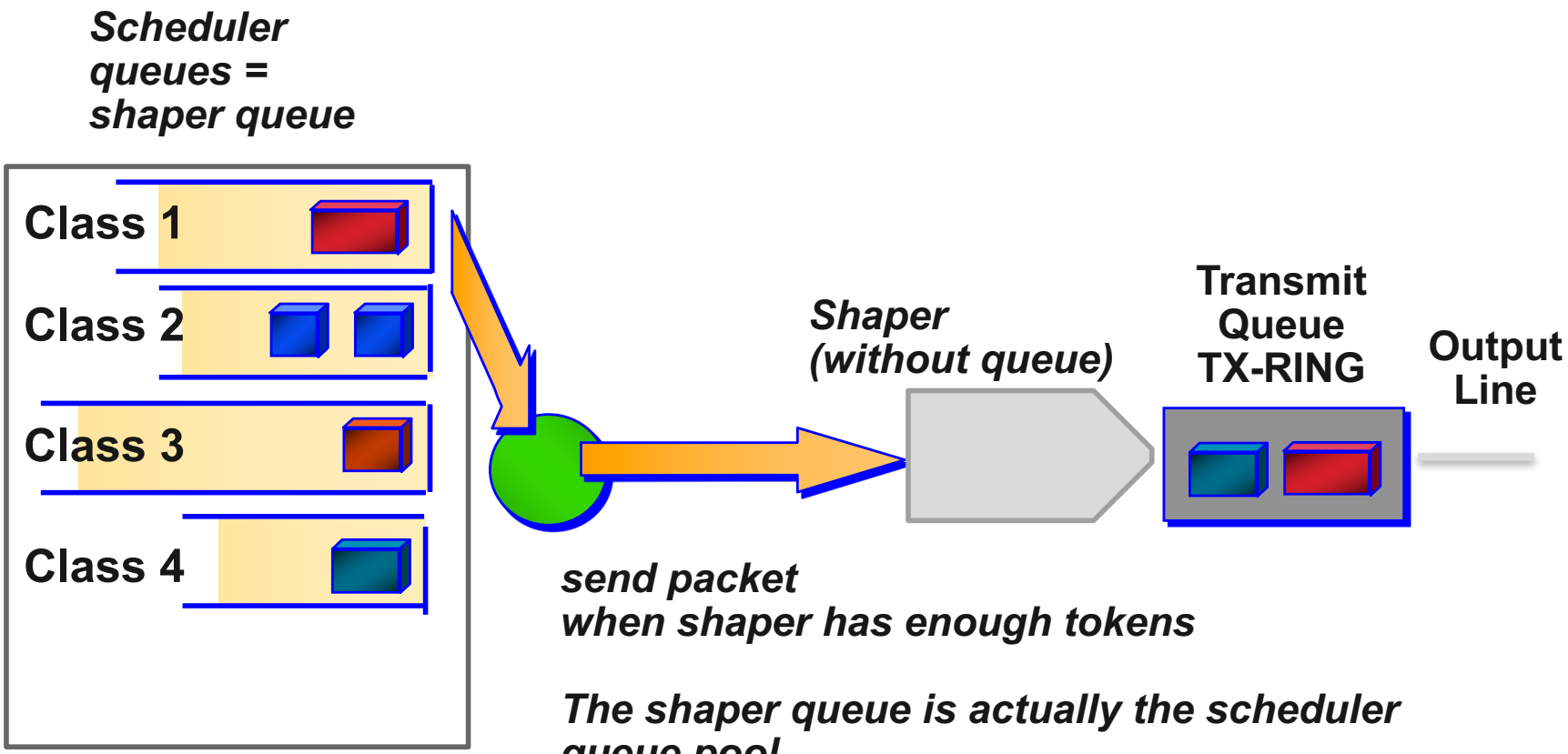
- When multiple classes within a single policy map are configured as priority classes, all traffic from these classes is queued to the same single, strict priority queue

- But they traverse different policers configured with the different priority bandwidth

Policy-map: Scheduling and Shaping

- Scheduler available bandwidth is the line rate
- But in a customer-provider network link the line rate may be higher than the contracted bandwidth.
- Traffic shaping allows customer to control the traffic going out on the link in order to ensure that the traffic conforms to policies contracted with the provider
- On the customer side QoS has to be applied on the shaped rate
- The shaper should drain packets from the scheduler rather than the TX-Ring

Policy-map: Scheduling and Shaping



send packet when shaper has enough tokens

The shaper queue is actually the scheduler queue pool

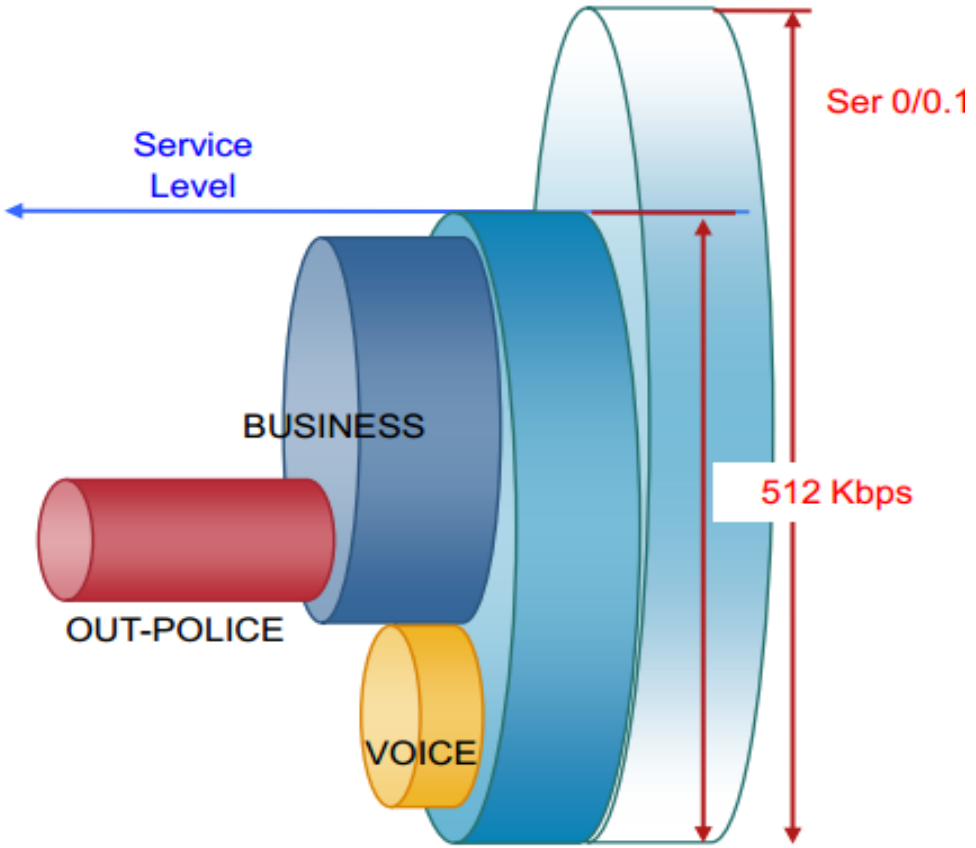
in case of LLQ, chose Tc of the shaper low as 10ms to avoid too much additional delay for priority class

Hierarchical Scheduling

+ **Requires shaping at the top level**

+ **Then scheduling level can be stacked embedding a n-level policy-map a (n-1)-level class**

```
policy-map GRANDCHILD
class BUSINESS-NO-MGMT
 police cir 128000
  conform-action transmit
  exceed-action set-frde-transmit
!
policy-map CHILD
class VOICE
 priority percent 25
class BUSINESS
 bandwidth remaining percent 66
 service-policy GRANDCHILD
!
policy-map PARENT
class class-default
 shape average 512000
 service-policy CHILD
```



Overall steps

- **Step 1 Define a traffic class by using the class-map command. A traffic class is used to classify traffic.**
- **Step 2 Create a traffic policy by using the policy-map command. A traffic policy (policy map) contains traffic classes and QoS features that will be applied to the traffic class.**
- **Step 3 Attach the traffic policy (policy map) to the interface by using the service-policy command**
  - **R1(config)#int f2/0**
  - **R1(config-if)#service-policy output myllq**

+ QOS architecture

QoS Architecture

+ Is the set of protocols, mechanisms and devices used to reserve and use bandwidth along a network path

+ We discuss:
  - Integrated Services Architecture (IntServ)
  - Differentiated Services Architecture (DiffServ)
  - Multi Protocol Label Switching (DiffServ-TE)

+ All of them are based on the concept of Collective pre-assignment (connection oriented) of resources

+ The first two are integrated in the IP layer; instead MPLS is another layer below IP

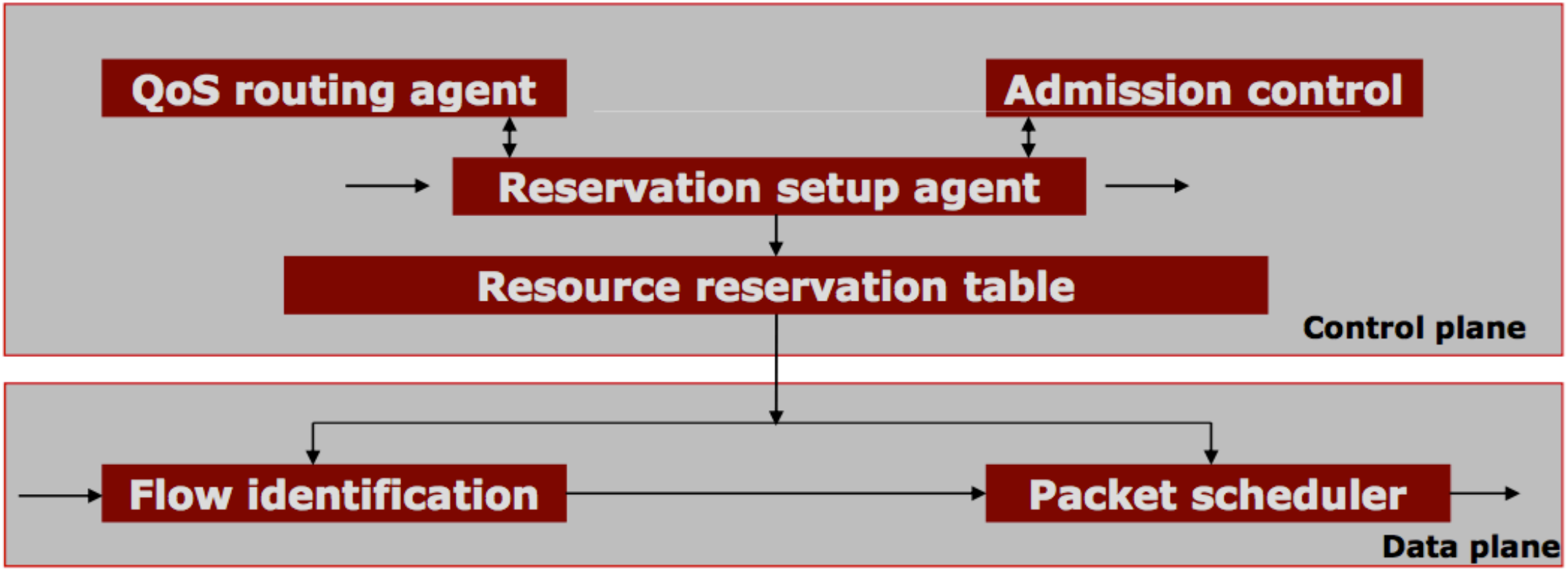
IntServ Architecture

+ **Network resources are provided according to an application's QoS request**

+ **Resource reservation on a flow-base**
  - **A distinguishable stream of related datagrams that results from a single user activity and requires the same QoS (e.g., a TCP connection or a UDP session)**
  - **Applications must be able to provide the flow specification**

+ **The Intserv architecture defines three major classes of service :**
  - **Guaranteed Service**
  - **Controlled Load**
  - **Best Effort**

IntServ Architecture



Call setup

+ **A flow requiring QoS guarantees must first be able to reserve sufficient resources at each network router on its source-to-destination path.**

+ **Call setup process requires the participation of each router on the path.**
  - **Determine the local resources required by the session**
  - **Consider the amounts of its resources that are already committed to other on-going flows**
  - **Determine whether it has sufficient resources to satisfy the per-hop QoS requirement of the flow at this router without violating QoS**

Call setup

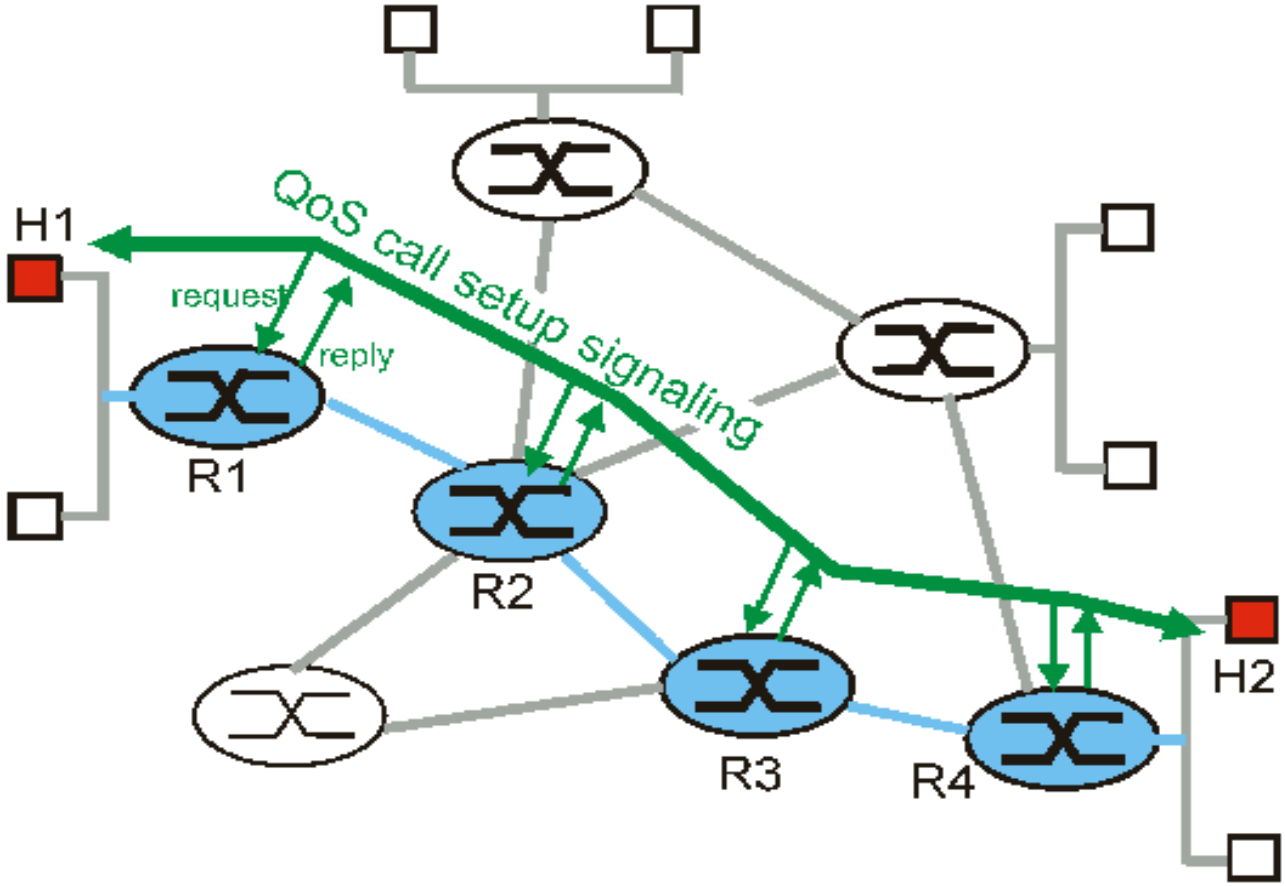


Figure 6.7-1: The call setup process

Traffic characterization and specification of the desired QoS

+ **Rspec (R for reserved)**
  - **Rspec defines the specific QoS being requested by a connection**
+ **Tspec (T for traffic)**
  - **Tspec characterizes the traffic the sender will be sending into the network, or the receiver will be receiving from the network**
  - **Traffic is characterized through the parameters of a dual token bucket**

Signaling for call setup

- A flow's Tspec and Rspec must be carried to the routers at which resources will be reserved for the flow.
- RSVP protocol is the signaling protocol.

Per-element call admission

+ **While receiving the Tspec and Rspec for a flow requesting a QoS guarantee, a router can determine whether or not it can admit the call.**

+ **The call admission decision depends on:**
  - **The traffic specification**
  - **The requested type of service**
  - **The existing resource commitments already made by the router to on-going sessions**

Per-element call admission (cont.)

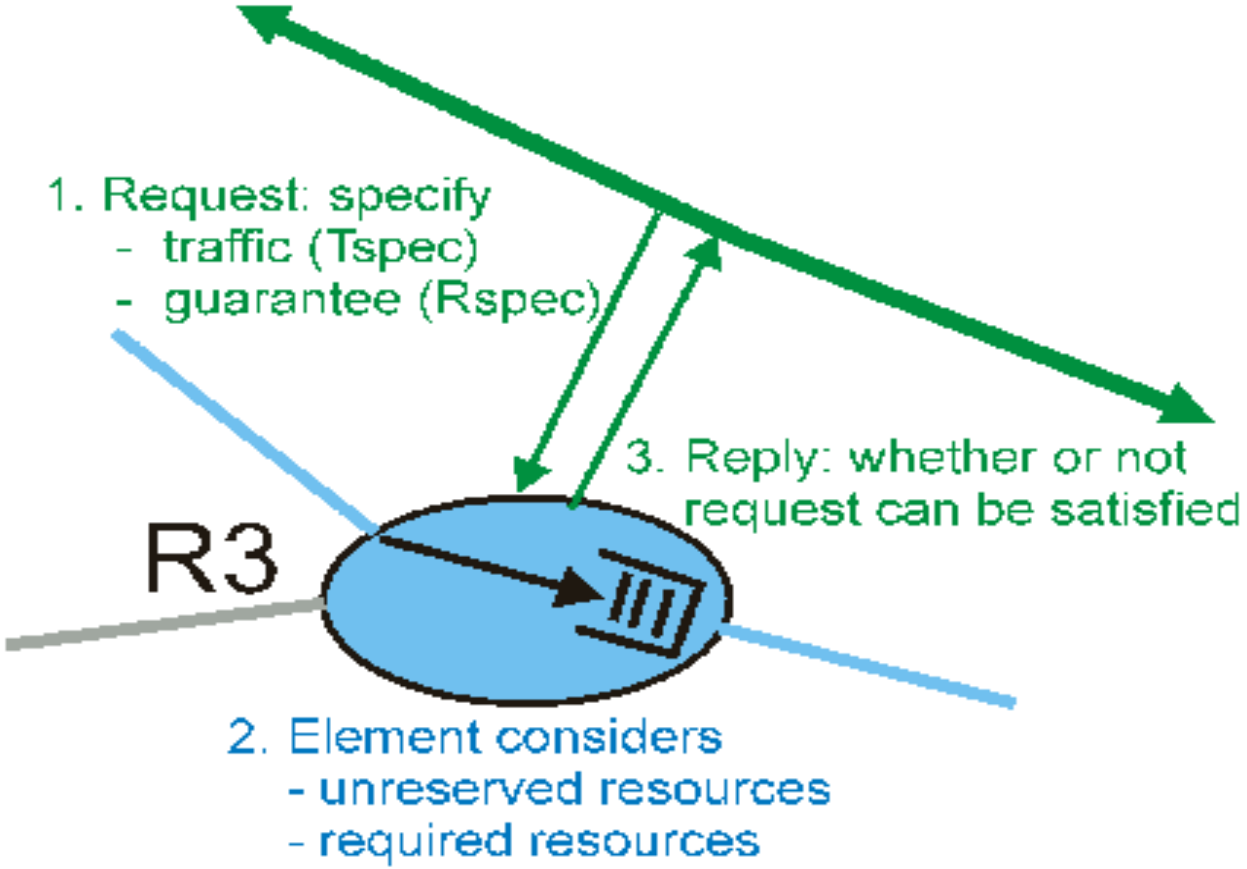


Figure 6.7-2: Per-element call behavior

Controlled Load Service

- A data flow will experience a quality of service closely to the QoS that a same flow would receive from an *unloaded network* element
- Using admission control to assure that this service is received even when the network element is *overloaded*
  - E.g. a new flow is accepted when the sum of the average rates of all accepted flow and the current flow is below the overall capacity dedicated to controlled load services
- Call setup only uses Tspec
- Allocation algos not defined

Guaranteed Service

+ **Packets of a flow will experience within the e2e path:**
  - **Assured level of bandwidth**
  - **Mathematically bounded end-to-end delay**
  - **No queuing losses for conforming packets**

Resource needed for guaranteed Delay

+ **Theorem**
  - **if**
    - flows are characterized and are enforced by a dual token bucket Tspec
    - WFQ queuing discipline used in all the routers of the network
    - a reserved rate R not less than average token bucket rate r ($R \geq r$)
  - **it can be proved that the maximum end-to-end queuing delay DMAXQ is mathematically upper bounded (Parekh and Gallager, 1993)**

Slide 74

Resource needed for guaranteed Delay

Maximum end-to-end delay is given by:

$$D_{MAX} = D_{FIX} + D_{MAXQ}$$

- D_{FIX} is related to fixed delays (transmission, propagation)
- D_{MAXQ} is the maximum end-to-end queuing delay

In a perfect fluid model the flow sees a dedicated wire of bandwidth R between source and destination

In this case the maximum end-to-end queuing delay should be:

$$D_{MAXQ} = \frac{b}{R}$$

b: bucket depth

R: reserved rate ($R \geq r$)

P→inf.

IntServ Guaranteed Delay

- To allow for deviations from the perfect fluid model two error terms are introduced:

$$D_{MAXQ} = \frac{b}{R} + \frac{C}{R} + D$$

C: rate dependent error term (e.g. datagram assembling from ATM cells)

D: rate independent error term (e.g. processing routing updates)

- Considering **peak rate** limitation **p** and **maximum packet size M**, the maximum end-to-end delay becomes:

$$D_{MAXQ} = \begin{cases} \frac{(b-M)(p-R)}{R(p-r)} + \frac{M+C_{TOT}}{R} + D_{TOT} & p > R \geq r \\ \frac{M+C_{TOT}}{R} + D_{TOT} & R \geq p \geq r \end{cases}$$

- Ctot, Dtot: sum of C and D parameters of the nodes of the path
- Lossless buffer size at node i = b+Csum(i)+Dsum(i)*R
  - Csum(i)=sums of C until node i

IntServ Traffic Classes

Template Components	Guaranteed Service	Controlled Load Service
End-to-End behavior	Guaranteed max. delay Guaranteed throughput No queuing losses	Approximates Best Effort over unloaded network
Typical Applications	Real time applications	Applications sensitive to network congestion
Network Elements involved	Fluid model using R (requested bandwidth) and B (buffer size)	Admission Control
Parameters requested	Tspec: r, b, p, M, m Rspec: R, S	Tspec: r, b, M, m (p is not required)
Exported Information	(C,D), i.e. values which measure deviation from the ideal fluid model	No exported information
Policing	M + min [p·T, r·T + b-M] Min. datagram length: m	r·T + b

RSVP Design Goals

+ **RSVP has been designed according to several goals:**
  - **Unicast and Multicast capabilities**
  - **Heterogeneous receivers support**
  - **Source and sub-stream filtering capabilities**
  - **Dynamic multicast group changes capability**
  - **Efficient use of resources**
  - **Protocol overhead limitation**
  - **Connectionless and dynamic routing environment adaptability**
  - **Modularity**

RSVP Design Principles

+ **The following design principles have been adopted:**
  - **Receiver initiated reservation**
  - **Soft-state**
  - **Reservation styles and merging**
  - **Opaque information transport**
  - **Independence from underlying routing protocol**

Soft states

- RSVP soft state is created and periodically refreshed by Path and Resv messages
- The state is deleted if no matching refresh messages arrive before the expiration of a "cleanup timeout" interval
- The state is deleted if no matching refresh messages arrive before the expiration of a "cleanup timeout" interval

RSVP Key concepts

+ **Flow Descriptor**
  - **obtained joining Filter Spec and Flow Spec**

Flow Descriptor	**Filter Spec**	◆ identifies packets of a flow ◆ updates classifier
	Flow Spec	◆ Service class ◆ Tspec (r,b,p,m,M) ◆ Rspec (R,S) ◆ updates scheduler

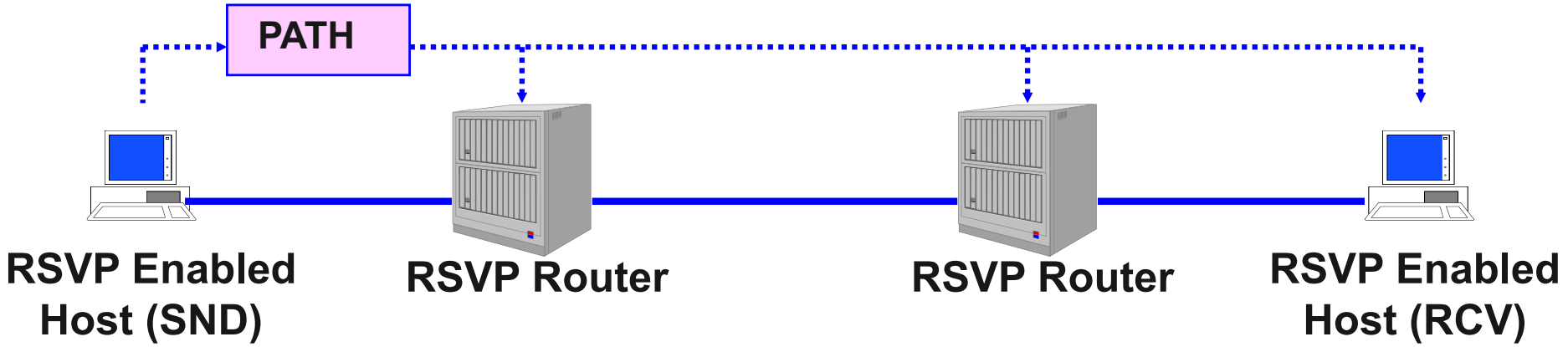
RSVP Messages

+ **RSVP messages are carried inside IP datagrams (protocol ID 46)**

+ **Seven message types:**
  - Path (downstream)
  - Resv (upstream)
  - PathErr (upstream)
  - ResvErr (downstream)
  - PathTear (downstream)
  - ResvTear (upstream)
  - ResvConf (downstream)

Unicast Reservation

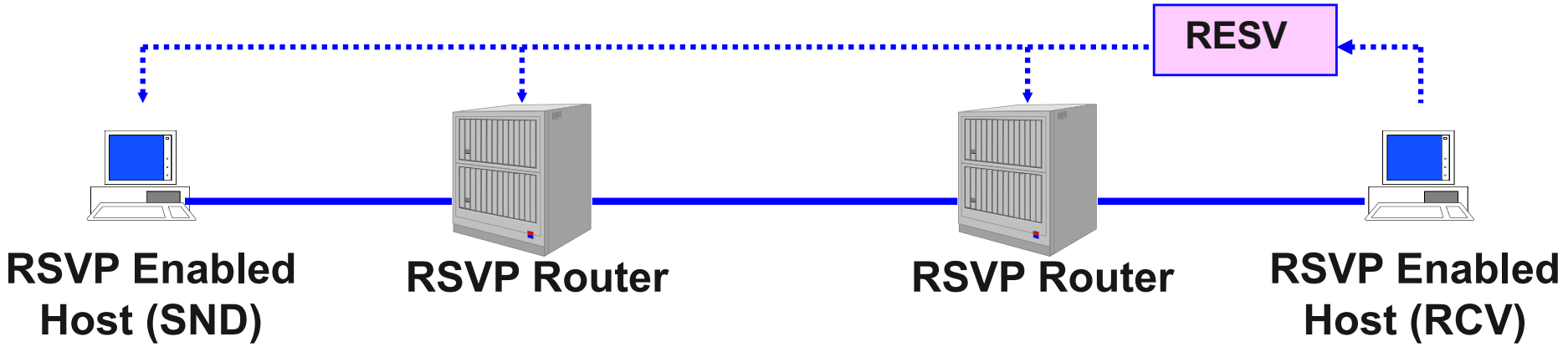
1st step: PATH messages are sent downstream



+ **At the reception of a PATH message**
  - RSVP router creates a path state associated to the corresponding session
  - RSVP router refeshes the timer associated to this path state
  - RSVP router updates Phop and Adspec objects and then forwards Path message towards next

Unicast Reservation

2nd step: RESV messages are sent back upstream



+ **At the reception of a RESV message an RSVP router**
  - **analyzes FlowSpec for Admission Control; if accepted a resv state is created**
  - **restarts the timer associated to the resv state**
  - **forwards Resv message to the previous node of the path (Phop)**

RSVP Messages

+ Path message

- **sent downstream from sender towards receiver(s)**
- **provides information about sender(s) Tspec and end-to-end path characteristics**
- **creates path states in each router along the path**
- **contains the following objects:**
  - Session: destination IP address, port and protocol ID
  - Phop: address of previous RSVP node
  - Sender_Template: IP address and port of the sender
  - Sender_Tspec: sender traffic characteristics (including dual token bucket parameters)
  - Adspec (optional): One Pass With Advertisement (OPWA) information updated by routers along the path
    - **Update MAX MTU if smaller**
    - **Parameters C and D of delay formula summed to the one just contained in the packet (Csum, Dsum)**

RSVP Messages

+ **Resv message:**
  - **sent upstream from receiver(s) towards sender(s)**
  - **carries reservation requests to the routers along the distribution tree**
  - **Resv messages originating from receivers of the same multicast group are merged together before being forwarded upstream**
  - **contains the following objects:**
    - Session: destination IP address, port and protocol ID
    - Reservation style: it can be FF, SE or WF
    - Flow descriptor: Filter spec and Flow spec (including rate R)
    - ResvConf (optional): IP address of the receiver, used to allow the sender to acknowledge reservation

Intserv Scalability Problem

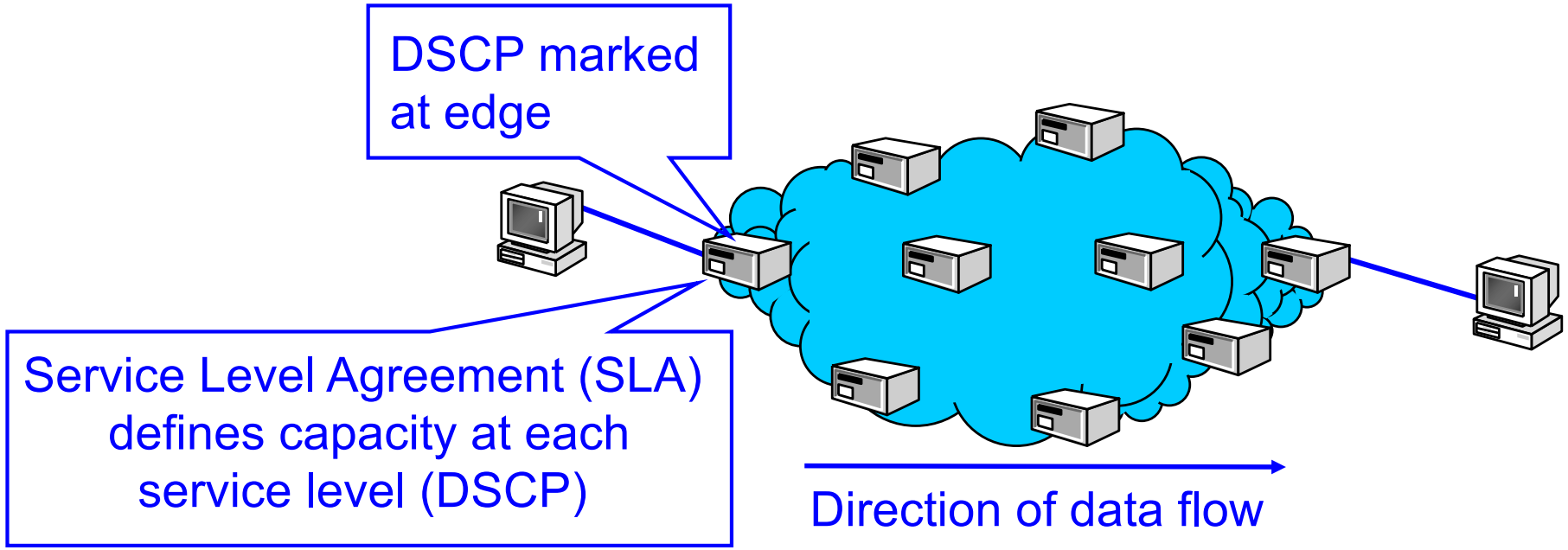
+ **RSVP per-flow reservation model implies large processing overhead in routers and great amount of traffic generation for periodic refreshes**

+ **Example:**
  - **ADPCM coding requires 32 kb/s for a voice channel. Neglecting packet overhead a single OC-12 interface of a backbone router (622 Mb/s) should support up to 20000 flows, implying that:**
  - **packet scheduler has to manage 20000 queues**
  - **up to 20000 states must be periodically refreshed**

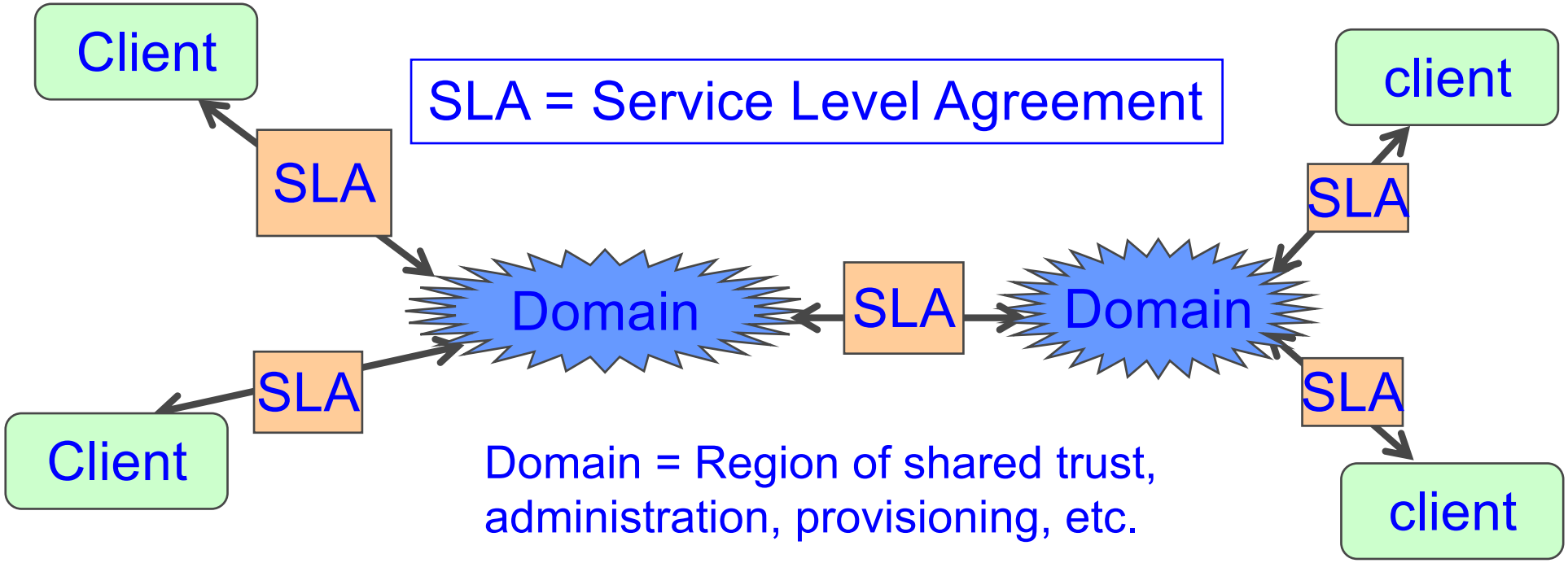
DiffServ Architecture

+ Scalability

- **A differentiated services mechanism must work at the scale of the Internet (e.g. millions of networks) and at the full range of speeds of the Internet (e.g., Gb/s links)**
  - push all the state to the edges
  - force all per-flow work to the edges
- **Edge-only state suggests that "simple" service indication must be carried in the packet: Diff Serv Code Point (DSCP) in the IP header**



Overall scenario for Diffserv QoS



+ **Domains provide their customers with the service specified in Service Level Agreement**

+ **Individual domains are free to manage the internal resources, to fulfill both internal and external obligations**

Service Level Agreement(SLA)

+ Service Level Agreement(SLA):

- A service contract between a customer and a service provider that specifies the forwarding service a customer should receive.
- A SLA may also specify traffic profiles and actions to traffic streams which are in- or out-of-profile.

+ Static SLA:

- norm at the present time.
- first instantiated at the agreed upon service start date and may periodically be renegotiated.

Per-hop Behavior(PHB)

+ SLA requirements for a traffic class can mapped in a limited set of forwarding behaviors, called Per-hop Behavior(PHB)

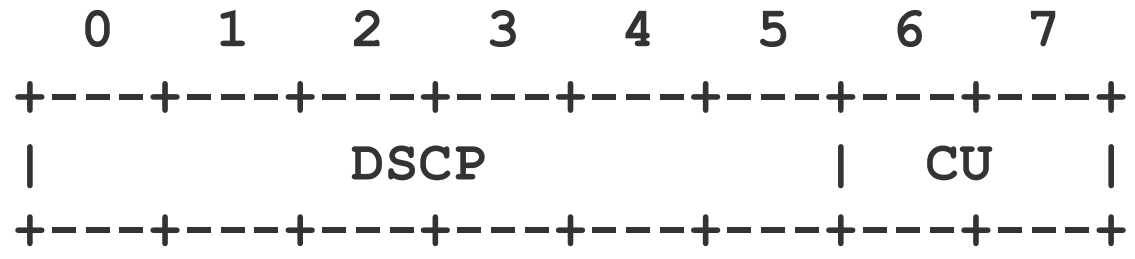
+ Per-hop Behavior(PHB)
  - is a description of the externally observable forwarding behavior of a DS node applied to a particular DS behavior aggregate.
  - PHBs may be specified in terms of their resource priority relative to other PHBs, or their relative observable traffic characteristics.
  - PHBs are implemented in nodes by buffer management and packet scheduling mechanisms

+ Packet requiring the same PHB are marked at the edge with the same IP DSCP

DiffServ Field in the IP header

+ **The DS field replaces the IPv4 TOS octet (and the corresponding IPv6 Traffic Class octet)**

+ **The DS field structure is the following**



DSCP: differentiated services codepoint
CU: currently unused

+ **The DS code point MUST be used as an index to a table**

DiffServ Field in the IP header

+ **Codepoint for the “Default” PHB (i.e. Best effort) is: “000000xx”**

+ **IP Precedence/Class selector code points (for backward compatibility)**

0	1	2	3	4	5	6	7
X	X	X	0	0	0	CU	

+ **NO backward compatibility for the DTR bits**

Standard PHB

+ Class selector compliant PHBs
+ **Expedited Forwarding** (EF) PHB
+ **Assured Forwarding** (AF) PHB group

Class selector compliant PHBs

+ 8 code points (“xxx000”)
+ At least two different PHBs
+ Any PHB selected by a higher code-point should give higher probability of timely forwarding than a PHB selected by a lower code-point
+ Code-points “11x000” must be mapped in a “better” PHB than “000000”

Expedited Forwarding PHB

+ The EF PHB can be used to build an end-to-end service characterized by
  - low-loss
  - low-latency, low-jitter
  - assured bandwidth

+ Queuing must be avoided
  - aggregate maximum arrival rate is less than that aggregate minimum departure rate

Expedited Forwarding PHB

+ To create the low-loss, low delay service

+ Nodes must be configured so that the aggregate has a “well-defined” (i.e. independent of other traffic on the node) minimum departure rate (EF PHB)

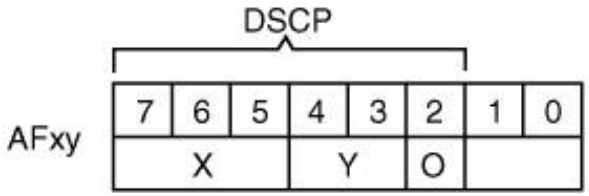
+ The aggregate must be configured (via policing and shaping) so that its arrival at any node is always less than the configured minimum departure rate

Expedited Forwarding PHB

+ **Mechanisms to implement EF PHB**
  - **Simple priority queue**
  - **CBWFQ**
  - **LLQ**
+ **Recommended code-point**
  - **101110**

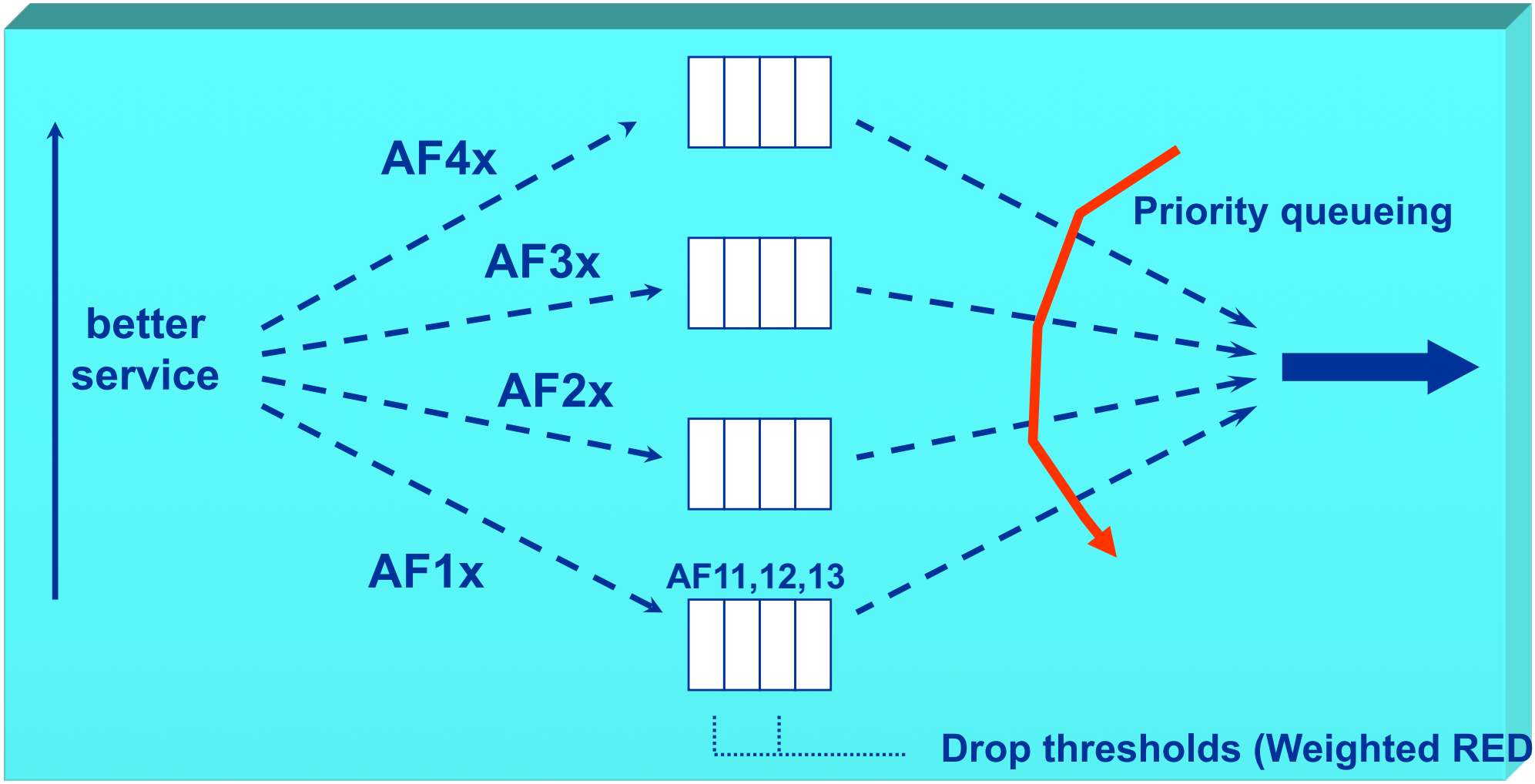
Assured forwarding PHB group

+ The AF PHB group provides delivery of IP packets in four independently *forwarded AF classes*
  - AFxy x=1,2,3,4
+ Within each AF class, an IP packet can be assigned one of three different levels of *drop precedence*
  - AFxy y=1,2,3
+ A DS node does not reorder IP packets of the same microflow if they belong to the same AF class
+ Packets of AF class x do not have higher probability of timely forwarding than class y packets, if x < y
+ Within an AF class, a packet with drop precedence *p* must not be forwarded with smaller probability than a packet with drop precedence *q*, if *p* < *q*



		DSCP Binary			DSCP Decimal
AF Class 1		x	y		
low drop pref	AF11	001	01	0	10
med drop pref	AF12	001	10	0	12
high drop pref	AF13	001	11	0	14
AF Class 2					
low drop pref	AF21	010	01	0	18
med drop pref	AF22	010	10	0	20
high drop pref	AF23	010	11	0	22
AF Class 3					
low drop pref	AF31	011	01	0	26
med drop pref	AF32	011	10	0	28
high drop pref	AF33	011	11	0	30
AF Class 4					
low drop pref	AF41	100	01	0	34
med drop pref	AF42	100	10	0	36
high drop pref	AF43	100	11	0	38

Assured forwarding PHB group



Service taxonomy: quantitative vs. qualitative

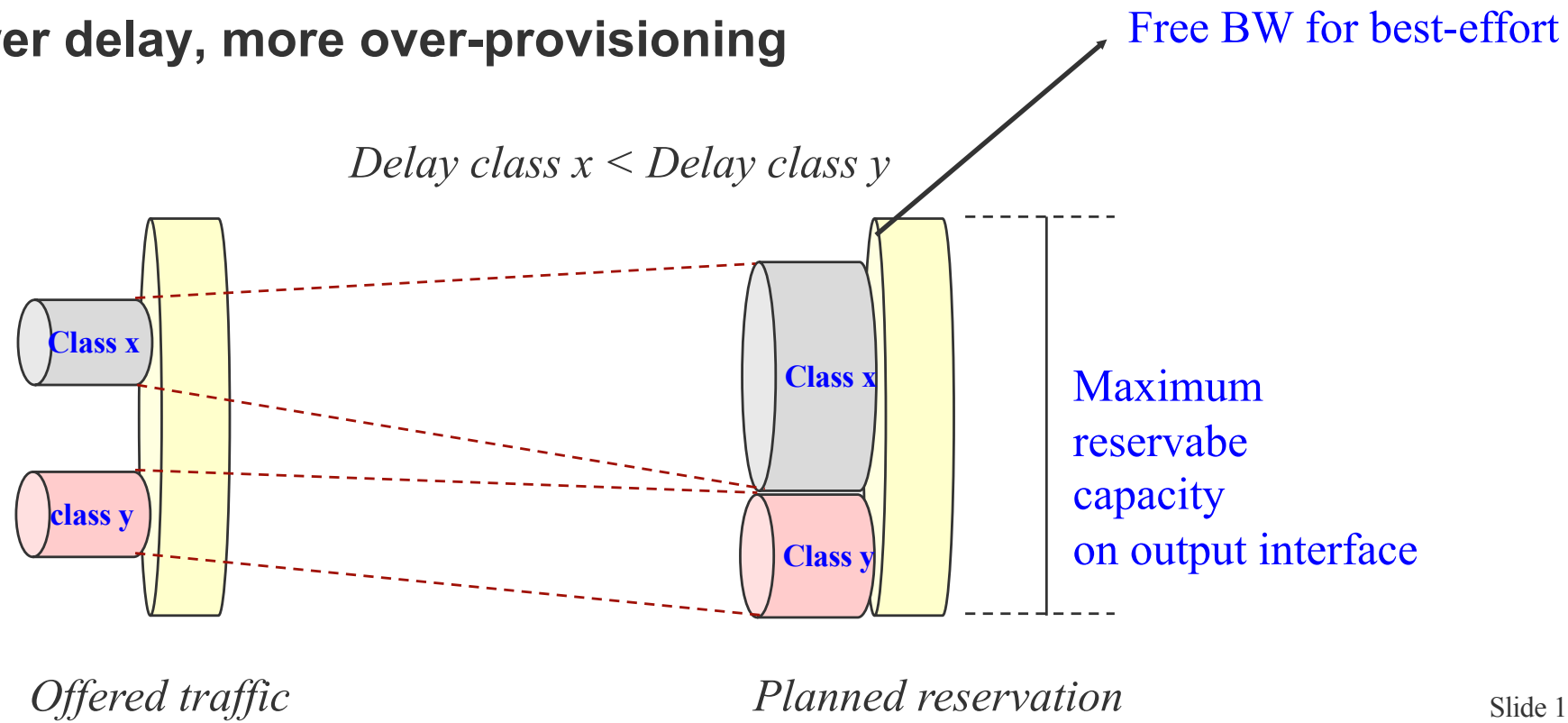
+ **Clearly qualitative (IntServ-controlled load , DiffServ EF)**
  - Level A traffic will be delivered with low latency
  - Level B traffic will be delivered with low loss
+ **Clearly quantitative (IntServ - guaranteed service)**
  - 90% of in profile traffic will experience no more than 50msec latency
  - 95% of in profile traffic will be delivered
+ **Not readily categorized (relative) (DiffServ - AF)**
  - Traffic offered at service level E will be allotted twice the bandwidth of service level F traffic
  - Traffic with drop precedence AF12 has a lower probability of delivery than traffic with drop precedence AF13

Multiple class over-provisioning 1/2

+ **Different classes with different bandwidth and delay requirements**

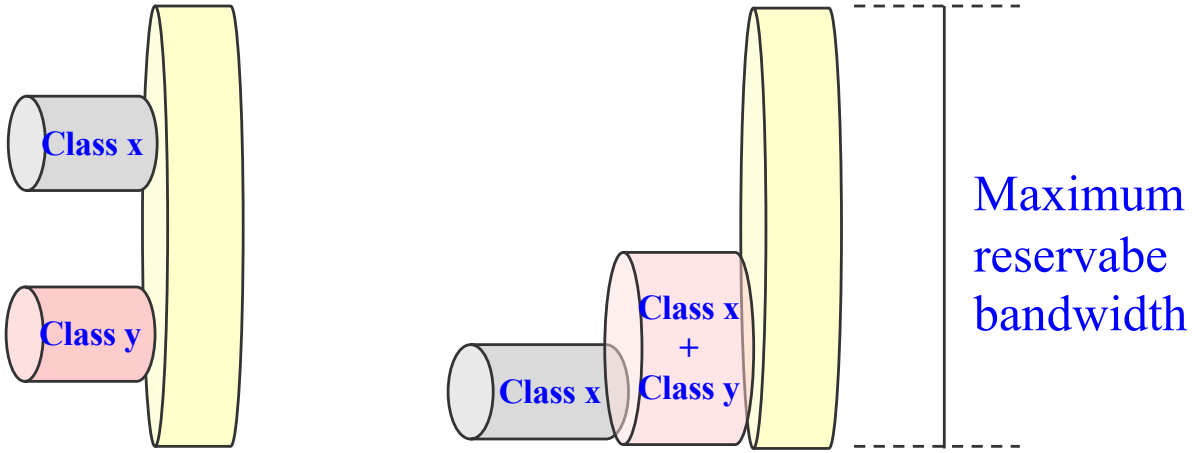
+ **Link BW distributed through over-provisioned bandwidth constraint realised through advanced scheduling (e.g., WDRR, WFQ, etc.)**

+ **Lower delay, more over-provisioning**



Multiple class over-provisioning 2/2

+ **Ingress shaping and Link BW distributed through priority queing**
  - **Condition: Sum of shaped offered traffic enougth lower than output capacity**
+ **Russian Dolls Model**
  - **Priority queueing leads class *x* to see all capacity, class *y* to see the maximum capacity less the one takens by *x* class**
+ **Problem: limit ingress traffic (e.g., shaping) to avoid starvation**
  - **This may involve resource usage inefficiency, since a class can not exploits temporary remaining capacity due to the shaping**



Offered traffic *Planned reservation*