



Quality of Service in IP Networks

Prof. Jean-Yves Le Boudec
Prof. Andrzej Duda
Prof. Patrick Thiran
LCA-ISC-I&C, EPFL

CH-1015 Ecublens
<http://icawww.epfl.ch>

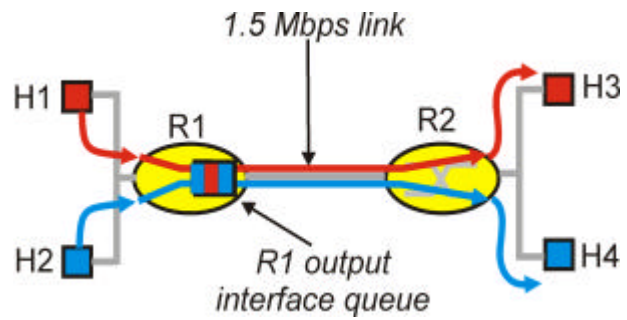
Contents

2

- ❑ Principles
- ❑ Traffic shaping
 - leaky bucket
 - token bucket
- ❑ Scheduling strategies
 - FIFO
 - Priority
 - Round Robin
 - Fair Queueing
 - RED
- ❑ IntServ
- ❑ DiffServ

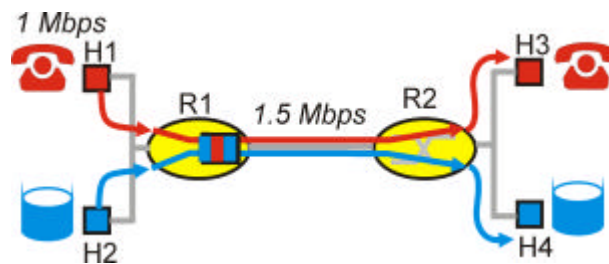
Improving QOS in IP Networks

- ❑ IETF groups are working on proposals to provide better QOS control in IP networks, i.e., going beyond best effort to provide some assurance for QOS
- ❑ Work in Progress includes Differentiated Services, and Integrated Services (RSVP)
- ❑ Simple model for sharing and congestion studies:



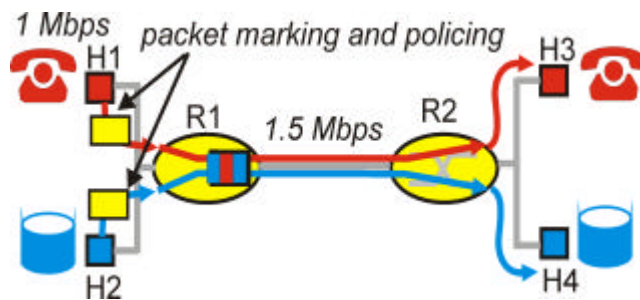
Principles for QOS Guarantees

- ❑ Consider a phone application at 1Mbps and an FTP application sharing a 1.5 Mbps link.
 - bursts of FTP can congest the router and cause audio packets to be dropped.
 - want to give priority to audio over FTP
- ❑ **PRINCIPLE 1: Marking of packets is needed for router to distinguish between different classes; and new router policy to treat packets accordingly**



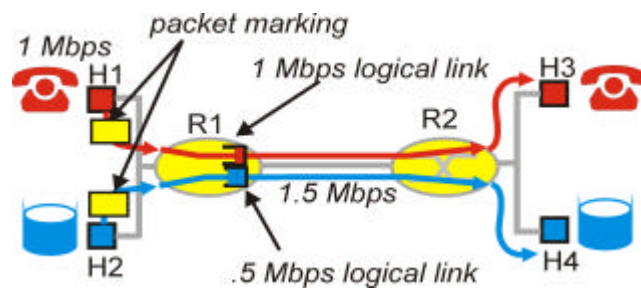
Principles for QOS Guarantees (more)

- ❑ Applications misbehave (audio sends packets at a rate higher than 1Mbps assumed above);
- ❑ **PRINCIPLE 2: provide protection (isolation) for one class from other classes**
- ❑ Require Policing Mechanisms to ensure sources adhere to bandwidth requirements; Marking and Policing need to be done at the edges:



Principles for QOS Guarantees (more)

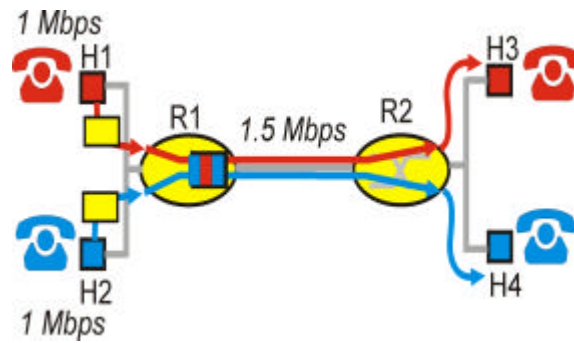
- ❑ Alternative to Marking and Policing: allocate a set portion of bandwidth to each application flow; can lead to inefficient use of bandwidth if one of the flows does not use its allocation
- ❑ **PRINCIPLE 3: While providing isolation, it is desirable to use resources as efficiently as possible**



Principles for QOS Guarantees (more)

7

- ❑ Cannot support traffic beyond link capacity
- ❑ **PRINCIPLE 4: Need a Call Admission Process; application flow declares its needs, network may block call if it cannot satisfy the needs**



Policing Mechanisms

8

☐ Three criteria:

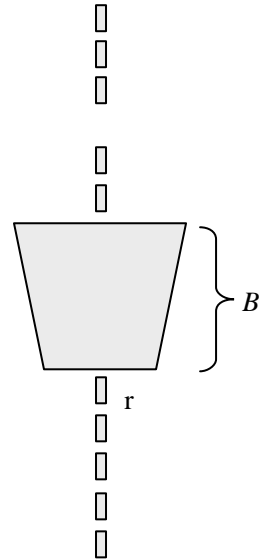
- (Long term) **Average Rate** (100 packets per sec or 6000 packets per min??), crucial aspect is the interval length
- **Peak Rate**: e.g., 6000 p p minute Avg and 1500 p p sec Peak
- (Max.) **Burst Size**: Max. number of packets sent consecutively, ie over a short period of time

Traffic shaping

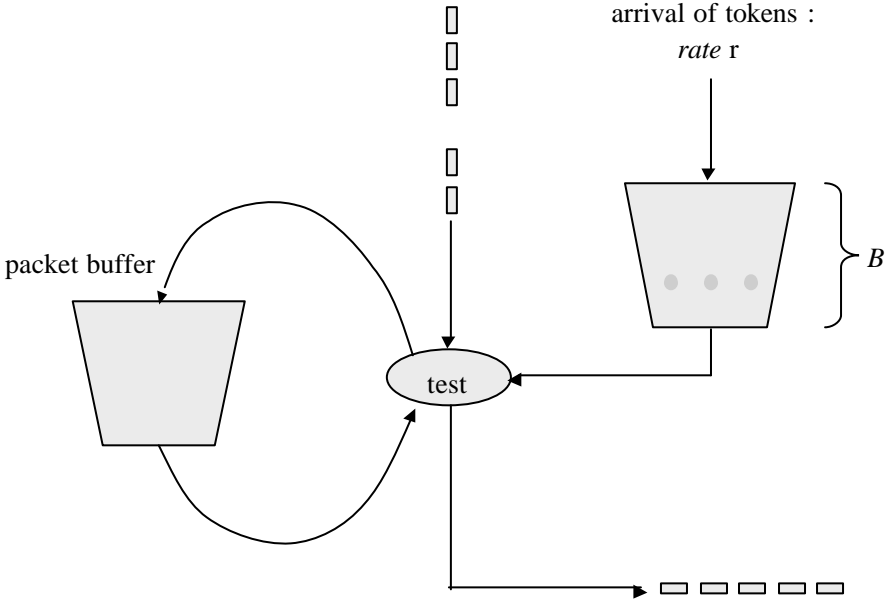
- ❑ How to prevent congestion?
 - it may result from burstiness
 - arrivals more deterministic, better performance
 - example : nbr of customers in D/D/1 vs. G/D/1
 - control the rate and burst size
 - traffic description
- ❑ Service contract
 - if the network knows the type of the traffic, it can reserve resources to support the traffic
 - contract between the source and the network
 - source: traffic description
 - network: QoS guarantee if the traffic conforms to the description
 - if the traffic is not conformant, penalty: reject a packet, no guarantees of the QoS (*traffic policing*)
- ❑ More details in Network Calculus course

Leaky bucket

- Limited size buffer with constant departure rate
 - r if buffer not empty
 - 0 if buffer empty
- Equivalent to the queue $G/D/1/N$
- Fixed size packets
 - one packet per clock tick
- Variable size packets
 - number of bytes per clock tick
- Packet loss if buffer filled



Token bucket



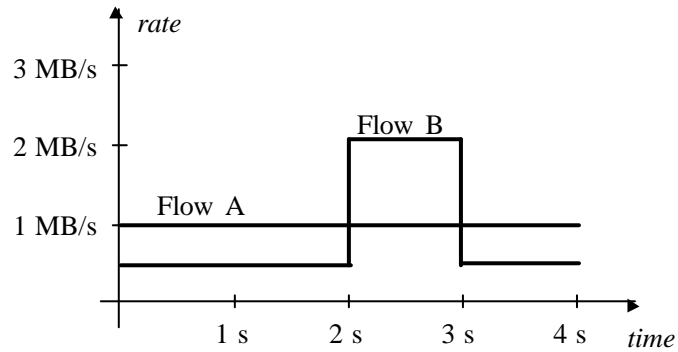
Token bucket

- ❑ Tokens generated with rate r
 - 1 token : 1 packet or k bytes
- ❑ Packet must wait for a token before transmission
 - no losses
 - allows limited bursts (a little bit more than B)
- ❑ When packets are not generated, tokens accumulate
 - n tokens - burst of n packets
 - if bucket filled, tokens are lost
- ❑ Mean departure rate : r

Burst duration

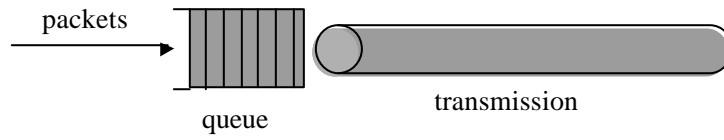
- ❑ Burst duration - S sec
- ❑ Size of the bucket - B bytes
- ❑ Maximal departure rate - p bytes/s
- ❑ Token arrival rate - r bytes /s
 - burst of $B + r S$ bytes
 - burst of pS
 - $B + rS = pS \rightarrow S = B/(r - p)$
- ❑ Example
 - $B = 250$ Kb, $p = 25$ Mb/s, $r = 2$ Mb/s
 - $S = 11$ ms

Traffic description



- ❑ Flow A : $r = 1 \text{ MB/s}$, $B = 1 \text{ byte}$
- ❑ Flow B : $r = 1 \text{ MB/s}$, $B = 1 \text{ Mbyte}$
 - during 2 s, the flow saves $2 \text{ s} \times 0.5 \text{ MB/s} = 1 \text{ MB}$

Scheduling strategies



- ❑ Scheduler
 - defines the order of packet transmission
- ❑ Allocation algorithms
 - bandwidth/delay
 - which packet chosen for transmission
 - buffers
 - which packet dropped

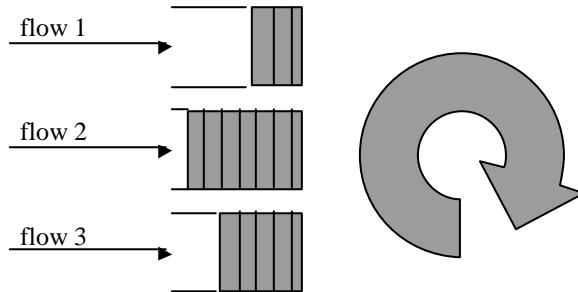
FIFO

- ❑ Last packets are dropped
- ❑ Current state of Internet routers
 - TCP adapt bandwidth based on losses
- ❑ Decouple the order of transmission and drop
 - RED (*Random Early Discard*) techniques
 - choose a packet randomly and drop it
- ❑ Allows to share bandwidth
 - proportionally to the offered load
- ❑ No isolation
 - elastic flows (rate controlled by the source e.g. TCP) may suffer from other flows
 - a greedy UDP flow may obtain an important part of the capacity
 - real time flows may suffer from long delays

Priority Queue

- ❑ Several queues of different priority
 - source may mark packets with priority
 - eg. ToS field of IP
- ❑ Problem
 - how to avoid everybody sending high priority packets?

Round Robin



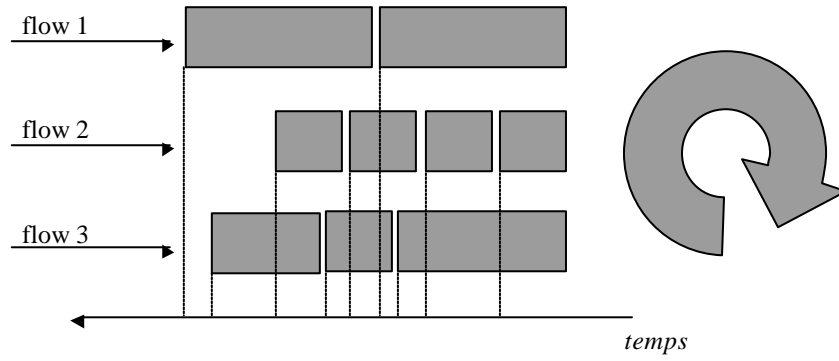
❑ Similar to *Processor Sharing or Time Sharing*

- one queue per flow
- cyclic service, one packet at a time

Characteristics

- ❑ It modifies the optimal strategy of sources
 - FIFO: be greedy - send as much as possible
 - RR: use your part the best
 - a greedy source will experience high delays and losses
- ❑ Isolation
 - good sources protected from bad ones
- ❑ Problem
 - flows sending large packets get more
 - cost of flow classification

Fair Queueing



❑ Round robin "bit per bit"

- each packet marked with the transmission instant of the last bit
- served in the order of instants

Weighted Fair Queueing

- ❑ Fair queueing
 - equal parts : $1/n$
- ❑ Weighted fair queueing
 - each flow may send different number of bits
- ❑ Example - weights w_i
 - flow 1 weight 2 flow 2 weight 1 flow 3 weight 3
 - $1/3$ $1/6$ $1/2$

$$x_i = C \frac{w_i}{\sum w_i}$$

C : link capacity

Rate guarantee

- ❑ Weights expressed as proportions (w_i - guaranteed weight)

$$x_i = C \frac{w_i}{\sum w_i}, \quad \sum w_i \leq 1$$

$$x_i \geq C \times w_i$$

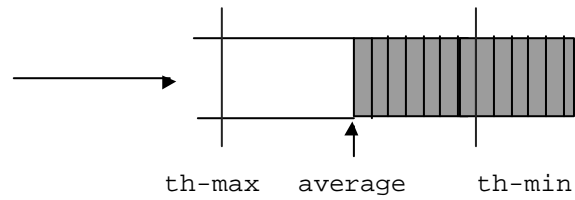
- ❑ Weights to guarantee a given rate

$$w_i = x_i / C$$

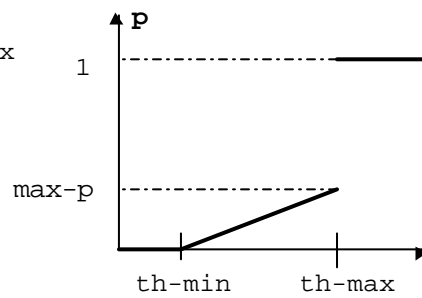
Random Early Detection

- ❑ Family of techniques used to detect congestion and signal sources
 - when a queue is saturated, packets are dropped
 - losses interpreted as congestion signals → decrease rate
- ❑ Idea
 - act before congestion and reduce the rate of sources
 - threshold for starting to drop packets
- ❑ Losses are inefficient
 - result in retransmissions - dropped packets should be retransmitted
 - enter Slow Start
- ❑ Synchronization of TCP sources
 - several packet dropped
 - several sources detect congestion and enter slow start at the same instant

RED



- ❑ Estimation of the average queue length
 - $\text{average} \leftarrow q \times \text{measure} + (1 - q) \times \text{average}$
- ❑ If $\text{average} \leq \text{th-min}$
 - accept the packet
- ❑ If $\text{th-min} < \text{average} < \text{th-max}$
 - drop with probability p
- ❑ If $\text{th-max} \leq \text{average}$
 - drop the packet



RED Characteristics

- ❑ Tends to keep the queue reasonably short
 - low delay
- ❑ Suitable for TCP
 - single loss recovered by *Fast Retransmit*
- ❑ Probability p of choosing a given flow is proportional to the rate of the flow
 - more packets of that flow, higher probability of choosing one of its packets

RED Characteristics

❑ Dynamic probability p

- $p\text{-tmp} = \text{max-p} \times (\text{average} - \text{th-min}) / (\text{th-max} - \text{th-min})$
- $p = \min(1, p\text{-tmp} / (1 - \text{nb-packets} \times p\text{-tmp}))$
- nb-packets : number of packets that have been accepted since last rejected packet
- p increases slowly with nb-packets
- nb-packets is uniformly distributed in $[1, 1/(p\text{-tmp})]$

❑ Example :

- $\text{max-p} = 0.02$
- If $\text{average} = (\text{th-max} + \text{th-min})/2$, one packet is rejected every 50 packets

