Advanced Computer Networks

QoS in IP networks

Prof. Andrzej Duda
duda@imag.fr

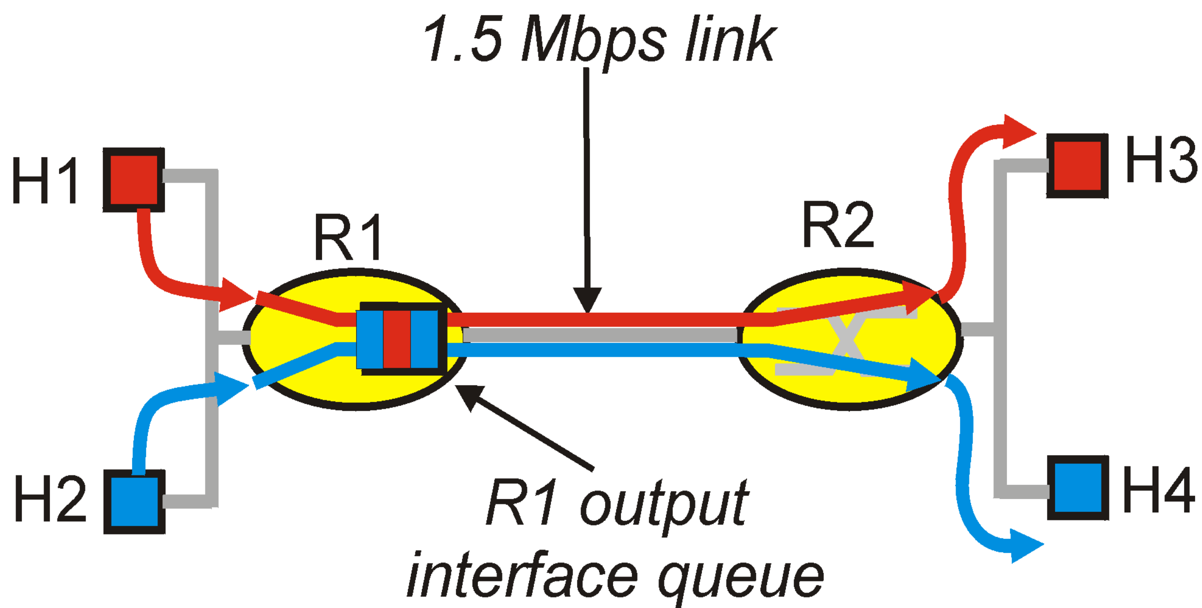
http://duda.imag.fr

Contents

- QoS principles
- Traffic shaping
  - leaky bucket
  - token bucket
- IntServ
- DiffServ

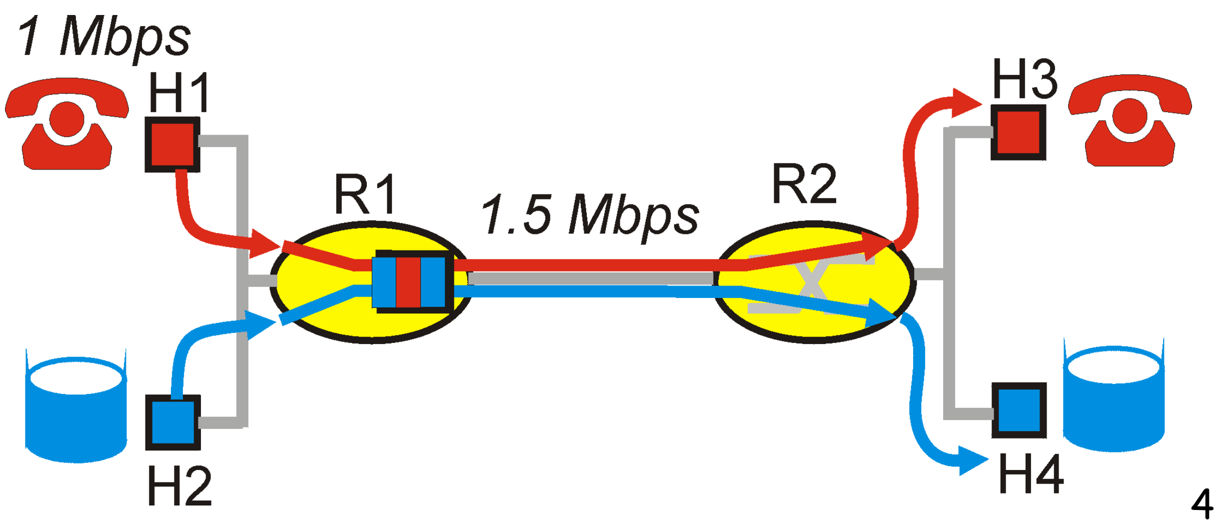
Improving QOS in IP Networks

- 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 Integrated Services, RSVP, and Differentiated Services
- Simple model for sharing and congestion studies:



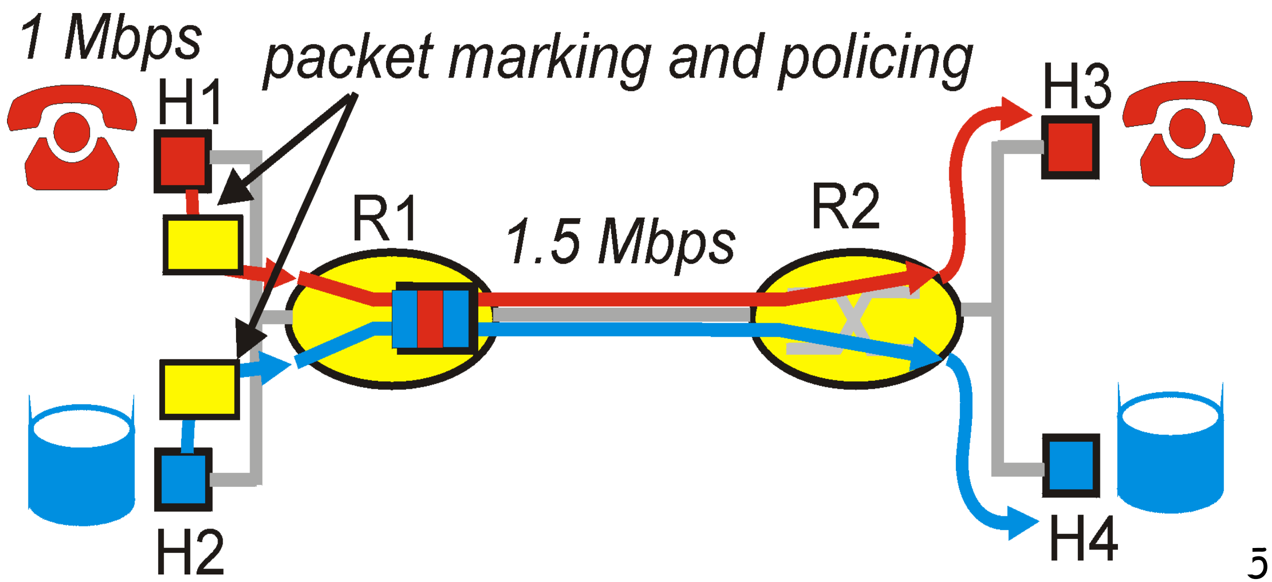
Principles for QOS Guarantees

- 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



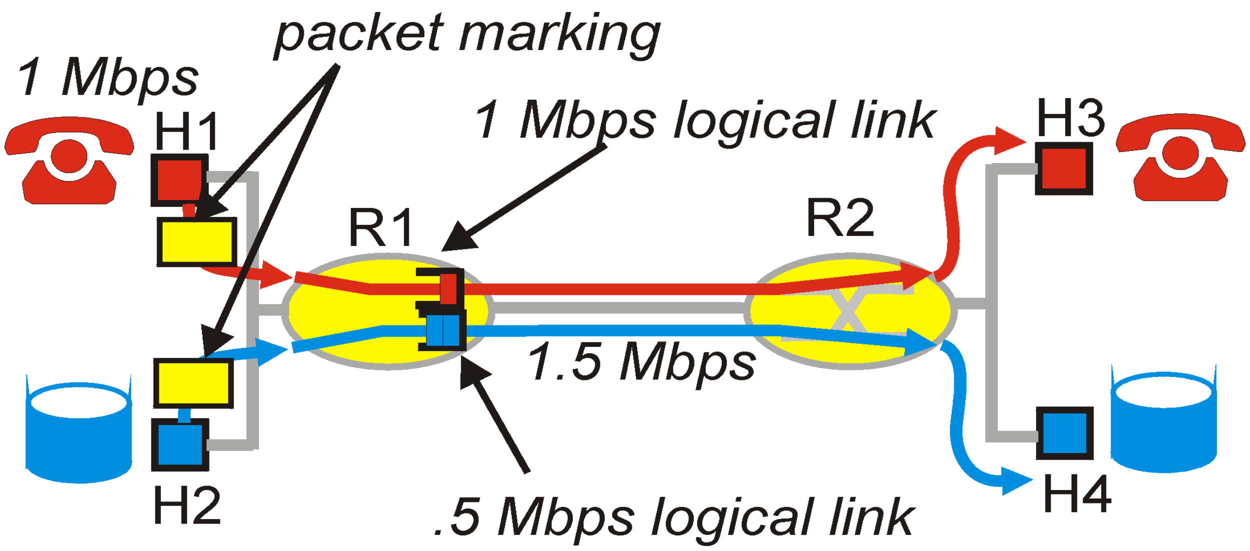
Principles for QOS Guarantees

- 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:



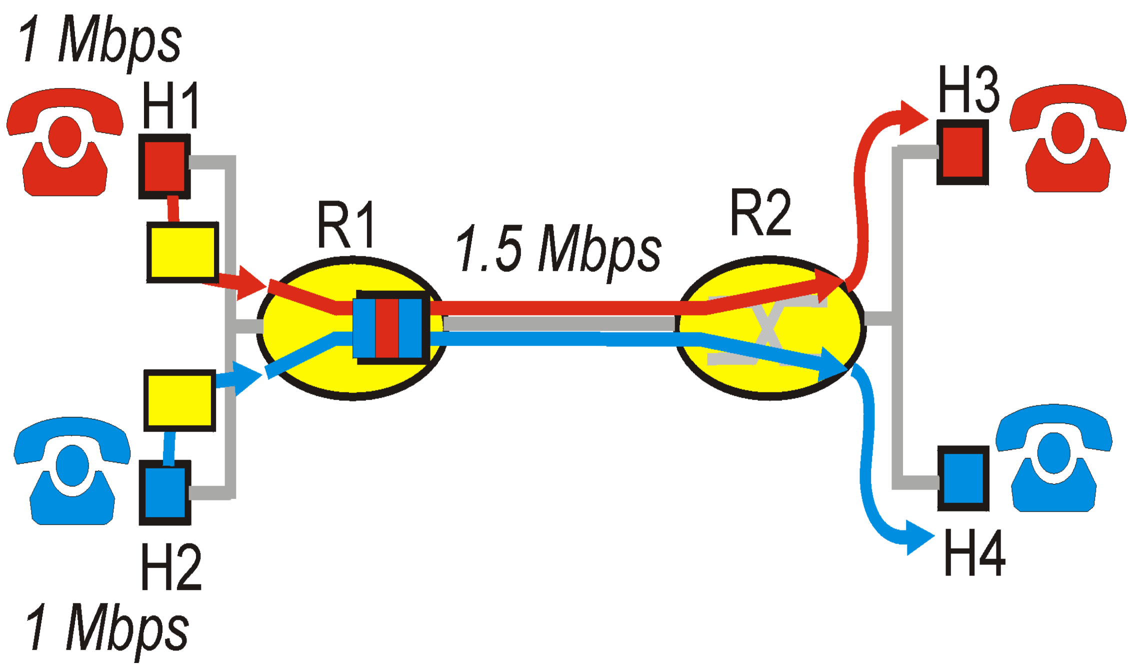
Principles for QOS Guarantees

- 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

- 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

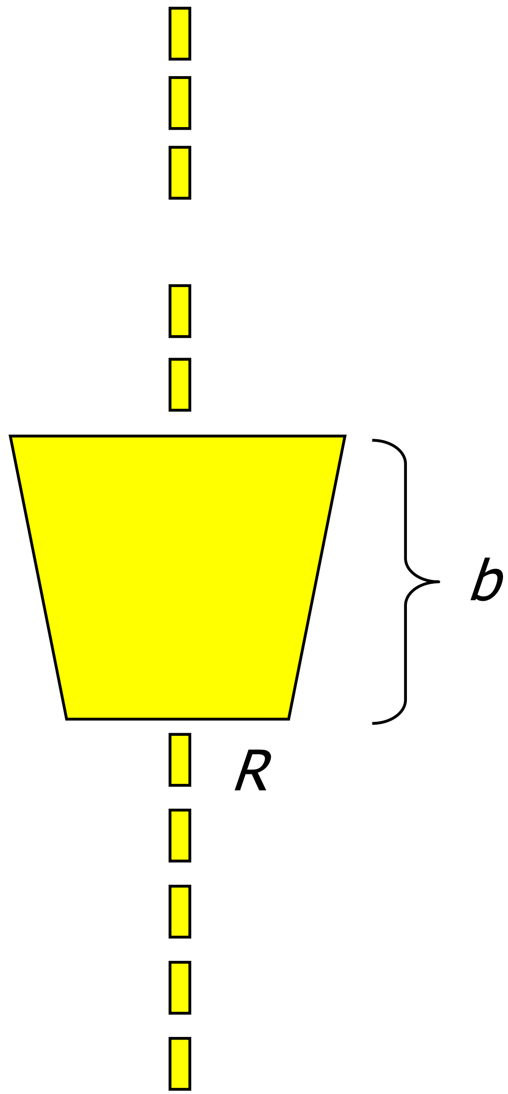


Traffic shaping

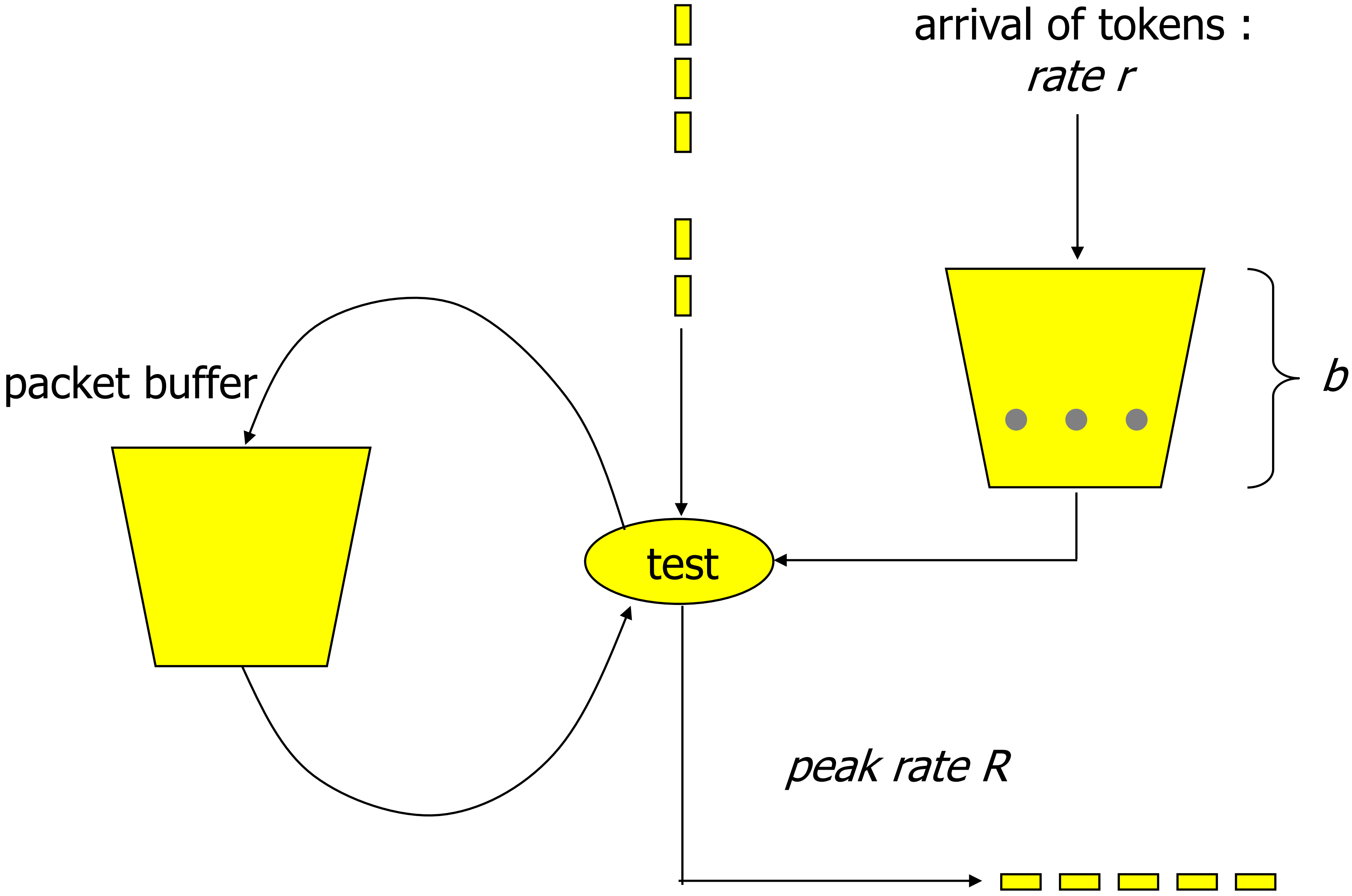
- How to prevent congestion?
  - it may result from burstiness
  - make arrivals more deterministic, obtain better performance
    - example : no. of clients in D/D/1 vs. G/D/1 or group arrivals vs. single arrivals
  - control the rate and burst size
    - traffic description - leaky bucket, token bucket
- 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 - leaky bucket, token bucket
    - network: QoS guarantee if the traffic conforms to the description
    - if the traffic is not conformant (leaky bucket, token bucket), penalty: reject a packet, no guarantees of the QoS *(traffic policing)*

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

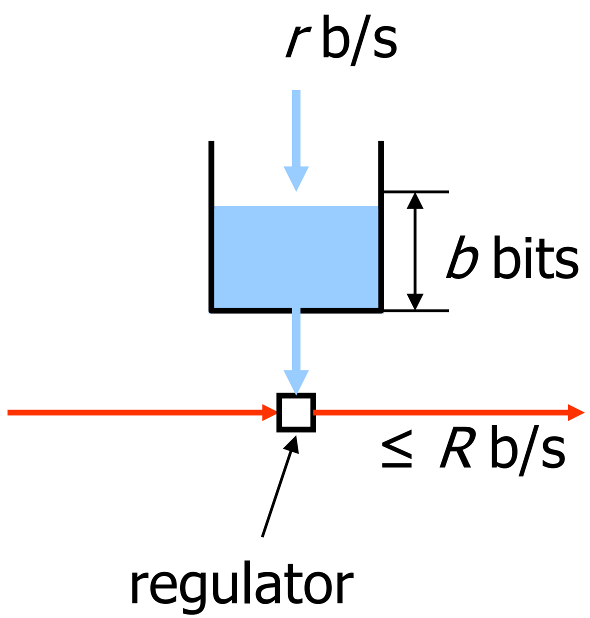


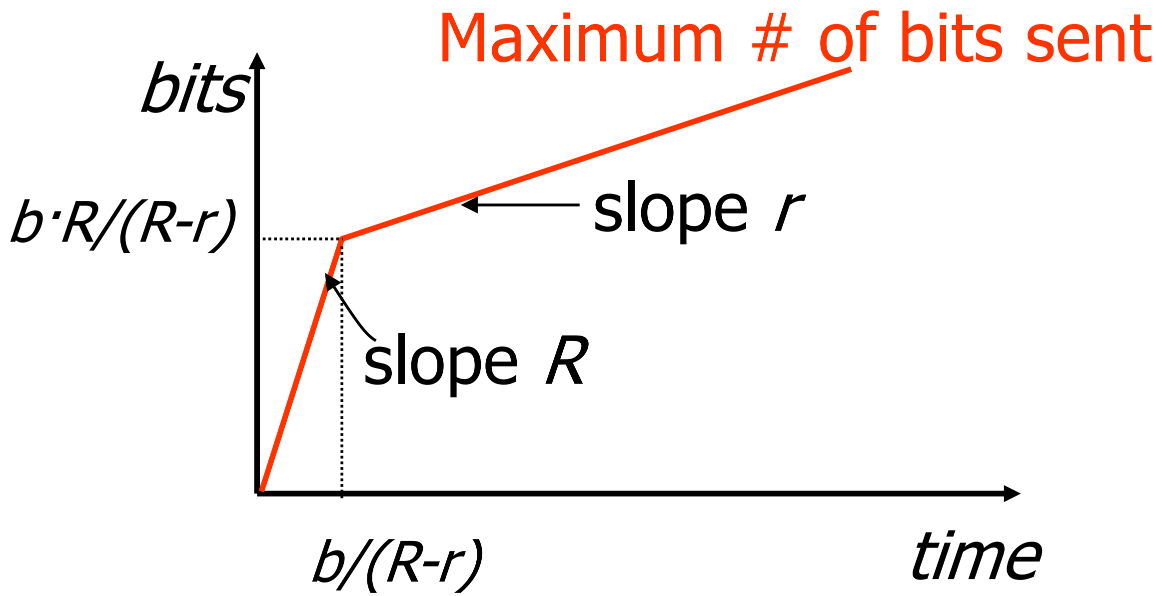
Token bucket



Characterizing Burstiness: Token Bucket

- Parameters
  - *r* – average rate, i.e., rate at which tokens fill the bucket
  - *b* – bucket depth (limits size of burst)
  - *R* – maximum link capacity or peak rate
- A bit (packet) can be transmitted only when a token is available





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*
- Delay limited by *b*/*r* (Little's formulae)

Example

- 25 MB/s link
- Network can support a peak rate R = 25 MB/s, but prefers sustained throughput of r = 2 MB/s
- Data generated
  - 1 MB each second, burst during 40 ms
- Example
  1. leaky bucket with b = 1 MB, R = 25 MB/s, r = 2 MB/s
  2. token bucket with b = 250 KB, R = 25 MB/s, r = 2 MB/s
  3. token bucket with b = 500 KB, R = 25 MB/s, r = 2 MB/s
  4. token bucket with b = 750 KB, R = 25 MB/s, r = 2 MB/s
  5. token bucket with b = 500 KB, R = 25 MB/s, r = 2 MB/s and leaky bucket with b = 1 MB, R = 10 MB/s

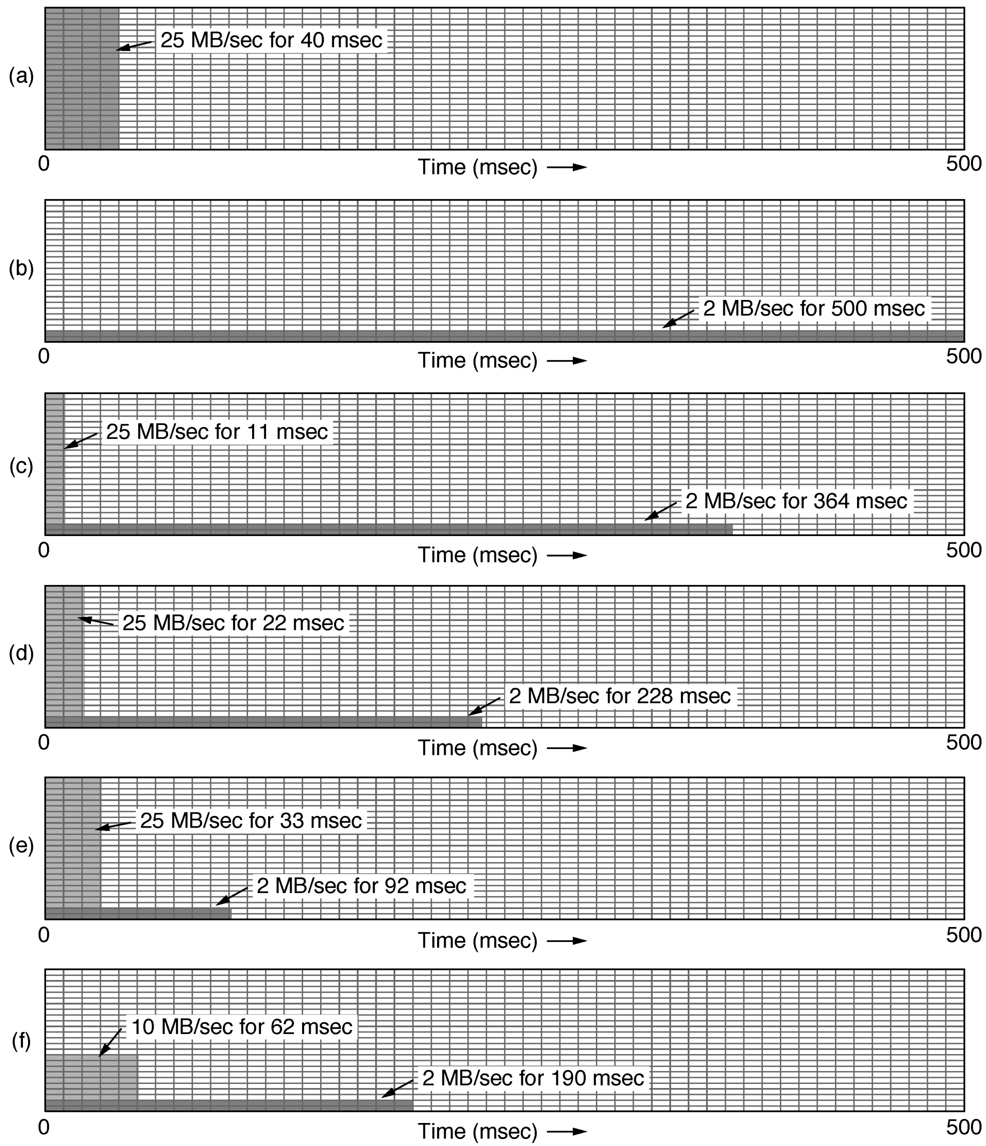
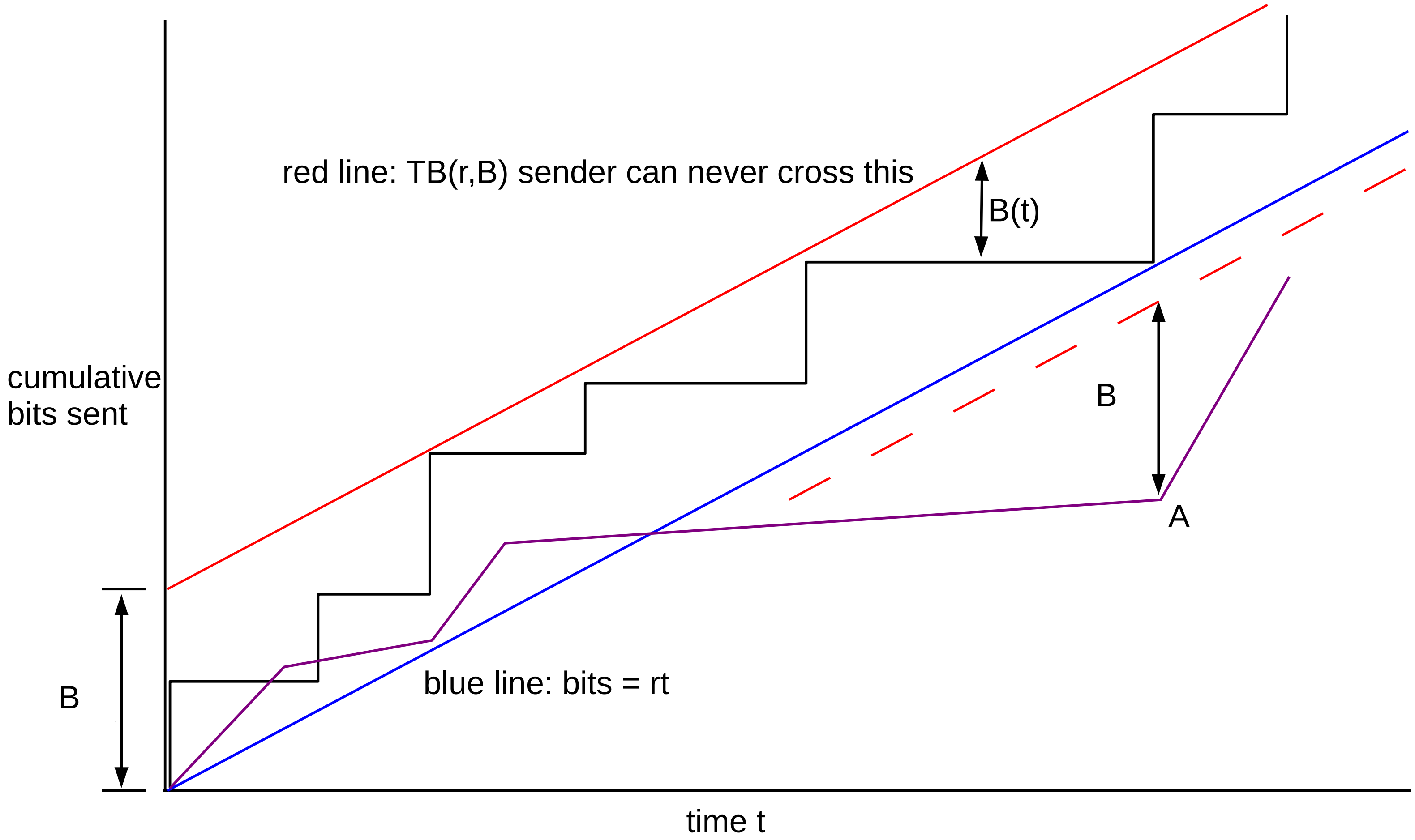


Fig. 5-25. (a) Input to a leaky bucket. (b) Output from a leaky bucket. (c) - (e) Output from a token bucket with capacities of 250KB, 500KB, and 750KB. (f) Output from a 500KB token bucket feeding a 10 MB/sec leaky bucket.

Burst duration

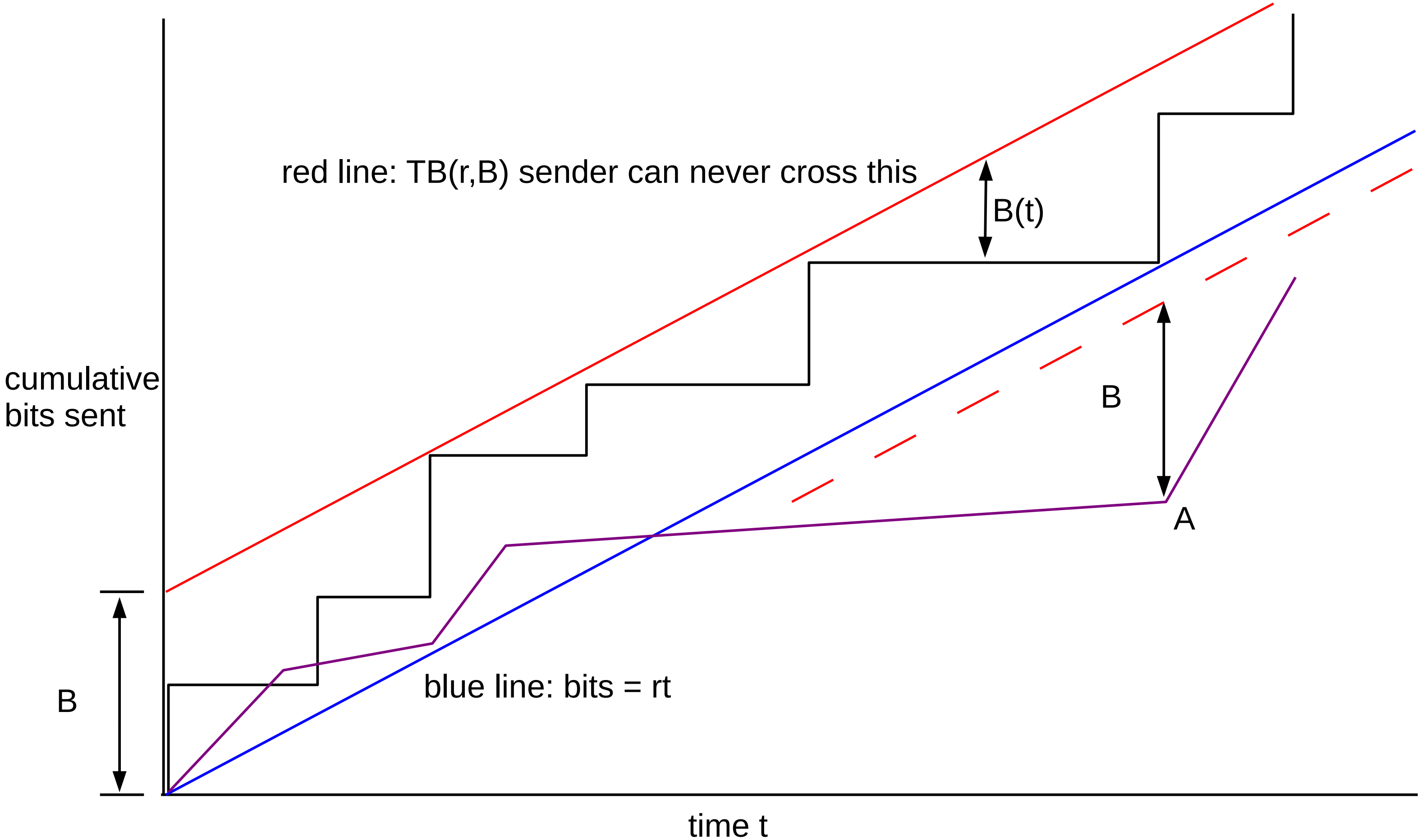
- Burst duration - S [s]
- Size of the bucket - b bits
- Maximal departure rate - R b/s
- Token arrival rate - r b/s
  - burst of $b + rS$ bits
  - burst of RS
  - $b + rS = RS \;\text{->}\; S = b/(R - r)$
- Example
  - b = 250 KB, R = 25 MB/s, r = 2 MB/s
  - S = 11 ms

Token Bucket – bucket depth B



- Solid red line: compliant sender may not cross
- Purple sender, by crossing below the blue line, cannot go back to the solid red line. The purple line cannot cross the dashed red line after falling “behind” at point A.

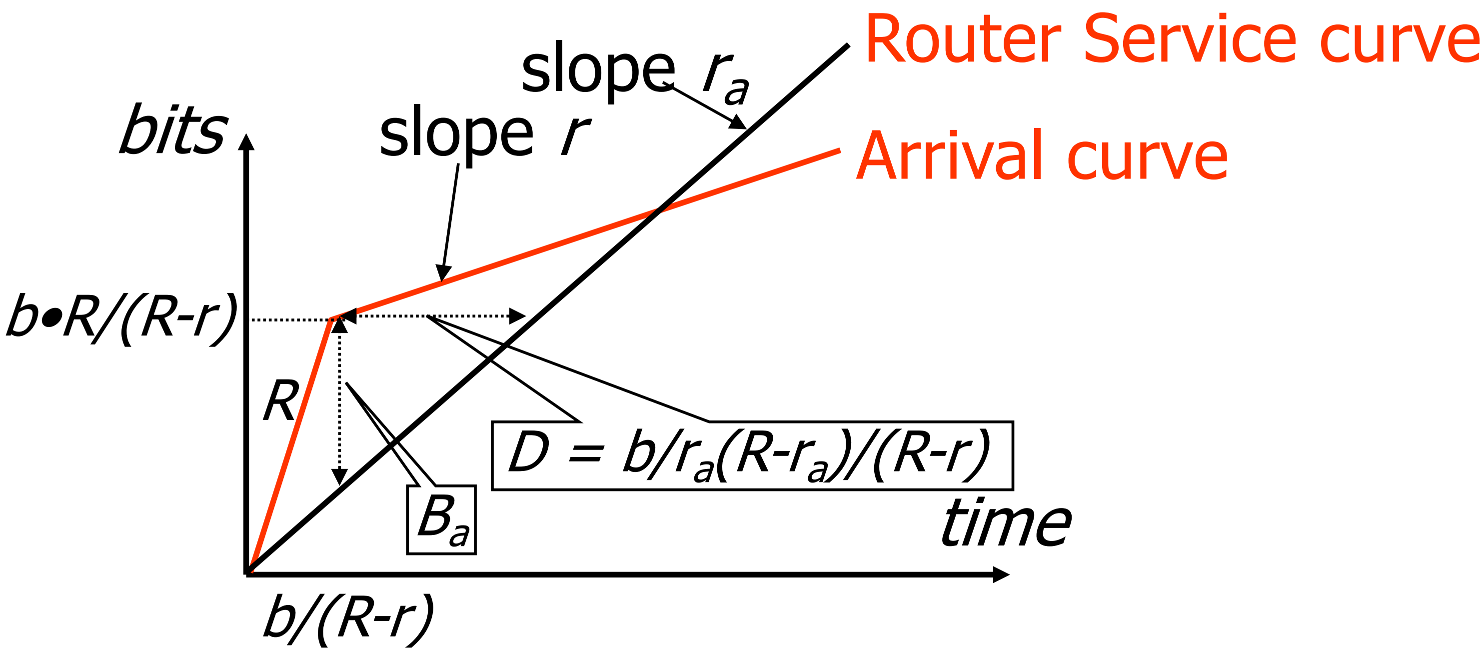
Token Bucket – bucket depth B



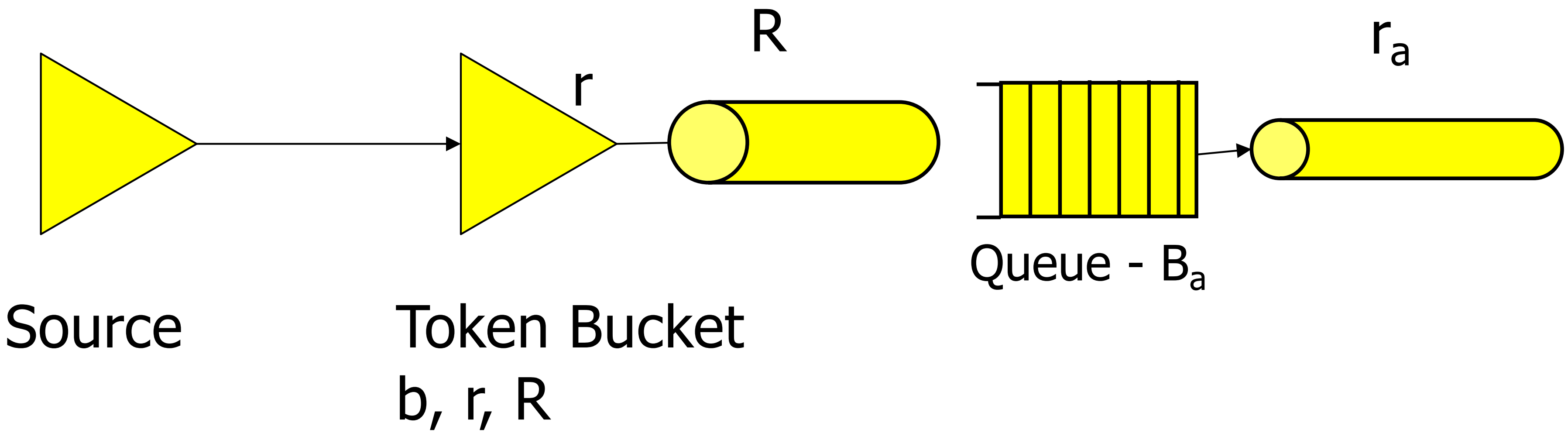
- bits(t) ≤ $rt + B$

QoS Guarantees: Per-hop Reservation

- End-host: specify
  - arrival rate characterized by token bucket with parameters *(b, r, R)*
  - the maximum tolerable delay *D*, no losses
- Router: allocate bandwidth r_a, buffer space B_a such that
  - no packet is dropped
  - no packet experiences a delay larger than *D*

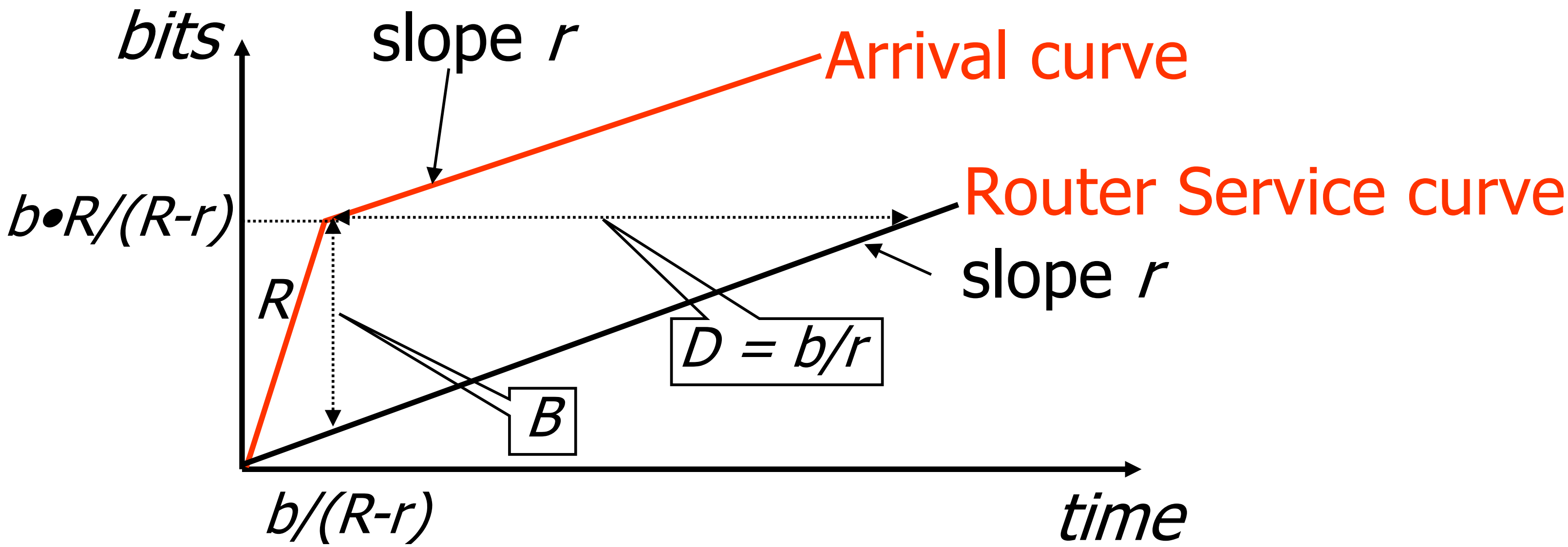


Token Bucket and a router

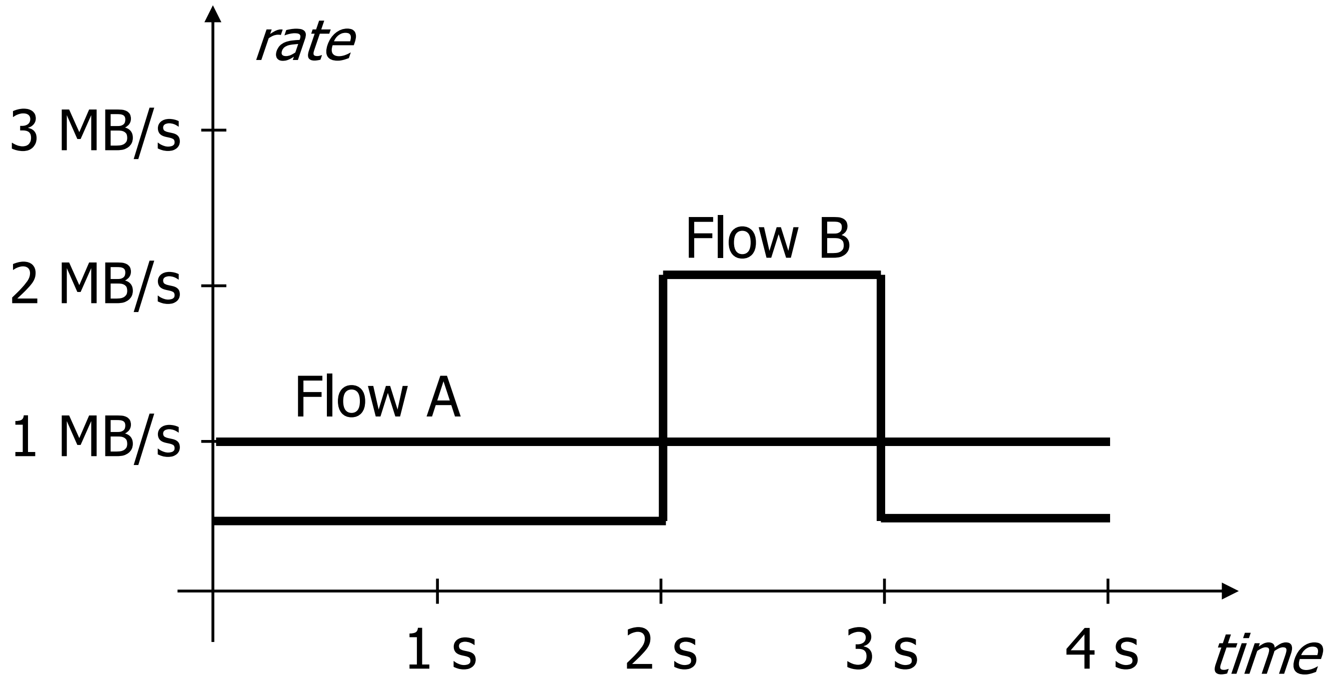


QoS Guarantees: Per-hop Reservation

- Router: if allocated bandwidth $r_a = r$, and buffer space B, then:
  - no packet is dropped
  - no packet experiences a delay larger than $D = b/r$

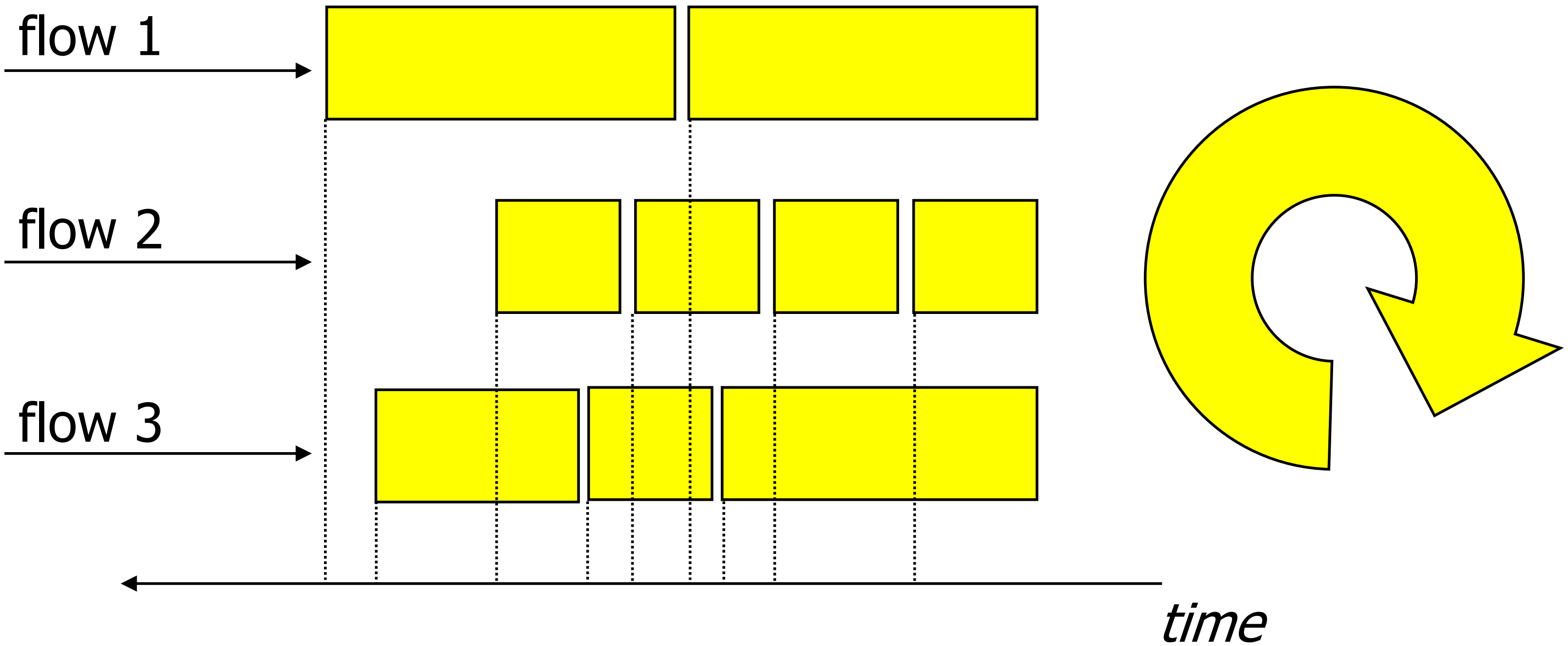


Traffic description



- Flow A : $r = 1$ MB/s, $b = 1$ B
- Flow B : $r = 1$ MB/s, $b = 1$ MB
  - during 2 s, the flow saves 2 s at 0.5 MB/s = 1 MB

Fair Queueing



- Round robin "bit per bit"
  - each packet marked with the transmission instant of the last bit
  - served in the order of the instants
  - allocates rates according to local max-min fairness

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	flow 2	flow 3
1/3	1/6	1/2

$r_i = C\,w_i$, *C:* link capacity

Rate guarantee

- Weights expressed as proportions (w_i - guaranteed weight)
  - If no packets of a given flow, unused capacity shared equally by other flows

$$r_i >= C\, w_i$$

  - Weights to guarantee a given rate

$$w_i = r_i / C$$

Delay guarantee

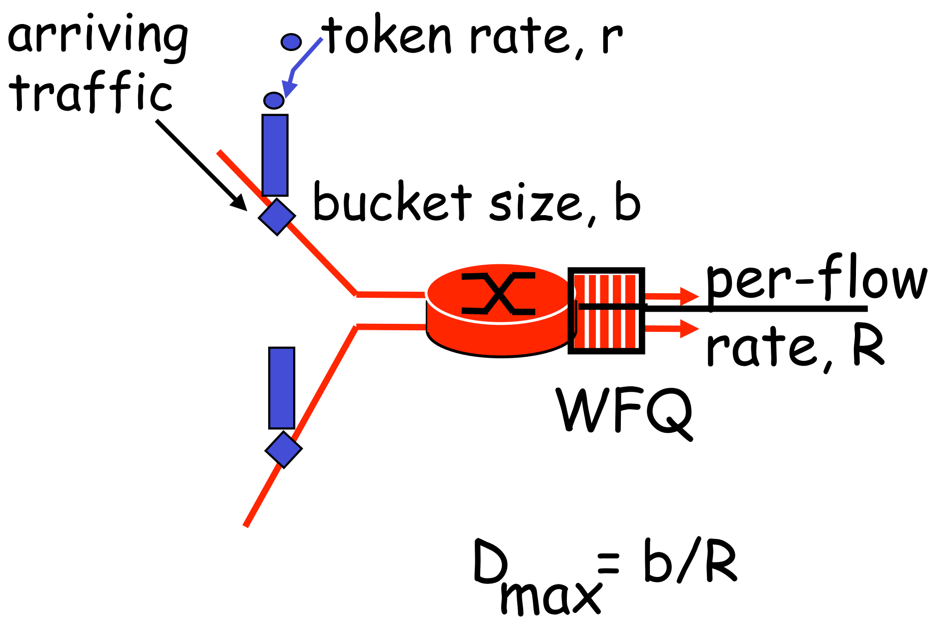
- Flow constrained by a token bucket
  - rate *r*, buffer of *b*
  - delay limited by *b/r*

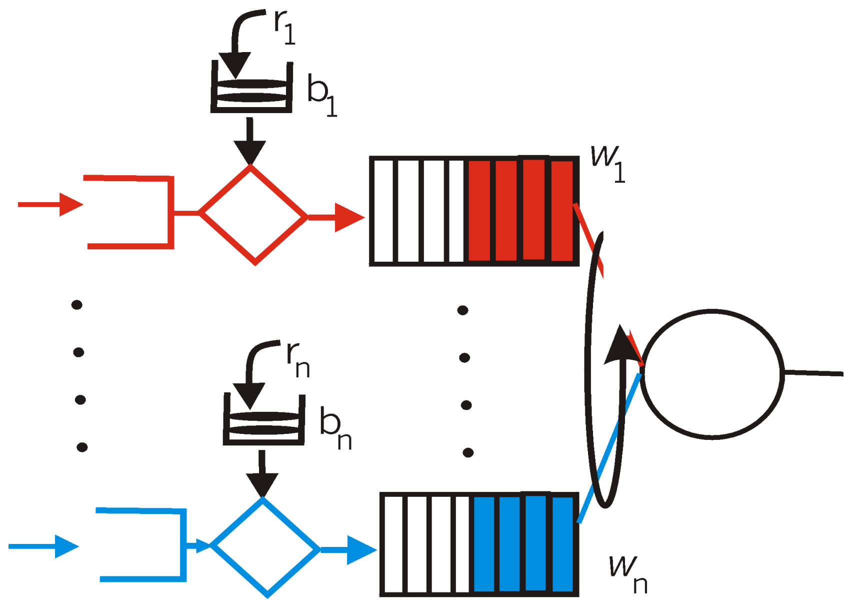
- If $r \leq r_i$ (the rate obtained is sufficient for the flow):

- delay limited by *b/r*

- total delay limited by *b/r*
  - if the packets pile up to the maximum size *b*, they only do so once - Pay Bursts Only Once

Delay guarantee



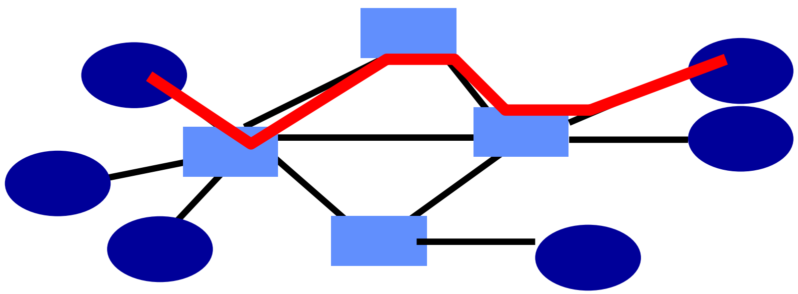


QoS architectures

- Integrated Services (IntServ)
  - per flow reservation at routers (RSVP protocol for reservation)
  - per flow scheduling
- Differentiated Services (DiffServ)
  - no reservation
  - classification at the border
  - scheduling per aggregated classes in the backbone

Reserving Resources End-to-End

- Source sends a reservation message
  - E.g., "this flow needs 5 Mbps"
- Each router along the path
  - Keeps track of the reserved resources
    - E.g., "the link has 6 Mbps left"
  - Checks if enough resources remain
    - E.g., "6 Mbps > 5 Mbps, so circuit can be accepted"
  - Creates state for flow and reserves resources
    - E.g., "now only 1 Mbps is available"

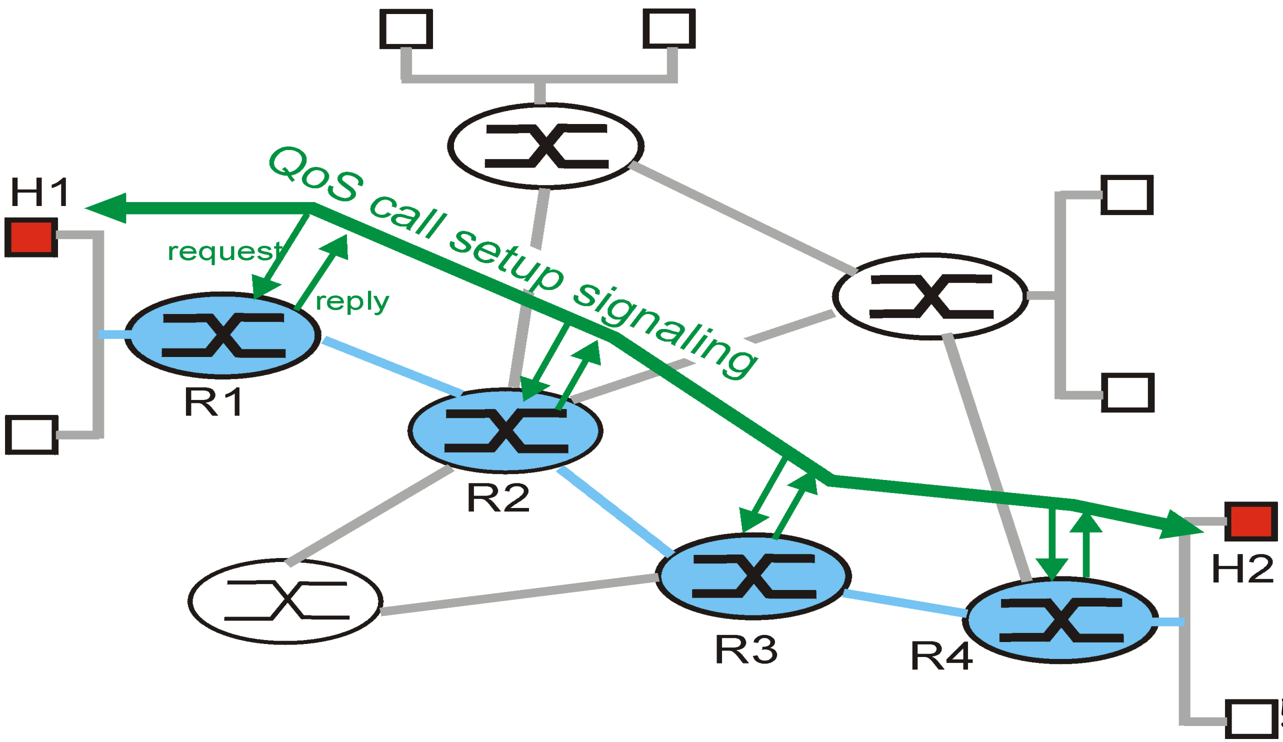


How to Specify Bursty Traffic

- Option #1: Specify the maximum bit rate. Problems?
  - Maximum bit rate may be much higher average
  - Reserving for the worst case is wasteful
- Option #2: Specify the average bit rate. Problems?
  - Average bit rate is not sufficient
  - Network will not be able to carry all of the packets
  - Reserving for average case leads to bad performance
- Option #3: Specify the burstiness of the traffic
  - Specify both the average rate and the burst size -> Token Bucket
  - Allows the sender to transmit bursty traffic
  - … and the network to reserve the necessary resources

Integrated Services

- An architecture for providing QOS guarantees in IP networks for individual application sessions
- Relies on resource reservation, and routers need to maintain soft state info, maintaining records of allocated resources and responding to new Call setup requests on that basis



Flow Admission

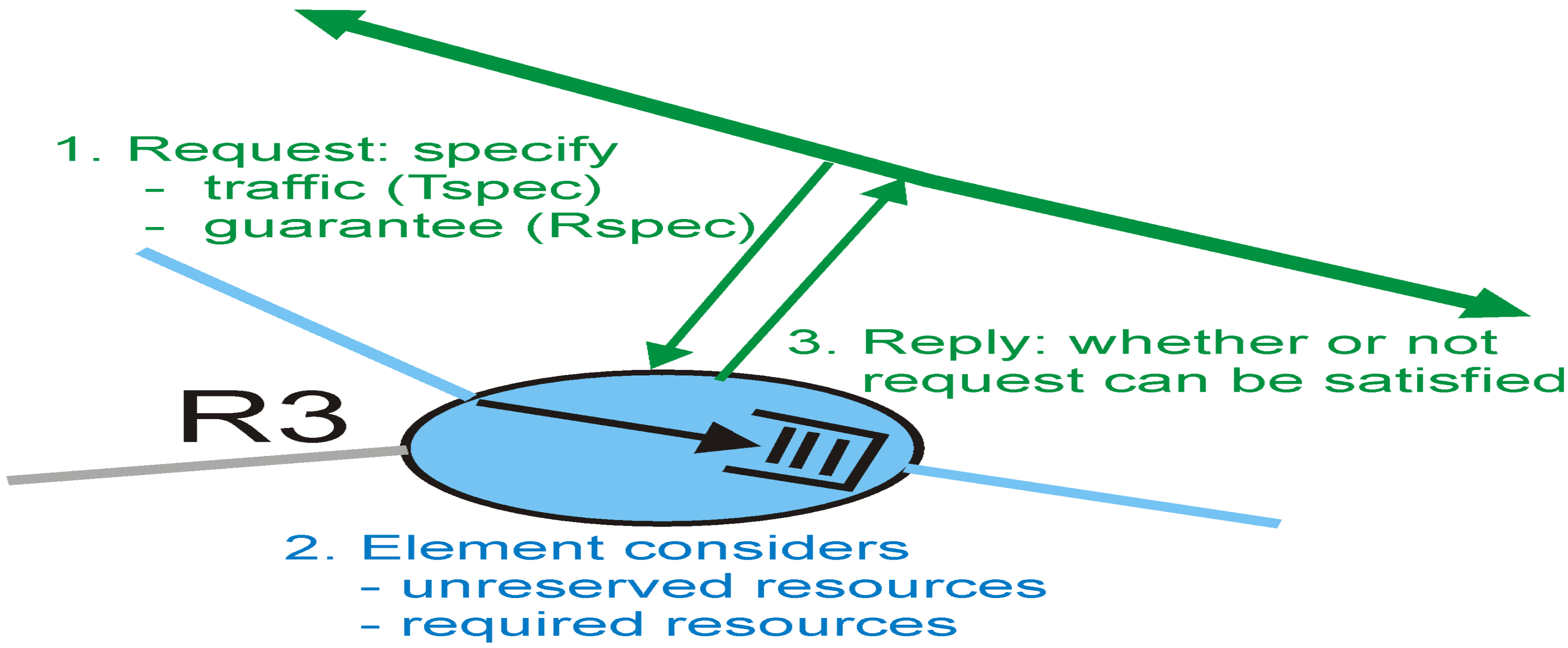
- Session must first declare its QOS requirement (T-spec) and characterize the traffic it will send through the network
- Routers check for resources and reserve them
- A signaling protocol is needed to carry QOS requirement to the routers where reservation is required

RSVP (Reservation Protocol)

- PATH message
  - **T-spec** - source traffic description
    - defines the traffic characteristics
    - token bucket: rate, capacity, and peak rate
  - packet from source to destination determines the return route
- RESV message
  - **R-spec**: if receiver wants to have better QoS (e.g. higher rate and jitter)
  - packet from destination to source follows the route established by PATH
  - reservations are done upon receiving this message

Flow Admission

- Flow Admission: routers will admit flows based on their T-spec and R-spec and base on the current resource allocated at the routers to other flows



Integrated Services: Classes

- **Guaranteed QOS**: this class is provided with firm bounds on queuing delay at a router; envisioned for hard real-time applications that are highly sensitive to end-to-end delay expectation and variance
  - rate and delay
- **Controlled Load**: this class is provided a QOS closely approximating an unloaded network; envisioned for today's IP network real-time applications which perform well in an unloaded network
  - rate

Problems with IntServ

- Scalability: per-flow state & classification
  - Aggregation/encapsulation techniques can help
  - Can overprovision big links, per-flow ok on small links
  - Scalability can be fixed - but no second chance
- Economic arrangements:
  - Need sophisticated settlements between ISPs
  - Contemporary settlements are primitive
    - Unidirectional, or barter
- User charging mechanisms: need QoS pricing
  - On a fine-grained basis

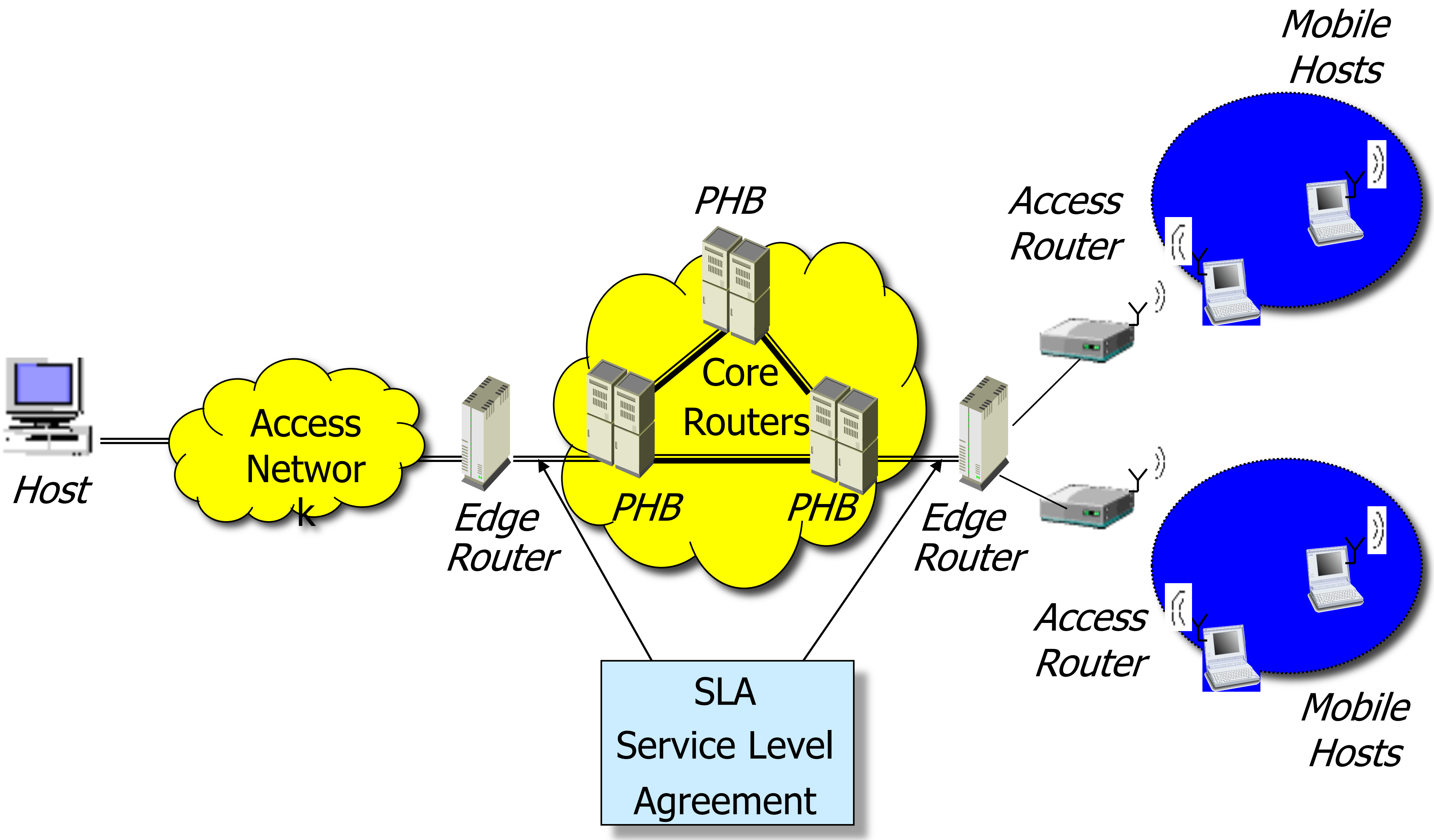
Differentiated Services

- Intended to address the following difficulties with Intserv and RSVP
  - **Scalability**: maintaining states by routers in high speed networks is difficult due to the very large number of flows
  - **Flexible Service Models**: IntServ has only two classes, want to provide more classes - *relative* service distinction (Platinum, Gold, Silver, …)
  - **Simpler signaling**: (than RSVP) many applications and users may only want to specify a more qualitative notion of service

Differentiated Services

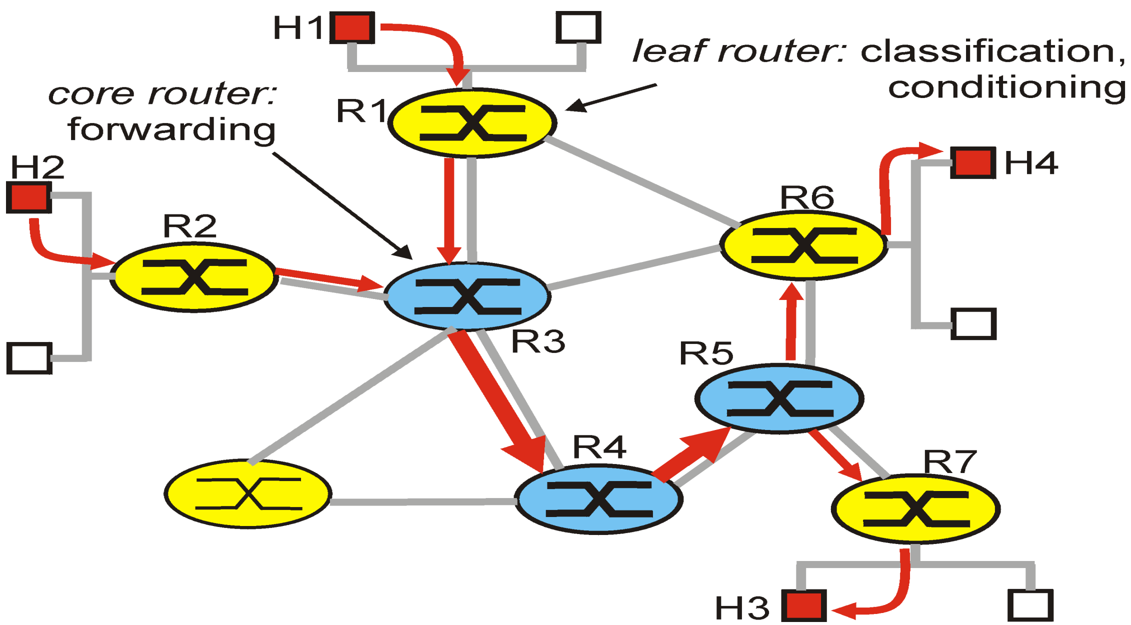
- Approach:
  - Only simple functions in the core, and relatively complex functions at edge routers (or hosts)
  - Do not define service classes, instead provide functional components with which service classes can be built

End-to-end *DiffServ* architecture



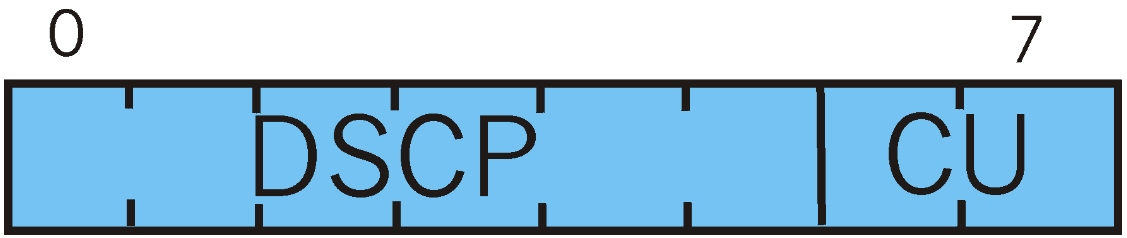
Edge Functions

- At DS-capable host or first DS-capable router
- **Classification**: edge node marks packets according to classification rules to be specified (manually by admin, or by some TBD protocol)
- **Traffic Conditioning**: edge node may delay and then forward or may discard



Classification and Conditioning

- Packet is marked in the Type of Service (TOS) in IPv4, and Traffic Class in IPv6
- 6 bits used for Differentiated Service Code Point (DSCP) and determine PHB that the packet will receive



Core Functions

- **Forwarding**: according to "Per-Hop-Behavior" or PHB specified for the particular packet class; such PHB is strictly based on class marking (no other header fields can be used to influence PHB)
- QoS, if sufficient provisioning

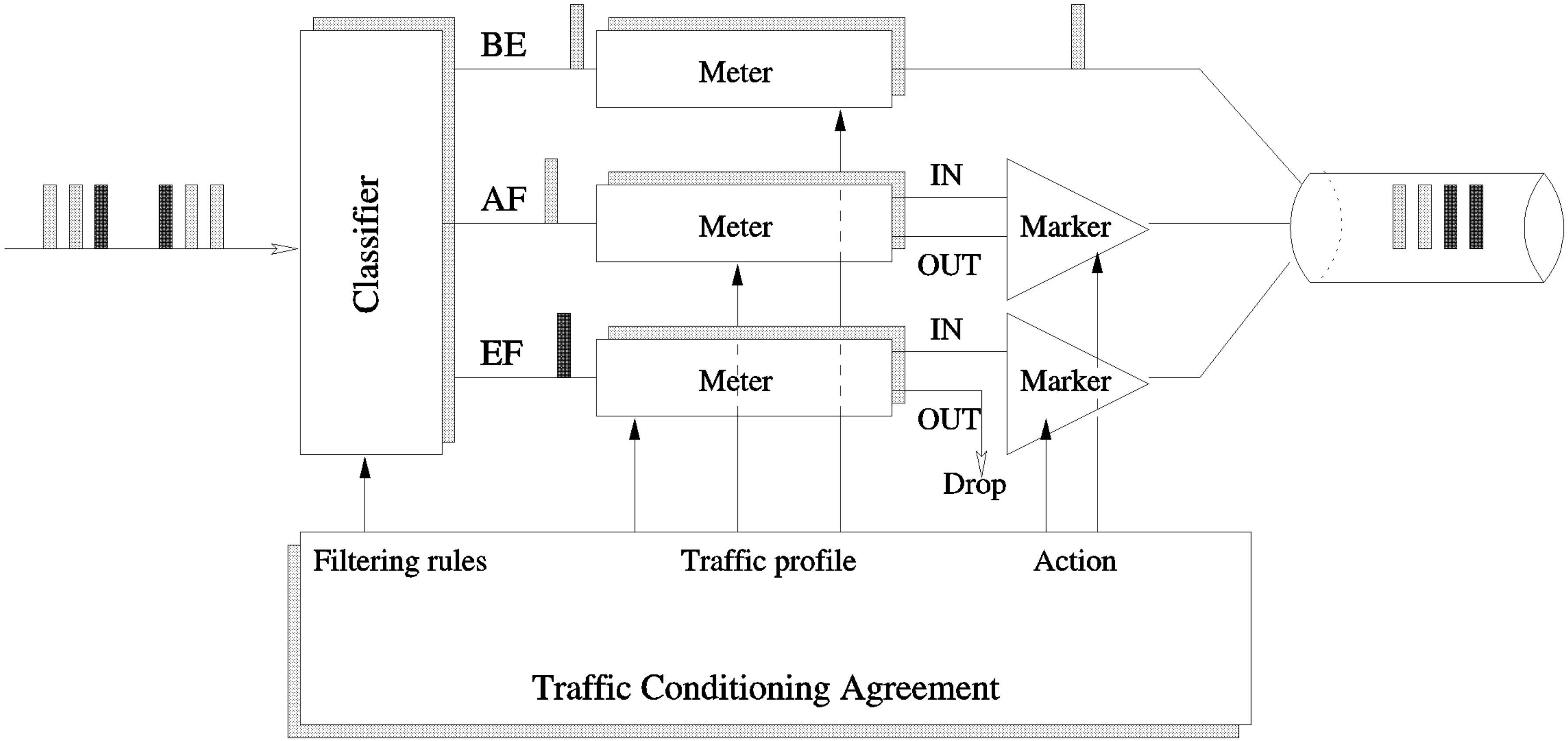
- BIG ADVANTAGE:

  No state info to be maintained by routers!

DiffServ service classes

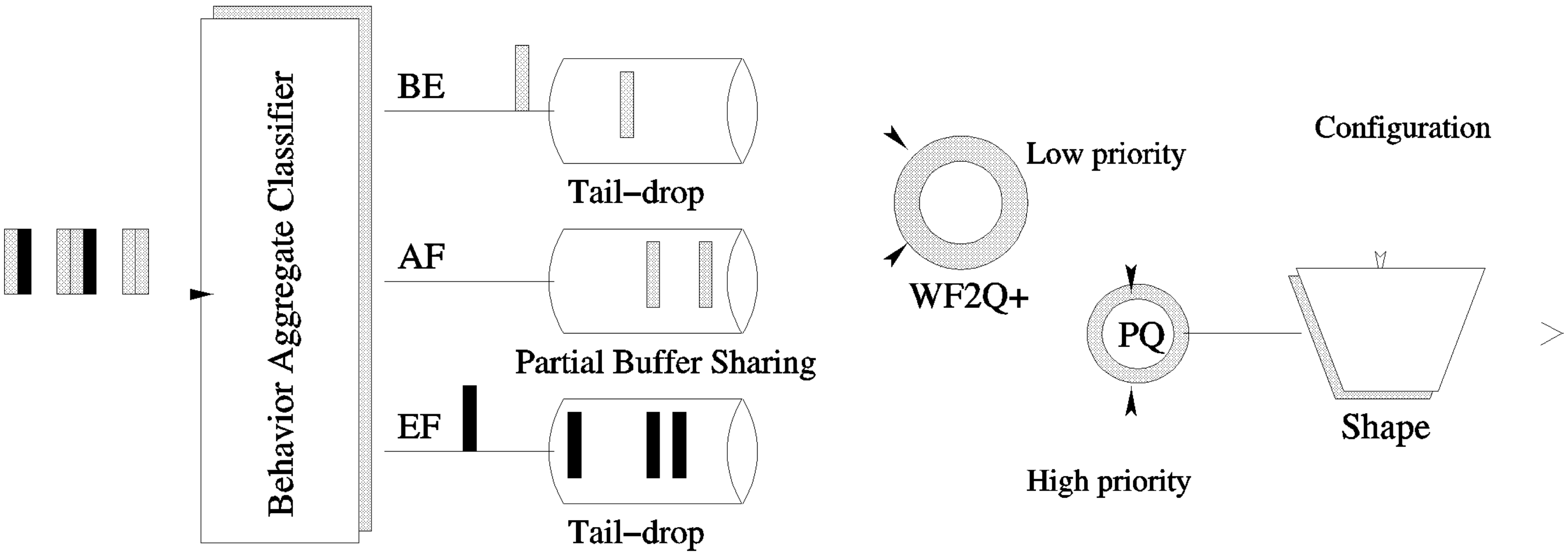
- Two main types of application
  - interactive (games, interactive distributed simulations, VoIP, device control)
    - delay, jitter
  - elastic (data transfer)
    - sustained throughput
- Traffic classes
  - EF (Expedited Forwarding)
    - short delay, small jitter
  - AF (Assured Forwarding)
    - minimal sustained throughput
    - 4 subclasses with 3 different drop probabilities (12 subclasses in total)
  - BE (Best Effort)

DiffServ - Edge router

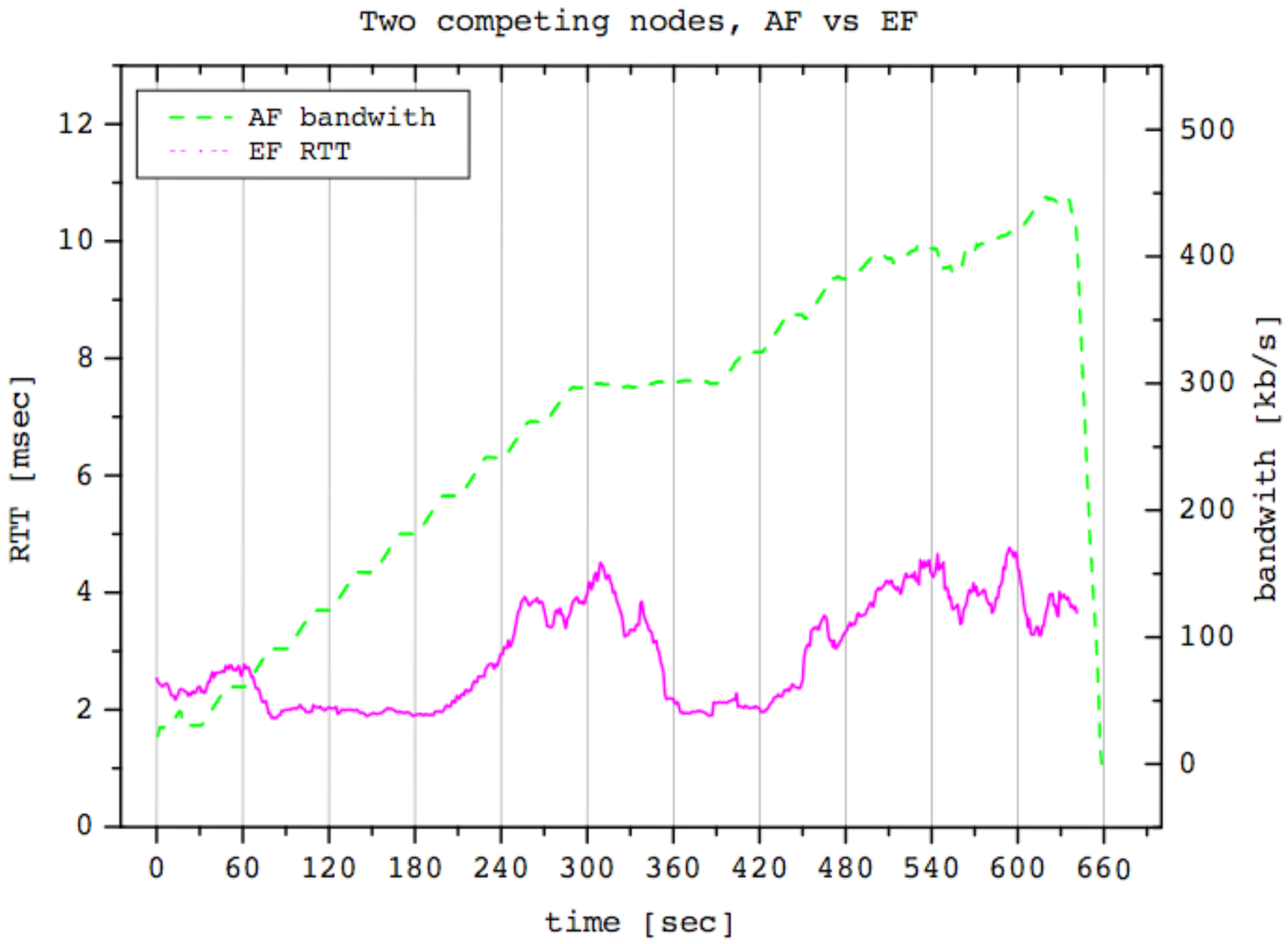


- Classification, metering, marking

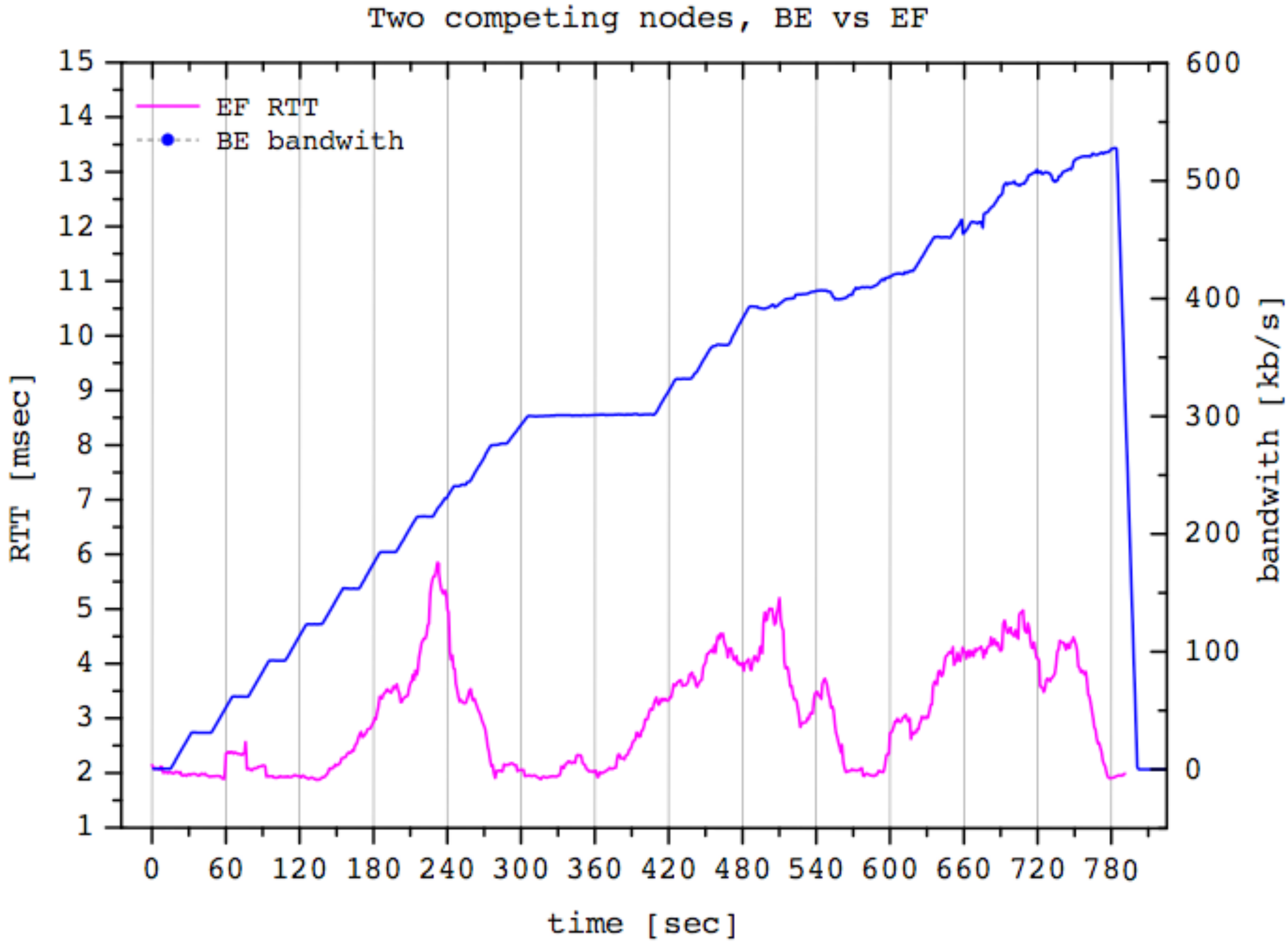
DiffServ - Core router



- Queue management and scheduling
  - EF: high priority
  - AF, BE: WFQ - Weighted Fair Queueing
- Traffic shaping



Two competing nodes, AF vs EF
AF bandwith
EF RTT
RTT [msec]
bandwith [kb/s]
time [sec]
0
2
4
6
8
10
12
0
100
200
300
400
500
0
60
120
180
240
300
360
420
480
540
600
660



Two competing nodes, BE vs EF
EF RTT
BE bandwith
RTT [msec]
bandwith [kb/s]
time [sec]
15
14
13
12
11
10
9
8
7
6
5
4
3
2
1
600
500
400
300
200
100
0
0
60
120
180
240
300
360
420
480
540
600
660
720
780

Facts to remember

- QoS in packet networks based on
  - scheduling algorithms
  - buffer management policies
- Traffic shaping helps to deal with QoS
  - limiting bursts
  - traffic description
  - traffic policing
- IETF models
  - *IntServ, DiffServ*