Policing Mechanisms

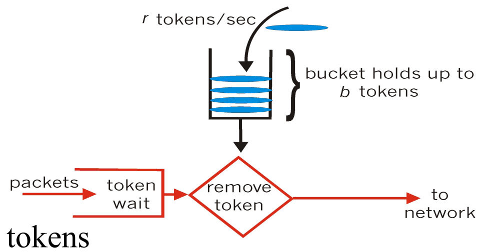
Goal: limit traffic not to exceed declared parameters

Three common-used criteria:

- *(Long term) Average Rate:* how many packets can be sent per unit time (in the long run)
  - crucial question: what is the interval length: 100 packets per sec or 6000 packets per min have the same average!
- *Peak Rate:* how many packets can be sent over a short time, e.g. 100 pps peak rate
- *Burst Size:* max. number of packets sent consecutively (with no intervening idle)

Policing Mechanisms

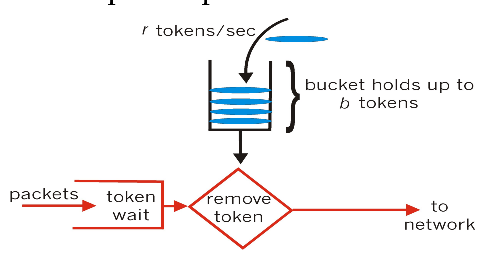
Token Bucket Filter: limits input to specified Burst Size and Average Rate.



- Bucket can hold b tokens
- Tokens generated at rate r *token/sec* unless bucket is full
- Over an interval of length t: number of packets admitted less than or equal to $(r\, t + b)$.
- A source can send a burst of no larger than b and at the average rate of no more r.
- Note: Actual mechanism works on *bytes* not *packets*.

Policing Mechanisms (con't)

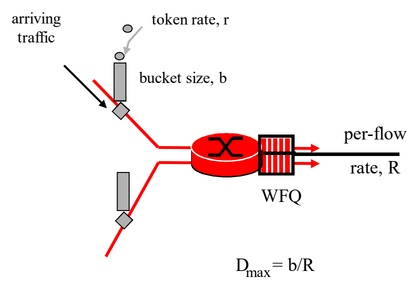
Token Bucket Filter: limits input to specified Burst Size and Average Rate.

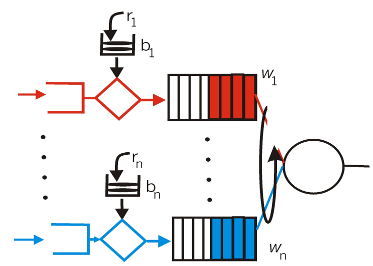


- Examples:
  - *Constant-rate* traffic of 1MBps flow: *r=1MBps, b=1B*
  - *Bursty-type* traffic of 0.5 MBps/2sec + 2 MBps/1sec: *r=1MBps, b=1MB*
- Note: the same flow can be described by different Token Bucket Filter; however explicit bandwidth description avoids over-allocation of network resources.

Policing Mechanisms (more)

- Token bucket and WFQ combined are used to provide guaranteed upper bound on delay, i.e., *QoS guarantee* !





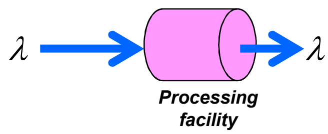
Call Admission

<u>Arriving session must:</u>

- Declare its QOS requirement
  - *R-spec*: defines service requested from network
    - Controlled-load or guaranteed (delay target)
- Characterize traffic it will send into network
  - T-spec: defines traffic characteristics
    - Token bucket filter (avg bw + burstiness)
- Signaling protocol: needed to carry R-spec and T-spec to routers (where reservation is required)
  - RSVP

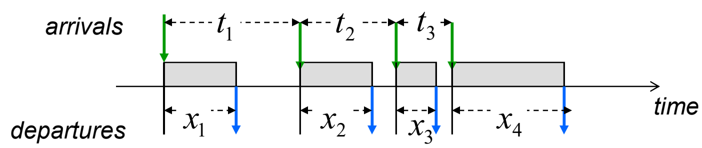
Call Admission

- Assume a processing facility with input rate of λ ["customers"/sec] and with processing time of the *i*-th customer of x_i [sec]:



We will call a traffic bursty, if $\lambda \cdot \bar{x} << 1$

- In our case, the processing facility is a link with a transmitter and the processing time is the message transmission time:



$$\text{arrival rate} \equiv \lambda = E(1/t_i); \qquad \text{link utilization} \equiv \rho = \frac{E(x_i)}{E(t_i)} = \frac{\bar{x}}{\bar{t}} = \lambda \cdot \bar{x}$$

Call Admission

- Consider the probability that an arrival finds all the *m* "servers" busy; i.e., p_m:

$$p_m = \frac{\rho^m / m!}{\sum_{n=0}^{m} \rho^n / n!},$$

- thus p_m is the probability that an arriving call will be lost and is referred to as *Erlang B formula.*
- *Erlang B formula* is extensively used in engineering of telephone system, as it allows to calculate the required number of trunks for some *Grade of Service* level.

Call Admission

- *Erlang B formula* is often written as:

$$B(C,\Gamma) \equiv \frac{\Gamma^C / C!}{\sum_{i=0}^{C} \Gamma^i / i!}$$

- where C is the number of trunks, Γ is the total offered load, and $B(C, \Gamma)$ is the probability of blockage (i.e., *Grade of Service).*
- Note that the actual *carried traffic,* $\Gamma_{carried}$*,* is $\Gamma_{carried} = \Gamma(1-B(C, \Gamma))$.
- Assume a system with 10 voice channels. A user uses his/her phone once every 2.5 [hours] for the duration of 6 [min], what is the number of users that can be supported in the system for 1% blocking probability?

$$\rho = \lambda \cdot \bar{x} = \frac{1}{2.5} \cdot \frac{6}{60} = 0.04\,; \quad B(10,\Gamma) = 0.01 \;\rightarrow\; \Gamma = 4.45[Erl] \;\rightarrow\; N = \left\lfloor \frac{4.45}{0.04} \right\rfloor = 111\ [users]$$

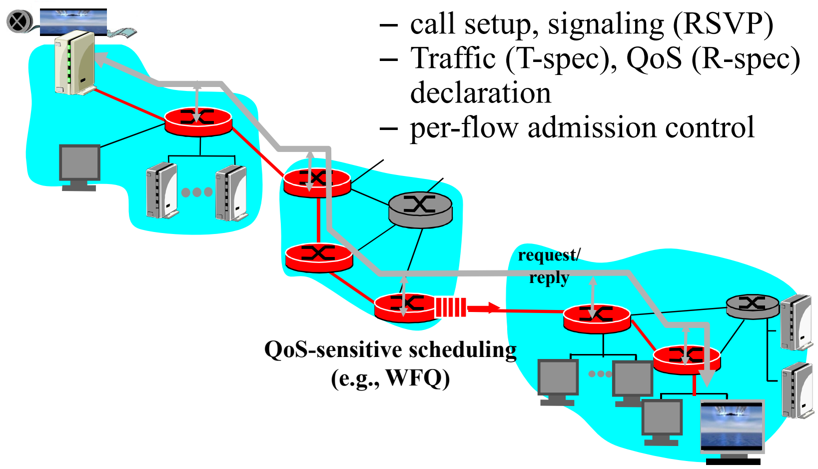
IETF Integrated Services

- Architecture for providing QOS guarantees in IP networks for *individual application sessions*
- Resource reservation: routers maintain state info of allocated resources, QoS req's
- Call setup: admit/deny new call setup requests

Question to consider: can newly arriving flow be admitted with performance guarantees while not violating QoS guarantees made to already admitted flows?

Intserv: QoS guarantee scenario

- Resource reservation
  - call setup, signaling (RSVP)
  - Traffic (T-spec), QoS (R-spec) declaration
  - per-flow admission control



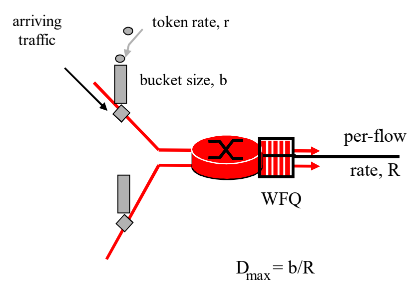
IntServ QoS: Service models [RFC2211, RFC2212]

Guaranteed service:

- worst case traffic arrival: token bucket policed source
- simple (mathematically provable) *bound* on delay [Parekh 1992, Cruz 1988]

Controlled load service:

"a quality of service closely approximating the QoS that same flow would receive from an unloaded network element."



Reservation Protocol (RSVP)

- Proposed Internet standard: *RSVP*
  - Performs signaling to set up reservations state for a session
  - A session is a simplex data flow sent to a unicast or multicast address
  - Multiple senders and receivers can be in the same session
    - Designed to support multicast
- Consistent with robustness of today's connectionless model
  - Uses soft state (refreshed periodically)
- Two messages: *PATH* and *RESV*
  - Source transmits *PATH* messages every 30 seconds
  - Destination responds with *RESV* message
- Does not specify how resources are to be reserved
  - Rather – a mechanism for communicating needs
- Separation of control (signaling) and data (forwarding)

RSVP Basic Operations

- Sender: sends *PATH* message via the data delivery path
  - Sets up the path state at each router (+ the address of previous hop)
- PATH also specifies
  - Source traffic characteristics
    - Use token bucket
- Receiver sends *RESV* message on the reverse path
  - Specifies the reservation style, QoS desired (*RSpec*)
  - Set up the reservation state at each router
- *RESV* specifies
  - Service requirements
  - Source traffic characteristics (from *PATH*)
  - Filter specification, i.e., what senders can use for the reservation
  - Based on this information, routers perform the reservation
- Things to notice
  - Receiver initiated reservation
  - Decouple routing from reservation

IETF Differentiated Services

Concerns with Intserv:

- Scalability: signaling, maintaining per-flow router state is difficult with large number of flows
- Flexible Service Models: Intserv has only two classes. Also was is needed are “qualitative” service classes
  - relative service distinction: Platinum, Gold, Silver

Diffserv approach:

- simple functions in network core, relatively complex functions at edge routers (or hosts)
- Don’t define service classes, provide functional components to build service classes

Diffserv Architecture

Edge router:

- per-flow traffic management

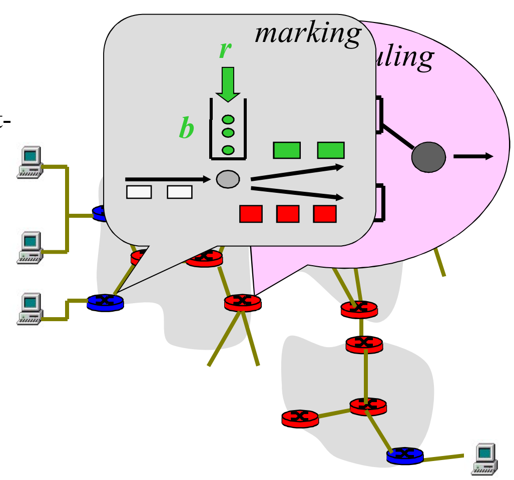
- marks packets as in-profile and out-profile

Core router:

- per class traffic management

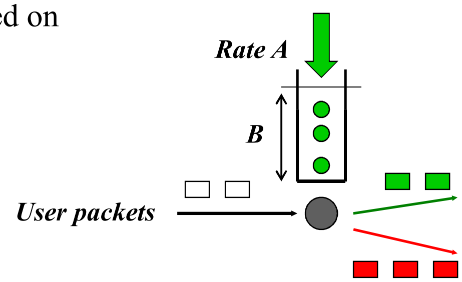
- buffering and scheduling based on marking at edge

- preference given to in-profile packets



Edge-router Packet Marking

- profile: pre-negotiated rate A, bucket size B
- packet marking at edge based on per-flow profile

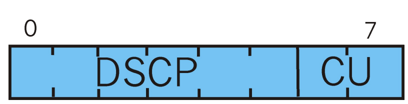


Possible usage of marking:

- class-based marking: packets of different classes marked differently
- intra-class marking: conforming portion of flow marked differently than non-conforming one

Classification and Conditioning

- Packet is marked in the Type of Service (TOS) in IPv4
- 6 bits used for *Differentiated Service Code Point* (*DSCP*) and determine *PHB* (*Per-Hop Behavior*) that the packet will receive
- 2 bits are used for ECN



Forwarding - Per Hop Behavior (Core Router Functionality)

- Examples of PHBs developed:
- *Expedited Forwarding (EF) PHB*: Low loss/latency
  - Sum of all rates to a particular interface is less than the interface's capacity
  - Sum of all the rates of EF packets in the domain is less than the capacity of the slowest link in the domain
  - Scheduling options: EF packets get priority; WFQ; etc
- *Assured Forwarding (AF) PHB*: 4 classes of traffic
  - each guaranteed minimum amount of bandwidth
  - each with three drop preference partitions
  - No latency/jitter guarantees