QoS traffic shaping

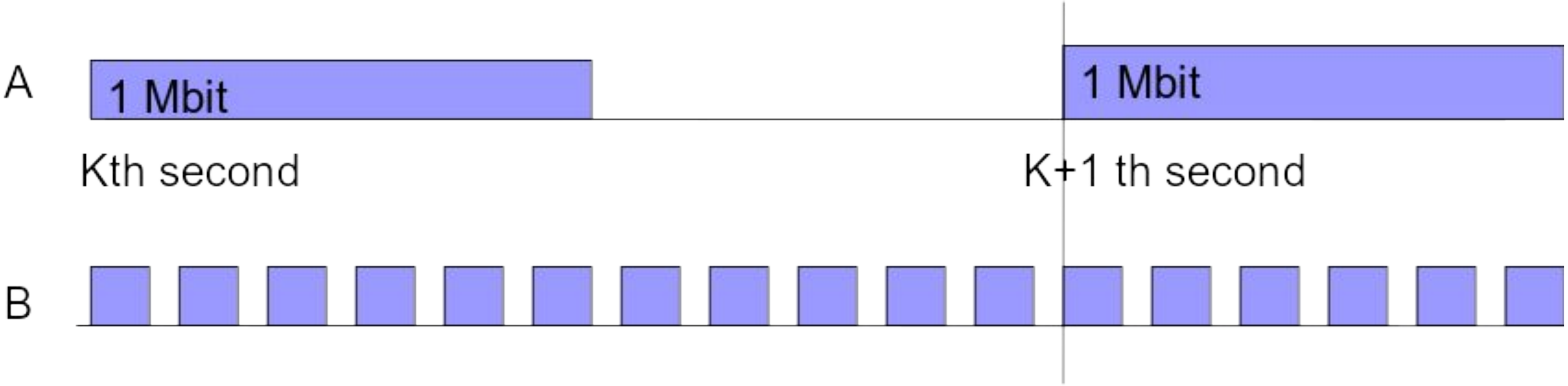
- In packet networks, admission control, reservation is not sufficient to provide QoS guarantees
- Need **traffic shaping** at the entry to network and within network
- Traffic shaping
  - Decides how packets will be sent into the network , hence regulates traffic
  - Decides whether to accept a flow's data
  - Polices flows

Traffic shaping

- Traffic shape
  - A way of a flow to describe its traffic to the network
- Based on traffic shape, network manager (s) can determine if flow should be admitted into the network
- Given traffic shape, network manager(s) can periodically monitor flow’s traffic

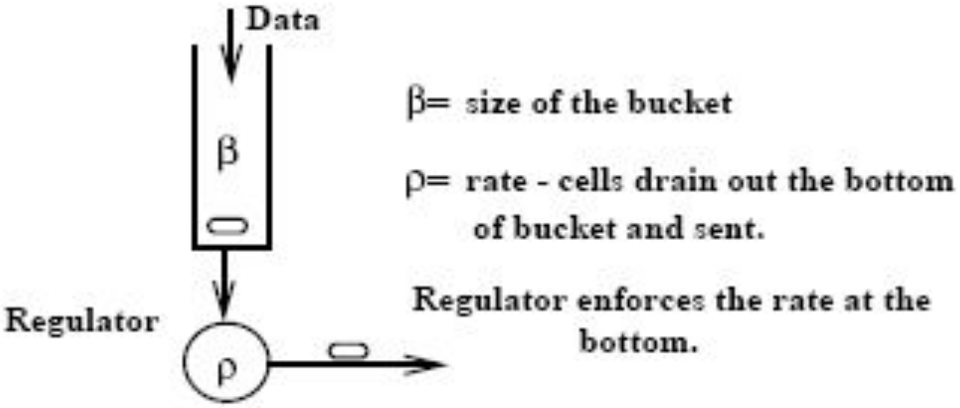
Traffic shaping example

- If we want to transmit data of 100 Mbps,
  - Traffic Shape A: Do we take 1 packet size of size 100 Mbit and send it once a second, or
  - Traffic Shape B: Do we take 1 packet of size 1 Kbit and send it every 10 microseconds?



Congestion control algos: Leaky bucket

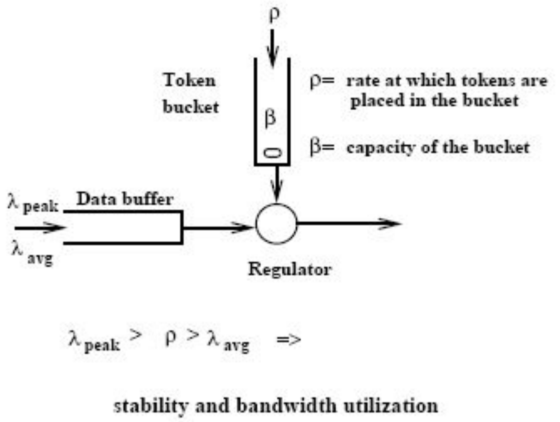
- Variable rate traffic comes in, leaves bucket at **fixed** rate
- If the bucket overflows packets are dropped
- Converts bursty traffic to uniform - avoiding congestion



Each flow has its own leaky bucket.

Congestion control algos: Token bucket

- Goal to fix with LB: don’t lose data
- Tokens added at regular intervals
- If there is a packet ready to send, remove tokens based on size
- TB discards tokens, not packets
- Allows for bursts - spend more tokens



Token bucket

- The effect of TB is different than Leaky Bucket (LB)
- Consider sending packet of size b tokens (b<β):
  - Token bucket is full – packet is sent and b tokens are removed from bucket
  - Token bucket is empty – packet must wait until b tokens drip into bucket, at which time it is sent
  - Bucket is partially full – let’s consider B tokens in bucket;
    - if b ≤ B then packet is sent immediately,
    - Else wait for remaining b-B tokens before being sent.

QoS: Integrated Services (IntServ)

- Defined service classes
  - Provides *guaranteed* service for intolerant applications
  - *Controlled load* for tolerant applications (e.g., buffered audio)
- Client *preemptively* and *actively* requests resources directly from the network using Resource Reservation Protocol (RSVP)
- Uses WFQ to isolate controlled load services from other traffic
- Uses token bucket

Problems with IntServ

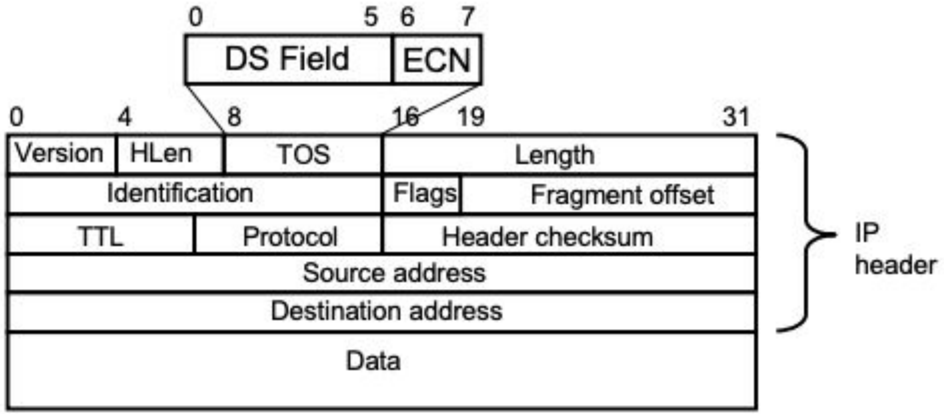
- Scalability: **per-flow state** & classification
  - Aggregation/encapsulation techniques can help
  - Can **overprovision** big links, per-flow ok on small links
  - Scalability can be fixed - but it's difficult
- Economic arrangements:
  - Need sophisticated settlements between ISPs
  - Contemporary settlements are primitive
    - Unidirectional, or barter
- User charging mechanisms: need QoS **pricing**
  - On a fine-grained basis

Differentiated services (DiffServe)

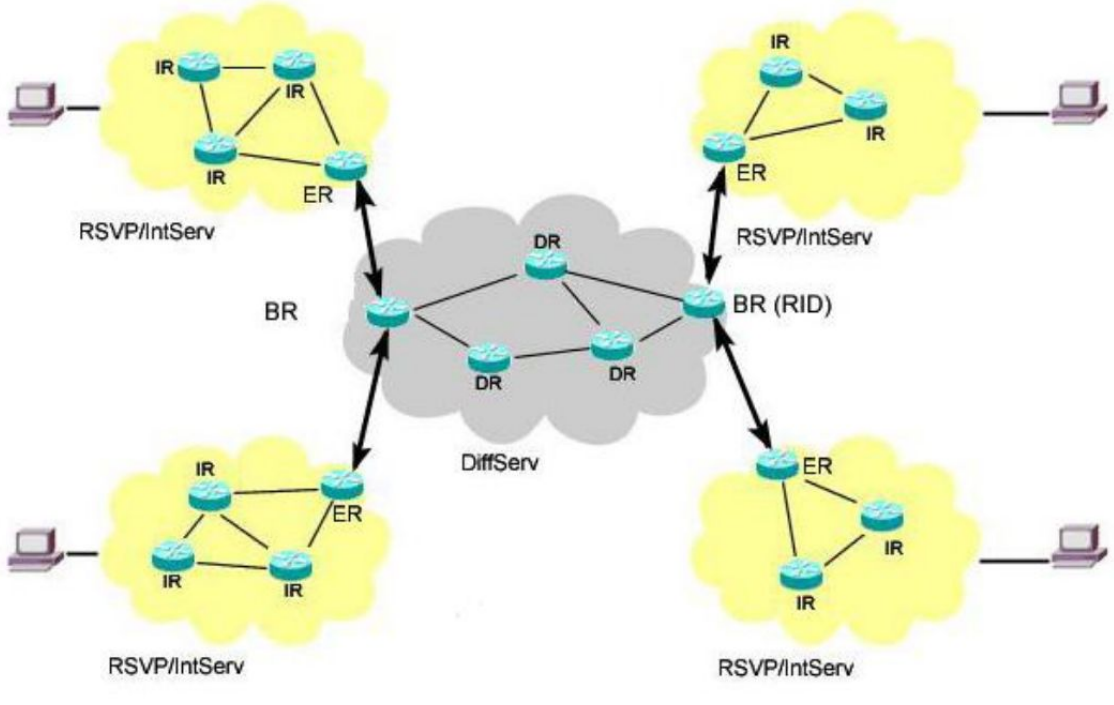
- How to know which packets get better service?
  - Bits in packet header *marked by network*, not client
  - No preemptive reservation
- Give some traffic better treatment than other
  - Application requirements: interactive vs. bulk transfer
  - Economic arrangements: first-class versus coach
- What kind of better service could you give?
  - Fewer drops
  - Lower delay
  - Lower delay variation (jitter)
- Deals with traffic in aggregate
  - Provides weaker service guarantees
  - But much more scalable

Differentiated services (DiffServe)

- Ingress routers - entrance to a DiffServ domain
  - Police or shape traffic
  - Set Differentiated Service Code Point (DSCP) in IP header
- Core routers
  - Implement Per Hop Behavior (PHB) for each DSCP
  - Process packets based on DSCP



Combining IntServe and DiffServe



QoS today

- End-to-end QoS across multiple providers/domains is not available today
- Issue #1: complexity of payment
    - Requires payment system among multiple parties
        - And agreement on what constitutes service
    - Diffserv tries to structure this as series of bilateral agreements …
        - … but lessens likelihood of end-to-end service
        - Architecture includes notion of “Bandwidth Broker” for end-to-end provisioning
            - Solid design has proved elusive
    - Need infrastructure for metering/billing end user

QoS today

- Issue #2: prevalence of overprovisioning
  - Within a large ISP, links tend to have plenty of headroom
  - Inter-ISP links are not over provisioned, however
- Is overprovisioning enough?
  - If so, is this only because access links are slow?
  - What about Korea, Japan, and other countries with fast access links?
  - Disconnect: ISPs overprovision, users get bad service
- Key difference: intra-ISP vs. general end-to-end

Exploiting lack of e2e QoS

- Suppose an ISP offers their own Internet service
  - E.g., portal (ala' Yahoo) or search engine (ala' Google)
- Then it's in their interest that service to Yahoo or Google is inferior
  - So customers prefer to use their value-added services
- ISP can
  - recognize traffic to competitor and demote it
  - charge competitor if they want well-provisioned paths
  - just not put effort/$ into high-capacity interconnects w/other ISPs; congestion provides traffic demotion directly
  - Works particularly well for large providers w/ lots of valuable content

QoS summary

- Basic mechanism for achieving better-than-best-effort performance: **scheduling**
  - Multiple queues allow priority service
  - **Fair queuing** provides isolation between flows
- IntServ provides per-**flow** performance guarantees
  - But lacks scalability
- DiffServ provides per-**aggregate** tiers of relative perf.
  - Scalable, but not as powerful
- Neither is generally available end-to-end today
- ISPs manipulating what services receive what performance raises issues of: **network neutrality**