Electrical Engineering E6761
Computer Communication Networks

Lecture 9
QoS Support

Professor Dan Rubenstein

Tues 4:10-6:40, Mudd 1127

Course URL:
http://www.cs.columbia.edu/~danr/EE6761

1

Overview

- Continuation from last time (Real-Time transport layer)
  - TCP-friendliness
  - multicast
- Network Service Models – beyond best-effort?
  - Int-Serv
    - RSVP, MBAC
  - Diff-Serv
  - Dynamic Packet State
  - MPLS

2

Review

- Why is there a need for different network service models?
- Some apps don't work well on top of the IP best-effort model
  - can't control loss rates
  - can't control packet delay
  - No way to protect other sessions from demanding bandwidth requirements
- Problem: Different apps have so many different kinds of service requirements
  - file transfer: rate-adaptive, but too slow is annoying
  - uncompressed audio: low delay, low loss, constant rate
  - MPEG video: low delay, low loss, high variable rate
  - distributed gaming: low delay, low variable rate
- Can one Internet service model satisfy all app's requirements?

3

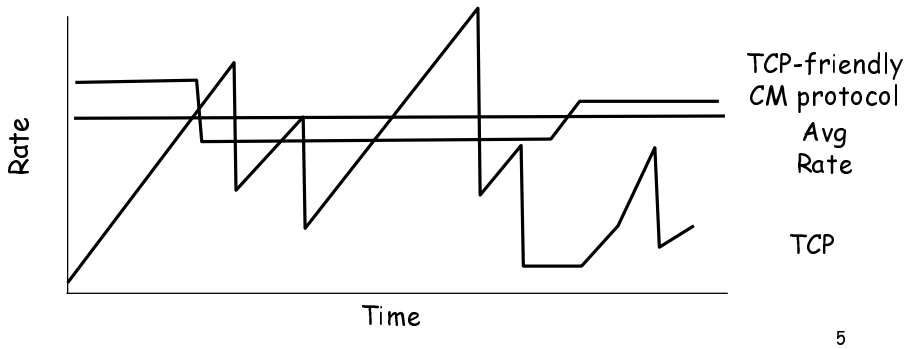
TCP-fair CM transmission

- Idea: Continuous-media protocols should not use more than their "fair share" of network bandwidth
- Q: What determines a fair share
- One possible answer: TCP could
  - A flow is TCP-fair if its average rate matches what TCP's average rate would be on the same path
  - A flow is TCP-friendly if its average rate is less than or equal to the TCP-fair rate
- How to determine the TCP-fair rate?
  - TCP's rate is a function of RTT & loss rate p
  - $Rate_{TCP} \approx 1.3 / (RTT \sqrt{p})$ (for "normal" values of p)
  - Over a long time-scale, make the CM-rate match the formula rate

4

TCP-fair Congestion Control

- Average rate same as TCP travelling along same data-path (rate computed via equation), but CM protocol has less rate variance



5

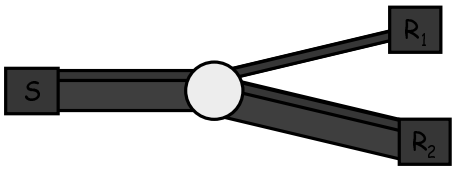
Multicast Transmission of Real-Time Streams

- Goal:
  - send same real-time transmission to many receivers
  - make efficient use of bandwidth (multicast)
  - give each receiver the best service possible
- Q: Is the IP multicast paradigm the right way to do this?

6

Single-rate Multicast

- In IP Multicast, each data packet is transmitted to all receivers joined to the group
- Each multicast group provides a single-rate stream to all receivers joined to the group

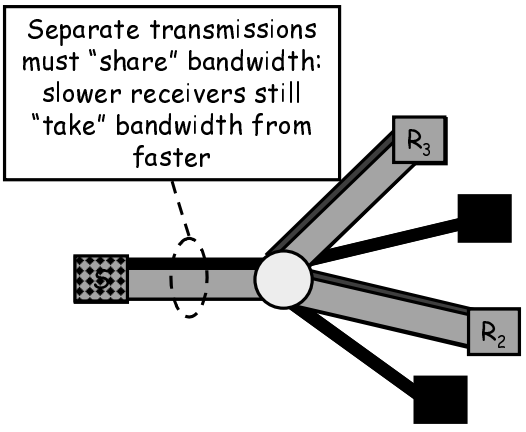


- R_2's rate (and hence quality of transmission) forced down by "slower" receiver R_1
- How can receivers in same session receive at differing rates?

7

Multi-rate Multicast: Destination Set Splitting

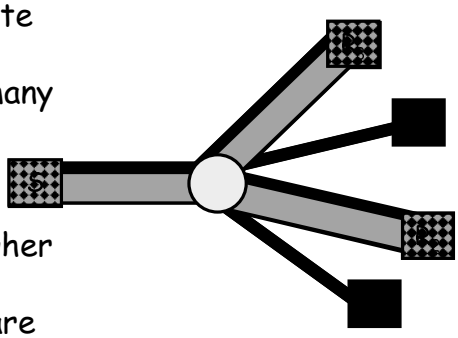
- Place session receivers into separate multicast groups that have approximately same bandwidth requirements
- Send transmission at different rates to different groups



8

Multi-rate Multicast: Layering

- Encode signal into layers
- Send layers over separate multicast groups
- Each receiver joins as many layers as links on its network path permit
- More layers joined = higher rate
- Unanswered Question: are layered codecs less efficient than unlayered codecs?



9

Transport-Layer Real-time summary

- Many ideas to improve real-time transmission over best-effort networks
  - coping with jitter: buffering and adaptive playout
  - coping with loss: forward error correction (FEC)
  - protocols: RTP, RTCP, RTSP, H.323,…
- Real-Time service still unpredictable
- Conclusion: only handling real-time at the transport-layer insufficient
  - possible exception: unlimited bandwidth
  - must still cope with potentially high queuing delay

10

Network-Layer Approaches to Real-Time

- What can be done at the network layer (in routers) to benefit performance of real-time apps?
- Want a solution that
  - meets app requirements
  - keeps routers simple
    - maintain little state
    - minimal processing

11

Facts

- For apps with QoS requirements, one of two options
  - use call-admission:
    - app specifies requirements to network
    - network determines if there is "room" for the app
    - app accepted if there is room, rejected otherwise
  - application adapts to network conditions
    - network can give preferential treatment to certain flows (without guarantees)
    - available bandwidth drops, change encoding
    - look for opportunities to buffer, cache, prefetch
    - design to tolerate moderate losses (FEC, loss-tolerant codecs)

12

Problems

- Call Admission
  - Every router must be able to guarantee availability of resources
  - may require lots of signaling
  - how should the guarantee be specified
    - constant bit-rate guarantee? (CBR)
    - leaky-bucket guarantee?
    - WFQ guarantee?
  - requires policing (make sure flows only take what they asked for)
  - complicated, heavy state
  - flow can be rejected ☹
- Adaptive Apps
  - How much should an app be able / willing to adapt?
  - if can't adapt far enough, must abort (i.e., still rejected)
  - service will be less predictable

13

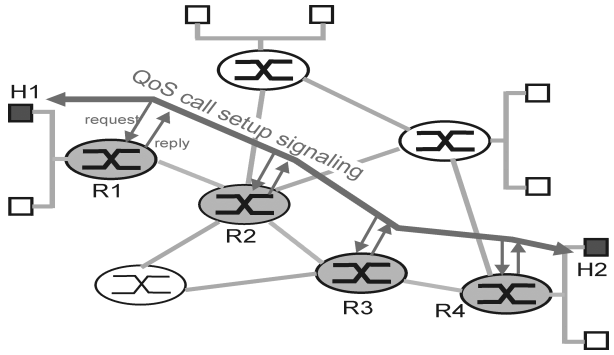
Comparison of Proposed Approaches

Name	What is it?	usage guarantees QoS	complexity
Int-Serv	Reservation framework	Y	high
RSVP	Reservation protocol	w/Int-Serv	high
Diff-Serv	Priority framework	N	low
MPLS	label-switching (circuit-building) framework	In future?	?

14

Integrated Services

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



Integrated Services: Classes

- **Guaranteed QOS**
  - provides 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
- **Controlled Load**
  - provides a QOS closely approximating that provided by an unloaded router
  - envisioned for today's IP network real-time applications which perform well in an unloaded network

16

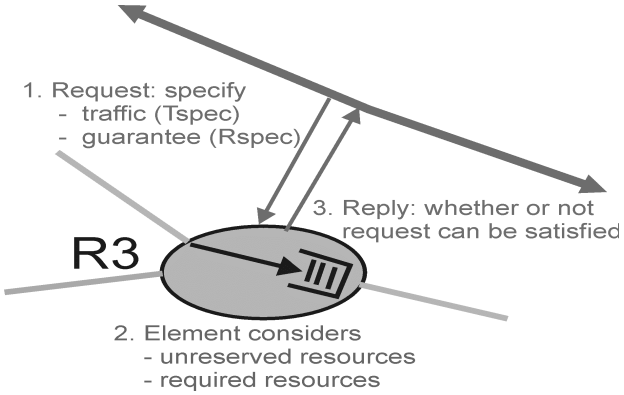
Call Admission for Guaranteed QoS

- Session must first declare its QOS requirement and characterize the traffic it will send through the network
  - **R-spec**: defines the QOS being requested
    - rate router should reserve for flow
    - delay that should be reserved
  - **T-spec**: defines the traffic characteristics
    - leaky bucket + peak rate, pkt size info
- A signaling protocol is needed to carry the R-spec and T-spec to the routers where reservation is required
  - RSVP is a leading candidate for such signaling protocol

17

Call Admission

- Call Admission: routers will admit calls based on their R-spec and T-spec and based on the current resource allocated at the routers to other calls.



18

T-Spec

- Defines traffic characteristics in terms of
  - leaky bucket model (r = rate, b = bucket size)
  - peak rate (p = how fast flow might fill bucket)
  - maximum segment size (M)
  - minimum segment size (m)
- Traffic must remain below M + min(pT, rT+b-M) for all possible times T
  - M instantaneous bits permitted (pkt arrival)
  - M + pT: can't receive more than 1 pkt at rate higher than peak rate
  - should never go beyond leaky bucket capacity of rT+b

19

R-Spec

- Defines minimum requirements desired by flow(s)
  - R: rate at which packets may be fed to a router
  - S: the slack time allowed (time from entry to destination)
  - modified by router
    - Let (R_{in}, S_{in}) be values that come in
    - Let (R_{out}, S_{out}) be values that go out
    - $S_{in} - S_{out}$ = max time spent at router
- If the router allocates buffer size β to flow and processes flow pkts at rate ρ then
  - $R_{out} = \min(R_{in}, \rho)$
  - $S_{out} = S_{in} - \beta/\rho$
- If any of the following conditions fail to hold, then reject flow
  - $\rho \geq r$ (rate bound)
  - $\beta \geq b$ (bucket bound)
  - $S_{out} < 0$ (delay bound)

20

Call Admission for Controlled Load

- A more flexible paradigm
  - does not guarantee against losses, delays
  - only makes them less likely
- only T-Spec is used
  - routers do not admit more than they can handle over long timescales
  - short time-scale behavior unprotected (due to lack of R-Spec)
- In comparison to QoS-Guaranteed Call Admission
  - more flexible admission policy
  - looser guarantees
  - depends on application's ability to adapt
    - handle low loss rates
    - cope with variable delays / jitter

21

Scalability: combining T-Specs

- Problem: Maintaining state for every flow is very expensive
- Sol'n: combine several flows' states (i.e., T-Specs) into a single state
  - Must stay conservative (i.e., must meet QoS reqmts of the flows)
  - Several models for combining
    - Summing: all flows might be active at the same time
    - Merging: only one of several flows active at a given time (e.g., a teleconference)

22

Combining T-Specs

- Given two T-Specs (r_1, b_1, p_1, m_1, M_1) and (r_2, b_2, p_2, m_2, M_2)
- The summed T-Spec is
  (r_1+r_2, b_1+b_2, p_1+p_2, $\min(m_1,m_2)$, $\max(M_1,M_2)$)
- The merged T-Spec is
  ($\max(r_1,r_2)$, $\max(b_1,b_2)$, $\max(p_1,p_2)$, $\min(m_1,m_2)$, $\max(M_1,M_2)$)
- Merging makes better use of resources
  - less state at router
  - less buffer and bandwidth reserved
  - but how to police at network edges?
  - and how common?
- Summing yields a tradeoff
  - less state at router
  - what to do downstream if flows split directions downstream?

23

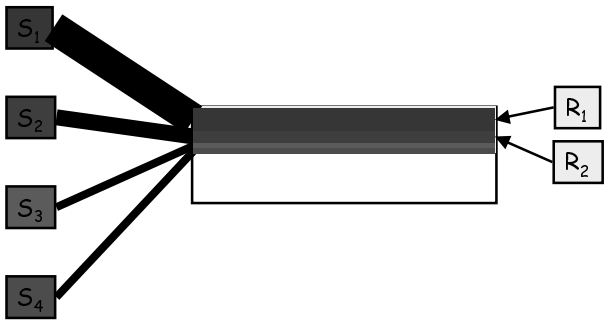
RSVP

- Int-Serv is just the network framework for bandwidth reservations
- Need a protocol used by routers to pass reservation info around
- Resource Reservation Protocol
  - is the protocol used to carry and coordinate setup information (e.g., T-SPEC, R-SPEC)
  - designed to scale to multicast reservations as well
  - receiver initiated (easier for multicast)
  - provides scheduling, but does not help with enforcement
  - provides support for merging flows to a receiver from multiple sources over a single multicast group

24

RSVP Merge Styles

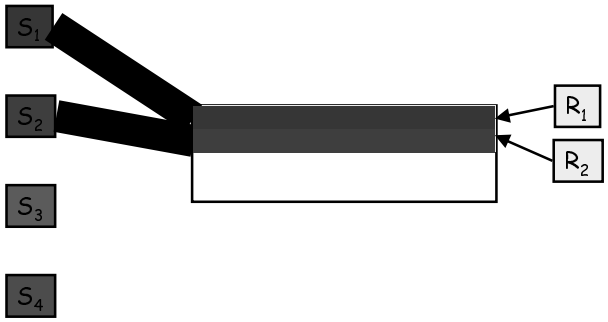
- No Filter: any sender can utilize reserved resources
  - e.g., for bandwidth:



25

RSVP Merge Styles

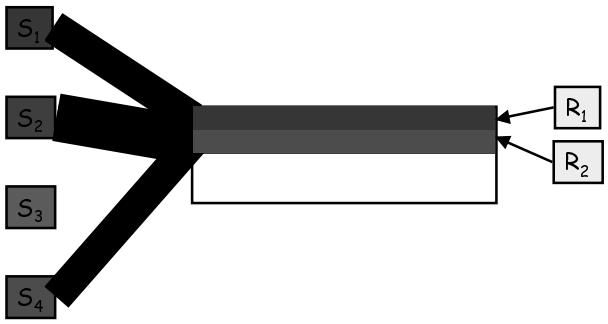
- Fixed-Filter: only specified senders can utilize reserved resources



26

RSVP Merge Styles

- Dynamic Filter: only specified senders can use resources
  - can change set of senders specified without having to renegotiate details of reservation



27

The Cost of Int-Serv / RSVP

- Int-Serv / RSVP reserve guaranteed resources for an admitted flow
- requires precise specifications of admitted flows
  - if over-specified, resources go unused
  - if under-specified, resources will be insufficient and requirements will not be met
- Problem: often difficult for apps to precisely specify their reqmt's
  - may vary with time (leaky-bucket too restrictive)
  - may not know at start of session
    - e.g., interactive session, distributed game

28

Measurement-Based Admission Control

- Idea:
  - apps don't need strict bounds on delay, loss – can adapt
  - difficult to precisely estimate resource reqmts of some apps
- flow provides conservative estimate of resource usage (i.e., upper bound)
- router estimates actual traffic load used when deciding whether there is room to admit the new session and meet its QoS reqm'ts
  - Benefit: flows need not provide precisely accurate estimates, upper bounds o.k.
  - flow can adapt if QoS reqmts not exactly met

29

MBAC example

- Traffic is divided into classes, where class j does not affect class i for j > i
- Token bucket classification (B_i, R_i)
- Let D_j be class j's expected delay
  - only lower classes affect delay
- $D_j = \sum_{i=1}^{j} B_i / (\mu - \sum_{i=1}^{j-1} R_i)$ (This is Little's Law!)
- Router takes estimates, d_j and r_j, of class j's delay and rate
- Admission decision: should a new session (β, ρ) be admitted into class j?

30

MBAC example cont'd

- New delay estimate for class j is
- $d_j + \beta / (\mu - \sum_{i=1}^{j-1} r_i)$ (bucket size increases)
- New delay estimate for class k > j is

$$d_k \underbrace{(\mu - \sum_{i=1}^{k-1} r_i) / (\mu - \sum_{i=1}^{k-1} r_i - \rho)}_{\text{delay shift due to increase in aggregate reserved rate}} + \underbrace{\beta / (\mu - \sum_{i=1}^{k-1} r_i - \rho)}_{\text{delay shift due to increase in bucket size}}$$

31

Problems with Int-Serv / Admission Control

- Lots of signalling
  - routers must communicate reservation needs
  - reservation done on a per-session basis
- How to police?
  - lots of state to maintain
  - additional processing load / complexity at routers
- Signalling and policing load increases with increased # of flows
- Routers in the core of the network handle traffic for thousands of flows
- Int-Serv approach does not scale!

32

Differentiated Services

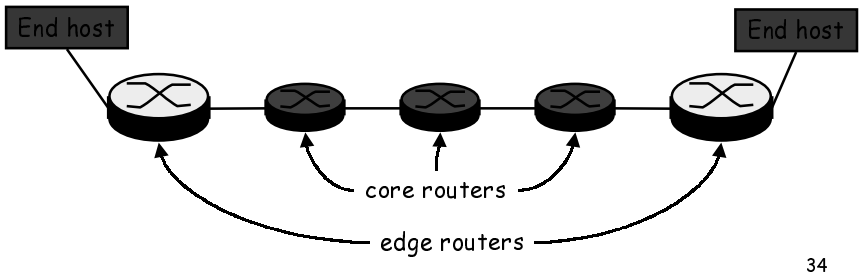
Intended to address the following difficulties with Intserv and RSVP;

- **Scalability**: maintaining states by routers in high speed networks is difficult sue to the very large number of flows
- **Flexible Service Models**: Intserv has only two classes, want to provide more qualitative service classes; want to provide '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

33

Differentiated Services

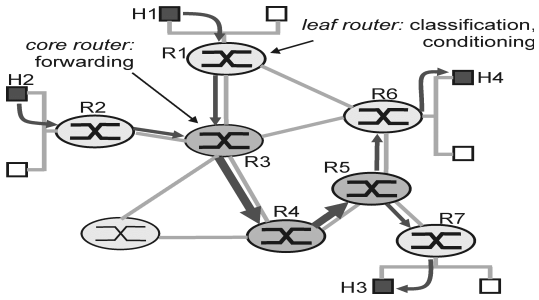
- Approach:
  - Only simple functions in the core, and relatively complex functions at edge routers (or hosts)
  - Do not define service classes, instead provides functional components with which service classes can be built



34

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



35

Core Functions

- **Forwarding**: according to "Per-Hop-Behavior" or PHB specified for the particular packet class;
  - strictly based on class marking
  - core routers need only maintain state per class
- BIG ADVANTAGE: No per-session state info to be maintained by core routers!
  - i.e., easy to implement policing in the core (if edge-routers can be trusted)
- BIG DISADVANTAGE: Can't make rigorous guarantees

36

Diff-Serv reservation step

- Diff-Serv's reservations are done at a much coarser granularity than Int-Serv
  - edge-routers reserve one profile for all sessions to a given destination
  - renegotiate profile on longer timescale (e.g., days)
  - sessions "negotiate" only with edge to fit within the profile
- Compare with Int-Serv
  - each session must "negotiate" profile with each router on path
  - negotiations are done at the rate in which sessions start

37

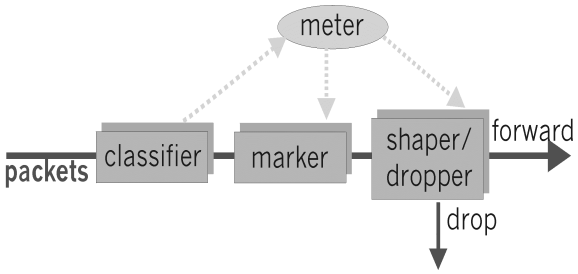
Classification and Conditioning

- 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
- 2 bits are currently unused

38

Classification and Conditioning at edge

- It may be desirable to limit traffic injection rate of some class; user declares traffic profile (eg, rate and burst size); traffic is metered and shaped if non-conforming



39

Forwarding (PHB)

- PHB result in a different observable (measurable) forwarding performance behavior
- PHB does not specify what mechanisms to use to ensure required PHB performance behavior
- Examples:
  - Class A gets x% of outgoing link bandwidth over time intervals of a specified length
  - Class A packets leave first before packets from class B

40

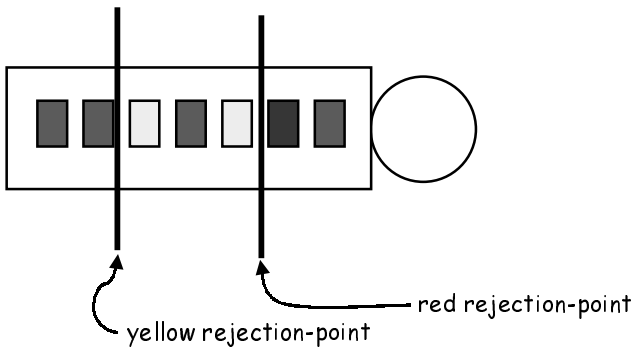
Forwarding (PHB)

- PHBs under consideration:
  - **Expedited Forwarding**: departure rate of packets from a class equals or exceeds a specified rate (logical link with a minimum guaranteed rate)
  - **Assured Forwarding**: 4 classes, each guaranteed a minimum amount of bandwidth and buffering; each with three drop preference partitions

41

Queuing Model of EF

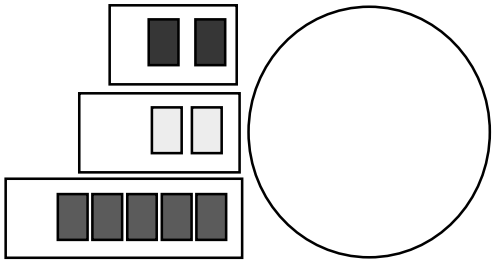
- Packets from various classes enter same queue
- denied service after queue reaches threshold
- e.g., 3 classes: green (highest priority), yellow (mid), red (lowest priority)



42

Queuing model of AF

- Packets into queue based on class
- Packets of lesser priority only serviced when no higher priority packets remain in system
- i.e., priority queue
- e.g., with 3 classes…



43

Comparison of AF and EF

- AF pros
  - higher priority class completely unaffected by lower class traffic
- AF cons
  - high priority traffic cannot use low priority traffic's buffer, even when low-priority buffer has room
  - If a session sends both high and low priority packets, packet ordering is difficult to determine

44

Differentiated Services Issues

- AF and EF are not even in a standard track yet… research ongoing
- "Virtual Leased lines" and "Olympic" services are being discussed
- Impact of crossing multiple ASs and routers that are not DS-capable

- Diff-Serv is stateless in the core, but does not give very strong guarantees
- Q: Is there a middle ground (stateless with stronger guarantees)

45

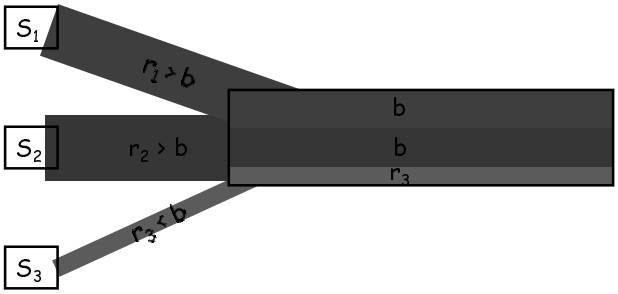
Dynamic Packet State (DPS)

- Goal: provide Int-Serv-like guarantees with Diff-Serv-like state
  - e.g., fair queueing, delay bounds
  - routers in the core should not have to keep track of individual flows
- Approach:
  - edge routers place "state" in packet header
  - core routers make decisions based on state in header
  - core routers modify state in header to reflect new state of the packet

46

DPS Example: fair queuing

- Fair queuing: if not all flows "fit" into a pipe, all flows should be bounded by same upper bound, b
- b should be chosen s.t. pipe is filled to capacity



47

DPS: Fair Queuing

- The header of each packet in flow f_i indicates the rate, r_i of its flow into the pipe
- r_i is put in the packet header
- The pipe estimates the upper bound, b, that flows should get in the pipe
- If $r_i < b$, packet passes through unchanged
- If $r_i > b$:
  - packet is dropped with probability b / r_i
  - r_i replaced in packet with b (flow's rate out of pipe)
- router continually tries to accurately estimate b
  - buffer overflows: decrease b
  - aggregate rate out less than link capacity: increase b

48

Summary

- Int-Serv:
  - strong QoS model: reservations
  - heavy state
  - high complexity reservation process
- Diff-Serv:
  - weak QoS model: classification
  - no per-flow state in core
  - low complexity
- DPS:
  - middle ground
  - requires routers to do per-packet calculations and modify headers
  - what can / should be guaranteed via DPS?
- No approach seems satisfactory
- Q: Are there other alternatives outside of the IP model?

49

MPLS

- Multiprotocol Label Switching
  - provides an alternate routing / forwarding paradigm to IP routing
  - can potentially be used to reserve resources and meet QoS requirements
  - framework for this purpose not yet established…

50

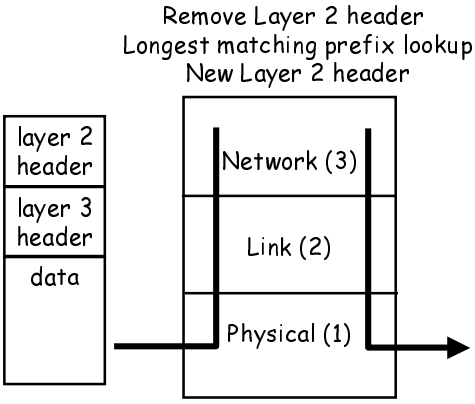
Problems with IP routing

- Slow (IP lookup at each hop)
- No choice in path to a destination (must be shortest path)
- Can't make QoS guarantees: session is forced to multiplex its packets with other flows' packets

51

MPLS

- Problem: IP switching is not the most efficient means of networking

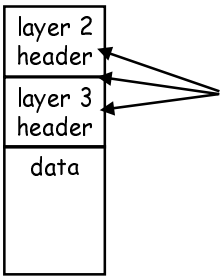


- The longest matching prefix lookup can be expensive
  - big database of prefixes
  - variable-length, bit-by-bit comparison
  - prefix can be long (32 bits)

52

Tag-Switching

- For commonly-used paths, add a special tag that quickly identifies packet destination interface
  - can be placed in various locations to be compatible with various link & network layer technologies
    - within layer 2 header
    - in separate header between layers 2 and 3 (shim)
    - as part of layer 3 header



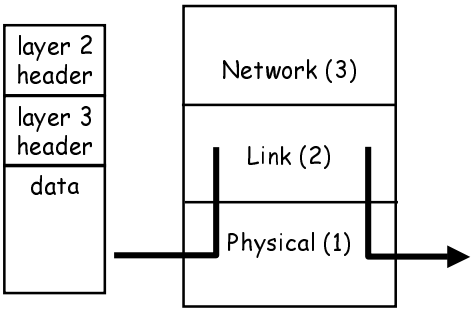
possible location of tag

- tag is a short (few-bit) identifier
- only used if there is an exact match (as opposed to longest matching prefix)

53

Tag switching cont'd

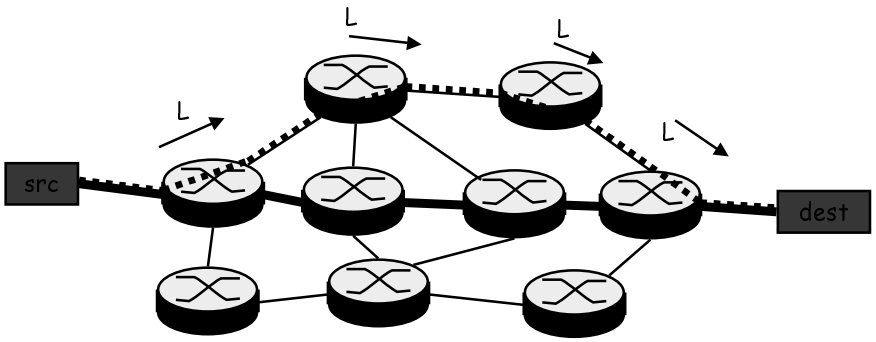
- Lookup using the small tag is much faster
  - often easy to do in hardware
  - often don't need to involve layer 3 processing



54

Circuiting with MPLS

- Can establish fixed (alternative) routes with labels

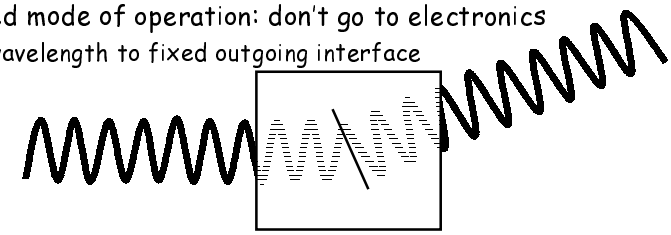


  - Note: can aggregate flows under one label
  - Also, can start labeling midway along path (i.e., router can set label)

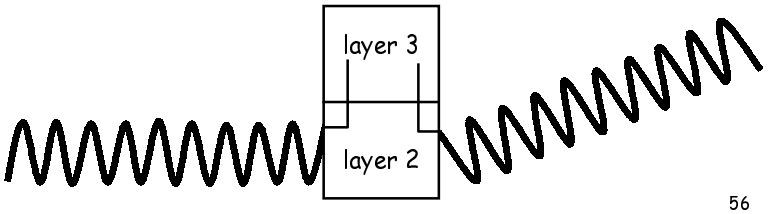
55

MPLS with Optical Nets

- Preferred mode of operation: don't go to electronics
  - map wavelength to fixed outgoing interface



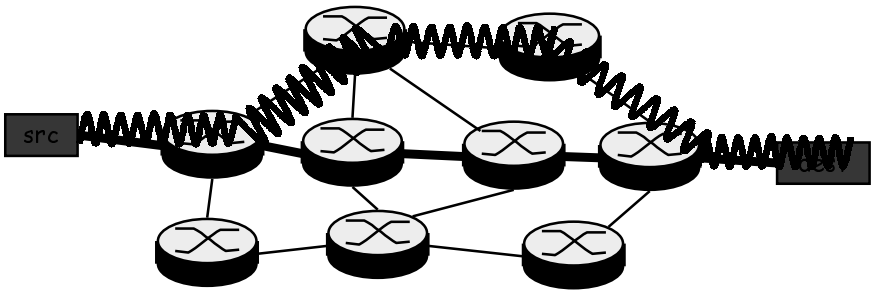
- IP-lookup requires electronics in the middle



56

All-Optical Paths via MPLS

- Reserve a wavelength (and a path) for a (set of) flow(s)



57

What won?

- IntServ Lost
  - Too much state and signaling made it impractical
  - unable to accurately quantify apps' needs in a convenient manner
- DiffServ is losing
  - Not clear what kind of service a flow gets if it buys into premium class
  - What / how many / when should flows be allowed into premium unclear
  - What happens to flows that don't make it into premium
- MPLS: still hot, but what does it change?
- The current winner: over-provisioning
  - Bandwidth is cheap these days
  - ISPs provide enough bandwidth to satisfy needs of all apps

58

Is over-provisioning the answer?

- Q: are you happy with your Internet service today?
- Problem: the peering points between ISPs
  - some traffic must travel between ISPs
  - traffic crosses peering points
  - ISPs have no incentive to make other ISPs look good, so they do not overprovision at the peering point
- Solutions:
  - ISPs duplicate content and buffer within their domain
- What to do about live / dynamically changing content?
- Will there always be enough bandwidth out there?
  - What is the next killer app?

59