The evolution of Internet QoS

Olivier Bonaventure

Department of Computing Science and Engineering
Université catholique de Louvain (UCL)
Place Sainte−Barbe, 2, B−1348, Louvain−la−Neuve (Belgium)

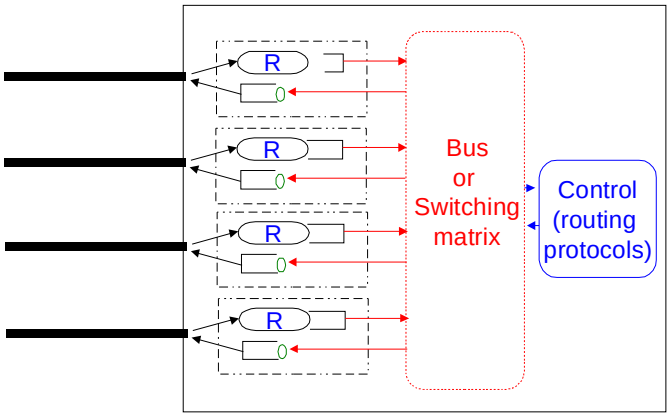
Email : Bonaventure@info .ucl.ac.be
URL : http://www.info.ucl.ac.be/people/OBO

The evolution of Internet QoS

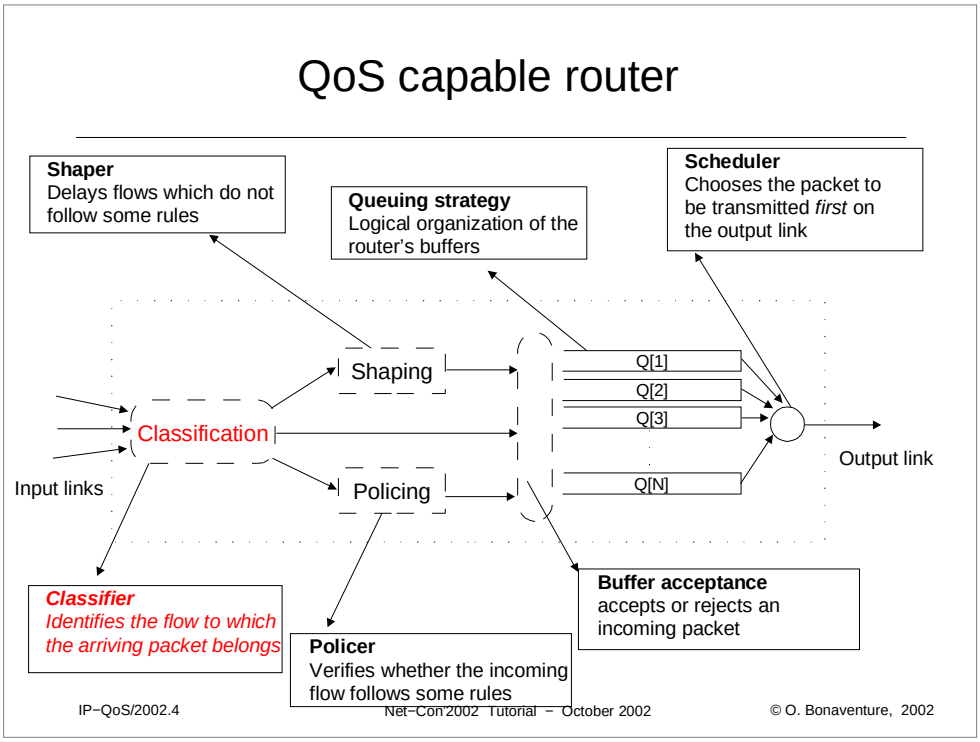
- Packet−level traffic control mechanisms
  - Classification
  - Shaping and Policing
  - Buffer acceptance algorithms
  - Scheduling algorithms
- Integrated Services
- Differentiated Services

A simple router

- Router with 4 input−output ports

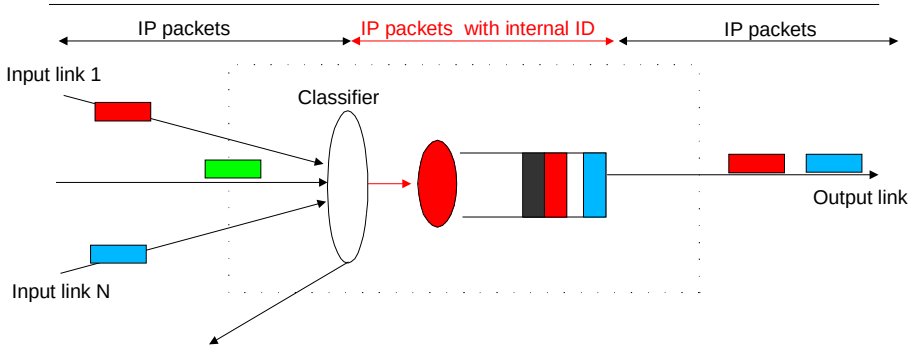


In this tutorial, we place, for pedagogical reasons, the traffic control functions on the router's output ports. It should however be noted that one some router architectures, some of these functions may be placed elsewhere.



In practice, the shaper could also be located on the output link, but we don't address this issue here to keep the picture simple and understandable.

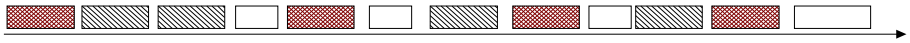
Classification



- Roles of the classifier
  - identify the flow to which an arriving packet belongs
    - identification can require complex operations
  - store this information internally so that other parts of the router will easily determine the flow of a packet
    - classification should be done at most once in each router

What is a flow ?

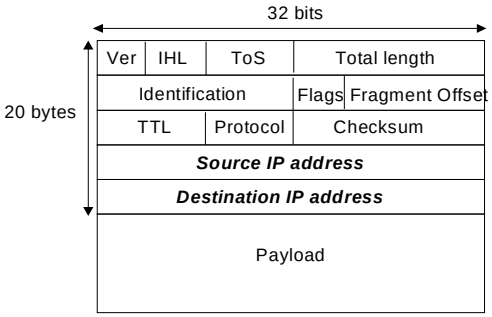
- Definition
  - a flow is a sequence of packets with one common "characteristic"
    - characteristic can be based on any field of the packets
    - a flow usually exist for some period of time



  - a layer−N flow is a sequence of packets with one common layer−N characteristic
    - layer two flow
      - e.g. ATM or frame relay circuits
    - layer three flow [IP related]
    - layer four flow [TCP or UDP related]
    - layer seven flow [application level flow]

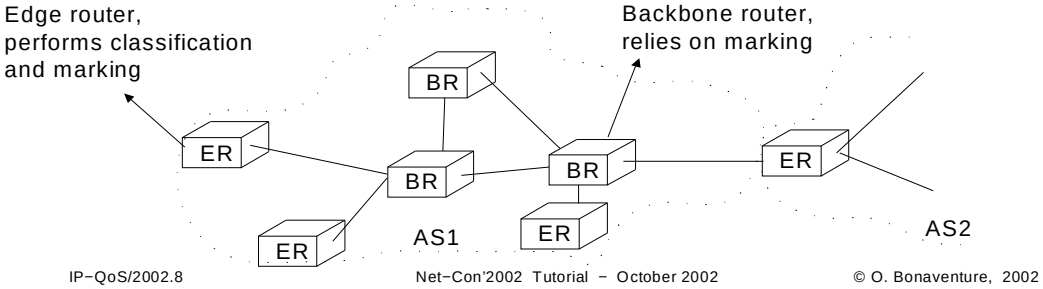
Layer−three flow

- Identification of layer−three flows
  - source and destination IP addresses with or without associated netmasks
    - e.g. all traffic from 138.48.0.0/16
  - all IP traffic with same route or BGP next hop
    - requires a route table lookup by the classifier



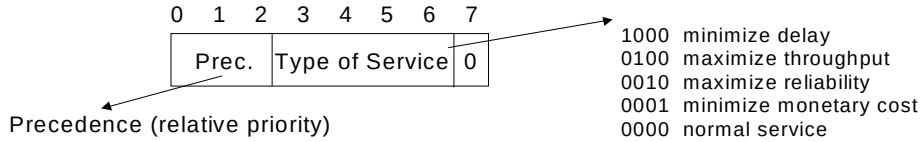
Layer−three flow (2)

- Layer−3 classification on each intermediate router can be expensive
- Alternative solution
  - perform classification at the ingress of the network
  - explicitly mark the classified packets
    - downstream (backbone) routers will rely on the marking without needing to classify each packet

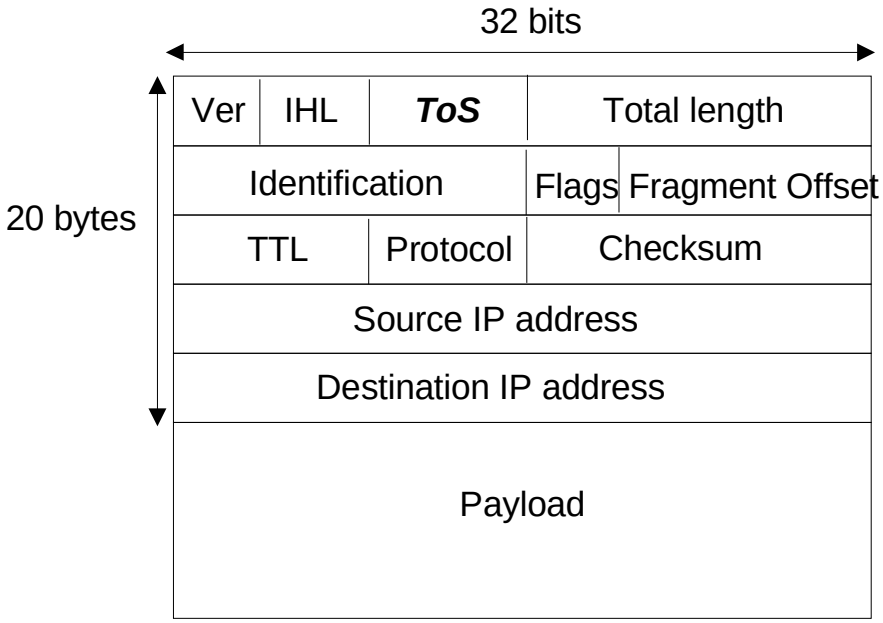


IP packet marking

- How can we mark an IP packet ?
  - Steal one field of the IP header
    - ToS : Type of Service Octet
    - defines the relative importance of the IP packet and the type of service required for this packet

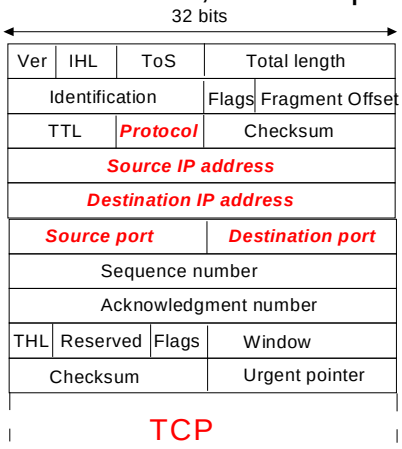


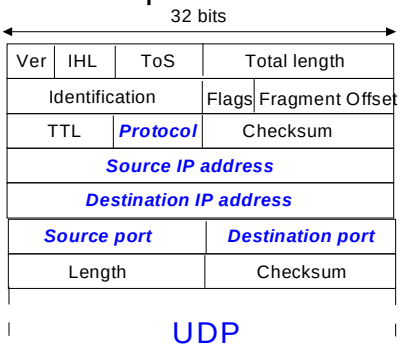
    - current status
      - definition of ToS Octet changed several times
      - Precedence is used in some networks
      - ToS field is rarely used
    - Using the ToS Octet for marking
      - advantage : easy to implement
      - disadvantage : limited number of marked flows



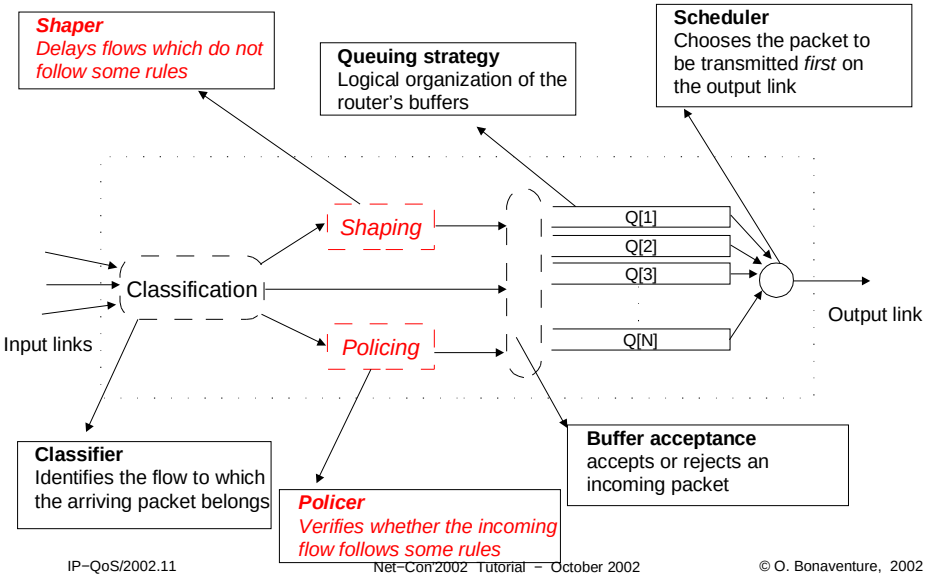
Layer−four flow

- Layer four flow identified by quintuple
  - source IP address, destination IP address, Protocol, source port, destination port





QoS capable router



In practice, the shaper could also be located on the output link, but we don't address this issue here to keep the picture simple and understandable.

Limiting rate of incoming flows

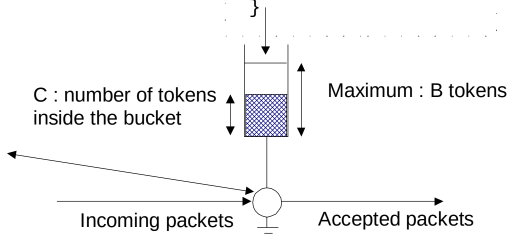
- How can we limit the rate of a flow of variable length packets on an input link ?
  1. Define flow rate for traffic contract
     - Which rate unit ?
     - Number of packets per unit of time
       - one 40 bytes packet per second versus one 1500 bytes packet per second
       - amount of information inside each packet must be considered
     - Number of bytes(bits) per unit of time
       - sounds better, but what appropriate unit of time ? one microsecond, one millisecond one second, one hour, one day...
  2. On packet arrival
     - If current rate is within contract, accept packet
     - Otherwise discard packet

Token Bucket

- Token bucket
  - R : average rate in bytes/sec
  - B : size of the token bucket

Bucket filling
Initialization
```
 C=B;
every 1/R second do
 {
  if(C<B)
     C=C+1;
 }
```

Arrival of packet P of length L
```
if (L <= C)
{ /* packet is accepted */
   C=C-L;
}
else
{
  /* packet is discarded */
}
```



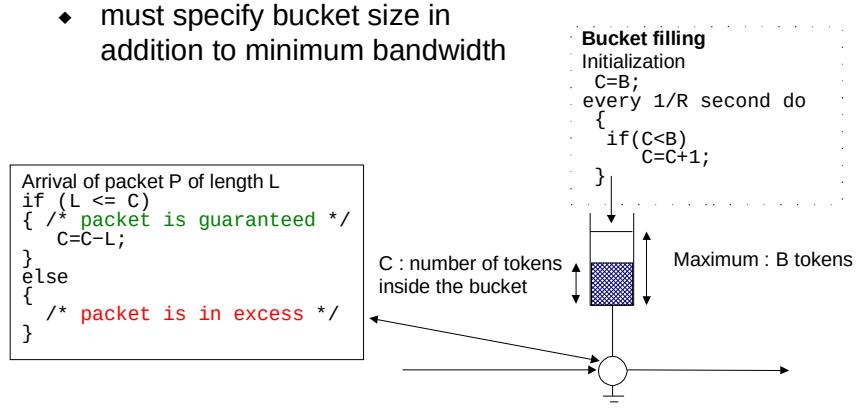
  - During a period of *T* seconds, the token bucket accepts at most B + (*T* ×R) bytes of traffic
    - worst case traffic at output of token bucket

Token Bucket (2)

- Advantages
  - can be used by a network provider to enforce a traffic contract since it provides a precise algorithmic definition for
    - conforming packets
    - non−conforming packets
  - provides a bound on the average rate
  - provides a bound on the maximum amount of traffic during any period of time
    - important to fix the size of buffers inside routers

Deterministic marking

- Principle
  - Modify token bucket to mark non−conforming packets instead of discarding them
    - must specify bucket size in addition to minimum bandwidth



This marker can also be modified to support more than three types of packets.

See J. Heinanen and R. Guerin, A Single Rate Three Color Marker, RFC 2697, Sept. 1999

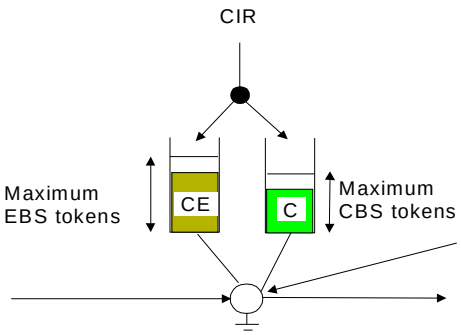
J. Heinanen and R. Guerin, A Two Rate Three Color Marker, RFC 2698, Sept. 1999

Extensions to token bucket

- Single rate three color marker
  - Committed Information Rate (CIR)
  - Committed Burst Size (CBS)
  - Excess Burst Size (EBS)

Bucket filling

```
Initialization
 C=CBS;
 CE=EBS;
every 1/CIR second do
 {
  if(C<CBS)
      { C=C+1; }
  else if (CE<EBS)
       { CE=CE+1 }
       else
       { /* nothing */ }
 }
```




```
Arrival of packet P of length L
if (L <= C)
{ /* packet is green */
   C=C-L; }
}
else if (L <= CE)
{ /* packet is yellow */
      CE=CE-L; }
else
{ /* packet is red */ }
```


See also

O.Bonaventure and S.De Cnodder. A rate adaptive shaper for differentiated services. Internet RFC2963, October 2000.

for a shaper that can be used to improve the performance of TCP with such markers

Cisco routers have a different way to implement this kind of token bucket with two burst sizes. See

S. Vegesna, IP Quality of Service, Cisco Press, 2001

Extensions to token bucket (2)

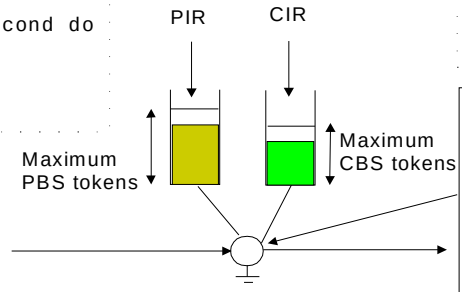
- Two rate three color marker
  - Committed Information Rate (CIR)
  - Committed Burst Size (CBS)
  - Peak Information Rate (PIR)
  - Peak Burst Size (EBS)

Peak Bucket filling

```
Initialization
 CP=PBS;
every 1/PIR second do
 {
  if(CP<PBS)
     CP=CP+1;
 }
```

Committed Bucket filling

```
Initialization
 C=CBS;
every 1/CIR second do
 {
  if(C<CBS)
     C=C+1;
 }
```

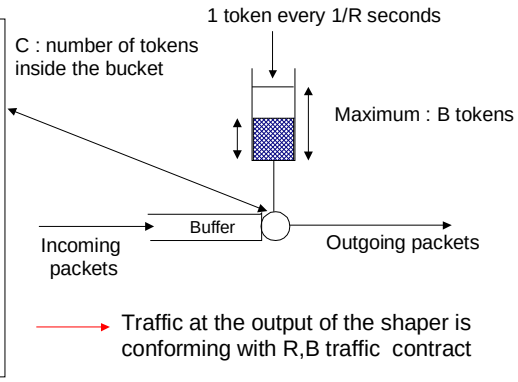



```
Arrival of packet P of length L
if (L < CP)
{ /* packet is red }
else if ( L < C )
   {/* packet is yellow */
     CP=CP-L; }
   else
   { /* packet is green */
     CP=CP-L;
     C=C-L;
   }
```

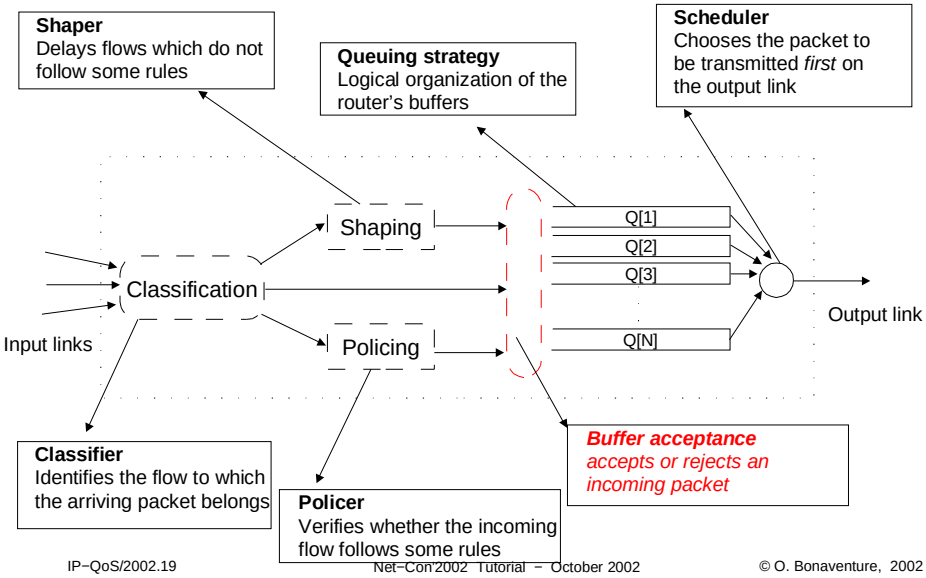
Token Bucket in shaping mode

- Problem
  - How can we ensure that one particular flow (e.g. after multiplexing) is conforming to a contract ?
    - utilize modified token bucket in shaping mode (shaper)

```
Arrival of packet of size L
if (L <= C)
{ /* packet arrived on time */
   C=C-L;
   transmit_packet();
}
else
{/* packet arrived too early
  * delay packet inside buffer
  * until it becomes conforming
  */
 while (C<L)
  { /* wait */  }
 /* now C=L and packet is
conforming */
 C=C-L;
 transmit_packet();
}
```

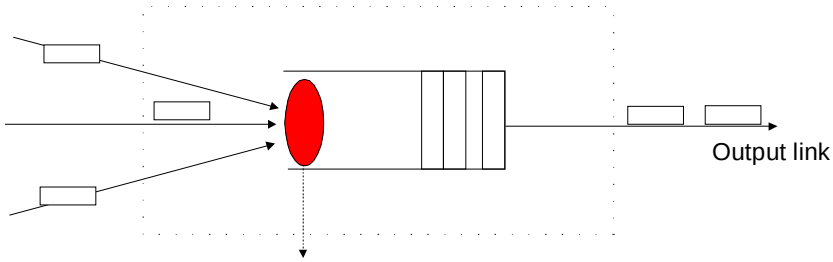


QoS capable router



Buffer acceptance algorithm

- Packet treatment inside router's output port



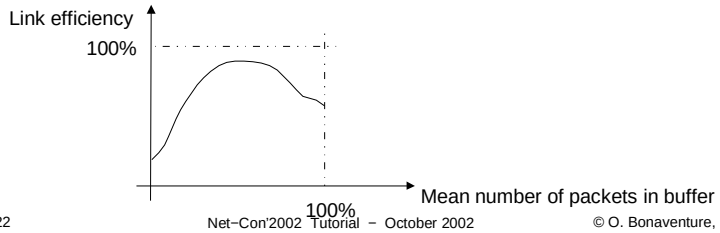
Buffer acceptance algorithm
When a packet arrives from an input link, the buffer acceptance algorithm decides whether this packet can be accepted inside the router's buffer

Buffer acceptance algorithms

- Two fundamental questions
  - When do we drop a packet ?
    - when the buffer is full
      - example : tail drop
    - when the buffer occupancy increases too much
      - example : Random Early Detection
  - Which packet should be dropped
    - The arriving packet (the packet at the tail of the queue)
      - but is this packet responsible for congestion ?
    - Another packet from the same flow as the arriving packet
      - this might help congestion control algorithms
    - A packet from some flow
      - not necessarily from the same flow as the arriving packet
    - The packet at the head of the queue
      - could improve the performance with TCP

Buffer acceptance algorithms (2)

- Objectives
  - control the amount of packets in the buffer to
    - efficiently support best−effort traffic
      - should provide a fair utilization of the routers buffers
    - provide protection among different flows
      - one flow should not prohibit other flows from having packets inside the router's buffers
    - achieve a good utilization of output link



Tail drop

- Simplest buffer acceptance algorithms
- Principle
  - when a packet arrives at a full buffer, the arriving packet is discarded
  - Advantages
    - easy to implement
    - can limit the number of packet losses for large buffer
  - Disadvantages
    - no distinction between the various flows
    - not the best solution for TCP traffic

Random Early Detection

- Goals
  - should be easily implemented in simple routers with a single logical queue
  - achieve a low, but non−zero, average buffer occupancy
    - low average occupancy provides low delay for interactive applications and ensure fast TCP response
    - non−zero average occupancy ensures an efficient utilization of the output link
  - approximate a fair discard of packets among the active flows without identifying them
  - discard packets in a TCP friendly way
    - we should avoid discarding bursts of packets since TCP reacts severely to burst losses

Random Early Detection (RED) was proposed in
S. Floyd, V. Jacobson, Random Early Detection gateways for congestion avoidance, IEEE/ACM Trans. Networking, V1, N4, 1993, pp. 397−413

Its utilization is recommended in
Braden at al., Recommendations on Queue Management and Congestion Avoidance in the Internet, RFC 2309, April 1998

See also : S. Floyd. Red (Random Early detection) queue management. available from http://www.aciri.org/floyd/red.html.

Random Early Detection (2)

- Principle
  - How can we detect congestion ?
    - measure average buffer occupancy by using a low−pass filter
    - buffer is considered congested when its average occupancy is above a configured threshold
      - threshold value usually around 10%− 20% of buffer size
  - What do we do in case of congestion ?
    - Probabilistic drop for incoming packet
      - drop will force TCP to slow down
      - drop probability should increase with congestion level
    - Why probabilistic drop ?
      - Avoid dropping burst of packets from single flow
      - Try to drop packets for each flow in proportion of network usage

Some papers in favor of RED

S. Doran. Red analysis. available from http://adm.ebone.net/~smd/red−1.html, 1998.

B. Reynolds. RED analysis for congested network core and customer egress. In Presented at NANOG, January 1999. available from http://engr.qual.net/papers/reddraft.html.

Some papers not really in favor of RED

M. May, J. Bolot, C. Diot, and B. Lyles. Reasons not to deploy RED. In IWQoS'99, London, June 1999. preprint available from http://199.2.52.7/PEOPLE/diot/.

G. Iannaccone, M. May, and C. Diot. Aggregate traffic performance with active queue management and drop from tail. ACM Computer Communications Review, 31(3):4−−13, July 2001.

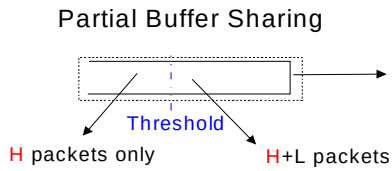
M. Christiansen, K. Jeffay, D. Ott, and F. Donelson Smith. Tuning RED for web traffic. In ACM SIGCOMM2000, August 2000.

A description of FRED (could be part of latests IOS versions)

D. Lin and R. Morris. Dynamics of random early detection. In SIGCOMM 97, pages 137−−145, Cannes, France, September 1997.

Packet discard preferences

- Problem
  - two types of packets
    - high and low priority packets
  - carry preferably high priority packets
- Solution
  - Discard less−important packets earlier than others




```
Arrival of packet
if (Packet.Type == H)
{ /* packet is high priority }
  if (Buf.Length < Buf.Size)
      accept_packet();
  else
      discard_packet();
}
else
{/* packet is low priority */
  if (Buf.Length < Buf.Threshold)
      accept_packet();
  else
      discard_packet();
}
```


Partial Buffer Sharing can easily be extended to support N different drop priorities.

Packet discard preferences (2)

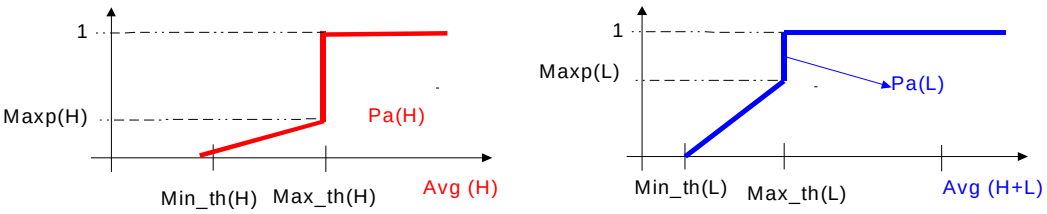
- Weighted RED
  - extension of RED to support several N packet discard preferences
  - Principle of the solution
    - N RED algorithms run in parallel
      - the first one decides the acceptance of priority N packets that should only be discarded in case of severe congestion
      - the second one decides the acceptance of priority N−1 packets that should be discarded earlier than high priority packets
      - ...
      - The Nth RED algorithm decides the acceptance of packets without any priority

WRED was initially proposed as RIO (RED with In/Out) in
Clark and Fang, Explicit Allocation of Best Effort packet delivery service, IEEE/ACM transactions on networking, August 1998, vol 6, N 4, pp.362−373

Several variants of RIO have been proposed and implemented since then.

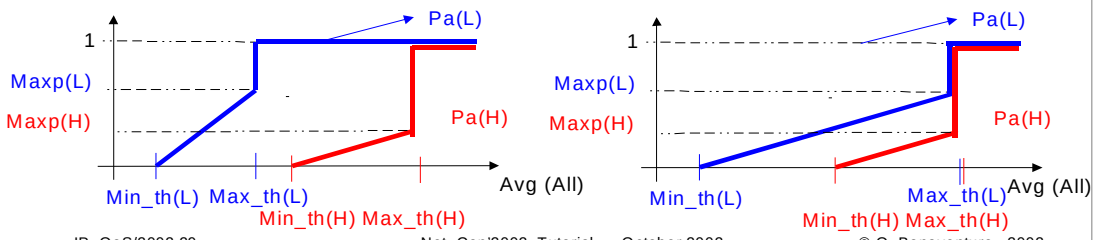
Weighted RED

- RED with In/Out
  - Initial proposal to support two priorities
  - Principles
    - compute two averages : Avg(H) and Avg(H+L)
    - Apply conservative RED for H packets with large thresholds
    - Apply aggressive RED for L packets with small thresholds



Weighted RED (2)

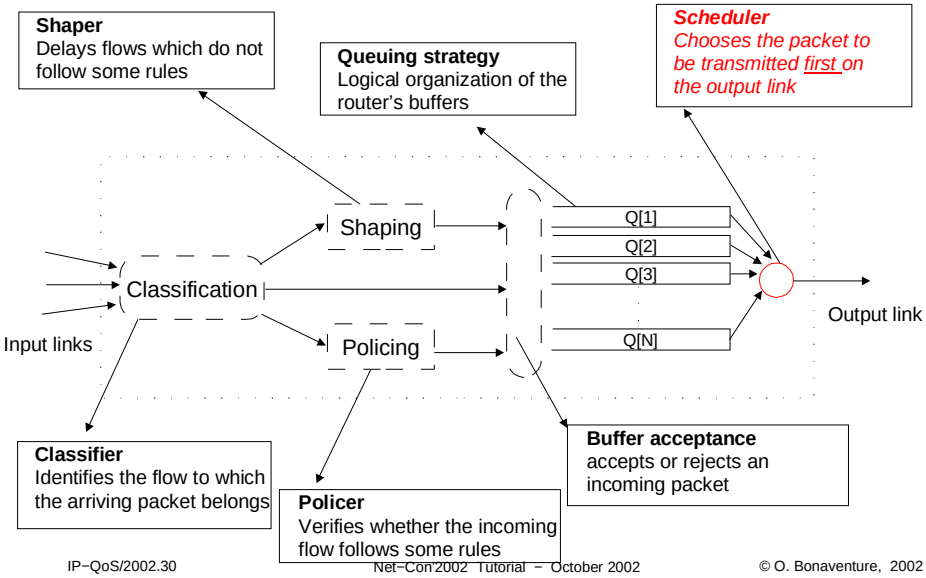
- Weighted RED
  - can be used to support N drop priorities
  - Principles
    - compute a single average for all packets in buffer
    - conservative RED algorithm for high priority packets
      - large values for min_th(H) and max_th(H)
    - aggressive RED for low priority packets
      - small values for min_th(L) and min_th(L), and higher drop prob.



The configuration guidelines for WRED on cisco routers propose to use the same value for max_th and maxp for all classes and to perform the differentiation only on the basis on min_th. See

http://www.cisco.com/univercd/cc/td/doc/product/software/ios120/12cgcr/qos_c/qcpart3/qcwred.htm

Scheduling algorithms



Scheduler

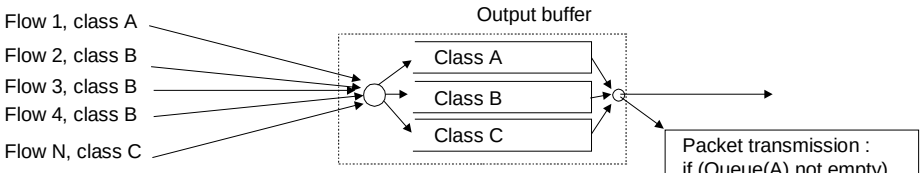
- Function
  - among all the logical queues containing at least one packet, select the packet that will be transmitted on the output link
- A scheduler should ...
  - be easy to implement in hardware
  - support best−effort and guaranteed services
  - provide fairness for best−effort traffic
  - provide protection
    - one flow should not be able to steal bandwidth from other existing flows
  - provide statistical or deterministic guarantees
    - bandwidth, delay

For more information on schedulers, see

H.Zhang. Service disciplines for guaranteed performance service in packet−switching networks. *Proceedings of the IEEE*, 83(10), October 1995.

Priority−based scheduler

- A simple priority scheduler



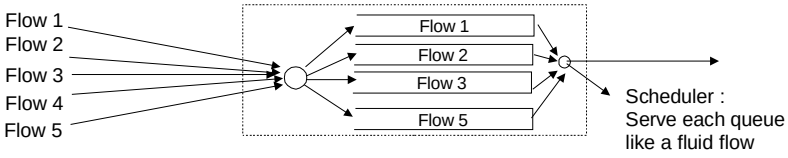

```
Packet transmission :
if (Queue(A) not empty)
 serve Queue(A);
else
  if (Queue(B) not empty)
    serve Queue(B);
  else
    if (Queue(C) not empty)
      serve Queue(C);
    else ....
```

- Advantages
  - easy to implement
  - packets in high priority class see low delay
- Disadvantages
  - no protection
  - a high priority flow can "kill" low priority flows
    - this might be desirable in some cases, but not always

→ Priority should be used with care...

Generalized Processor Sharing

- Generalized Processor Sharing (GPS)
  - ideal work−conserving scheduler
    - weight W[i] associated with Queue[i]
    - each queue is served by the scheduler as if it contained a fluid flow
    - at time t, Queue[j] is served at rate
    - $$rate = link_{rate} \times \left(\frac{W[j]}{\sum_{i = activequeues} W[i]} \right)$$
    - a queue is active if it contains something



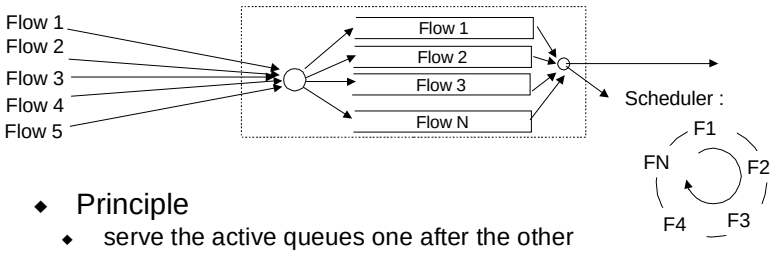
Generalized Processor Sharing (2)

- Advantages
  - provides per−flow bandwidth guarantee
    - through one GPS scheduler
    - through a network of GPS schedulers
  - provides per−flow delay guarantee for token−bucket (R,B) constrained flows
    - through one GPS scheduler
    - through a network of GPS schedulers

    $delaybound = \frac{B}{R}$
  - provides bound on buffer utilization $bufferbound = B$
  - provides protection among the different flows
    - a flow cannot jeopardize the guarantees for another flow
  - trivial guarantee on delay jitter ($[0,D_{max}]$)
- Disadvantage
  - Mathematical scheduler not implementable

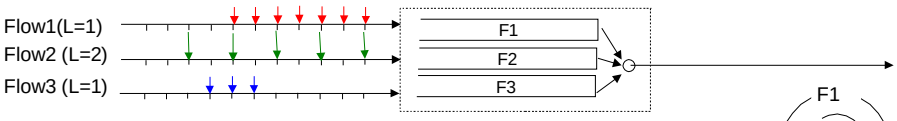
Round Robin

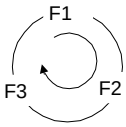
- Round Robin



  - Principle
    - serve the active queues one after the other
  - Advantages
    - can be easily implemented in hardware
    - provides protection for best−effort traffic
    - provides fair distribution of bandwidth with fixed−size packets
      - but fairness is only provided at timescales larger than schedule
  - Disadvantages
    - unfairness with variable length packets

Round Robin : example



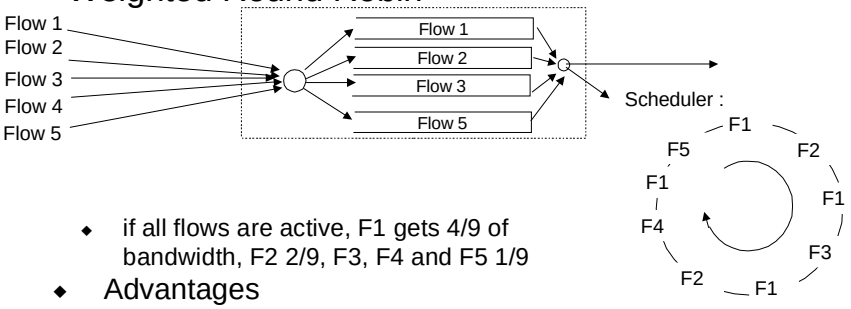


Flow1 and Flow3 send packets of size 1
Flow 2 sends packets of size 2

T	In F1	In F2	In F3	Q(F1)	Q(F2)	Q(F3)	Scheduled
0	P10[1]	P20[2]	–	P10	P20	–	F1:P10
1	P11[1]	–	–	P11	P20	–	F2:P20
2	P12[1]	P22[2]	–	P11,P12	P22	–	F2:P20 (cont)
3	P13[1]	–	–	P11,P12,P13	P22	–	F1:P11
4	P14[1]	P24[2]	–	P12,P13,P14	P22,P24	–	F2:P22
5	P15[1]	–	P35[1]	P12,P13,P14,P15	P24	P35	F2:P22 (cont)
6	P16[1]	P26[2]	P36[1]	P12,P13,P14,P15,P16	P24,P26	P35,P36	F3:P35
7	–	–	P37[1]	P12,P13,P14,P15,P16	P24,P26	P36,P37	F1:P12
8	–	P28[2]	–	P13,P14,P15,P16	P24,P26	P36,P37	F2:P24
9	–	–	–	P13,P14,P15,P16	P26,P28	P36,P37	F2:P24
10	–	P2A[2]	–	P13,P14,P15,P16	P26,P28	P36,P37	F3:P36

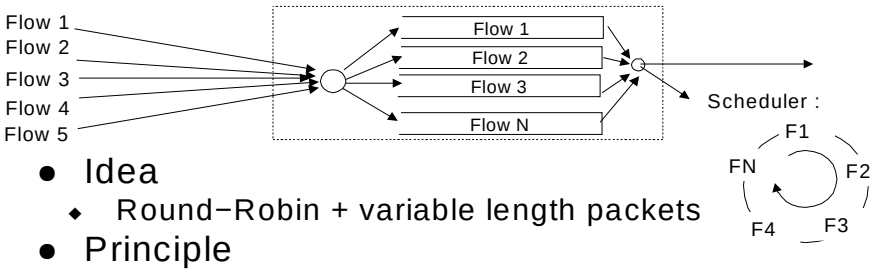
Weighted Round Robin

- Weighted Round Robin



  - if all flows are active, F1 gets 4/9 of bandwidth, F2 2/9, F3, F4 and F5 1/9
- Advantages
  - easy to implement with short schedule
  - different weights provide different bandwidths
  - inter−flow protection
  - Deficit Round−Robin can be extended to support weights
- Disadvantages
  - a long schedule is required to support many flows with small bandwidth, but a long schedule is complex...

Deficit Round Robin

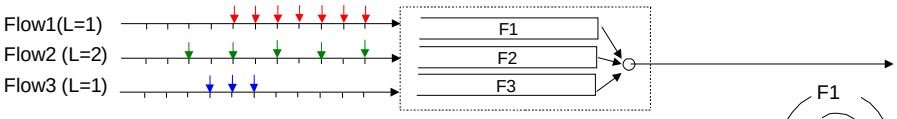


- Idea
  - Round−Robin + variable length packets
- Principle
  - associate counter d[i] to each queue
  - increase d[i] by *quantum* every time queue[i] is visited
    - if first_packet of queue[i] larger than d[i]
      { packet stays in queue[i]; }
    - else
      {
      packet is transmitted on output link;
      d[i]=d[i]− packet length;
      if queue[i] is empty { d|i]=0; }
      }

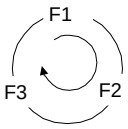
Deficit Round Robin is described in

M.Shreedhar and G.Vargese. Efficient fair queueing using deficit round robin. In *Proc. ACM SICOGMM'95*, pages 231−−242, 1995.

Deficit Round Robin : example



Flow1 and Flow3 send packets of size 1
Flow 2 sends packets of size 2



T	Inc.	D[1]	D[2]	D[3]	Q(F1)	Q(F2)	Q(F3)	Scheduled
0	F1	0+1−1	0	0	P10	P20	−	F1:P10
1	F2,F1	0+1−1	0+1	0	P11	P20	−	F1:P11
2	F2	0	1+1−2	0	P12	P22	−	F2:P20
3	−	0	0	0	P13	P22	−	F2:P20(cont)
4	F1	0+1−1	0	0	P14	P22,P24	−	F1:P13
5	F2,F3	0	0+1	0+1−1	P14,P15	P22,P24	P35	F3:P35
6	F1	0+1−1	0	0	P14,P15,P16	P22,P24,P26	P36	F1:P14
7	F2	0	1+1−2		P15,P16	P22,P24,P26	P36,P37	F2:P22
8	−	0	0	0	P15,P16	P24,P26	P36,P37	F2:P22(cont)
9	F3	0	0	0+1−1	P15,P16	P24,P26	P36,P37	F3:P36
10	F1	0+1−1	0	0	P15,P16	P24,P26	P37	F1:P15
11	F2,F3	0	0+1	0+1−1	P15,P16	P24,P26	P37	F3:P37
...								

Weighted Fair Queuing

- Objective
  - Define an implementable approximation for GPS
- Idea
  - simulate GPS on a per−packet basis
  - serve the packets in (approximately) the same order as the one they would be served with GPS
- How to do this ?
  - Compute time at which GPS would serve each packet (finish time)
  - Serve packets in order of finish times

A.Parekh and R.Gallagher. A generalized processor sharing approach to flow control : the single node case. *IEEE/ACM Transactions on Networking*, 1(3):346−−357, 1993.

A.Parekh and R.Gallagher. A generalized processor sharing approach to flow control − the multiple node case. *IEEE/ACM Transactions on Networking*, 2(2):137−−150, 1996.

Virtual Clock

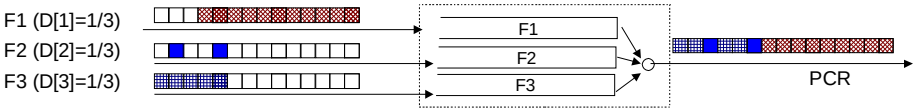
- Approximation of GPS
  - Idea
    - associate one timestamp to each arriving packet
    - scheduler selects among all the queued packets the packet with the smallest timestamp
  - First algorithm
    - D[i] : bandwidth associated with Queue[i]
    - V[i] : state variable associated with Queue[i]
    - Arrival of a P bytes long packet in Queue[i]
      - V[i] = V[i] + (P / D[i])
      - associate V[i] with the packet
    - Scheduler
      - select the packet with the smallest timestamp for transmission

Virtual Clock was proposed in

L.Zhang. VirtualClock: A new traffic control algorithm for packet switching. *ACM Transactions on Computing Systems*, 9(2):101−−124, May 1991.

Virtual Clock (2)

- Example



T	V(F1)	Q(F1)	V(F2)	Q(F2)	V(F3)	Q(F3)	Scheduled
0	max(0,0)+3	3	0	–	0	–	F1
1	max(3,1)+3	6	0	–	0	–	F1
2	max(6,2)+3	9	0	–	0	–	F1
...							
8	max(24,8)+3	27	0	–.	0	–	F1
9	max(27,9)+3	30	max(0,9)+3	12	max(0,9)+3	12	F2
10	max(30,10)+3	30,33	12	–	max(12,10)+3	12,15	F3
11	33	30,33	12	–	max(15,11)+3	15,18	F3
12	33	30,33	max(12,12)+3	15	max(18,12)+3	18,21	F2
13	33	30,33	15	–	max(21,13)+3	18,21,24	F3
14	33	30,33	15	–	max(24,14)+3	21,24,27	F3
....							

SCFQ / Virtual Spacing

- Approximation of GPS
  - Principle
    - associate one timestamp to each arriving packet
    - scheduler selects packet with smallest timestamp
  - Algorithm
    - D[i] : bandwidth associated to Queue[i]
    - V[i] : state variable associated to Queue[i]
    - V : state variable associated to the scheduler
      - at all time, V is equal to the timestamp of the packet being transmitted
    - Arrival of a packet of P bytes in Queue[i]
      - V[i] = max(V[i], V) + (P / D[i])
      - V[i] is associated to the arriving packet
    - Scheduler
      - select the packet with the smallest timestamp for transmission

For more information on SCFQ, see

J.Roberts. Virtual spacing for flexible traffic control. *International Journal of Communication Systems*, 7:307−−318, 1994.

J.Roberts, U.Mocci, and J.Virtamo, editors. *Weighted Fair Queueing*, chapter6, pages 173−−187. Number 1155 in Lecture Notes in Computer Science. Springer Verlag, 1996.

S.Golestani. A self−clocked fair queuing scheme for broadband applications. In *IEEE INFOCOM94*, pages 636−−646, 1994.

Guarantees

- Schedulers supporting per−flow bandwidth guarantees and protection between flows
  - GPS
  - WFQ/PGPS
  - SCFQ
  - Deficit−WRR
  - These guarantees are independent of the behavior of the guaranteed flows and of the behavior of other flows
    - one flow cannot jeopardize the bandwidth guarantees provided to other flows
    - this implies a good buffer acceptance mechanisms

Guarantees (2)

- Schedulers supporting delay guarantees
  - delay guarantees are only available for token bucket (R,B) limited flows
    - but guarantee does not depend on behavior of other flows
  - Delay through a series of n schedulers (ignoring the fixed delays)
    - GPS $\frac{B}{R}$
    - WFQ / PGPS $\frac{B + n \times P_{max}}{R} + \sum_{i=1}^{n} \frac{P_{max}}{C_i}$
    - Virtual Clock $\frac{B + n \times P_{max}}{R} + \sum_{i=1}^{n} \frac{P_{max}}{C_i}$
    - SCFQ $\frac{B + n \times P_{max}}{R} + \sum_{i=1}^{n} \frac{K_i \times P_{max}}{C_i}$

C_i : output link rate on ith switch

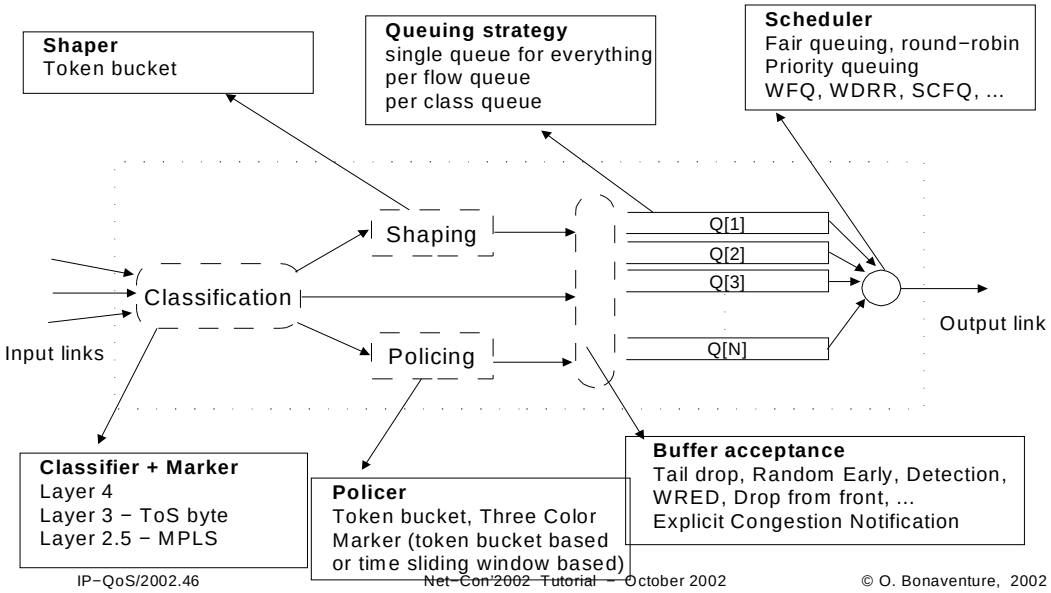
K_i : number of different flows on ith switch

P_{max} : maximum packet size

Source : H. Zhang, Service disciplines for guaranteed performance service in packet switching networks, Proc. IEEE, Vol 83, No 10, October 1995

A good textbook with a good description of schedulers is
S. Keshav, An engineering approach to computer networking : ATM networks, the Internet and the Telephone network, Addison Wesley, 1997

Packet level traffic control mechanisms

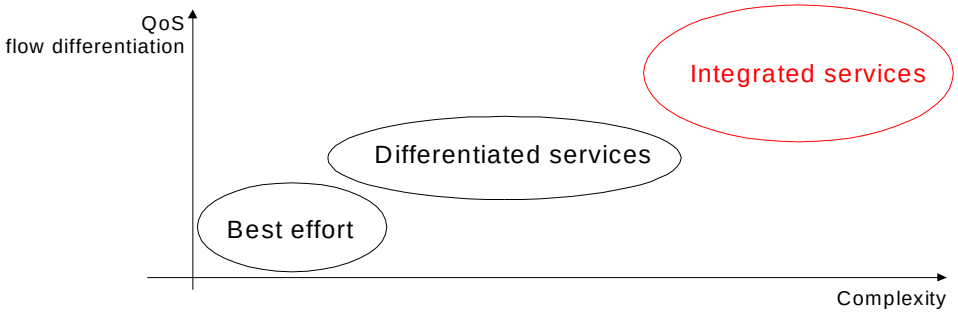


The evolution of Internet QoS

- Packet−level traffic control mechanisms
- Integrated Services
  - Architecture
  - RSVP : Resource reSerVation Protocol
  - The Services
    - Guaranteed Service
    - Controlled Load
    - Null
- Differentiated Services

Services

- What kind of services can we build with all these traffic control mechanisms ?



Integrated services
The "hard" approach for QoS

- Basic hypothesis
  - Some specific applications require QoS
    - delay guarantees
    - bandwidth guarantees
  - QoS will be provided to layer 4 flows
    - Each layer 4 (TCP or UDP) flow will inform the network about its QoS requirements
    - The network will accept or reject the flow based on its requirements and the current state of the network
  - QoS should support unicast and multicast
  - QoS flows should be allowed to coexist with best-effort flows in the same network
  - The existing routing protocols are left unchanged

The Intserv architecture was proposed in

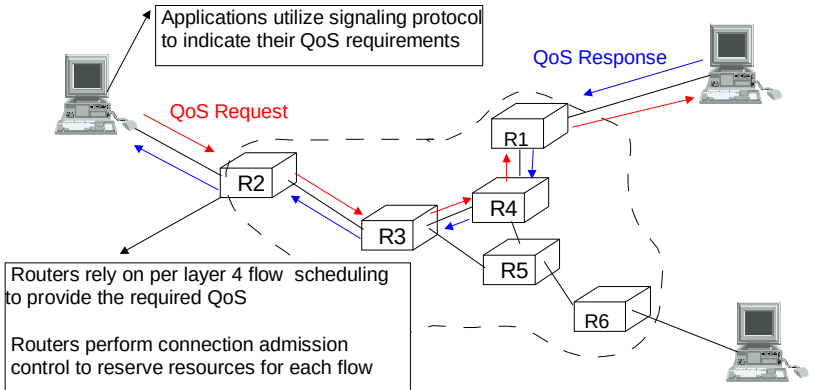
R.Braden, D.Clark, and S.Shenker. Integrated services in the Internet architecture : an overview. Internet RFC 1633, July 1994.

See also

P.White and J.Crowcroft. Integrated services in the Internet : the next stage in Internet : state of the art. *Proceedings of the IEEE*, 85(12):1934−−1946, December 1997.

Integrated services network

- Provision of integrated services in a network



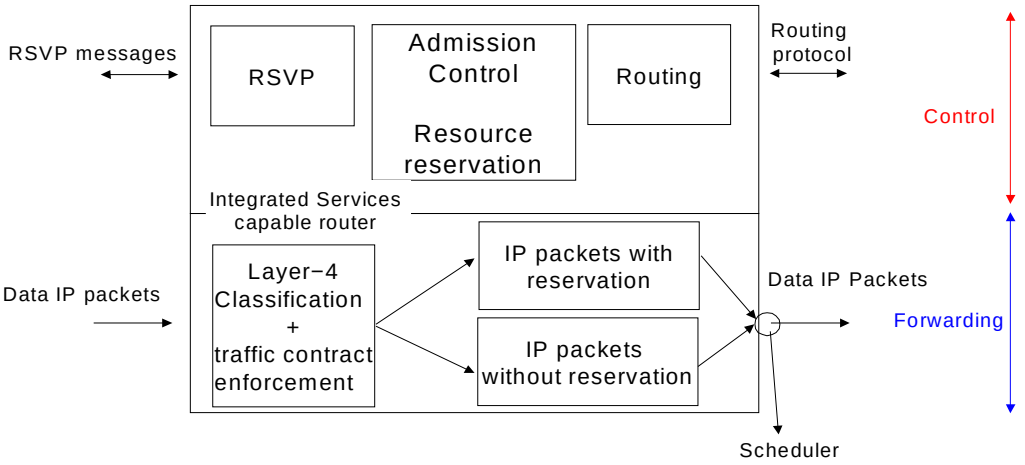
Routers rely on per layer 4 flow scheduling to provide the required QoS

Routers perform connection admission control to reserve resources for each flow

Normal IP routing is used to select path towards destination. IP routing is not changed to support Integrated Services

An Integrated services router

- Model of an Integrated Services router



The evolution of Internet QoS

- Packet−level traffic control mechanisms
- Integrated Services
  - Architecture
  - RSVP : Resource reSerVation Protocol
  - The Services
    - Guaranteed Service
    - Controlled Load
    - Null
- Differentiated Services

RSVP

- RSVP : Resource Reservation Protocol
- Objectives
  - Support the establishment of unidirectional flows in IP networks
    - different types of flows
      - initially layer 4 flows
      - today also MPLS flows (to be discussed in second part)
  - Suitable for IP unicast
    - should take into account the possibility of route changes without requiring any modification to routing protocols
  - Suitable for IP multicast
    - should fully support the IP multicast service model including dynamic groups

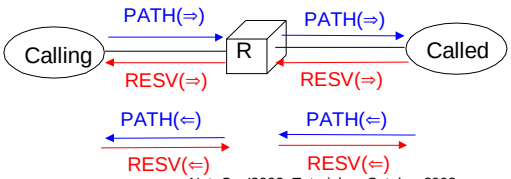
R.Braden, Ed., L.Zhang, S.Berson, S.Herzog, and S.Jamin. RFC 2205: Resource ReSerVation Protocol (RSVP) −−− version 1 functional specification, September 1997.
F.Baker, J.Krawczyk, and A.Sastry. RFC 2206: RSVP management information base using SMIv2, September 1997. Status: PROPOSED STANDARD.
A.Mankin, Ed., F.Baker, B.Braden, S.Bradner, M.O'Dell, A.Romanow, A.Weinrib, and L.Zhang. RFC 2208: Resource ReSerVation Protocol (RSVP) −−− version 1 applicability statement some guidelines on deployment, September 1997. Status: INFORMATIONAL.
J.Wroclawski. RFC 2210: The use of RSVP with IETF integrated services, September 1997. Status: PROPOSED STANDARD.

RSVP (2)

- Principles of operation
- Two important RSVP messages
  - PATH
    - used by sender to inform routers and receivers of the new flow and its required resources
      - no resources are reserved due to reception of PATH
  - RESV
    - used by receiver to actually reserve resources for the flow specified in the PATH message
      - resources are reserved for the IP packets sent by the sender towards the receiver along the path taken by the PATH message
- RSVP messages are sent inside IP packets

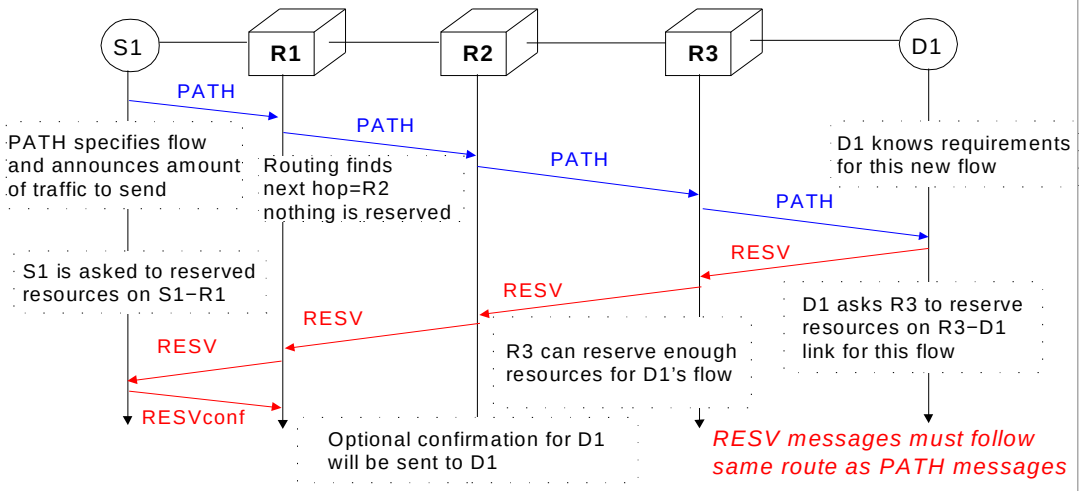
RSVP (3)

- RSVP flows
  - RSVP is used to associate reservations to unidirectional layer 4 flows
    - RSVP flow defined as the set of packets with same
      - IP destination address
      - IP Protocol Number (e.g. TCP or UDP)
      - [Transport−level] Destination Port Number
    - A typical bi−directional voice conversation requires
      - One RSVP flow from calling to called user
      - One RSVP flow from called user to calling



RSVP : example

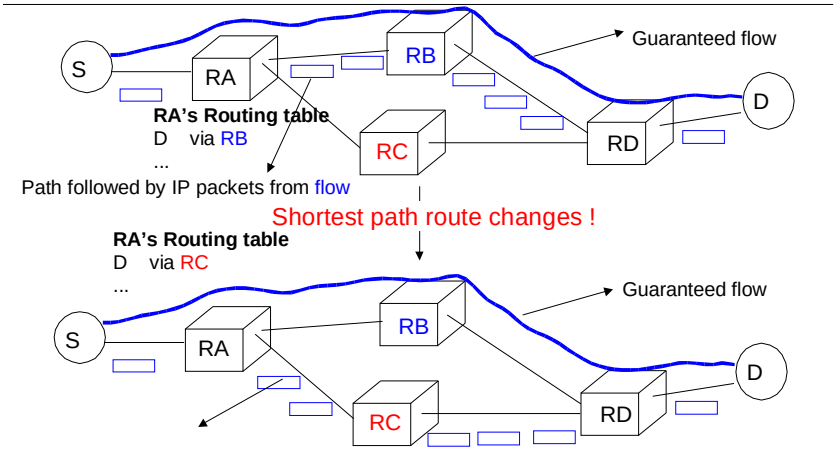
- Unicast flow establishment with RSVP



It should be noted that RSVP messages always follow exactly the same route as the regular IP packets that will be send by the source and destinations.

RSVP messages are directly encapsulated inside IP packets. Usually, the IP router alert option is used to ensure that those packets are intercepted by the transit routers.

The problem of route changes



- Either IP packets should continue to follow the guaranteed path or the guaranteed path should change to reflect the new path for the IP packets

Dealing with route changes with RSVP

- Principle
  - A flow established with RSVP lasts for some time
  - To remain alive, a flow must be refreshed regularly
  - RSVP only considers unidirectional flows
- Endsystems behavior
  - regularly send RSVP messages to maintain flow
- Routers behavior
  - maintain some state information for each flow
  - state is removed and flow released if no signaling messages have been received during some time

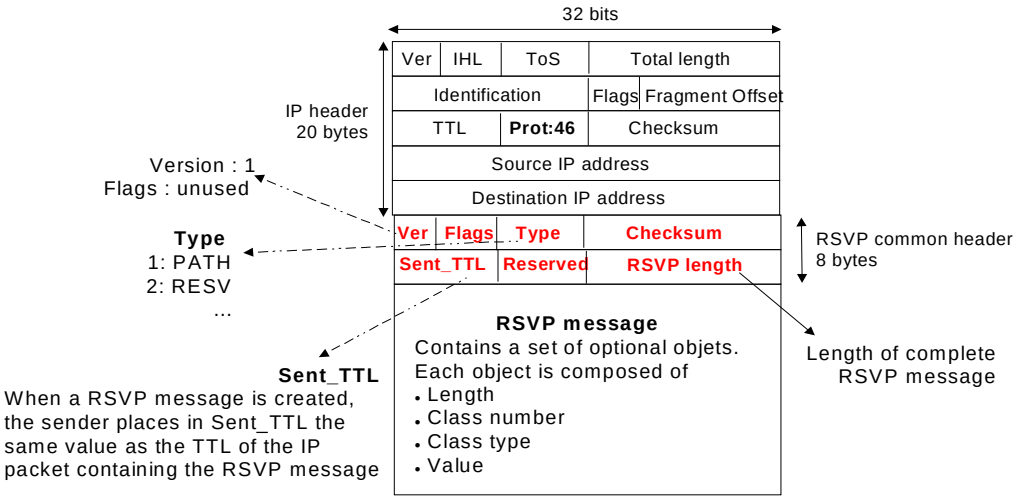
State maintenance

- RSVP routers maintain some per−flow state
- Possible solutions for state maintenance
  - Hard−state
    - traditional solution used in circuit−switched network
    - state is created at flow establishment and removed at flow tear down
    - if intermediate router crashes, state+reservation are lost
  - Soft−state (solution chosen by RSVP)
    - a timer is associated with each per−flow state
      - state is removed upon expiration of timer
    - hosts periodically retransmit PATH and RESV message
      - timer is resent upon reception of PATH/RESV message
    - if intermediate router crashes, state automatically reset
    - if route changes, new reservations/states are established on new path and all reservations/states expires

RSVP : detailed example

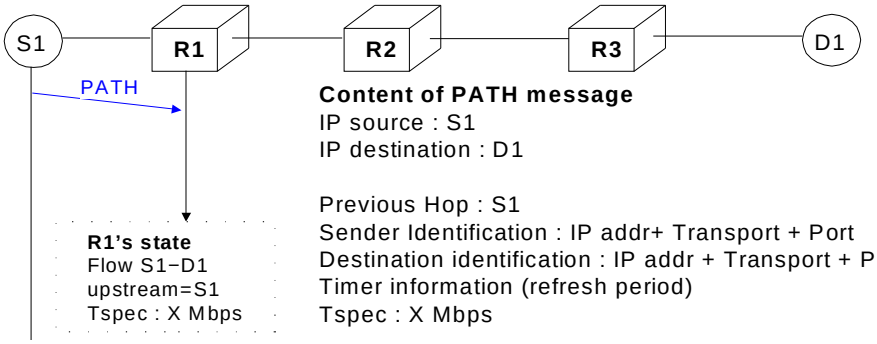
- Issues to consider
  - How to encapsulate RSVP messages in IP packets ?
  - How to ensure that the reservation messages follow the same route as the PATH messages
    - cannot simply rely on IP for this
      - remember : we do not change anything to routing protocols
  - What kind of information is stored in the state of the intermediate routers ?

RSVP : packet format



RSVP : detailed example (1)

- What happens inside the routers ?
  - S1 announces a flow towards D1



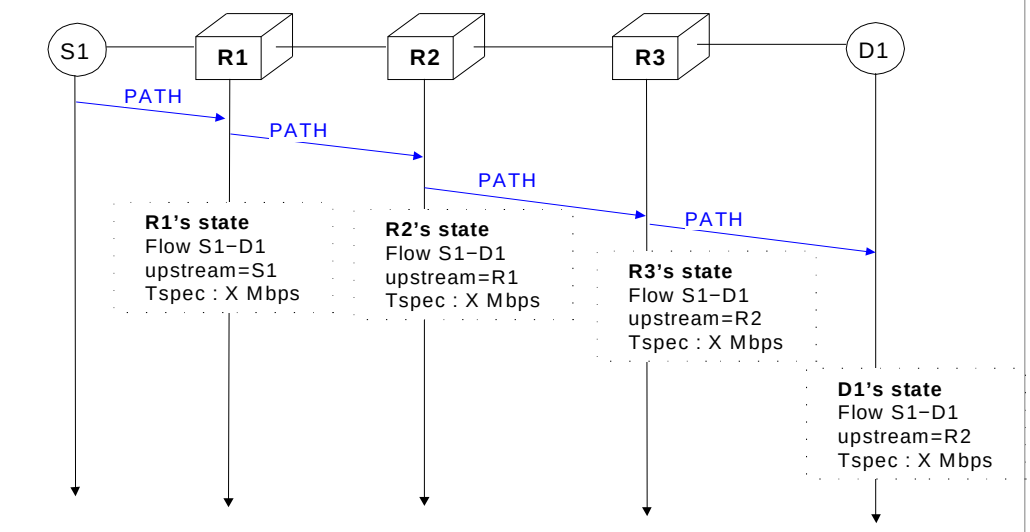
Content of PATH message
IP source : S1
IP destination : D1

Previous Hop : S1
Sender Identification : IP addr+ Transport + Port
Destination identification : IP addr + Transport + Port
Timer information (refresh period)
Tspec : X Mbps

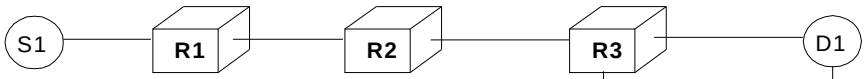
R1's state
Flow S1−D1
upstream=S1
Tspec : X Mbps

Behavior of R1
Content of PATH message inserted in R1's
RSVP message updated and sent downstream
PHOP replaced by R1

RSVP : detailed example (2)



RSVP : detailed example (3)



Content of RESV message
IP source : D1
IP destination : R3

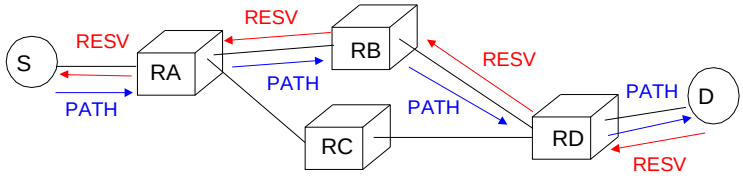
Previous Hop : D1
Destination identification : IP addr + Transport + Port
Information about Sender
Timer information (refresh period)
Rspec : Y Mbps

RESV

R3's state
Flow S1−D1
upstream=R2
Tspec : X Mbps
Rspec : Ymbps

RESV messages are transmitted hop−by−hop by RSVP capable routers to ensure that the reservation follows the same route as the corresponding PATH

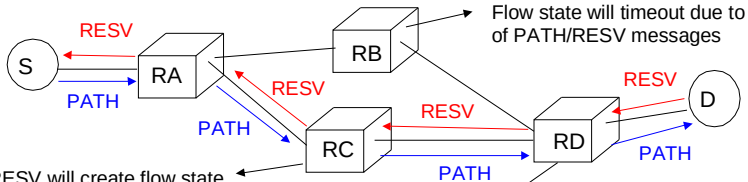
RSVP and route changes



Shortest path route changes !

S will re−send its PATH message every *Refresh* seconds (default: 30 sec)

D will re−send its RESV message every *Refresh* seconds (default: 30 sec)

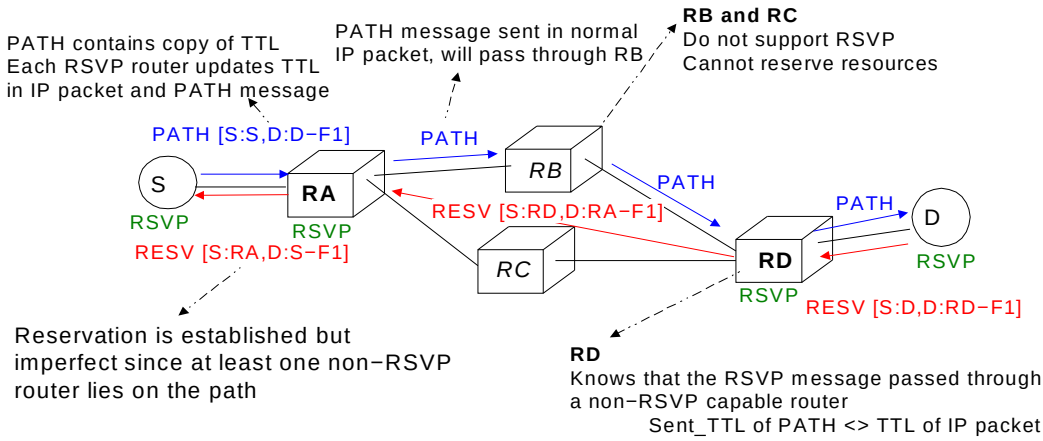


PATH/RESV will create flow state
Flow F1,
upstream=RA
...

PATH/RESV will update flow state
Flow F1,
upstream=RC

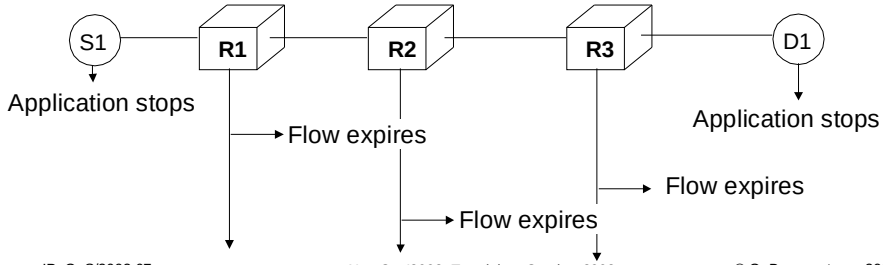
Smooth deployment of RSVP

- Problem
  - Can we utilize RSVP in heterogeneous network ?



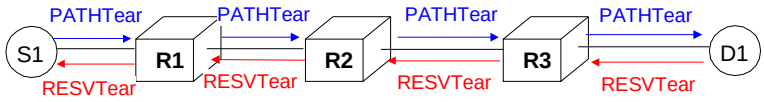
RSVP flow release

- How can we release RSVP flows ?
  - Sender and receiver stopping transmitting PATH and RESV messages
    - flow state will timeout inside routers
    - not the best solution since routers are expected to maintain flow state about five times longer than refresh period without receiving PATH/RESV messages
      - default refresh period is 30 seconds



RSVP flow release (2)

- A better solution
  - Define two new specific signaling messages
    - PATHtear
      - used by sender to announce to network end of a flow
    - RESVtear
      - used by receiver to announce to network end of a reservation



    - state timeout remains necessary since all RSVP messages (including PATHTear and RESVTear) are delivered as simple datagrams subject to packet loss

RSVP

- Some comments on RSVP
  - allows the establishment of flows with QoS guarantees in pure datagram networks
    - a flow is unidirectional for RSVP
  - soft−state approach to deal with route changes
    - soft−state must be regularly refreshed, which increases the signaling load
    - but many experts propose utilizing explicit routes that remain fixed for the duration of a flow
  - complete support of IP multicast
    - heterogeneous model is nice in theory...
      ... but isn't this model too complex in practice ?
  - Supports MPLS and layer 4 flows

RSVP : comments

- The initial RSVP goal
  - define a signaling protocol to support large scale multicast reservations for layer−4 flows
- Best design decision
  - extensibility of the protocol and its mechanisms
- What RSVP is becoming today
  - a generic signaling protocol that can be used to establish any flow (layer 4, MPLS, SDH, ...)
  - support for multicast and unicast
  - soft−state can be completed by acknowledgments to improve reliability

It should be noted that the IETF is currently considering the development of new signaling protocols to replace RSVP within the NSIS working group.

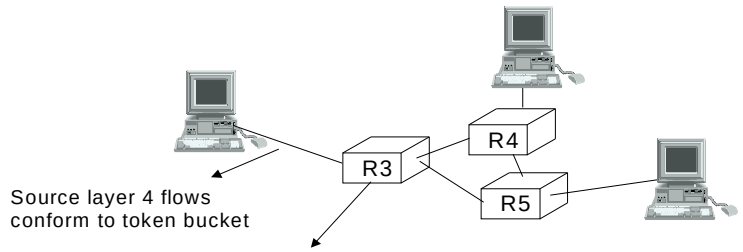
See M. Brunner (Editor), Requirements for QoS Signaling protocols, Internet draft, draft−ietf−nsis−req−02.txt, work in progress, May 2002

The evolution of Internet QoS

- Packet−level traffic control mechanisms
- Integrated Services
  - Architecture
  - RSVP : Resource reSerVation Protocol
  - The Services
    - Guaranteed Service
    - Controlled Load
    - Null
- Differentiated Services

Guaranteed Service

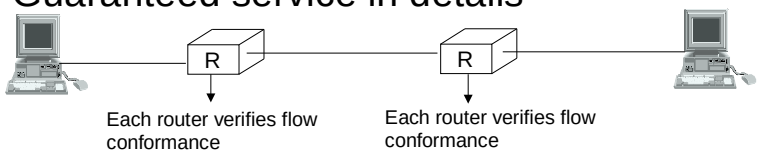
- Idea
  - provide a service as close as possible to a physical link between source and destination(s)
    - Ideal service would be fluid flow
    - the closest service we can provide in reality is an approximation to fluid flow with a firm (mathematically provable) guarantee on end−to−end delay and no losses



S.Shenker, C.Partridge, and R.Guerin. Specification of Guaranteed quality of service. Internet RFC 2212, September 1997.

Guaranteed service (2)

- Guaranteed service in details



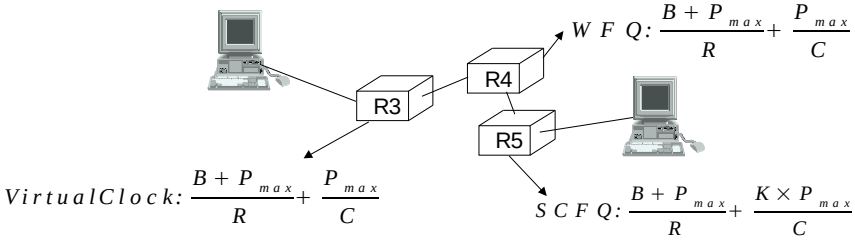
  - Guarantees are provided for conforming packets
    - conformance defined by token bucket
    - guarantee on maximum delay guarantee
      - upper−bound on end−to−end network delay
    - no guarantee on average delay
    - no guarantee on delay jitter
    - loss guarantee
      - packet loss due to queue overflow cannot occur
      - implies reservation of buffers at intermediate routers
      - packet loss due to random bit errors cannot be avoided
  - non−conforming packets treated as best−effort

Guaranteed Service (2)

- Tspec, specified by sender in PATH messages
  - M = maximum packet size (bytes)
    - packets larger than M are considered as best−effort, IP fragmentation is not supported for guaranteed service flows
  - Token Buckets
    - Peak rate token bucket
      - bucket depth = M
      - p = peak rate of flow (bytes/sec)
      - during T seconds, flow will not send more than M+p×T bytes
    - Average rate token bucket
      - b = bucket depth (bytes)
      - r = token bucket rate (bytes/sec)
      - during T seconds, flow will not send more than b+r×T bytes
    - During T sec, traffic ≤ M+min[p×T, r×T+b−M]
    - m = minimum policed unit (bytes)
    - packets smaller than m are considered of size m by token bucket

GS : delay bound

- What is the delay bound provided by GS ?
  - Fluid flow with GPS
    - transmission and processing delay
    - queueing delay ≤ b/r
  - Packet flows
    - transmission and processing delay
    - queueing delay depends on scheduler
    - what about heterogeneous networks ?

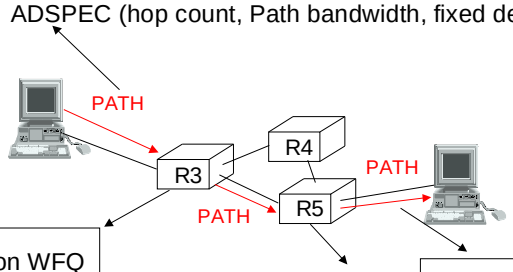


GS : delay bound (2)

- How can we support different schedulers ?
  - Compute delay bound of each scheduler as a delta compared to the GPS bound
  - queueing delay bound = $\frac{b}{r} + \frac{C}{r} + D$
    - C : rate dependent error term
    - D : rate independent error term
  - Each scheduler advertises its C and D error terms in ADSPEC of PATH messages
    - ADSPEC is modified by each intermediate router
    - ADSPEC contains $C_{tot} = \sum_i C_i \quad D_{tot} = \sum_i D_i$

GS : delay bound (3)

PATH message contains
TSPEC
ADSPEC (hop count, Path bandwidth, fixed delay, MTU, Ctot, Dtot)



R3 relies on WFQ
ADSPEC updated

$$C_{tot} = P_{max}$$

$$D_{tot} = \frac{P_{max}}{C}$$

R5 relies on SCFQ
ADSPEC updated

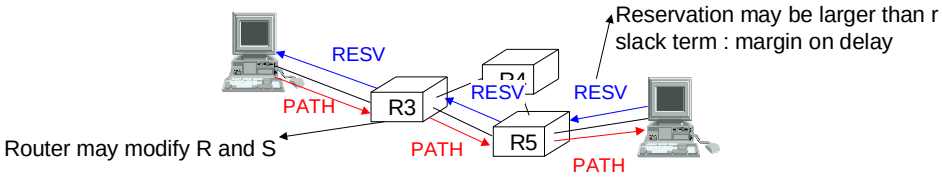
$$C_{tot} = P_{max} + P_{max}$$

$$D_{tot} = P_{max} + \frac{K \times P_{max}}{C}$$

Destination knows
- MTU for whole path
- Ctot
- Dtot
- Fixed Delay
- Min bandwidth of links in path

GS : delay bound (4)

- RESV message
  - destination specifies its reservation in RSPEC
    - Token bucket parameters (r, b, p, M, m)
    - R : rate (bytes per second)
      - R ≥ r
      - can be used to request a smaller queuing delay
    - S : Slack term (microseconds)
      - can be used by the destination when the announced delay is smaller than what is needed
      - intermediate routers could then benefit from part of this slack term to reduce the reservation attached to this flows



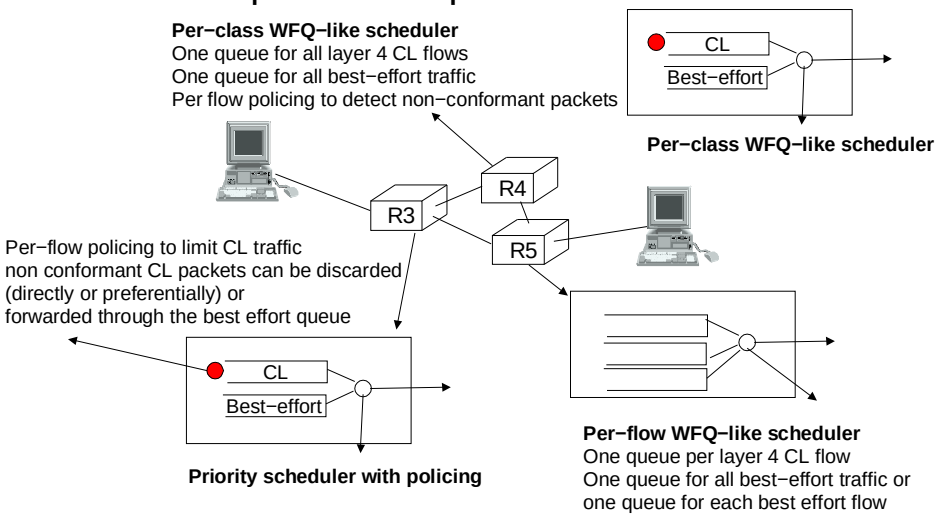
Controlled Load

- Idea
  - provide a service with the same performance as best−effort service in a *lightly loaded and uncongested* network
- Details
  - no deterministic guarantee on losses
    - but packet losses should be almost as rare as with GS
  - no guarantee on end−to−end delay or delay jitter
    - but on average queuing delay through intermediate routers should be low for conforming packets
- Utilization
  - applications requiring bandwidth reservation

J.Wroclawski. Specification of the Controlled−Load network element service. Internet RFC 2211, September 1997.

Controlled Load implementation

- Several possible implementations

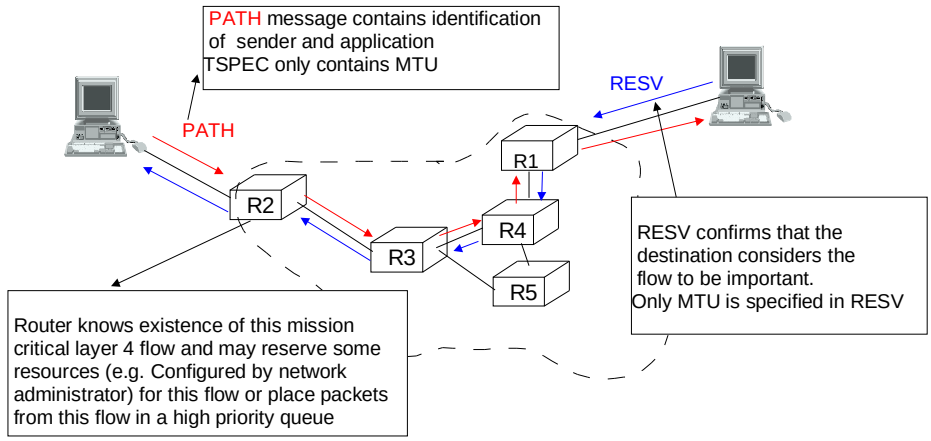


The Null service

- Analysis
  - In many enterprises, some applications are much more critical than others
    - e−commerce
    - real−time banking
    - industrial process control
  - Managers want the network to provide a better service for these applications than normal ones
    - but these applications cannot define a traffic contract
- Idea of the Null service
  - Applications use RSVP signaling to inform network of the existence of critical layer 4 flows
  - Network provides a better best effort service for these critical applications

Y.Bernet, A.Smith, and B.Davie. Specification of the null service type. RFC2997, November 2000.

Provision of the Null service

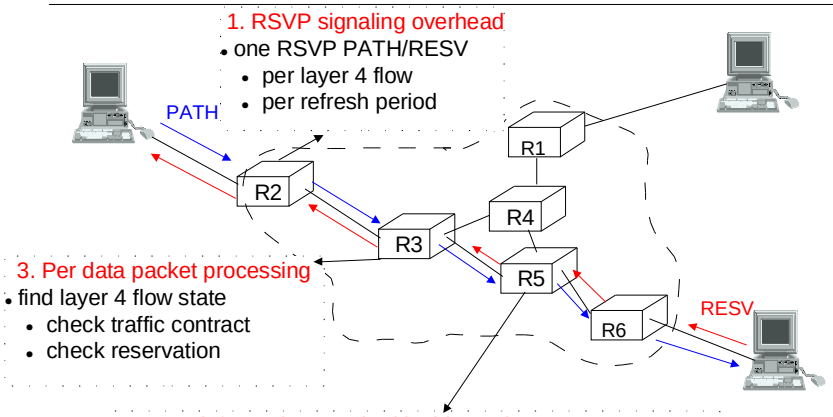


- With the Null service, a layer−4 flow will not receive strong QoS commitments, but the network will usually provide a better best−effort service

A critique of Integrated services

- Advantages
  - provides per layer 4 flow QoS guarantees
  - GS with delay/bandwidth guarantees
  - CL with bandwidth guarantees
- Drawbacks
  - Requires each intermediate router to perform some operations for each layer 4 flow
    - RSVP message processing
    - per layer 4 flow classification
      - classification can become complex for multicast !
    - per layer 4 flow policing/queueing/scheduling
      - a backbone router may see thousands of simultaneous flows !
  - Not all applications are able to express precisely their traffic and QoS requirements

Scaling issues with integrated services

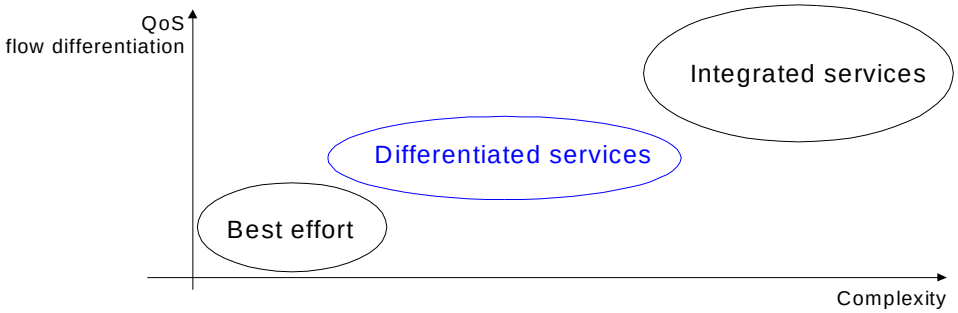


The evolution of Internet QoS

- Packet−level traffic control mechanisms
- Integrated Services
- Differentiated Services
  - Architecture
  - The Per−Hop and Per−Domain Behaviors
    - Class Selector (CS)
    - Expedited Forwarding (EF)
    - Assured Forwarding (AF)
    - Bulk Handling
  - Case Study
  - Improving the scalability of IntServ with DiffServ

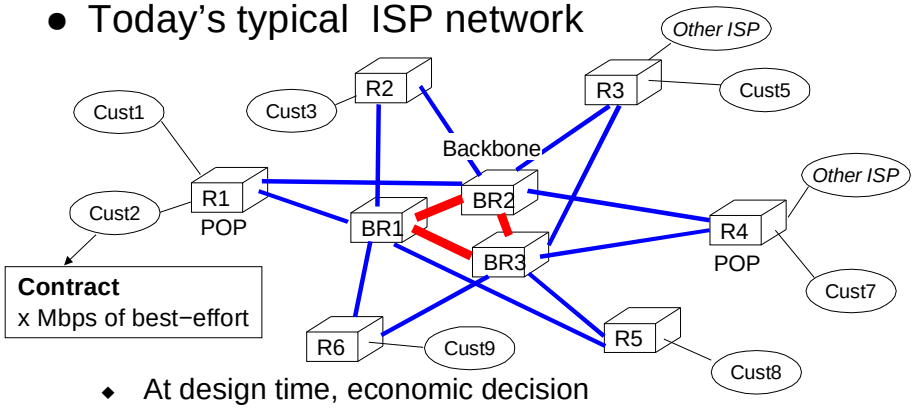
Services

- What kind of services can we build with all these traffic control mechanisms ?



Today's best effort IP networks

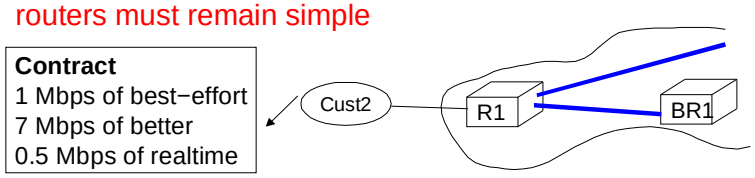
- Today's typical ISP network



  - At design time, economic decision
  - For each customer, we know subscribed amount of traffic, but usually not precisely destination
  - Follow closely the load on links and contracts
    - upgrade links when load becomes too high

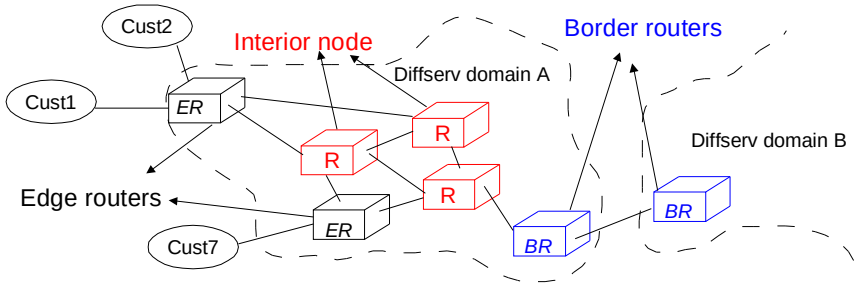
A pragmatic approach to QoS

- Idea
  - Allow customers to specify contracts for a few classes of traffic
    - Example classes
      - best effort, better than best effort, realtime, ...
    - different QoS commitments associated with each class
      - average delay
      - average losses
  - ISP configures its routers to efficiently support the requirements of these traffic classes
    - routers must remain simple



Differentiated Services : Architecture

- Principles
  - Network is divided in two parts



  - Complex mechanismes are implemented only on boundary nodes (border and edge routers)
  - Interior nodes are simple to operate at high speeds
  - No change to existing routing protocols

The architecture is described in

S.Blake, D.Black, M.Carlson, E.Davies, Z.Wang, and W.Weiss. An architecture for differentiated services. Internet RFC 2475, December 1998.

See also

D.Grossman. New terminology and clarification for diffserv. RFC3260, April 2002.

The MIB is being finalized, see

F.Baker, K.Chan, and A.Smith. Management information base for the differentiated services architecture. Internet draft, draft−ietf−diffserv−mib−16.txt, work in progress, November 2001.

One of the original papers on DiffServ

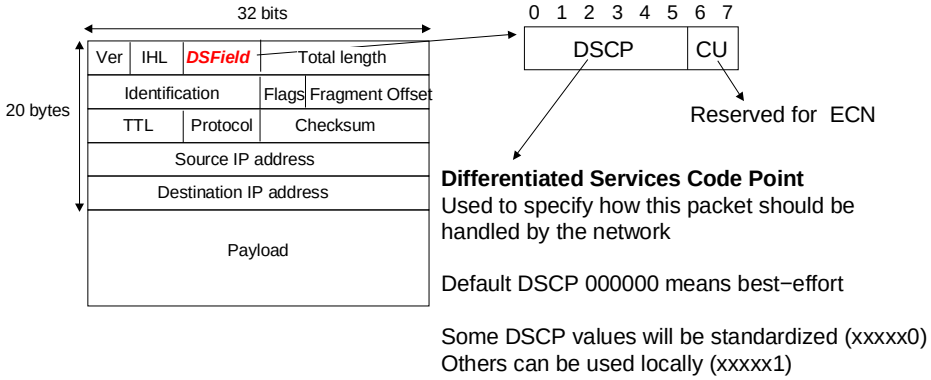
K.Nichols, V.Jacobson, and L.Zhang. A two−bit differentiated architecture for the internet. Internet draft draft−nichols−diff−svc−arch−00.txt, work in progress, available from ftp://ftp.ee.lbl.gov/papers/dsarch.pdf, November 1997.

Differentiated Services : Architecture (2)

- Provision of different types of service
  - The type of service is indicated explicitly inside each IP packet
    - Marking can be performed by
      - customer
      - edge router
      - border router
  - Interior routers only rely on this indication to provide the different types of services
    - complexity of interior routers depends on number of different services, not on number of layer x flows

Differentiated Services : Packet Marking

- Packet marking
  - redefine the semantics of the rarely used ToS byte inside IP header



Differentiated Services Code Point
Used to specify how this packet should be handled by the network

Default DSCP 000000 means best−effort

Some DSCP values will be standardized (xxxxx0)
Others can be used locally (xxxxx1)

See

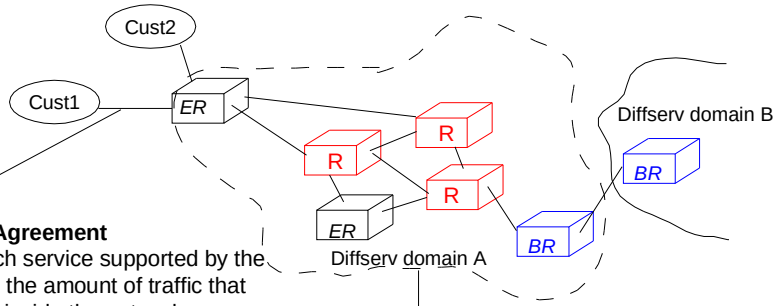
K.Nichols, S.Blake, F.Baker, and D.Black. Definition of the differentiated services field (DS field) in the IPv4 and IPv6 headers. Internet RFC 2474, December 1998.

For previous usage of the Diffserv field, see

P.Almquist. Type of service in the internet protocol suite. Internet RFC 1349, 1992.

F.Baker. Requirements for IP version 4 routers. Internet RFC 1812, June 1995.

Provision of Differentiated Services

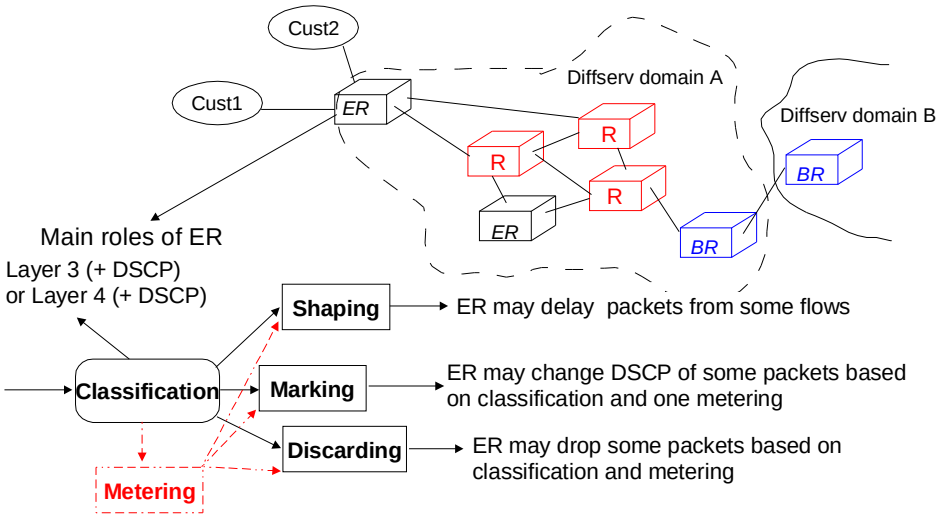


Service Level Agreement
specifies for each service supported by the diffserv domain the amount of traffic that Cust2 can send inside the network (possibly with some specification of destination)

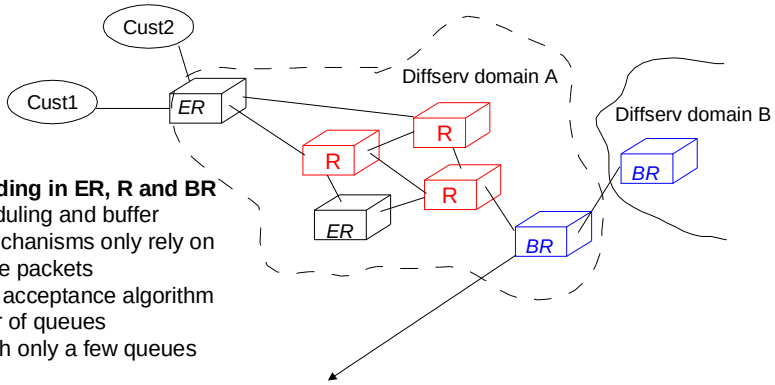
Network will commit to some performance figures
- packet loss rate over some period
- end to end delay at key points
- network reliability
- ...

Diffserv Codepoints
Network provider defines the services that he wants to provides and associates one or several DSCP to each service

Provision of Differentiated Services (2)



Provision of Differentiated Services (3)



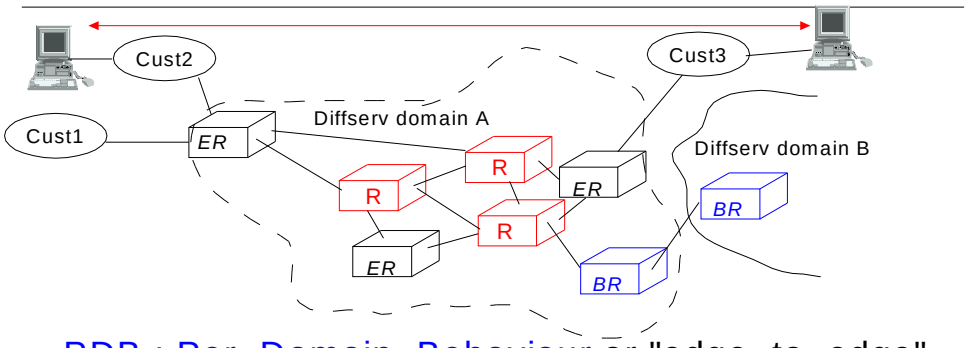
Packet forwarding in ER, R and BR
Queuing, scheduling and buffer acceptance mechanisms only rely on the DSCP of the packets
− simple buffer acceptance algorithm
− small number of queues
− scheduler with only a few queues

Roles of Border Routers
Supports same functions as ER, but usually on higher bandwidth links and with interdomain traffic
May also need to remark traffic when Diffserv domains A and B utilize different DSCPs for the same service

The evolution of Internet QoS

- Packet−level traffic control mechanisms
- Integrated Services
- Differentiated Services
  - Architecture
  - The Per−Hop and Per−Domain Behaviors
    - Class Selector (CS)
    - Expedited Forwarding (EF)
    - Assured Forwarding (AF)
    - Bulk Handling (BH)
  - Case Study
  - Improving the scalability of IntServ with DiffServ

The Differentiated services



PDB : Per−Domain−Behaviour or "edge−to−edge" service provided by a Diffserv network depends on :

- *Traffic conditioning functions* (shaping, policing, marking,...) located only on edge and border routers
- *Packet handling functions* (queueing, scheduling, buffer acceptance) located on each router
- PHB : per hop behavior in Diffserv terminology

K.Nichols and B.Carpenter. Definition of differentiated services per domain behaviors and rules for their specification. RFC 3086, April 2001.

The Per−Hop Behaviors

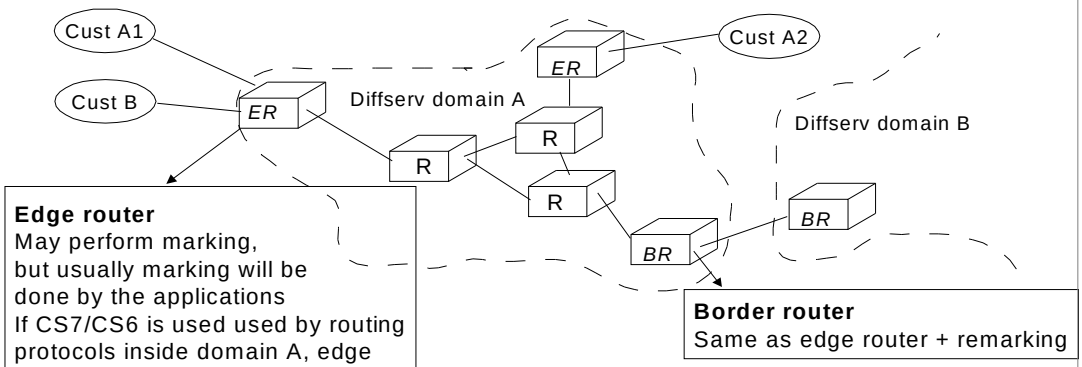
- Defined PHBs
  - Best−effort service
    - classical router behavior
  - Class Selector
    - backward compatibility with Precedence field
  - Expedited forwarding
    - should provide across a diffserv domain a service with
      - low packet loss ratio
      - low end−to−end packet delay
      - low delay jitter
  - Assured forwarding
    - reserve different amount of resources (bandwidth, buffers) for 4 classes of traffic in each router
    - classes of traffic should be served independently

Class Selector PHB

- Objective
  - Backward compatibility with IP Precedence
  - 8 different class selectors
- Definition
  - Use of class selector should yield to at least 2 independently forwarded traffic classes
  - Router should give a higher probability of timely forwarding to CS_x packets than CS_y packets if x>y
  - Packets with CS6 and CS7 should receive a better treatment than best−effort traffic
    - several routing protocols use CS6/7 IP packets

The Class Selector PHB corresponds to packets with DSCP=xxx000 (i.e. The three high order bits are used like the former Precedence bits)

Sample CS implementations



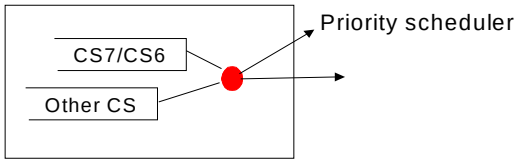
Edge router
May perform marking,
but usually marking will be
done by the applications
If CS7/CS6 is used used by routing
protocols inside domain A, edge
routers should block or remark
incoming packets with CS7/CS6

Border router
Same as edge router + remarking

Per Hop Behavior
Rely on scheduler to timely serve packets
with higher Class Selectors

Sample CS implementations (2)

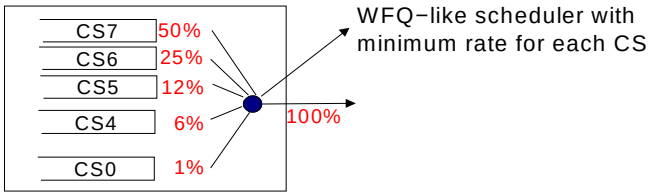
- Priority−based implementation



  - Advantages
    - Simple to implement
    - CS7/CS6 are timely served
  - Drawbacks
    - CS7/CS6 traffic may starve other traffic
      - assume BGP uses CS7 packets and BGP router reloads...
      - be careful in case of DoS attacks with CS7/CS6
    - Difficult to generalize to N CS codepoints
      - but only CS7/CS6 are used in practice

Sample CS implementations (3)

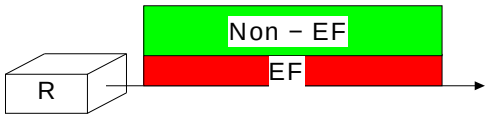
- WFQ−based implementation



  - Advantages
    - Starvation is not possible anymore
    - Packets with high CS are timely served
  - Drawbacks
    - more configuration is required than with PQ
    - needs some knowledge of traffic mix

EF : Expedited Forwarding

- Objective
  - Provide low delay, low loss, low jitter service
- Principle



  - On each node, the administrator should be able to configure a rate R_{EF} for EF traffic
  - The EF traffic should be served at rate $\geqslant R_{EF}$ independently of the intensity of any other traffic through the node

The original definition

V.Jacobson, K.Nichols, and K.Poduri. An expedited forwarding PHB. Internet RFC2598, June 1999.

The updates to clarify (notably mathematically the EF specification)

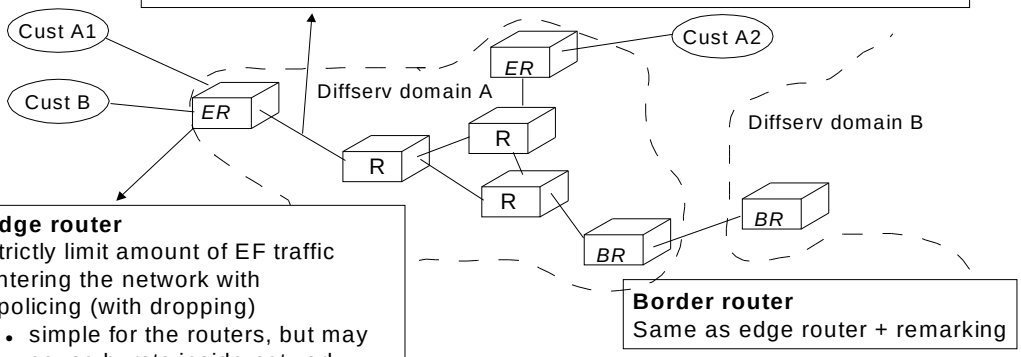
B.Davie, A.Charny, F.Baker, J.Bennet, J.−Y. Leboudec, K.Benson, A.Chiu, W.Courtney, S.Davari, V.Firoiu, C.Kalmanek, K.K. Ramakrishnan, and D.Stiliadis. An expedited forwarding PHB. RFC 3246, March 2002.

G.Armitage, A.Casati, J.Crowcroft, J.Halpern, B.Kumar, and J.Schnizlein. A delay bound alternative revision of RFC2598. RFC3248, March 2002.

A.Charny, F.Baker, J.Bennet, B.Davie, K.Benson, A.Chiu, W.Courtney, S.Davari, V.Firoiu, C.Kalmanek, K.K. Ramakrishnan, and D.Stiliadis. Supplemental information for the new definition of the EF PHB. RFC 3247 March 2002.

Sample EF implementations

On each link
Provision link so that amount of EF traffic is limited to small fraction of bandwidth



Edge router
Strictly limit amount of EF traffic entering the network with
- policing (with dropping)
  - simple for the routers, but may cause bursts inside network
- shaping (preferred)
  - complex for the routers, but reduces bursts inside network

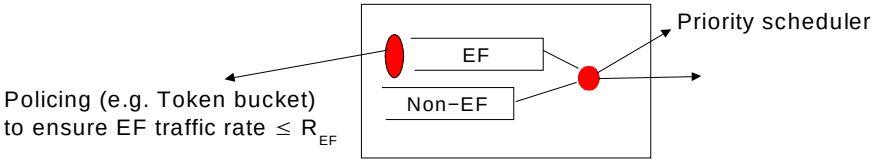
based on the customer's SLA

Border router
Same as edge router + remarking

Per Hop Behavior
Determine R_{EF} for each link at provisioning
Configure scheduler on each node so that EF traffic is always served at rate $\geqslant R_{EF}$

Sample EF implementations (2)

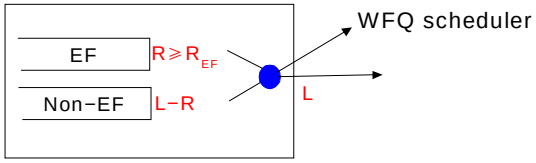
- Priority−based implementation



  - Comments
    - provides smallest possible delay for EF packets
    - traffic on each link should be regularly monitored to ensure that EF rate is not larger than R_{EF}
      - if the network is well engineered, no packets will be dropped by EF policer on interior nodes
      - if EF packets are dropped inside interior nodes, this might be due to overs ubscription or network problems (DoS, configuration errors, ...)
      - EF drops should be monitored

Sample EF implementations (3)

- WFQ−based implementation



  - Comments
    - provides larger queueing delay than PQ
      - average delay and delay jitter will decrease when R increases
    - policing is not required
      - policing (at R_{EF}) should be used when R >> R_{EF}
    - traffic on each link should be regularly monitored to ensure that EF rate is not larger than R_{EF}
      - EF drops from EF queue should be monitored

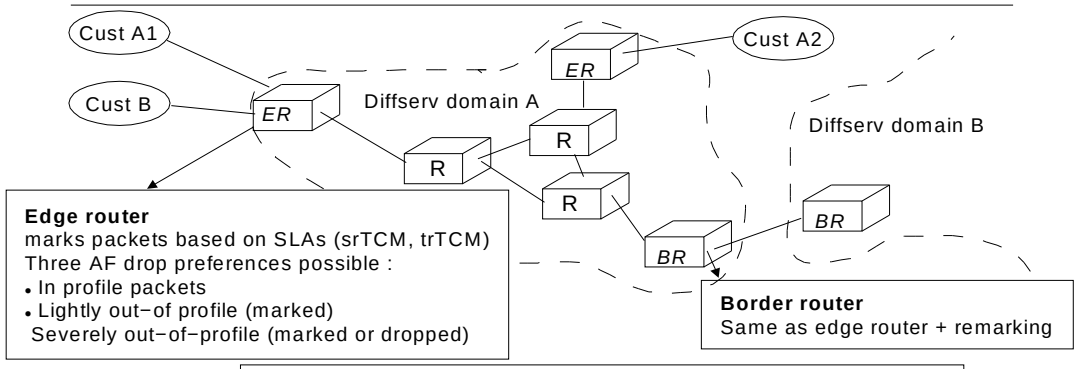
AF : Assured Forwarding

- Goal
  - provide differently provisioned services on top of a single IP network
    - business service with large over provisioning
    - paid dialup service correctly provisioned
    - under provisioned free dialup service
- Principle
  - Color assigned to each traffic class
  - edge routers color packets based on their class
  - network resources are divided into a few classes
    - business service has a lot of resources to avoid congestion
    - paid dialup has fewer resources, but can benefit from unused business resources
    - free dialup service utilizes the leftover resources

J.Heinanen, F. .Baker, W.Weiss, and J.Wrocklawski. Assured forwarding PHB group. Internet RFC 2597, June 1999.

N.Seddihg, B.Nandy, and J.Heinanen. An assured rate per−domain behaviour for differentiated services. Internet draft, draft−ietf−diffserv−pdb−ar−00.txt, work in progress, February 2001.

Sample AF implementation



Edge router
marks packets based on SLAs (srTCM, trTCM)
Three AF drop preferences possible :
- In profile packets
- Lightly out−of profile (marked)

Severely out−of−profile (marked or dropped)

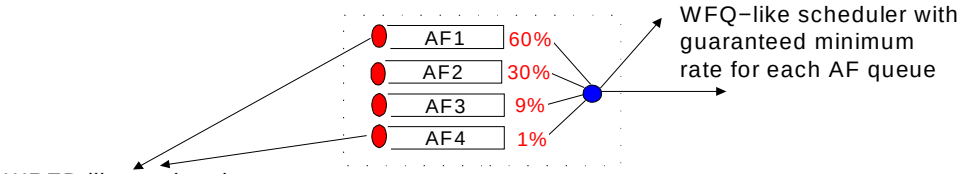
Border router
Same as edge router + remarking

Per Hop Behaviour
Four AF classes can be supported
- AF1 should rarely see congestion
- AF2 should see more congestion than AF1
- AF3 should see more congestion than AF1 and AF2

...
Inside each AF class
- first drop severely out−of−profile packets
- then drop lightly out−of−profile packets
- only drop in−profile packets if dropping out−of−profile is not enough

Sample AF implementation (2)

- Typical AF implementation



WRED like packet dropper
Inside AFX class,
If buffer becomes full, first drop AFX3 packets
If dropping AFx3 packets is not enough, also drop AFX2 packets
If buffer still becomes full, drop also AFX1 packets

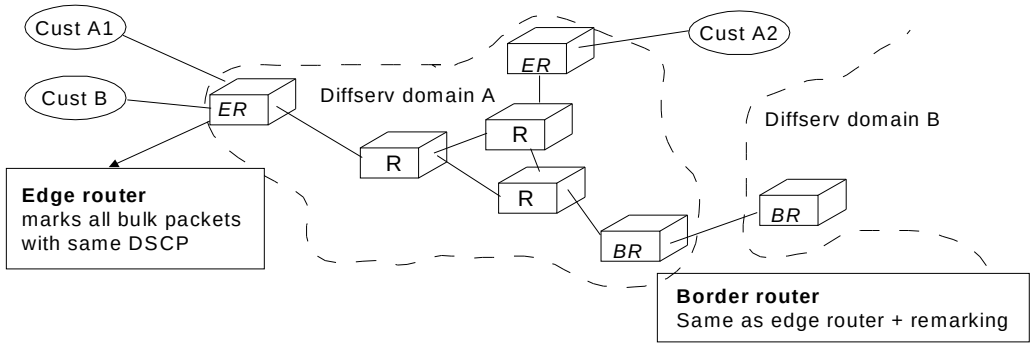
- Comments
  - a network does not need to support the 4 AF classes and the 3 drop preferences inside each class
  - weight associated to each queue depends on expected amount of AF traffic in each class

Bulk Handling PDB

- Objectives
  - Force bulk traffic to be handled during off−peak
    - Non−important applications should only consume bandwidth during off−peak hours
      - example : netnews for ISP or company
    - Non−important users should be discouraged to utilize network resources during peak hours
      - example : free ISP user
- Deployment
  - based on Class Selector PHB
  - based on Assured Forwarding PHB

B.Carpenter and K.Nichols. A bulk handling per−domain behavior for differentiated services. Internet draft,draft−ietf−diffserv−pdb−bh−00.txt, work in progress, October 2000.

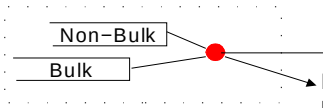
Supporting Bulk Handling



Per Hop Behaviour
Serve BH packets only if nothing needs to be served

Supporting Bulk Handling (2)

- Priority−based implementation

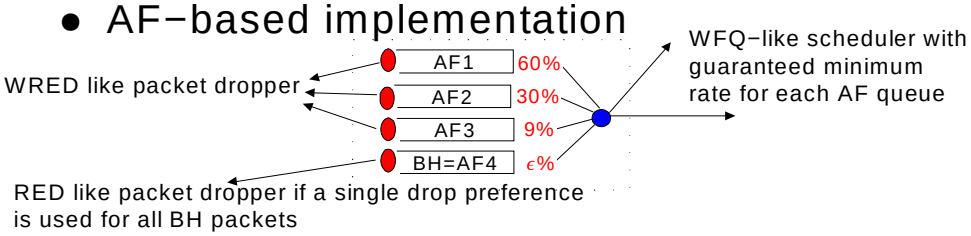


Priority scheduler with
low priority for Bulk queue
May cause starvation

- Comments
  - BH packets will only be served when no other packets needs to be transmitted
    - nice from a network utilization point of view
  - may cause starvation that could lead to problems with TCP−based applications
    - after N unsuccessful retransmissions of the same packet during starved periods, TCP will terminate the connection
    - some applications may not be ready to deal with terminated connections

Supporting Bulk Handling (3)

- AF−based implementation



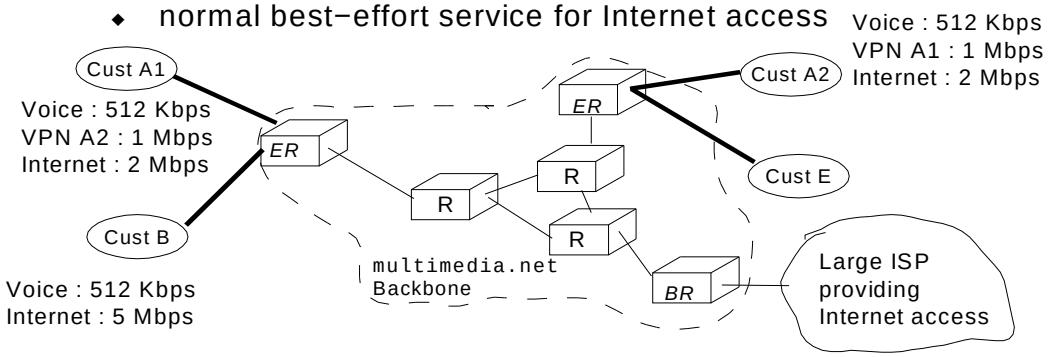
  - Comments
    - Assumes that AF4 class can be used
      - set rate of BH queue at lowest value supported by scheduler
        - BH packets always utilize some bandwidth, but this is usually better than PQ with starvation for TCP if AF4 queue is large
      - utilize RED−like packet dropper for AF4 queue
    - If all four AF classes or queues are already used, alternative is
      - use AF43 for BH packets and place BH packets in AF4 queue and drops them as soon as queue fills

The evolution of Internet QoS

- Packet−level traffic control mechanisms
- Integrated Services
- Differentiated Services
  - Architecture
  - The Per−Hop and Per−Domain Behaviors
    - Class Selector (CS)
    - Expedited Forwarding (EF)
    - Assured Forwarding (AF)
    - Bulk Handling
  - Case Study
  - Improving the scalability of IntServ with DiffServ

Case study
multimedia.net

- Case study
  - ISP multimedia.net wants to provide
    - low−delay, low loss service for voice over IP
    - low loss business critical data service for IPSEC VPNs
    - normal best−effort service for Internet access



    - internally, OSPF and BGP traffic should be preserved

multimedia.net
AF−based solution

- Principle
  - Define four AF classes
    - AF1 will be used for voice traffic
      - shaping instead of policing at edge routers
    - AF2 will be used for VPN traffic inside backbone
      - policing with marking at edge routers, but amount of out−of−profile packets will be limited (for example maximum 2 times AF2 rate)
    - AF4 will be used for best−effort traffic
      - edge routers will mark best−effort traffic
  - How to map routing protocols into classes ?
    - OSPF
      - low volume expect during reload and requires low delay
      - proposal : place OSPF packets in AF2 queue
    - BGP
      - no packets should be lost and delay should be low
      - iBGP session can produce bursty traffic when peers come up
      - use AF3 for BGP packets (if BGP routing is not needed for VPN)

Back to schedulers

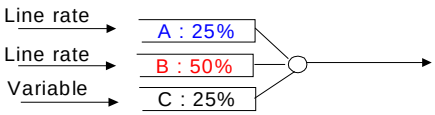
- Bandwidth distribution for most schedulers

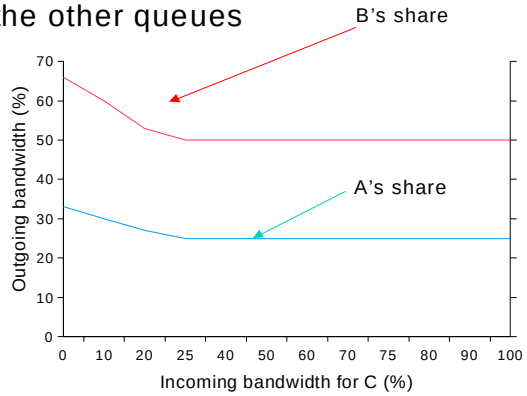
$$rate[i] = Bandwidth \times \left(\frac{(Weight[i])}{\left(\sum_{j=queues} Weight[j] \right)} \right)$$

  - provided all queues are active
- What happens if a queue is not active ?
  - The bandwidth unused by this queue is distributed among the active queues in proportion to their weight
    - few schedulers allow to control how the leftover bandwidth is distributed among the active queues

Back to schedulers (2)

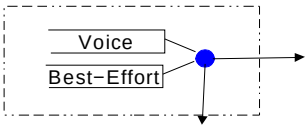
- Consequence
  - the bandwidth allocated by a scheduler to a queue is thus function of
    - the weight associated to the queue
    - the amount of traffic in the other queues





Supporting Voice with AF

- Is it possible to use AF for Voice traffic ?
- Characteristics of Voice traffic
  - requires a low delay
  - volume of traffic is bounded by sources or policing
- Example
  - Voice traffic consumes 10% of bandwidth

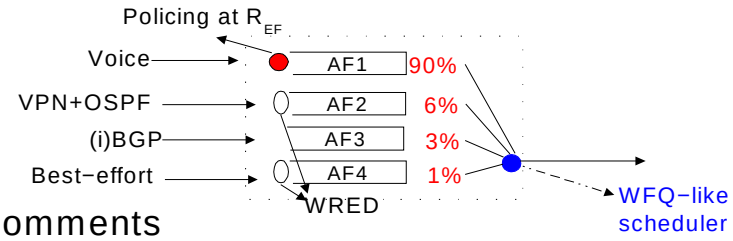


WFQ−like scheduler

- Bandwidth−based rate allocation
  - Voice : 10 % of bandwidth
  - Best−Effort : 90% of bandwidth
  - no voice packets will be lost
  - delay for voice will be high
- Delay−based rate allocation
  - Voice : 95% of bandwidth
  - Best−effort : 5% of bandwidth
  - only works if Voice traffic is bounded

multimedia.net
AF implementation

- Principle
  - Rely on a correctly configured WFQ scheduler



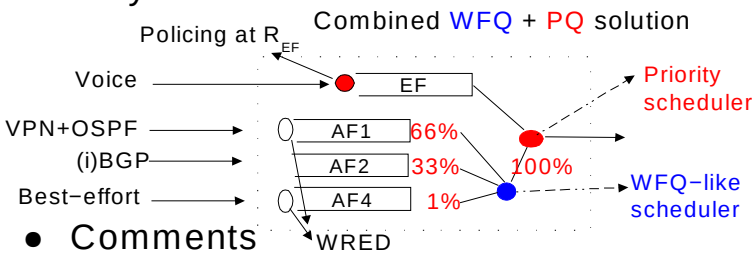
  - Comments
    - Assuming R_{EF}=10%, AF1 packets will never stay in queue
      - R_{AF1} = 6% of total + 60% of bandwidth unused by AF1
        - all OSPF packets should be in−profile
      - R_{AF2} = 3% of total + 30% of bandwidth unused by AF1
        - but (i)BGP will not consume so much bandwidth
      - Best−effort queue will utilize leftover bandwidth
    - Feasibility depends on granularity of scheduler weights

multimedia.net
AF+EF solution

- Principles
  - Utilize EF for voice and AF for the remaining
    - EF will be used for voice traffic
      - shaping at edge routers
  - Define three AF classes
    - AF1 will be used for VPN traffic
      - policing with marking at edge routers, but amount of out−of−profile packets will be limited (for example maximum 2 times AF2 rate)
    - AF2 will be used for iBGP traffic
      - edge and border routers will remark AF2 packets as Best−Effort
    - AF3 remains available for future usage
    - AF4 will be used for best−effort traffic
      - edge routers will mark best−effort traffic

multimedia.net
AF+EF implementation

- Principle
  - Rely on two schedulers



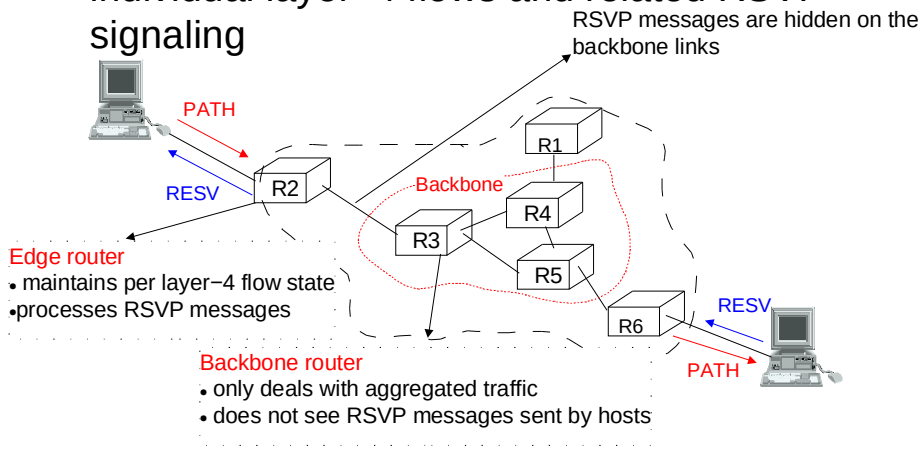
  - Comments
    - EF will obtain low delay thanks to PQ scheduler
    - WFQ scheduler will first serve VPN traffic
      - all OSPF packets should be in profile
    - (i)BGP traffic will be served earlier than best−effort
      - no SLA for iBGP and thus no WRED
      - but since amount of VPN and (i)BGP traffic is limited, best−effort will often be served

The evolution of Internet QoS

- Packet−level traffic control mechanisms
- Integrated Services
- Differentiated Services
  - Architecture
  - The Per−Hop and Per−Domain Behaviors
    - Class Selector (CS)
    - Expedited Forwarding (EF)
    - Assured Forwarding (AF)
    - Bulk Handling
  - Case Study
  - Improving the scalability of IntServ with DiffServ

How can we improve Intserv's scaling ?

- Backbone routers should not be aware of individual layer−4 flows and related RSVP signaling



This solution is described in

RFC3175 Aggregation of RSVP for IPv4 and IPv6 Reservations. F. Baker, C. Iturralde, F. Le Faucheur, B. Davie. September 2001. (Format: TXT=88681 bytes) (Status: PROPOSED STANDARD)

Some information on this topic is also available in

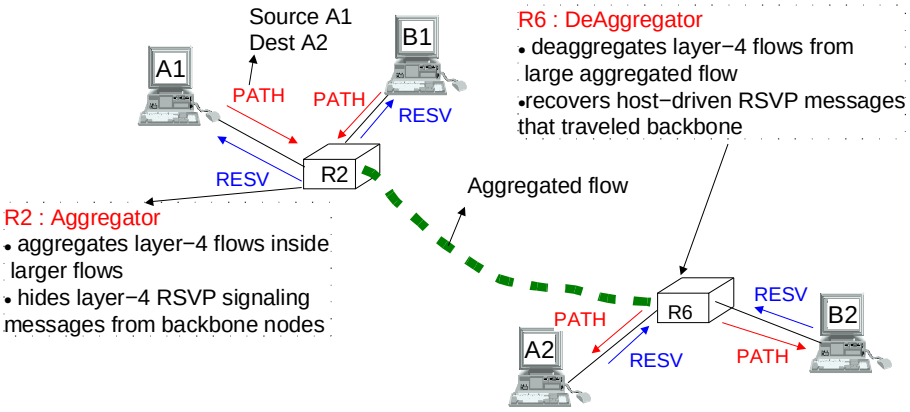
RFC2998 A Framework for Integrated Services Operation over Diffserv Networks. Y. Bernet, P. Ford, R. Yavatkar, F. Baker, L. Zhang, M. Speer, R. Braden, B. Davie, J. Wroclawski, E. Felstaine. November 2000. (Format: TXT=76378 bytes) (Status: INFORMATIONAL)

An important point to note about this solution is that it is targeted at a single backbone network. This implies that all routers show in the figure are part of the same autonomous system. If this solution was used to provide Integrated services across interdomain boundaries, then the border routers would need to process and maintain state for the RSVP messages of all Intserv flows that pass through them. This severely limits the scalability of this solution.

Reducing state information

- Basic solution
  - Integrated services and RSVP are used in the access network
    - allows to provide good QoS for layer−4 flows
    - number of flows is not too high in access network
  - Differentiated services is used inside backbone
    - differentiated services will only be used for the aggregated flows that cross the backbone
    - differentiated services is much more scalable than integrated services inside the backbone

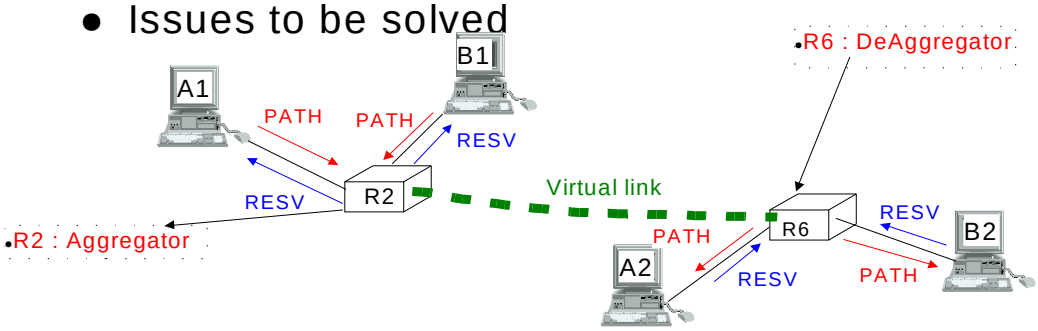
Intserv over a Diffserv backbone



- From the point of view of RSVP running on the hosts, the end−to−end path appears as containing one (virtual) link between R2 and R6
  - RSVP on hosts is not aware of the existence of backbone routers

Intserv over a Diffserv backbone (2)

- Issues to be solved



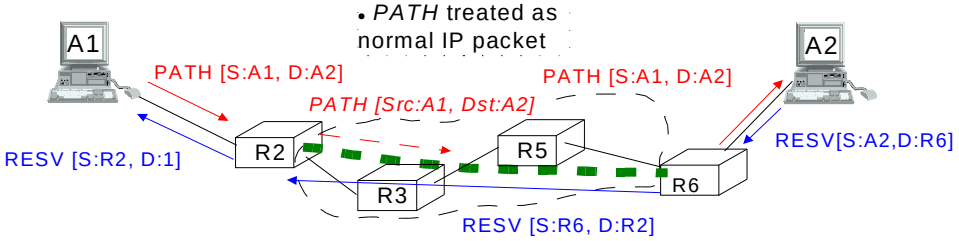
  - How to hide the end−to−end RSVP messages inside backbone ?
  - How to establish the virtual link R2−R6 ?
  - How to reserve bandwidth inside backbone ?
  - How to map GS/CL onto AF/EF/... ?

Hiding RSVP inside backbone

R2 : Aggregator
- PATH message is placed inside IP packet with Router Alert option and other Prot. Than RSVP
- PATH message is sent towards R6

Backbone router
- *PATH* treated as normal IP packet

R6 : Deaggregator
- router notices *RSVP* message
- PATH message placed in IP packet with Prot. = 46 and sent towards A2



R2 : Aggregator
- RESV message received from downstream
- new RESV message sent to A1

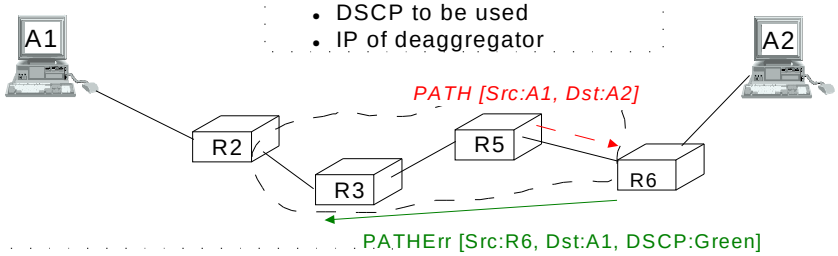
Backbone router
- RESV message with Dest = R2 ignored by R3 & R5

R6 : Deaggregator
- router received *RSVP* message
- new RESV message generated and sent towards R2 accross backbone

RSVP messages are normally sent as IP packets with Router Alert option and `Protocol` field set to 46

How to establish Virtual Link ?

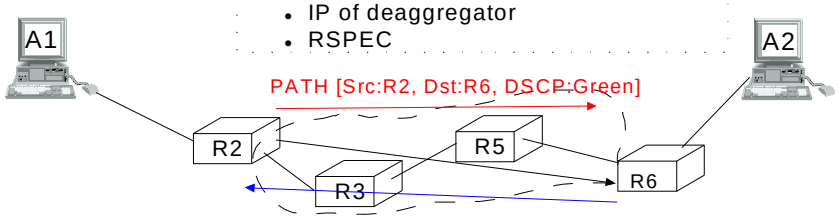
R6 : Deaggregator
- PATH received without existing virtual link
- Send PATHErr
  - DSCP to be used
  - IP of deaggregator



R2 : Aggregator
- PATHErr intercepted
- New reservation needs to be established
  - PATH message to R6 with
    - DSCP proposed by R6
    - TSPEC is guessed by R2

How to establish Virtual Link (2) ?

R6 : Deaggregator
- PATH received from R2
- Aggregated reservation to be established
- Send RESV message
  - DSCP to be used
  - IP of deaggregator
  - RSPEC



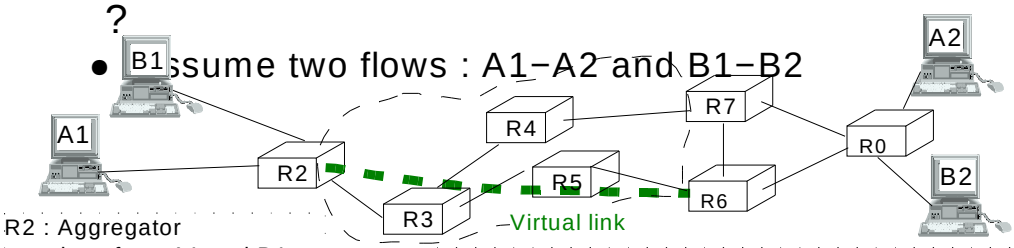
R2 : Aggregator
- virtual link is established with reservation
- confirmation with RESVCon
- R2 knows that some traffic can use DSCP=Blue to exit through R6

RESVConf

Virtual link : all packets with DSCP=Blue

Utilization of Virtual Link

- How can aggregator be sure that some flows will exit through a given deaggregator ?
  - Assume two flows : A1−A2 and B1−B2



R2 : Aggregator
- packets from A1 and B1 marked with DSCP=Green

R3 : Backbone router
- forwards packets on its own
- packets from A1 and B1 may not follow same path as PATH messages sent by R2 to R6

  - some tunneling (like MPLS) is preferable to ensure that packets follow virtual link

Service mapping : Controlled Load

- How to map Controlled Load onto a differentiated service ?
  - Expedited Forwarding (EF)
    - EF can provide a virtual link type of service but
      - nonconformant packets will be delayed or discarded by EF while Controlled Load expect them to be treated as best effort
    - not suitable for Controlled Load unless nothing else exists
  - Assured Forwarding (AF)
    - can provide a guaranteed bandwidth if network is well managed and not oversubscribed
    - conformant packets from RSVP flows will be marked with a low discard probability
    - nonconformant packets from RSVP flows will be marked a higher discard probability

For a discussion of this mapping, see :

J. Wroclawski, A. Charny, Integrated Service Mappings for Differentiated Services Networks, Internet Draft , draft−ietf−issll−ds−map−00.txt , March 2000

Service mapping : Guaranteed Service

- How to map Guaranteed Service onto a differentiated service ?
  - Assured Forwarding
    - AF does not provide any delay guarantee
  - Expedited Forwarding
    - the most natural solution, but Guaranteed Service requires a mathematically proven upper bound on the delays through the virtual link
      - propagation delay
      - shaping delay since EF assumes shaping at the edge
      - queuing delay inside the diffserv network
        - some work is being done to find such bounds, but the risk is that they might be too conservative to be useful in practice

Thank you

Questions and comments can be sent to

Olivier Bonaventure

Computing Sciences and Engineering Dept.
Université catholique de Louvain (UCL)
Place Sainte−Barbe, 2, B−1348, Louvain−la−Neuve (Belgium)
Email : Bonaventure@info.ucl.ac.be
URL : http://www.info.ucl.ac.be/PEOPLE/obo

This half−day tutorial is based on a two−days tutorial on "Traffic control and QoS in IP networks" that was given several times during the last few years.

Some of these tutorials were recorded and those recordings are available upon request by sending an email to Bonaventure@info.ucl.ac.be .