Quality of Service in the Internet: MPLS, Intserv and Diffserv
Broad outline

- MultiProtocol Label Switching (MPLS)
- Integrated Services (intserv): RSVP
- Differentiated Services (diffserv)
- Tying it all together

Overview paper:


covers intserv, diffserv, MPLS
MultiProtocol Label Switching (MPLS)
Resources

**Keshav**: not covered

**Standards**:  
- Primary: RFC3031  

**Advocate**: http://www.mplsforum.org/

**Tutorials**:  
- Papers:  
- Courses: http://www.nanog.org/mtg-9905/ppt/mpls/

Label Switching

**Label switching**: Labels of fixed length select state in switches that indicate how to switch packet (port, QOS, new label).

Benefits:
- **Simplify core network**:  
  - switches in network core merely lookup & swap fixed-length labels  
  - complicated multi-field longest prefix matching to determine labels is done at edges  
    - need only be done once  
    - is done where there are relatively few flows
- Traffic management
- Provide new services, *e.g.* Virtual Private Networks
Multiprotocol Label Switching

**Multiprotocol Label Switching (MPLS)**:

- Initiated by desire to send IP over ATM, e.g. to interconnect routers.
  (Original technique to interconnect routers was to emulate broadcast LAN on ATM, but that required large numbers of point-to-point VCs, unproven ATM routing protocols, & obscured QOS benefits of ATM)
- Advocated by Cisco, IBM, Ipsilon and others – some called “tag switching”
- Generalized in IETF to multiple protocols: IP/IPX/NetBEUI/etc over ATM/Frame Relay/Ethernet/etc

† Like ATM, MPLS also has its detractors, e.g. “Much Preaching, Little Substance”
Comparison with ATM

Similarities:
• Tag switching – fast (no complicated longest prefix matching), resource guarantees
• Hierarchy of tags: ATM has 2 levels: VPs aggregate VCs; Variable packet lengths allow more than 2 MPLS labels to be stacked

Differences (IP flavour):
• Packets have variable length
• “Routing” can be done using established IP protocols (e.g. BGP), rather than newer ATM protocols (e.g. P-NNI).
  Often described as “layer 3 routing + layer 2 switching”, where:
  • “Routing”: General determination of which port to send packets so that they reach their destination, satisfy QOS requirements, and minimise network load.
  • “Switching”: Using a label to index a table to determine which port to send a particular packet at a specific instant. Tables may vary over time as routing protocols discover better routes.
MPLS terminology

Label Edge Router (LER) – pushes (+) / pops (-) labels
Label Switched Router (LSR) – switches using MPLS labels
Label Distribution Protocol – enables an upstream router (closer to source) to be allocated a label by a downstream router.
Label Switched Path (LSP) – similar to ATM VC/VP
Forwarding Equivalence Class (FEC): A set of packets that are all treated the same way by a router

Not necessarily by all routers – e.g. packets with different DAs may belong to same FEC in the midst of the network
Conventional IP routers determine FEC at each hop, MPLS routers (LERs) determine FEC only at entry to domain.
MPLS label stack entries

<table>
<thead>
<tr>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
</tbody>
</table>

++-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ Label
|                 Label                 | Exp |S|     TTL       | Stack
+++-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ Entry

20b **Label.** Values $[16,2^{20}-1]$ identify FECs. Other special values:

0: **IPv4 Explicit Null Label:** End of LSP, next protocol is IPv4
1: **Router Alert Label:** bring pkt to router’s attention, next label=FEC
2: **IPv6 Explicit Null Label**
3: **Implicit Null Label:** For LDP only, not for packets
4-15: **Reserved**

3b **Experimental:** obsolete – was Class Of Service

1b **Bottom of stack:** =1 for last label in stack

8b **Time-To-Live,** adjusted by switches & returned to IP @ egress

This is the format of the “shim layer”, added when the link layer cannot convey labels (more shortly…)
Stacks of labels

Packet can have multiple labels (in a “stack”), each indicating how it should be switched in different domains.

Labels can be pushed onto the stack at:

- source (source routing), or
- ingress router to domain

  - Helps scalability/hierarchical routing: Domain routes aggregate (common outer label of stack) rather than individual flows.
  - At exit to domain, the common label is popped, and future routing is done according to next label in the stack.
  - Pushing a label may extend the packet length beyond the MTU ⇒ LERs need to know what the MTU is, and be prepared to fragment packets if label pushing extends length beyond MTU

Labels are popped off the stack by egress routers

Last label indicates protocol: IP, IPX, etc
Pushing & popping *multiple* labels

Transit network (middle):
- pushes on label *(blue)* as packets enter
- this label is used for routing within this network
- pops off label as packets depart
How are labels carried?

Labels carried by lower layer whenever possible

In ATM: Use VPI/VCI (e.g. each carries one MPLS label; VPI for outer label & VCI for next label, etc)

• (Assuming there is a dedicated ATM link between MPLS routers. If they are connected by a SVC, then they cannot use the ATM header to carry this information.)
• Need to be careful about MPLS labels not being mapped into reserved ATM VPI/VCI values, e.g. as used for signalling
• If VPI/VCI is used for labelling, then it can’t be used to distinguish different packets (e.g. coming from different sources & passing through same egress router), e.g. input:

```
in the  VPI/VCI  the cat  beginni  sat on
```

⇒ “cell interleave problem”. Solved with output using either:

- separate labels
- reassembly prior to forwarding (“VC merge”)
In Frame Relay: Use 10b DLCI (<20b ⇒ limited label space)
TTL & CoS carried in first 32b label stack entry

In other protocols (e.g. Ethernet & PPP) use shim layer
Always have at least one shim layer header to indicate end of stack & network-layer protocol
MultiProtocol Lambda Switching (MP\(\lambda\)S) / Generalized MPLS (GMPLS)

Extension of MPLS for use on all-optical networks

Labels correspond to wavelengths (lambdas)

Wavelength conversion is difficult; MPLS ideas mainly used for signalling to establish lightpath across an optical network.

See RFC 3945 for more on GMPLS
Label Distribution

Labels are allocated by downstream router & distributed to upstream router using:

- a Label Distribution Protocol (LDP – RFC 3036), or
- extending routing protocols (e.g. BGP – RFC 3107), or
- signalling (e.g. RSVP) protocols to carry labels.
Setting up Label Switched Paths

What causes a Label Switched Path to be set up? Either:

- Data driven: On demand by source transmitting packets
- Control driven: When routing topology changes, or explicit signalling request to establish a LSP

How is the LSP determined?:

- Layer 3 hop-by-hop routing, of either:
  - Initial packets that are transmitted & stimulate data-driven setup
  - Control signals
- an Explicit Route (source route) specified by the source
Traffic Engineering with MPLS

IP routing protocols (e.g. OSPF/IS-IS) select shortest paths (number of hops), often loading them excessively while underutilising longer paths.
Traffic Engineering attempts to even-out the loadings.

“constraint-based routing”: Routing that has multiple constraints (not just finding the shortest path):
• meet multiple QOS requirements e.g. jitter, BW, loss
• meet policy requirements e.g. distribute load, ↓$
More variables increases router computation load

Route instability can occur, with routes changing excessively often because of changing variables.
MPLS in use

$ traceroute nytimes.com
traceroute to nytimes.com (199.239.137.245), 30 hops max, 38 byte packets
1 eebu4s1.uwn.unsw.EDU.AU.92.171.149.in-addr.arpa
2 129.94.255.181
3 gigether0-2-3.bb1.a.sydney.aarnet.net.au
4 gigether0-2-3.bb1.a.sydney.aarnet.net.au
5 pos2-0.bb1.a.pao.aarnet.net.au
6 p4-4-1-0.r00.plalca01.us.bb.verio.net
7 p16-0-1-1.r21.plalca01.us.bb.verio.net
   MPLS Label=433170 CoS=3 TTL=1 S=0
8 p64-0-0-0.r21.mlpsca01.us.bb.verio.net
   MPLS Label=260096 CoS=3 TTL=1 S=0
9 xe-1-2-0.r20.mlpsca01.us.bb.verio.net
10 p16-1-1-1.r20.nycmny01.us.bb.verio.net

... [traced on June 2, 2005. IP addresses & delay measurements omitted]
   “mlpsca” = Milpitas, California – not a MPLS typo]
RSVP outline

General architecture
RSVP is separate from routing protocols
Soft state: Definition, Benefits, Costs
How RSVP supports multicast

Messages
Formats
PATH & RESV

Using RSVP
merger & filters
scalability
RSVP over reservationless networks
Resources

Overview papers:
C. Metz: “IP QOS: Traveling first class on the Internet”, *IEEE Internet Computing*: 84-8, 1999
covers intserv, diffserv

RSVP:
*Keshav*: pp. 471-5
*Standards*: RFCs 2205-2216
*Advocate*: http://www.isi.edu/div7/rsvp/
*Tutorials:*
*IEEE Net. Mag.*, 7(5):8-18, Sep. 1993
Chapter 6 of Jha and Hassan
Resource reSerVation Protocol (RSVP)

- Offers 2 types of service:
  - **Guaranteed service** [RFC 2212]: Provides firm end-to-end delay bounds
  - **Controlled-load service** [RFC 2211]: No firm delay bounds, but service shouldn’t deteriorate as network load increases.

Both involve admission control.

- Separate from routing protocols →
- Uses “soft state” →
- Designed to support group communication →
RSVP is separate from routing protocols

Designed to be separate from the routing protocol

- Simplifies RSVP
- Allows operation with existing routing protocols
- May hinder ability to provide QOS guarantees (Route that provides necessary QOS depends on the connection’s session’s profile & requirements of traffic ⇒ depends on signalling)
Soft and hard state

RSVP uses soft state (PATH and RESV messages are periodically resent)
Compare with ATM, which uses “hard state” …

Under normal conditions:
(“normal”: systems using state info remain available, e.g. haven’t crashed)

- **Hard state** persists until it is explicitly released.
- **Soft state** persists only for a moderately short interval.
  - It must be refreshed if it is to persist longer.
  - Failure to refresh will cause implicit release.

i.e. soft state is *often* refreshed, whereas hard state is not.

[Keshav pp. 108-9]
Soft state: Evaluation

Benefits:
✓ **Adaptive:** routing path changes redirect endpoint refresh messages, creating new state.
✓ **Robust:** Packets can be lost and switches can crash, but service will continue until either
  • there is no path (service is impossible) or
  • end-system loses state
✓ Soft state reflects the end-to-end arguments: State is needed until end system dies ⇒ sensible to store it in the end-system.
✓ Soft state simplifies signalling, since switches needn’t explicitly adapt to changes, or recover from failures.

Costs:
✗ Bandwidth to refresh the state
✗ Need authentication to prevent session hijacking by injecting fake state info.
RSVP supports *group* communication

Designed at the outset to support group communication:

1. **Receiver oriented** *(c.f. sender-oriented ATM)*
   - Receivers initiate reservations *(source initiates a multicast group, but receivers initiate their connection to the group)*
   - Merger of reservations enables many receivers without many requests imploding at source
   - Reservations can be heterogeneous, according to differing requirements
   - It is the *receiver* that experiences QOS & usually pays for service

2. State information is continually refreshed →

3. Reservations are for “sessions” →
RSVP supports group communication (continued)

2. State information is continually refreshed (soft state), reflecting updates to group membership. Multicast membership changes more often than unicast ⇒ continuous refreshing is appropriate.

3. Reservations are for “sessions”: traffic to a particular multicast group (not from a particular source)
   - A session may have multiple sources, e.g. videoconference
   - Reserving resources for sessions, rather than sources, allows admission of more sessions: often only one or a few sources transmit at any instant.
Outline
RSVP message format: Header

Common header (for all message types):

```
  0             1             2             3
+-------------+-------------+-------------+-------------+
| Vers | Flags|  Msg Type   |        RSVP Checksum      |
+-------------+-------------+-------------+-------------+
|   Send_TTL  | (Reserved)  |        RSVP Length        |
+-------------+-------------+-------------+-------------+
```

Flags: only 1b ever used to control refresh overhead – see RFC 2961

Message types:

- 1 = Path
- 3 = PathErr
- 5 = PathTear
- 2 = Resv
- 4 = ResvErr
- 6 = ResvTear
- 7 = ResvConf

TTL: Compared with IP’s TTL to detect non-RSVP hops on path.
RSVP message format: Objects

“objects” follow the header and have the following form:

```
+-------------+-------------+-------------+-------------+
|      Length (bytes)       |  Class-Num  |    C-Type   |
+-------------+-------------+-------------+-------------+
|                                                       |
+-------------+-------------+-------------+-------------+
//                  (Object contents)                  //
+-------------+-------------+-------------+-------------+
```

Class-Num and Class-Type essentially define the object type

**INTEGRITY** object that optionally follows header is for message authentication
**RSVP message types**

<table>
<thead>
<tr>
<th>Transmission direction</th>
<th>Commands</th>
<th>Responses</th>
<th>Errors</th>
</tr>
</thead>
<tbody>
<tr>
<td>Downstream</td>
<td>Path, PathTear</td>
<td>ResvConf</td>
<td>ResvErr</td>
</tr>
<tr>
<td>Upstream</td>
<td>Resv, ResvTear</td>
<td>PathErr</td>
<td></td>
</tr>
</tbody>
</table>

**Path**: Gathers information about path from source

**Resv**: Reserves resources

**Tear**: Release resources before soft-state timeout

**Err**: Error interpreting message, or honouring reservation
PATH message

4 roles:
• records network capacity on path from source (AD_SPEC)
• informs receivers of source’s capabilities (SENDER_TSPEC)
• refreshes state in routers (e.g. sent every 30 seconds)
• marks the path from the source (RSVP_HOP), for return of RESV …

<Path Message> ::= <Common Header> [ <INTEGRITY> ]
  <SESSION>
  <RSVP_HOP>
  <TIME_VALUES>
  [ <POLICY_DATA> ... ]
  [ <sender descriptor> ]

<sender descriptor> ::= 
  <SENDER_TEMPLATE>
  <SENDER_TSPEC>
  [ <ADSPEC> ]

(Receivers use IGMP to join group, informing routers of route for PATH msg)
† Larger ⇒ lower overhead, but less responsive to network changes.
Receivers may send RESVs with period α # rx for scalability ⇒ router needs to know period.

† Larger ⇒ lower overhead, but less responsive to network changes.
Receivers may send RESVs with period α # rx for scalability ⇒ router needs to know period.
RESV message

Role: makes reservation, acts as ack for PATH message

- No explicit ack for RESV: Receiver knows of success through payload delivery
- RESV error messages (in case reservation isn’t possible)

<Resv Message> ::= <Common Header> [ <INTEGRITY> ]
<SESSION>
<RSVP_HOP>
<TIME_VALUES>
[ <RESV_CONFIRM> ]  \( IP \text{ address of receiver requesting confirmation} \)
[ <SCOPE> ]  \( \text{Lists targets of this RESV} \)
[ <POLICY_DATA> ... ]
<STYLE>  \( \text{Indicates if RESV should apply } \forall \text{ or some sources in sess} \)
<flow descriptor list>

<flow descriptor list> ::= <empty> |  
<flow descriptor list>  
<flow descriptor>  \( \text{Defines flow & source that it applies to} \)
Outline
Merger of reservations

Routers that are branches in the multicast tree can merge reservations from multiple receivers

- Prevents “implosion” of multiple requests, helping scaling to large numbers of receivers
- S3 need not reserve additional bandwidth for C (seeking baseband layer) when it already carries the baseband layer for B.

B sends RESVs every 30sec for baseband + enhancement

C sends RESVs every 30sec for baseband

S3 sends RESVs every 30sec for:
  - baseband (for B&C)
  - enhancement (for B)
Filters

RSVP provides different “styles” of reservations, e.g.:

- **Shared Explicit**: Packets from any source in group can be transmitted using the reservation. Useful for videoconference:
  - Video from one (speaking) participant at a time (plus, perhaps, previous speaker)
  - Don’t want to have to reserve bandwidth for all participants.

- **Filtered reservations**: Receiver sends signals to switch indicating which source it wishes to receive from. Reservations from multiple receivers can still be merged.

A-E seek access to all sources (2, 7, 9, 10), but only one at a time.
- Reservations ensure access to all.
- Filters specify which source.
  - e.g. S3 reserves on link to S2 capacity for 2 channels now; 3 in worst case.
- Switches can merge reservations
  - e.g. S2 needn’t reserve 5 channels for A-E on link to S1
RSVP scalability

- Scales *well* in terms of *source overhead* for increasing numbers of receivers.
- Scales *poorly* in terms of *switch overhead* for increasing numbers of sessions through that switch:
  Each switch on the end-to-end path must process signalling, not just to establish & release the state, but also continuous refresh messages.
RSVP is best at edges of network, where there are few flows
RSVP over reservationless networks

May need to connect networks supporting reservations with reservationless networks (LANs, or IP)

- RSVP messages still pass over these networks
- QoS can only be assured if the reservationless networks are not the bottleneck

Common arrangement: interconnect RSVP networks with a **diffserv** network ⇒

- edges (where scalability isn’t critical) have per-flow state
- network core only has state for aggregates (e.g. all TV packets)

Need to map between services:

<table>
<thead>
<tr>
<th>intserv</th>
<th>diffserv</th>
</tr>
</thead>
<tbody>
<tr>
<td>Guaranteed</td>
<td>Expedited (premium)</td>
</tr>
<tr>
<td>Controlled load</td>
<td>Assured</td>
</tr>
</tbody>
</table>

For details, see RFC 2998
Outline
diffserv references

Keshav: not covered

Standards:
  Primary: RFC2475
  Others: RFC 2597 (Assured Forwarding),
          RFC 3246 (Expedited Forwarding)
  http://www.ietf.org/html.charters/diffserv-charter.html

Introductory papers:
Diffserv terminology

Per Hop Behavior (PHB): Defines the behaviour of a switch when forwarding certain packets, e.g. what service they receive.

Behavior Aggregate (BA): A group of packets passing through a common network point that have a common label (DSCP) so that they experience the same PHB at that point.

Different BAs experience different behavior ⇒ “Differentiated Services”

Differentiated Services Code Point (DSCP): An IP header field.
IP headers carry diffserv labels

DiffServ information is carried in an 8b IP header field:
- IPv4: “Type of Service”
- IPv6: “Traffic Class”

```
+---+---+---+---+---+---+---+---+
|          DSCP         |  ECN  |
+---+---+---+---+---+---+---+---+
```

Routers can remark the DSCP field:
- In particular, first-hop routers remark when sources set it to indicate traditional Type of Service
- DSCP interpretation can be customized for specific domains ⇒ remarking at domain boundary
Diffserv entities

Domains: Contiguous network regions, with common administrative ownership, set of service provisioning policies and PHB definitions.

Boundary node: Point of entry (ingress) or exit (egress) from domain. They condition traffic & set DSCP labels (like MPLS LER)

Interior nodes: Within a domain, but not a boundary node
Forward transit traffic, with service determined by label (like MPLS LSRs)
May mark locally generated traffic (may also be done by source)
Service Level Agreements

Service Level Agreements (SLAs) encompass:

- accounting and billing
- availability levels
- Traffic Conditioning Agreement:
  - Traffic profile that source must abide by (e.g. Leaky Bucket)
  - Performance metrics that network will deliver (delay, throughput, loss)
  - Defines how out-of-profile packets will be handled

Each source makes an agreement with their local domain
Domains make agreements with each other in order to satisfy their commitments to sources.
Agreements are often static and made manually.
May also be made dynamically through a signalling protocol, e.g. RSVP.
Per Hop Behaviors

Best-effort service (DSCP=000000)

**Assured Forwarding** (AF) (aka “assured service”) (DSCP=$x_1x_2x_3y_1y_20$)
- “Assured” in that this traffic gets lower loss rate than best effort
- Excess traffic is converted to best effort
- Theoretically assured for a *source* (not S-D pair), but that leads to inefficient provisioning, since source may transmit to arbitrary destination – allocate resources on all possible links?!

12 types (AF$xy$) ($x_1x_2x_3=001,010,011$ or $100$, $y_1y_2=01,10$ or $11$†)
- 4 classes ($x$), each assigned separate resources (buffer & “bandwidth”)
- 3 drop precedences ($y$) per class. Prob(drop)$\propto 1/y$

**Expedited Forwarding** (EF) (aka “premium service”) (DSCP=101110)
- low loss (like AF) *and* low delay, e.g. for voice/video
- Traffic entering network must be rate limited

† $y_1y_2=00$ is reserved for backward compatibility with traffic that uses IP precedence
Bandwidth Brokers

Each Domain has a Bandwidth Broker (BB), responsible for admission control for that domain (c.f. ATM/MPLS/RSVP where admission control is distributed amongst switches in domains). Signalling between BBs is done using a protocol such as RSVP or LDAP (Lightweight Directory Access Protocol – RFC 1777). BBs can aggregate multiple requests before forwarding an aggregated request to the next BB.

Fig. from Xiao 99
Multicasting complicates diffserv

Diffserv is *source* oriented

But source may not know addresses of multicast members & membership may change rapidly

⇒ difficult for network to anticipate load that a multicast flow will create

⇒ difficult to assure performance

For details, see Internet Draft draft-bless-diffserv-multicat-03.txt: R. Bless & K. Wehrle: IP Multicast in Differentiated Services Networks
<table>
<thead>
<tr>
<th></th>
<th>Intserv</th>
<th>diffserv</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unit of operation</td>
<td>individual sessions</td>
<td>aggregates</td>
</tr>
<tr>
<td>⇒ granularity of isolation &amp; guarantees</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Signalling frequency</td>
<td>frequent (each flow ctsly refreshed)</td>
<td>rarely (SLAs)</td>
</tr>
<tr>
<td>Based on</td>
<td>reservation for <em>rx</em></td>
<td>prioritization of <em>sources</em></td>
</tr>
<tr>
<td>Able to guarantee absolute service to end-users?</td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>Service scope</td>
<td>end-to-end</td>
<td>domains</td>
</tr>
<tr>
<td>(end-to-end requires extensive network upgrade)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Complexity</td>
<td>routers must process signalling for individual flows ⇒ poor scalability</td>
<td>simple internal nodes, boundary nodes must classify &amp; condition traffic</td>
</tr>
<tr>
<td>Compatibility with multicast traffic</td>
<td>good</td>
<td>problematic</td>
</tr>
</tbody>
</table>
## Tying it all together

<table>
<thead>
<tr>
<th>Role of RSVP</th>
<th>MPLS</th>
<th>RSVP</th>
<th>diffserv</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Terminology</strong></td>
<td>Label Switched Router</td>
<td>Label Edge Router</td>
<td>Forwarding Equiv. Class</td>
</tr>
<tr>
<td>may distribute labels</td>
<td>reserves for rx</td>
<td>may reserve for source/domain</td>
<td></td>
</tr>
</tbody>
</table>

> RSVP can be used with *all* of the above (not intended to confuse you!)
The end.

Next week: We’ll discuss the exam and review the course.