Quality of Service (QoS) Architectures in IP Networks

- Integrated Services (Intserv)
  - History
  - RSVP (ReSource Reservation Setup Protocol) and Intserv architecture

- Differentiated Services (Diffserv)
  - History
  - Diffserv architecture
    » Boundary Node Architecture
    » Per-Hop-Behaviors (PHB)
      • Expedited Forwarding (EF) PHB
      • Assured Forwarding (AF) PHB

- QoS Service Models in IP Networks
  - Better than Best Effort service
  - Virtual Leased Line service
  - Intserv/diffserv service
  - Quantitative Assured service

- Hybrid Use of Intserv and Diffserv
**Integrated Services (Intserv) - History**

- RSVP (Resource Reservation Setup Protocol) first appeared as a JSAC paper (L. Zhang et al., 1993)

- The growing demand for an integrated services Internet for support of delay- and loss-sensitive applications has resulted in RFC 1633 “Integrated Services in the Internet Architecture: An overview” (1994) that proposes RSVP as the signaling protocol for Intserv

- Two service definitions have been specified by the IETF
  - Guaranteed Service (RFC 2212, 1997)
    » Assured level of bandwidth, a firm end-to-end delay bound, and no queuing loss for conforming packets: emulates a virtual circuit (e.g., ATM VBR VC)
  - Controlled Load (RFC 2211, 1997)
    » No firm quantitative guarantees: service equivalent to that seen by a best-effort flow on a lightly loaded network

- RSVP functional specification finalized (RFC 2205, 1997)

- Guaranteed and controlled-load services have been formally specified for use with RSVP (RFC 2210, 1997)
Why resource reservation in IP networks?

- Basic reason: Demand exceeds resources
- Best effort model implies “equal” share
  - Goes down as the number of users increases
  - No guarantees
- What options if I need a minimum guarantee
  - Ensure resources “always” exceed demand
    » Typically long time scale (capacity planning)
    » Loose guarantees
  - Set aside (reserve) specific amount of resources for a given user
    » Ask for resources Signalling
    » Allocate resources Call admission
    » Monitor usage Flow control, policing
    » Enforce allocation Scheduling, buffer management
What is RSVP

- RSVP is a signalling protocol to request allocation of resources to a flow in an IP network.

- Major characteristics of RSVP:
  - Application initiated (fine granularity of reservation).
  - Designed for scalability to a very large multicast group.
    - Receiver oriented model, e.g., as with ATM LIJ.
  - Reservations are for simplex flows.
  - Support for heterogeneous reservations (multicast) and renegotiation of reservations.
  - Allows sharing of reservations across multiple flows.
  - Soft state approach for simple error recovery.
Some Differences with ATM Signalling

- Receiver versus sender oriented reservations
- Simplex versus duplex connections
- Soft state versus hard state
- Support for
  - Sharing
  - Renegotiation
  - Heterogeneity
- No distinction between UNI and NNI, i.e., same protocol for hosts and routers (end-to-end perspective)
Overview of RSVP Role and Interfaces

- RSVP aware application
- RSVP daemon
- Routing interface
- Routing protocol/table
- Admission control
- Control path
- Local resources management
- Packet classifier API
- Packet classifier
- Link scheduler
- Data path
- Socket API
- RTP
**Intserv - Use of RSVP and Architecture**

1. **PATH message** from sender contains the Traffic Specification (TSpec) that profiles the data flow to be sent.

2. **PATH Message** follows the "downstream" data route to receiver(s). Each RSVP-enabled router installs PATH state and forwards PATH message to next hop on route to receiver(s).

3. **RSVP Message** goes "upstream" following the Source Route provided in PATH message. Each RSVP-enabled router makes the allocation and forwards PATH message, or rejects it and returns an error back to receiver.

4. **RSVP message** contains resource reservation request, which contains TSpec from sender, RSpec with QoS level (controlled or guaranteed), and "Filter Spec" (transport and port) for "Flow-Descriptor".

5. Receiver cannot make a reservation request until it receives PATH message (to show the way "upstream" to receiver).

6. **PATH and RESV messages** are passed through non-RSVP routers transparently, although these routers are weak links in the chain of resource reservations.
**Some RSVP Definitions**

- **Session**: set of packets addressed to a particular destination and transport protocol
- **Flow descriptor**: Flowspec + Filter spec
  - Flowspec: Reservation request (RSpec & TSpec)
  - Filter spec: Sender address and TCP/UDP port number
- **RSVP flow**: Session + Filter spec
- **Reservation style**: distinct/shared; explicit/implicit

<table>
<thead>
<tr>
<th>Sender selection</th>
<th>Distinct</th>
<th>Shared</th>
</tr>
</thead>
<tbody>
<tr>
<td>Explicit</td>
<td>Fixed-Filter (FF) style</td>
<td>Shared-Explicit (SE) style</td>
</tr>
<tr>
<td>Implicit</td>
<td>none defined</td>
<td>Wildcard-Filter (WF) style</td>
</tr>
</tbody>
</table>
Reservation Styles

- **Wildcard-filter style.** When a receiver uses the wildcard-filter style in its reservation message, it is telling the network that it wants to receive all flows from all upstream senders in the session and that its bandwidth reservation is to be shared among the senders.

- **Fixed-filter style.** When a receiver uses the fixed-filter style in its reservation message, it specifies a list of senders from which it wants to receive a data flow along with a bandwidth reservation for each of these senders. These reservations are distinct, that is, they are not to be shared.

- **Shared-explicit style.** When a receiver uses the shared-explicit style in its reservation message, it specifies a list of senders from which it wants to receive a data flow along with a single bandwidth reservation. This reservation is to be shared among all the senders in the list. Shared reservations, created by the wildcard filter...
Example
Fixed Filter

- (distinct reservation and explicit sender selection) The Filterspec of each FF reservation installed at an interface consists of a single sender only. The effective Flowspec of the reservation installed is the maximum of all FF reservation requests received on that interface for that particular sender. In cases where the interface connects to a shared medium LAN Resv messages from multiple next hops may be received.

- The Flowspec of the FF Resv message unicast to the previous hop of a particular sender is given by the maximum Flowspec of all reservations installed in the router for that particular sender.

<table>
<thead>
<tr>
<th>Send</th>
<th>Reserve</th>
</tr>
</thead>
<tbody>
<tr>
<td>A: S1(4b)</td>
<td>S1(4b), S2(5b)</td>
</tr>
<tr>
<td>B: S2(5b), S3(b)</td>
<td>S1(3b), S3(b)</td>
</tr>
</tbody>
</table>
**Wildcard Filter**

- (shared reservation and wildcard sender selection) The Filterspec of each WF reservation installed at an interface is wildcard and matches on any sender from upstream. The effective Flowspec installed is the maximum from all WF reservation requests received on that particular interface. The Flowspec of each WF Resv message unicast to a previous hop upstream is given by the maximum Flowspec of all WF reservations installed in the router. More strictly speaking, only WF reservations whose "Scope" applies to the interface out of which the Resv message is sent are considered for this second merging process.
**Shared Reservation Explicit Sender**

- (shared reservation and explicit sender selection) The Filterspec of each SE reservation installed at an interface contains a specific set of senders from upstream and is obtained by taking the union of the individual Filterspecs from each SE reservation request received on that interface. The effective Flowspec installed is the maximum from all SE reservation requests received on that particular interface. The Filterspec of a SE Resv message unicast out of an interface to a previous hop upstream is the union of all senders whose previous hop is via that interface and who are contained in the Filterspec of at least one SE reservation in the router. Likewise the Flowspec of this SE Resv message is given by the maximum Flowspec of all SE reservations whose Filterspecs contain at least one sender whose previous hop is via that interface.

```
<table>
<thead>
<tr>
<th>Send</th>
<th>Reserve</th>
</tr>
</thead>
<tbody>
<tr>
<td>A: S1(3b)</td>
<td></td>
</tr>
<tr>
<td>B: (S2, S3)(3b)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>(S1, S2)(b)</td>
</tr>
<tr>
<td></td>
<td>(S1, S3)(3b)</td>
</tr>
<tr>
<td></td>
<td>(S2)(2b)</td>
</tr>
<tr>
<td></td>
<td>(S1, S2)(b)</td>
</tr>
<tr>
<td></td>
<td>(S1, S3)(3b)</td>
</tr>
</tbody>
</table>
```
RSVP Messages

- PATH: sets up state along path followed by packets
- RESV: Request for reservation back along setup path
- PATH_TEAR: Explicit removal of state along path
- RESV_TEAR: Explicit removal of reservation
- RESV_ERR: Reservation failure & errors
- PATH_ERR: Path error
- RESV_CONFIRM: Reservation confirmation*

* Not an end-to-end guarantee
**Basic RSVP Operation**

- Setup path in the network through PATH messages from sender(s)
- Reserve resources on path through RESV messages from receiver(s)
RSVP Soft State

- Soft state means that any RSVP related state (PATH and RESV states) is temporary (timer)
  - Needs to be *refreshed* in order to stay
  - Will (eventually) disappear if not refreshed

- Refresh and state clean-up
  - Send new message every $t$ sec (30sec)
  - If fail to receive any refresh in $n*t$ sec, delete state ($n=3$)

- Simplification of error recovery as you know you won’t stay forever in a bad state
  Don’t need to be “too” picky in taking care of all error scenarios

- Trade-off between additional protocol overhead and simpler processing rules
  - RSVP messages are sent *unreliably*
RSVP PATH Message

- From sender to receiver(s) to establish path state in network elements (routers) between sender and receiver(s)
- Addressed directly to destination
  - Router alert option ensures interception at each RSVP hop
- Includes
  - Previous hop (PHOP): The previous RSVP aware entity on the path
  - Sender template: Filter spec for the sender
  - Sender TSpec: Traffic characteristics of sender
  - Sender ADSPEC: Used to capture path characteristics
- PATH message is processed and updated at each RSVP aware hop on the path
  - Create or refresh path state
  - Update ADSPEC
Format of Sender TSpec

<table>
<thead>
<tr>
<th>31</th>
<th>2423</th>
<th>16</th>
<th>15</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>reserved</td>
<td>7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>reserved</td>
<td>6</td>
<td></td>
</tr>
<tr>
<td>127</td>
<td>0</td>
<td>5</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- Token bucket rate \([r]\) (32-bit IEEE floating point number)
- Token bucket size \([b]\) (32-bit IEEE floating point number)
- Peak data rate \([p]\) (32-bit IEEE floating point number)
- Minimum policed unit \([m]\) (32-bit integer)
- Maximum packet size \([M]\) (32-bit integer)

0:  Message format version number
7:  Overall length (7 words without header)
1:  Service header, service number 1 (default/global information)
6:  Length of service 1 data (6 words without header)
127: Parameter ID, parameter 127 (Token_Bucket_TSpec)
0:  Parameter 127 flags (none)
5:  Parameter 127 length (5 words without header)
**TSpec Conformance Rules**

- Limitation on when and how much data to send
  \[ \forall t \ A(t) \leq M + \min[pt, rt+b-M] \]

- Limitation on minimum packet size
  - Per packet processing time, e.g., lookup, must be smaller than packet transmission time
    \[ \tau \leq \frac{m}{c}, \quad c = \text{input link speed} \]
    - Packet smaller than \( m \) are counted as being of size \( m \)
      Lower effective rate if smaller packets are used
      \[ m' \leq m \Rightarrow r' = r \frac{m'}{m} \]

- Limitation on maximum packet size (\( M \))
  - Bound the maximum transmission time of packets
  - Packets larger than \( M \) are deemed non-conformant
Content of ADSPEC

- Two components
  - General parameters
  - Service specific parameters (absent ineligible service) possible means for coordinating service selection between receivers

- General parameters
  - Global break bit
  - Hop count for Integrated Services network elements
  - Path bandwidth estimate
  - Minimum path latency
  - Path MTU

- Service specific parameters
  - Service break bit
  - Overrides of general parameters (MTU, bandwidth estimate)
  - Additional service specific quantities
Processing of PATH Messages

- Generate PATH message (sender template, Tspec, ADSPEC)
- Query routing to identify NHOP(s) (forwarding)
- PATH state processing
  - Create PATH state (TSpec, PHOP, NHOP, etc.) if not present
  - Update ADSPEC (hop count, MTU, latency, bandwidth, etc.)
  - Refresh state if PATH state present
  - Send immediate refresh if state changed
Why Do We Need a PHOP?

- Multiple reasons
  - Asymmetric routes
  - Non-IS hops/clouds (tunneling)
One Pass With Advertising (OPWA)

- **Benefits**
  - Discover characteristics of path before making reservation, i.e., ADSPEC
    - Improves odds of successful reservation (bandwidth estimates)
    - Hints to receiver for improved configuration (playback buffer based on latency estimate)
    - Can compensate for path characteristics during reservation phase (Guaranteed Service queueing delay adjustment)

- **Disadvantage**
  - Path selection (routing) done without knowing exact service requirements
RSVP RESV Message

- From receiver to sender(s) to reserve resources
- Sent hop-by-hop using PHOP information
- Includes
  - NHOP, i.e., where it came from (interface address)
  - Reservation style (FF, SE, WF) and scope object if needed
    » Scope object: List of senders to which implicit reservation applies
  - Reservations style and flow descriptor list
    » Sender(s) to which reservation applies (filter spec’s)
    » Flowspec (service specific parameters)
      • RSpec, i.e., QoS specific requirements
      • TSpec, i.e., sender traffic to which reservation applies
  - RESV_CONFIRM (optional)
- RESV message processing at each hop
  - Create or modify RESV state
  - Merging of RESV message (avoids RESV implosion)
  - Forward upstream (PHOPs) after successful reservation
Processing of RESV Messages

- Generate RESV message (Flowspec, Filter spec)
- Query admission control (new/modified reservation)
- RESV state processing
  - Create RESV state (Flowspec, filter spec, etc.) if not present
  - Coordinate merging of Flowspec’s (from multiple RESV)
  - Refresh state if RESV state present
    » Send immediate refresh if state changed
  - Send RESV_CONFIRM if necessary
    » End point or larger reservation already in place
  - Determine where to propagate RESV messages
    » Filter spec (and scope object if necessary)
    » list of PHOPs
  - Blockade state (avoidance of “killer” reservation)
**Killer Reservation Problem**

- Large unsuccessful reservation prevents smaller one(s)
- Blockade state installed to block offending reservation

Blockade state initiated by propagating RESV_ERR
RESV_ERR must propagate all the way

Timer controls duration of blockade state
RSVP as an MTU Discovery and Traffic Negotiation Mechanism

- **MTU discovery**
  - PATH: Propagate min(M,MTU)
  - RESV: Propagate M=min(Mi)
- **Traffic negotiation**
  - Receiver specifies desired TSpec, e.g., smaller b and larger r
- **Some dangers**
  - Lowest common denominator rules...
Merging of Flowspec and Sender TSpec

- Two types of merging operations
  - Least Upper Bound (LUB): As “good” as each Flowspec for any individual parameter (merging of reservations)
    - Flowspec TSpec1: $r_1=1000$, $b_1=3500$, $p_1=10,000$, $m_1=64$, $M_1=1500$
    - Flowspec TSpec2: $r_2=800$, $b_2=5000$, $p_2=10,000$, $m_1=32$, $M_2=1000$
    - Merged TSpec : $r=1000$, $b=5000$, $p_1=10,000$, $m=32$, $M=1000$
  - Greatest Lower Bound (GLB): As big a reservation as needed given aggregate traffic (comparing reservation to sum of sender traffic)
    - Sender_TSpec: $r_1=1000$, $b_1=3500$, $p_1=10,000$, $m_1=64$, $M_1=1500$
    - Flowspec_TSpec: $r_2=800$, $b_2=5000$, $p_2=10,000$, $m_1=32$, $M_2=1000$
    - Merged_TSpec : $r=800$, $b=3500$, $p_1=10,000$, $m=64$, $M=1000$
Intserv and RSVP - Pros/Cons

- **RSVP**
  - Reservations are “soft”, i.e., they need to be refreshed
  - Network control protocol, does not carry data
  - Reservations are receiver-based for multicast receiver groups
  - Can traverse non-RSVP routers

- RSVP is intentionally decoupled from both resource reservation (e.g., call admission) and routing, i.e., RSVP ensures delivery of the traffic specification and reservation objects without interpreting them

- Intserv provides the highest level of IP QoS but at the expense of complexity and overhead
  - Call processing (processing of PATH and RESV messages)
  - Packet classification
  - Packet scheduling (e.g., WFQ)

- Who is going to pay for the reservations? Who qualifies?

- RSVP/Intserv is not a good fit for core networks which would carry thousands of sessions; marginal deployment today
RSVP - New Developments

- RSVP tunnels
  - Internet draft (1999)
  - Works with other IP tunneling mechanisms, e.g., IP in IP or IPSec
  - Resource allocation on an aggregate basis where tunnels may be established between gateways as opposed to the original design of allocation between hosts on a per-microflow basis
  - Requires tunnel/RSVP management at the gateways
  - Scalable in the core due to reduced number of sessions

- RSVP/Diffserv interworking
  - RSVP at the edge, diffserv in the core for scalability
  - Microsoft supports this model