RSVP Working Group

Seattle IETF Meeting

March 29, 1994

RSVP OVERVIEW

Bob Braden

USC Information Sciences Institute

OUTLINE

- Introduction
- Service Model
- Protocol Mechanisms
- Unsettled Issues

INTRODUCTION

RSVP: an Internet setup protocol for *resource reservation,* to achieve specific end-to-end *quality of service* (QoS) for logical packet streams, or *flows.*

Acknowledgments

- MANY PEOPLE HAVE CONTRIBUTED TO RSVP DESIGN . . .

  Lixia Zhang, Scott Shenker, Deborah Estrin,

  Dave Clark, John Wroclawski, Steve Deering,

  Sugih Jamin, Shai Herzog, Bob Braden,

  Craig Partridge, Daniel Zappala, and Steve Berson

RSVP Assumptions

- IP MULTICASTING IS FUNDAMENTAL

  -- Generalizing 1->1 to 1->n is not effective design paradigm. Instead, make unicast a special case.

- SUPPORT HETEROGENEOUS SENDERS & RECEIVERS

  -- Important to handle diversity.

- SCALE TO VERY LARGE # OF RECEIVERS

RSVP Assumptions (cont'd)

- DECOUPLED FROM ROUTING PROTOCOLS

  -- Can use existing routing protocols.

  -- Can use future routing protocols -- e.g, CBT or ESL or ...

- SUPPORT '*CHANNEL SWITCHING'*

  -- idea from Jon Crowcroft: each receiver can independently select, *within the network*, which subflow it wants to receive.

- TRUTH, JUSTICE, and the INTERNET WAY:

  -- robustness, modularity, extensibility, ...

Two Fundamental Ideas

- **RECEIVERS CONTROL RESERVATIONS**
  - -- Simplicity and directness: receivers know what they want.
  - -- Matches IP multicasting, which is receiver-oriented.
  - -- Scales well to large number of heterogeneous receivers.
- **SOFT STATE IN ROUTERS**
  - -- End systems must periodically refresh router state
  - -- Route change => refresh msg sets up state on new route

RSVP SERVICE MODEL: OUTLINE

- **Reservation Request => Flow Descriptor**
- **Session -- Unit Of Reservation**
- **Sender Selection**
- **Reservation Model**
- **Reservation Styles**

Elementary Reservation Request

Flow Descriptor = (Q, F) pair.

- **Q = FLOWSPEC**

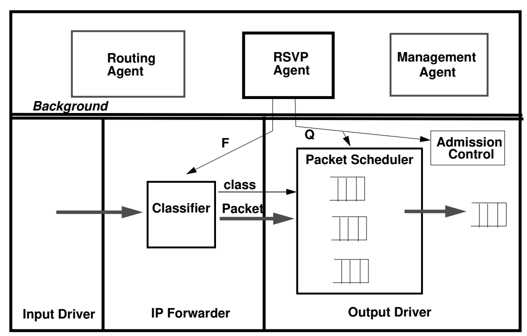
  -- Defines desired QOS (bandwidth, delay, burstiness...)

  => resources

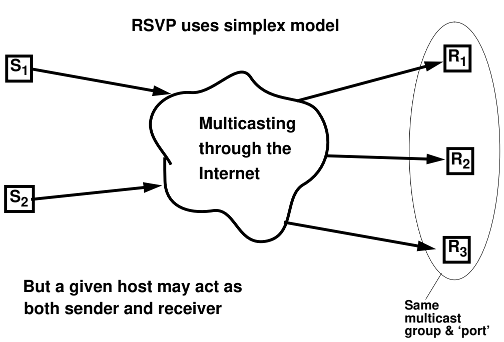
- **F = FILTERSPEC**

  --Controls Classifier, to select subset of data packets to receive this QOS.

Router Model



RSVP Session



RSVP Session

- **Sessions are independent**
- **Session = (Dest. Address, Reservation Id)**

  Dest. address: unicast (pt-pt) or multicast

  Reservation Id = 32 bit number

  - Generalized receiver 'port'

  - Supplied by application, a 'cookie' to RSVP

Sender Selection

- **Receiver can set Filterspec to select sender(s)**

  Filter may select on:

  {Src-IP-addr/Src-'port', Dst-IP-addr/Dst-'port'}

  or:

  {Dst-IP-addr/Dst-'port' } (Sender is '*wildcard*')

Reservation Model

- **Receiver initiates request: FD = (Q, F)**
- **FD Passed on reverse routes upstream to sender(s) selected by F.**
- **Each node: local Admission Control decision on Q => (YES)/NO**

  (*One-pass* reservation algorithm)

Reservation Style

At a given node, and for a given session:

- **Share reservation among all downstream receivers**
- **May/may not share among upstream senders**

=> *Reservation Style:*

-- Currently 3 types, more may be added in future

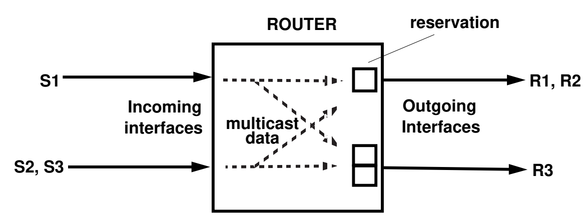
o WILDCARD FILTER -- WF

o FIXED FILTER -- FF

o DYNAMIC FILTER -- DF

Reservations and Interfaces

(Assume a given session)



In reality: same physical interface may be both incoming and outgoing interface for RSVP.

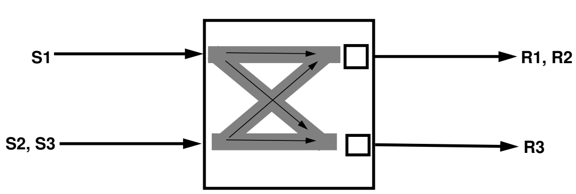
Reservation Styles

- **Wildcard Filter (WF) Style**

  Filter: { Dst-IP-addr / Dst-port }

  Create single reservation 'pipe', shared among senders;

  e.g., audio.

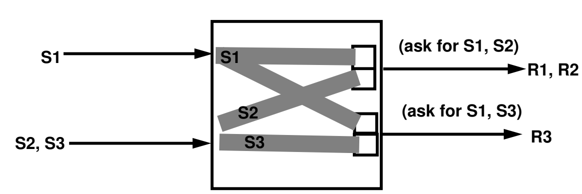


Reservation Styles

- **Filter (FF) Style**

  -- Each receiver explicitly lists senders it wants to see (=> *filterspecs*)

  -- Different sender's reservation channels are *not shared*-- e.*g., video*.



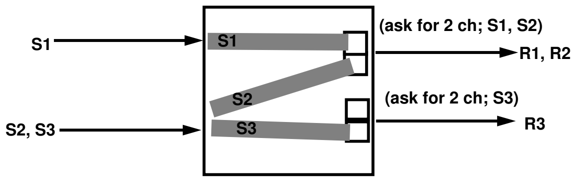
Reservation Styles

- **Dynamic Filter (DF) Style**

  Channel-switching style

  Reserve n non-shared ‘channels’ of capacity *(e.g., for video)*

  Choose/change selection of particular senders to use these.



(Never reserve more channels than number of senders upstream)

Reservation Styles -- Summary

A receiver may issue:

- **WF -- Wildcard Filter reservation**

  => (Q, F(*))

- **FF -- Fixed Filter reservation**

  => $(Q_1, F(S_1))$, $(Q_2, F(S_2))$, . . .

- **DF -- Dynamic Filter reservation**

  => n, Q, { $F(S_1)$, $F(S_2)$, ... $F(S_k)$ },

  $0 \le K \le N$

RSVP PROTOCOL -- Outline

- **RSVP Messages**
- **Merging**
- **Soft State**

RSVP Messages

RSVP Messages are:

- **Sent as datagrams directly over IP.**
- **Periodically resent**

  -- to refresh state along the path, and

  -- to replace lost messages.

- **Given some preferred class of service.**

RSVP Message Types

- **RESV -- *Reservation***

  -- Follow reverse data paths to specified senders

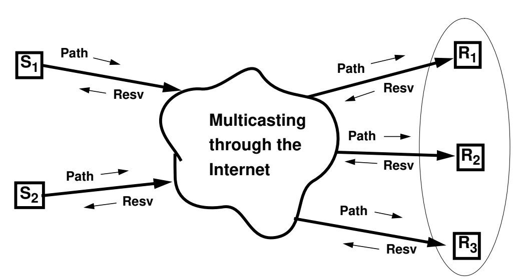
  -- Leave *reservation state* in each router

- **PATH**

  -- Follow forward data paths

  -- Leave *path state* in each router, to route RESV messages.

RSVP Messages

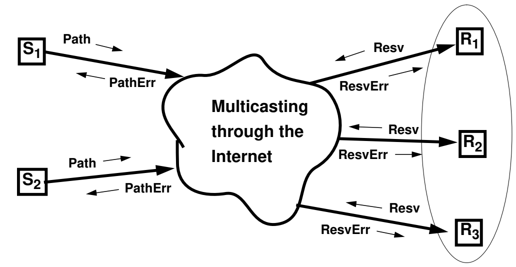


RSVP Message Types

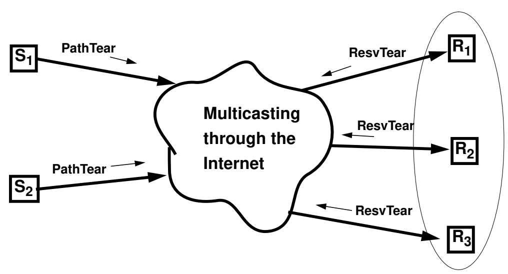
- **PathErr, ResvErr -- *error messages***
- **PathTear, ResvTear -- *teardown messages***

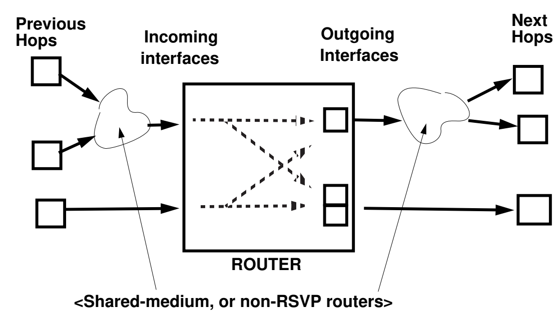
  -- Optimization: remove state immediately.

Error Messages



Teardown Messages





Merging

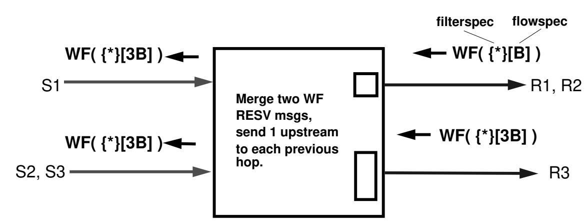
To limit RSVP traffic:

- ***Merge* PATH, RESV messages at each router.**

  -- Same session, reservation style
- **Objective: one RSVP message in each direction**

  per session per refresh period.

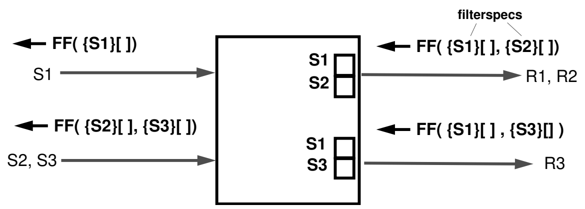
Merging Example: WF Resv



E.g. audiocast: only one WF message arrives at each sender!

Merging Example: FF Reservation

Example: Fixed Filter Style:



Managing Soft State

- **Merging requires gathering message state at each node**

  => Forward (merged) message at end of current *refresh period*.

- **State that is not refreshed times out after TTL interval.**
- **TTL interval > 3 * refresh period, to allow loss of individual RSVP messages.**

Path Message Format

```
+-------------+-------------+-------------+-------------+
| Vers |  1   |    Flags    |       RSVP Checksum       |
+-------------+-------------+-------------+-------------+
|                       DestAddress                     |
+-------------+-------------+-------------+-------------+
|                         DestSID                       |
+-------------+-------------+-------------+-------------+
|                      Refresh Period                   |
+-------------+-------------+-------------+-------------+
|                      State TTL Time                   |
+-------------+-------------+-------------+-------------+
|                   Previous Hop Address                |
+-------------+-------------+-------------+-------------+
|      ///////////////      |          SD Count         |
+-------------+-------------+-------------+-------------+
|                  Sender Descriptor List               |
+-------------+-------------+-------------+-- ...
```


PATH Messages

Sender Descriptor: (Template, FlowSpec)

-- Template: format of data packets (bit-mask, bit-string)

To block propagation of RESV with erroneous filter upstream.

-- Flowspec: Characterize ‘maximum’ traffic than can be sent.

To prevent over-reservation. Assumes that can compute max(flwsp1, flwsp2).

RESV Messages

- o **Session ID**
- o **Refresh period, TTL interval**
- o **Receiver address**

  To establish ownership of receiver-specific reservation state (DF), and to prevent looping RESV messages.

- o **Reservation Style**
- o **Number of DF reservations**
- o **List of Flow Descriptors: (FilterSpec, FlowSpec)**

Resv Message Format

```
+-------------+-------------+-------------+-------------+
| Vers |  2   |    Flags    |       RSVP Checksum       |
+-------------+-------------+-------------+-------------+
|                       DestAddress                     |
+-------------+-------------+-------------+-------------+
|                         DestSID                       |
+-------------+-------------+-------------+-------------+
|                      Refresh Period                   |
+-------------+-------------+-------------+-------------+
|                      State TTL Time                   |
+-------------+-------------+-------------+-------------+
|                     Next Hop Address                  |
+-------------+-------------+-------------+-------------+
|                       RecvAddress                     |
+-------------+-------------+-------------+-------------+
| Dynamic Reservation Count |          FD Count         |
+-------------+-------------+-------------+-------------+
|                   Flow Descriptor List                |
+-------------+-------------+-------------+-- ...
```

Host Model (API)

Host must:

- Register as a sender (if appropriate)

  => RSVP starts sending PATH msgs

- Join multicast group (if a receiver)
- Request reservation (if a receiver)

  => RSVP starts sending RESV msgs

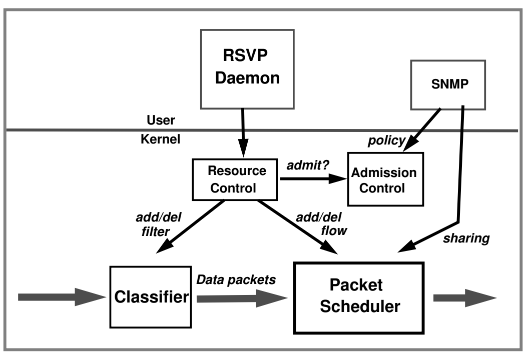
Host may:

- Change reservation request at any time

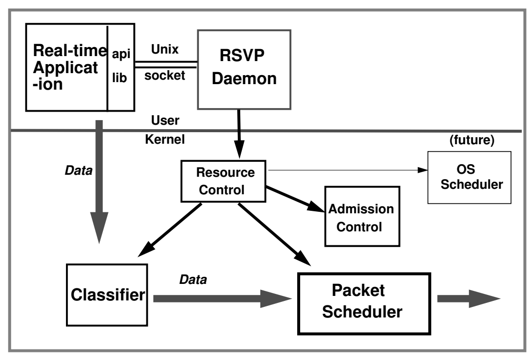
Interfaces

- API
- Traffic Control kernel functions
  - o Add/delete Flowspecs (=> admission control)
  - o Add/delete Filterspecs
  - o Abstract operations on Flowspecs
- Routing
  - o 'Tell me the route to...'
- Forwarding

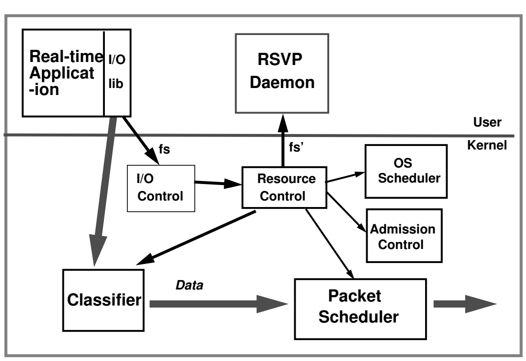
Router Implementation



Host Implementation



Host: Integrated I/O Model



Open Issues

- Route Pinning

  Handle route-flapping

- Flowspec Comparison
- Filterspec Encoding-
- Path messages: good/evil ?
- Authenticated User Identification field

More Open Issues

- **'Global' Updates**

  In steady-state, suppress per-session refresh traffic.

- **Core Aggregation**

  Reduce RSVP state in core networks

- **Flow Id**

  Make classifier more efficient

- **Hardened Channels**

RSVP R&D History

- **Lixia Zhang (PARC): simulated protocol**
- **Sugih Jamin (USC/PARC): First implementation**

  -- RSVP within modified SunOS 4.1.1 kernel [DARTnet]

  -- Kernel included CSZ packet scheduler

  -- Demonstrated in DARTnet June 8, 1993.
- **Shai Herzog (ISI): new RSVP implementation**

  -- Daemon process

  -- Demonstrated for ARPA Oct 26, 1993

  -- Tested with CSZ kernel December 1993

  -- Further development by Steve Berson (ISI)