RSVP and Integrated Services

ETSI VoIP Workshop
Sophia-Antipolis, France
June 8, 1999

Bob Braden

University of Southern California
Information Sciences Institute
Marina del Rey, California

Outline

- o Introduction
- o Integrated Services
- o RSVP
- o Current Status and Work in Progress
- o Int-Serv and Diff-Serv

Introduction: Why Integrated Services?

In 1991 . . .

- **The Internet was just beginning to be commercialized**
- **The World Wide Web had not happened yet.**
- **The MBONE had not happened yet.**
- **BUT: Workstation vendors were planning multimedia capability.**

The prospect of major multimedia (M/M) traffic seemed to pose a future problem for the Internet.

M/M in the Internet

Internet characteristics:

- **Only Best Effort (BE) service**
  - **There were priority bits in the IP header, but no sensible way to determine who could set them.**
- **Routers used FIFO scheduling**
- **TCP backed-off from congestion**
  - **This prevented congestion collapse of the Internet.**
- **IP multicast was available but not yet deployed**

M/M in the Internet

M/M characteristics:

- **o Sensitive to delay/jitter ("real-time")**
- **o Need fairly (or very) low packet loss**
- **o Continuous streams of UDP packets -- no congestion control**
- **o Multicast expected to dominate.**

Expected two kinds of M/M usage:

- **(1) Many video teleconferences**
- **(2) Broadcast video & audio**
- **(but NOT: Voice over IP!)**

M/M in the Internet

Requirements:

(1) Protect real-time M/M traffic from effects of network congestion.

(2) Protect TCP from real-time M/M streams, which do not adapt to congestion.

Possible solution: make M/M applications adaptive

- **Did not know how to do this for multicast**

  (currently an active research area!)

- **Cannot satisfy all end users if voice/picture quality varies.**

=> Need to add QoS to Internet

Internet QoS: Possible Approaches

- **ST-2**
  - Product of 1970s ARPAnet research on packetized speech
  - Virtual-circuits with QoS
    - !> 'hard state' violated Internet religion
    - !> Incompatible with IP
  - Sender-oriented multicast trees
    - Scaling problem for large multicast groups
- **New approach => Integrated Services**

Internet Integrated Services

Compatible extension of Internet architecture

- o BE service must always be available
  - -- Reservations not suitable for short TCP connections
  - -- Some M/M applications will be adaptive
  - -- Over-provisioning will sometimes be good enough
- o Multicast delivery is fundamental
- o Traffic = continuous packet streams

So state setup overhead is OK

Integrated Services needed:

o New (end-to-end) IP service models

o Admission control& traffic control in routers &hosts

-- Fine grained micro-flows, for isolation

E.g., make reservation for a subflow defined by:
{ Destination(Addr , Port), Sender(Addr , Port), Protocol Id }

-- If amount of state becomes a problem in core of Internet, can aggregate reservations.

o Signaling protocol -- RSVP.

Two Service Models

1. **Guaranteed service (RFC 2212)**

   -- Receiver can compute maximum queueing delay

   -- 'Hard' guarantee: high assurance, high cost

   -- For non-adaptive applications

   -- Can be thought of as a Weighted Fair Queueing (WFQ) service.

Int-Serv Service Models (2)

2. Controlled Load service (RFC 2211)

-- Simulates '*unloaded* network'

-- 'Soft' guarantee: lower assurance & lower cost

-- For adaptive applications

-- Can be thought of as simple priority service with admission control.

(In both cases, service specs describe service without constraining implementations)

Int-Serv Parameter Sets

1. TSpec: sender->receiver(s) in Path message

describes data packet stream:

Tspec(r, b, p, m, M)

r: Token bucket rate

b: Token bucket depth

p: Peak transmission rate (or infinity)

m: Minimum packet size

M: Maximum packet size

Int-Serv Parameter Sets

2. Flowspec: sender <- receiver in Resv message

- **Requests QoS: Tspec + Rspec**
  - **Rspec = (R, S) for Guaranteed Service**
    - **R = Requested bandwidth**
    - **S = Slack term**
  - **Rspec = ( ) for Controlled Load Service**

3. Adspec: sender -> receiver in Path message

- **Gathers path characteristics for Guaranteed Service**

RSVP

RSVP: designed to be the 'signaling' protocol to set up Integrated Services reservation state in hosts and routers.

Note: People sometimes use "RSVP" as a metaphor for "Internet integrated services"

RSVP History

o 1990: Receiver-oriented soft-state design invented in meeting of End-to-End research group.

o 1991-1993: Early research phase

Lixia Zhang (PARC/UCLA),
Deborah Estrin (USC/ISI),
Scott Shenker (Xerox PARC),
Sugih Jamin (USC/PARC), Daniel Zappala (USC).

o 1993: DARPA project at ISI to turn research idea into real protocol

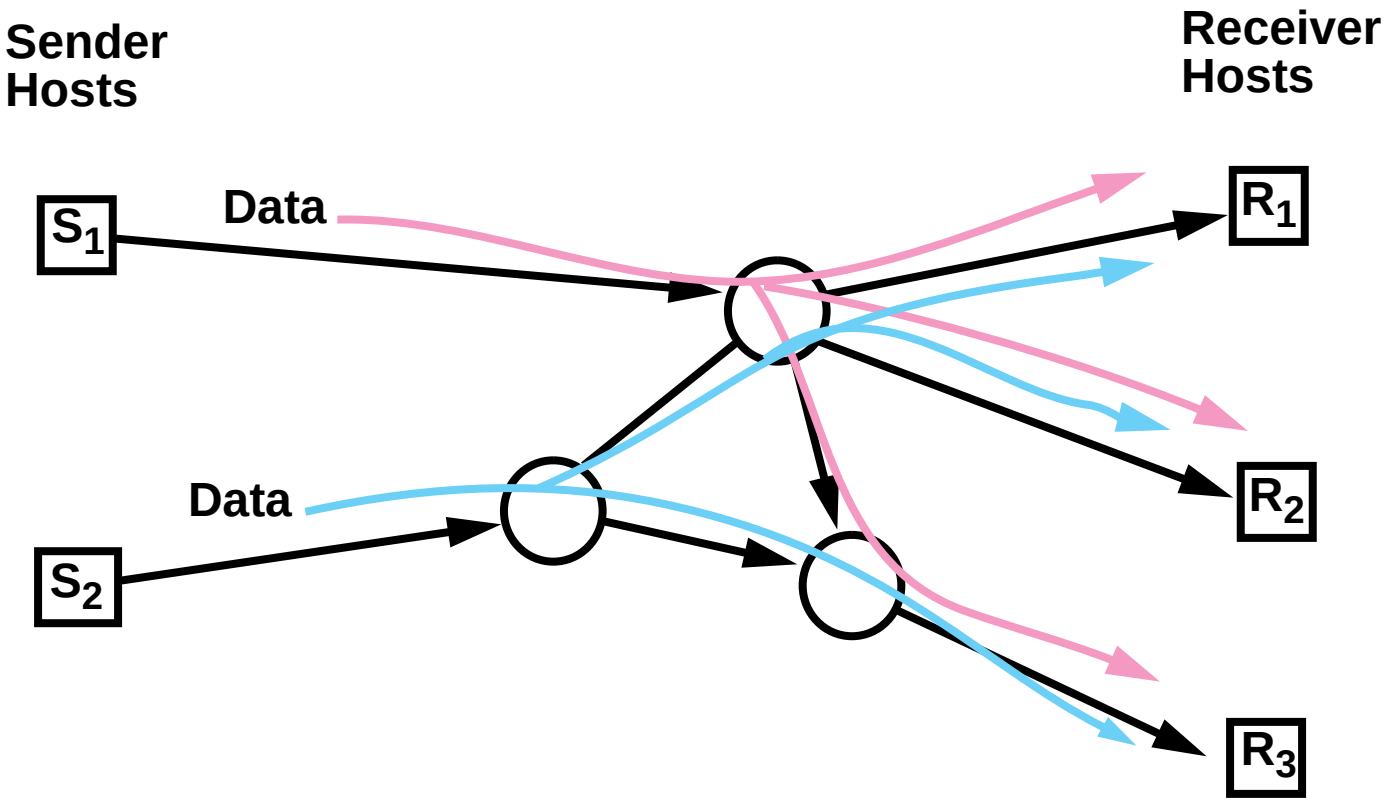
o Nov 1993: First RSVP BOF at IETF (Houston IETF); WG formed

o Sept 1997: Version 1 of RSVP becomes Proposed Standard

RSVP Requirements

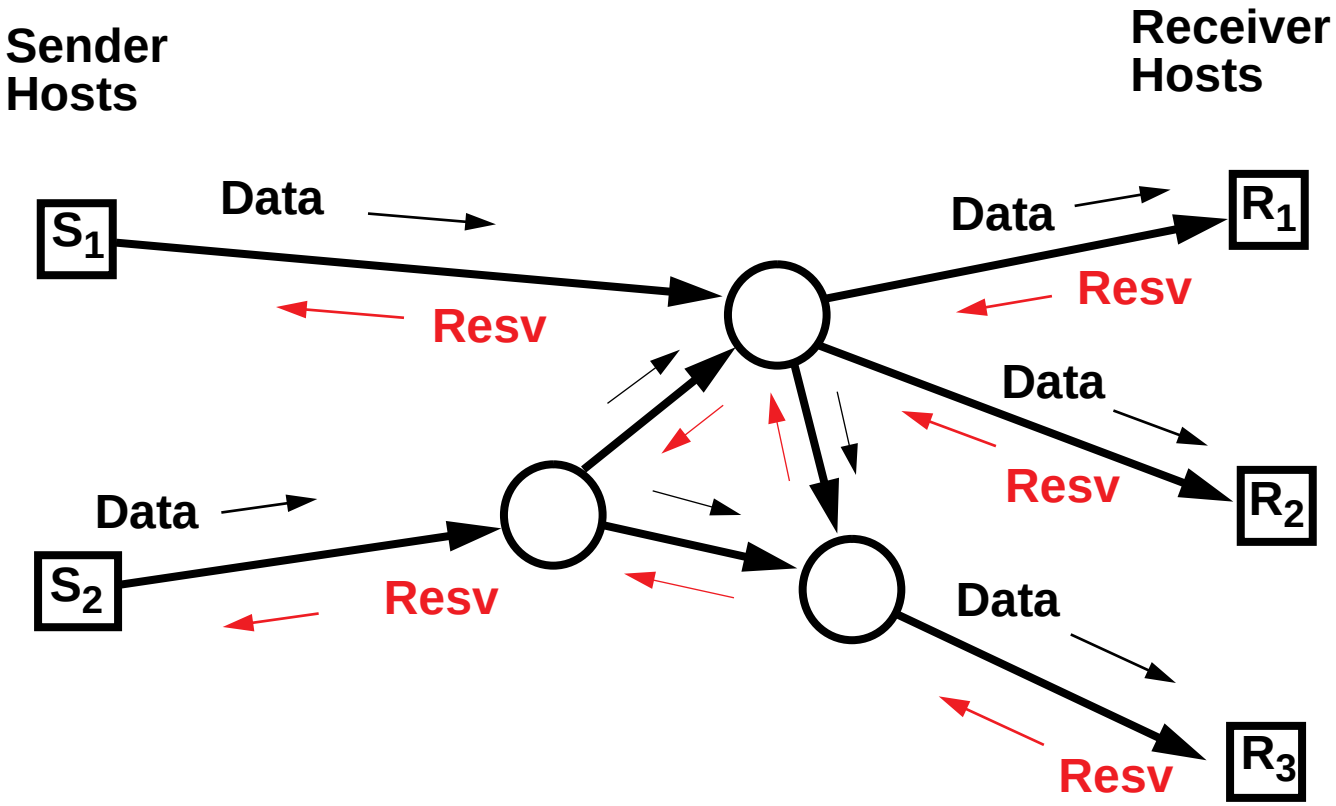
1. End-to-end protocol: requests come from apps
2. Signaling for fine-grained flows
3. IP multicast support designed in at the start.
4. Receiver-oriented setup, to support large multicast groups.

Supporting Multicasting



Multipoint-to-multipoint flows to a multicast group

Receiver Orientation



Reservation messages originate in receivers

Primary RSVP Design Decisions

1. **Signaling distinct from routing**
   (modularity, deployability)
2. **Soft state (robustness, simplicity)**
3. **Transparent operation across non-RSVP routers**
   (deployability)
4. **Support shared and distinct reservations**
5. **Support heterogeneous reservations**

1. Signaling Distinct From Routing

A. Plausible: signal along PATH, but route in DOMAIN

B. Otherwise, scaling and stability problems

C. Chose an incremental approach:

Phase 0: Standard routing with RSVP
(but reservation may fail unnecessarily)

Phases 1, 2... Augment routing architecture to better support reservations

Augment Routing for Reservations

[Hard problem! We still don't really have scalable approach that always finds a good path]

Scheme A:

- **o QoS Routing -- dynamic metric**
  - ***Find a route with enough resources for request.***
    - **But this approach is not scalable outside AD**

Scheme B:

- **o QoR (Quality of Route) Routing -- static metric**
  - ***Find a route that is more* likely *to have enough resources***

- **o Dynamic alternate path routing --**
  - **actively recovery from admission control failure**

In either case, may also need Route pinning

2. Soft State

o Times out unless periodically refreshed.

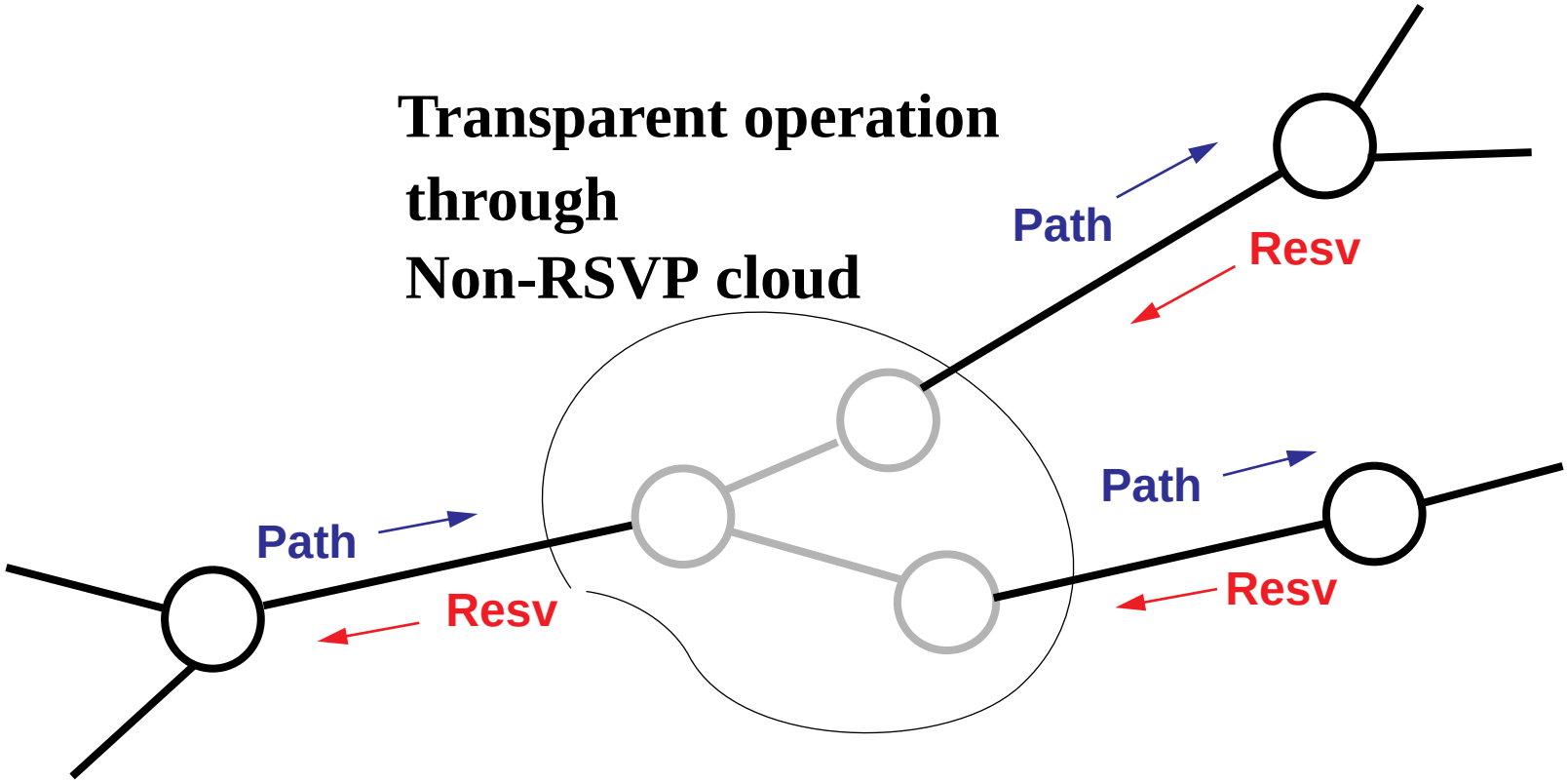
o State changes propagate immediately, but only as far as needed.

o Original and refresh messages are identical
  -- Can automatically adapt to route changes.
  -- Robust, because self-correcting

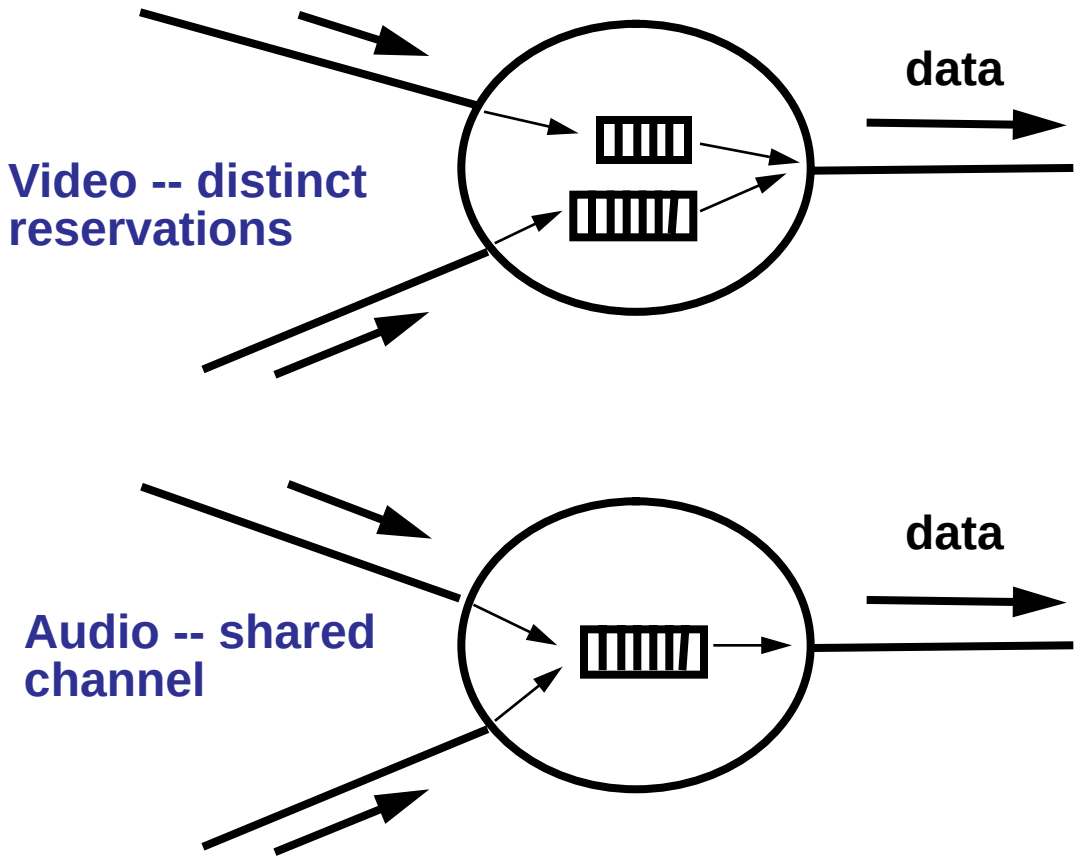
o Simplicity: incremental setup --
  -- To modify state: just send new message

As close to the connectionless religion as possible...

3. Non-RSVP Routers

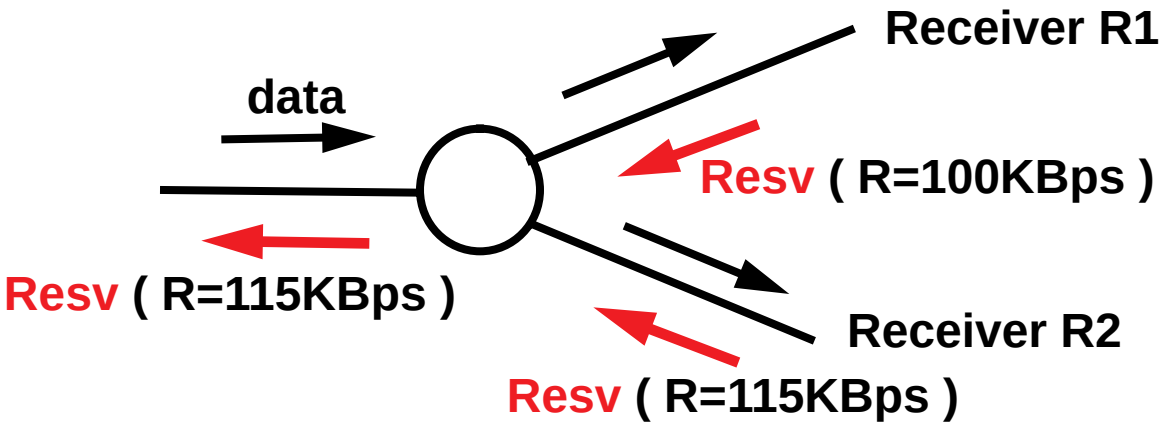


4. Shared and Distinct Reservations



5. Heterogeneous reservations

=> *Merging* Reservations



Merging & Heterogeneity

Why heterogeneous reservation requests?

(a) Using Guaranteed service: Downstream receivers want same bound on queuing delay, but routers have different properties => different R requests.

Guaranteed Service model => NO PACKETS ARE DROPPED.

(b) Some downstream paths really don't have enough bandwidth for the application stream.

This is a bug in the way the application is using int-serv.

For example, may need hierarchical encoding using multiple multicast groups.

Merging & Heterogeneity

But there is a bad result of merging:

Possible denial-of-service
<<<Killer Reservations>>>

Example: R2 asks for R=9999KBps, so merged reservation fails upstream. Then R1 loses!

This is a fundamental problem with any multicast QoS that allows heterogeneity.

But if you don't allow heterogeneity, you significantly restrict the kinds of service models you can support.

RSVP adopted a partial solution to Killer Reservations; not simple!

RSVP Design Issues

1. RSVP message reliability
2. Limiting refresh overhead
3. Path messages
4. Reservation Styles
5. Loop prevention
6. Response to admission failures

1. RSVP Message Reliability

Refresh messages provide basic reliability mechanism for RSVP signaling.

- **Simple, but then RSVP packets themselves must have ~ lossless QoS.**
- **Experience: not adequate**

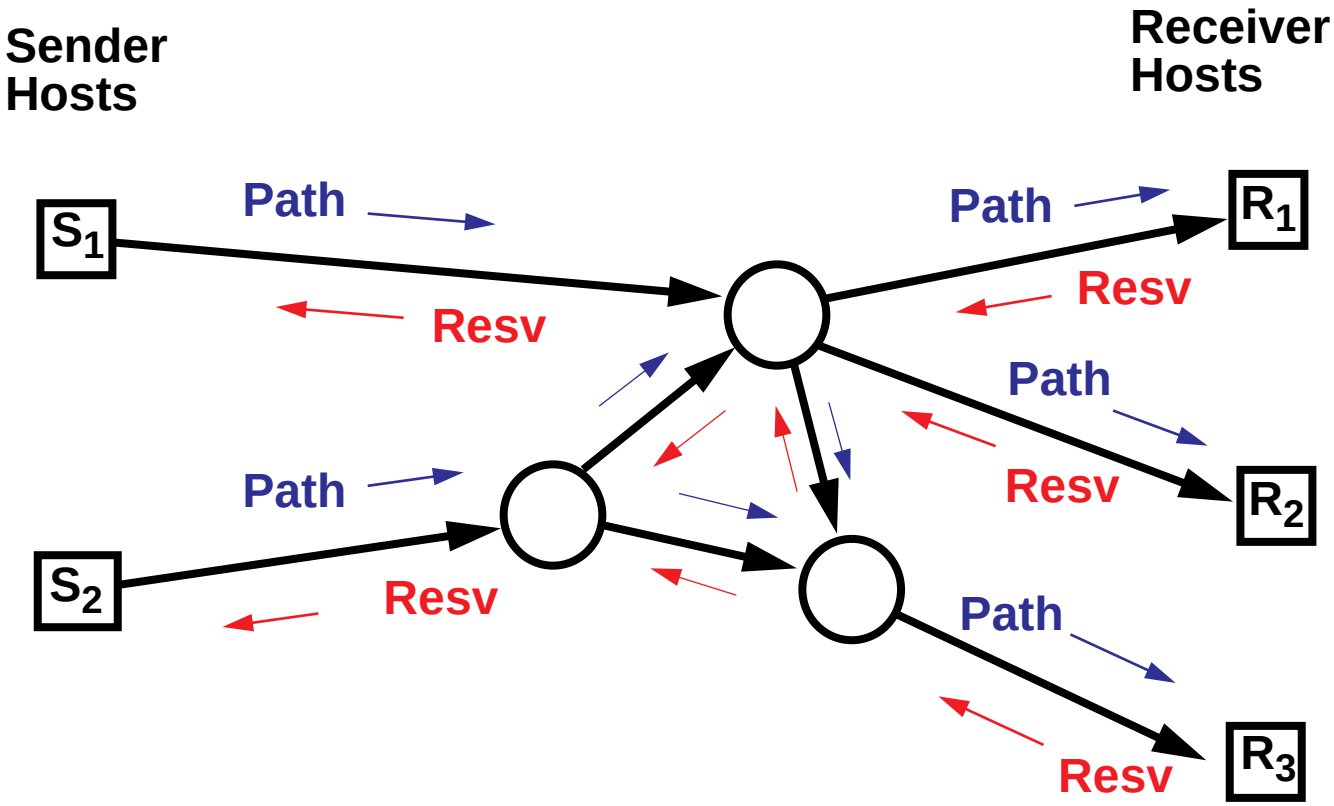
  Possible 30 sec delays in making or tearing down a reservation

  *** Not so easy to provision enhanced QoS class for signaling.**
  - **-- Non-Int-Serv clouds**
  - **-- Router CPU scheduling may not be adequate**
- **RSVP WG is now designing a reliable-delivery (ACK) mechanism for RSVP messages.**

2. Limit Refresh Overhead

- **Refresh bandwidth, CPU can be significant.**
  10,000 sessions => O(300Msg/s), O(300Kbps)
- **"Local repair" provides immediate adaptation to route changes.**
  - **-- Routing signals RSVP when a route changes;**
    RSVP then triggers immediate local refreshes.
  - **-- Eventual refresh message ensures correct final state.**
- **RSVP WG of IETF working on refresh reduction**
  - **-- To refresh unchanged state, nodes periodically exchange aggregated refresh messages.**

3. Path Messages



Path messages follow data downstream, steer Resv's upstream

Path Messages

- Introduced to route reservation messages upstream.

- Also carry information on data stream and path --
  Adspec: MTU, total propagation delay,
  Guaranteed-service path characteristics)

4. Reservation Styles

Parameters controlling multipoint reservations

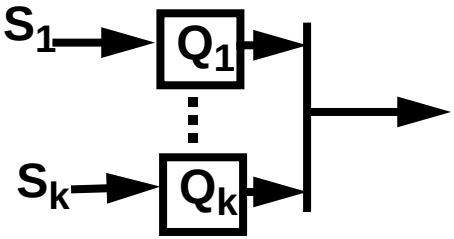
o Choose shared/distinct reservation

"Wildcard Filter": shared by all upstream senders

WF(Q, *)

"Fixed Filter": distinct to particular senders

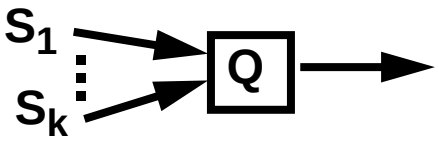
FF((Q1, S1), (Q2, S2), ... (Qk, Sk))



o Later added:

"Shared Explicit"

SE(Q, S1, ... Sk)



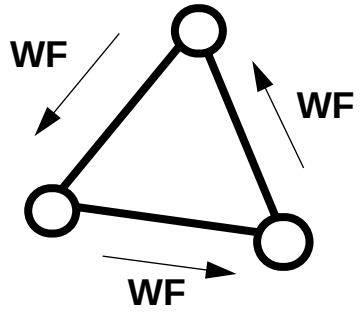
Other, more complex, styles have been suggested.

5. Loop Prevention

o **WF style: single shared reservation for all senders.**

o **Seems to scale well: *Resv* msg size and classifier state are independent of # of senders**

o **But it *doesn't work ...* can have subtle kind of loop: self-refreshing**



Had to add explicit sender list to WF reservation message, after all

6. Response to Admission Failures

There is a variety of possible policies

o Intertwined with anti-killer-reservation heuristic.

o In RSVP V1, policies are:

-- Reservation is left in place downstream from failure point, but

-- Request is not propagated upstream from failure point.

IETF Standards Status of Int-Serv

- **Service Definitions**

  Controlled Load -- RFC 2211, Sept 1997

  Guaranteed -- RFC 2212, Sept 1997

- **RSVP v1**

  RFC 2205, RFC 2207, RFC 2210, Sept 1997

- **RSVP API**

  Two versions: Winsock2, XOPEN.

- **RSVP MIB**

  RFC 2206

Two Other Important Areas

o Link-Layer mappings of Int-Serv: ISSLL WG

-- RSVP over ATM: RFC 2382
-- RSVP over IEEE802 nets (Ethernet, Token Ring)
-- RSVP over low-bandwidth serial links (ISDN)

o Policy Control: RAP WG

User authentication, accounting. ...)

-- Off-loading Policy Control from routers and hosts into centralized policy servers, using COPS protocol.

RSVP Standards Work in Progress

1. **RSVP Diagnostic Message**

   Travels towards sender, records state

2. **RSVP MD5 Integrity**

   Hop-by-hop integrity of RSVP messages

3. **RSVP Reservations in IP/IP Tunnels**

4. **Modifications to RSVP v1 soft state mechanism**

   -- Reliable message delivery

   -- Refresh reduction

RSVP as General Signaling Protocol

- RSVP is a soft-state setup mechanism that is designed to be easily extensible

  RSVP message = {header} {typed "object"}*

- RSVP has already been adapted:

  -- ISSLL Subnet Bandwidth Manager (SBM) [Layer 2]

- There is interest in extending RSVP for other purposes.
- Label-switched path setup for MPLS

  To avoid inventing another signaling protocol

- VPN setup

  Ditto

Acceptance and Deployment

o Many end-system and router vendors are building Int-Serv/RSVP products

o Int-Serv is not being widely deployed in the general Internet

-- Need Policy Control

-- Need for aggregation: “RSVP doesn’t Scale”

Integrated Services will be deployed first in intranets and other local environments, where scaling and Policy Control are much less challenging.

RSVP Scaling

Original design point: Aggregate BW limited # flows.

E.g., 1000 M/M conference sessions @ ~ 1 Mbps

=> 1Gbps BW, 1MB state.

New prospect:

30,000 Internet telephone sessions @ 30Kbps

=> 1Gbps BW, 30MB state.

RSVP protocol engineering could squeeze out some factors of 2, at the expense of protocol and implementation complexity.

Solution: QoS flow aggregation in "middle" of net => RSVP state aggregation

Int-Serv Flow Aggregation

- o In core where there is large amount of Int-Serv state:
  - -- Don't isolate individual flows.
  - -- Map flows into a fixed number of service classes
  - -- Let RSVP messages pass through transparently
  - -- Keep complete RSVP & TC state (only) in border routers.
- o Multicast makes it harder.
- o Details being worked out in ISSLL WG.

Differentiated Services

- Purpose: allow ISPs to sell, and customers to buy, various grades of Internet service.
- Explicitly avoid per-app-flow state in network; per-flow state only at the “edges”
- When possible, allocations are made on a long time scale and for large aggregates.
  <Provisioning replaces reservation>

Int-Serv and Diff-Serv

- Diff-serv provides a natural mechanism for aggregating Int-Serv flows in core of network.
  - Int-serv services end-to-end,
  - Diff-serv in the 'middle'.
- There is active work in ISSLL WG to define 'Int-Serv-over-Diff-Serv' service.

CONCLUSIONS

My crystal ball says:

- **The future will see some combination of IntServ and DiffServ widely deployed.**
- **RSVP will be widely used for End-End signaling.**
- **The RSVP protocol framework will be adapted to a variety of signaling tasks.**
- **The technical future of the Internet is still unfolding.**