REVIEW RSVP FUNCTIONAL SPECIFICATION

draft-ietf-rsvp-spec-05.xx

Bob Braden
USC Information Sciences Institute

RSVP Working Group

Danvers IETF Meeting

April 5, 1995

RSVP Functional Spec

- **draft-ietf-rsvp-spec-04.xx -- November 1994**

  -- Prior to San Jose meeting

- **draft-ietf-rsvp-spec-05.xx -- March 1995**

  -- Incorporates changes agreed in San Jose

  -- Major rewrite.

Functional Spec (05) Outline

1. INTRODUCTION
2. PROTOCOL MECHANISMS
3. EXAMPLES
4. FUNCTIONAL SPECIFICATION
5. MESSAGE PROCESSING RULES *[new]*

APPENDIX A: OBJECT DEFINITIONS *[new]*

APPENDIX B: UDP ENCAPSULATION *[new]*

APPENDIX C: DF STYLE *[moved from main text*]

1. INTRODUCTION

1.1 Data Flows

Introduce terminology:

sender, receiver,
DestAddress, session,
generalized port.

1.2 Reservation Model

1.3 Reservation Styles

1.1 Data Format

- **RSVP Session = (DestAddress,**
  generalized dest port **)**

- **DestAddress: unicast or multicast IP addr**

- ***Generalized dest port*****: further demuxing info.**

  Could be any transport- or application-layer info,
  but currently allow only UDP/TCP dest port.

(Note: RSVP Id is gone)

1.2 Reservation Model

- **Introduce: *flowspec, filter spec, flow descriptor***
- **Filter spec: only (sender addr, port, protocol id)**
  (Address family is implied).
- ***Merging* of flowspecs**
- **Reservation *style*.**
- ***One-pass*, *OPWA* reservation models**

Elementary Reservation Request

Flow Descriptor = (Q, F) pair.

- **Q = FLOWSPEC**

  -- Controls packet scheduler

  -- Contains *Rspec* (desired QOS) and *Tspec* (expected traffic level).

- **F = FILTERSPEC**

  --Controls Classifier, to select subset of data packets to receive service defined by Q.

Filter Spec

- Filter spec: select sender(s) =>
  (sender IP address, *generalized source port*)
  (Address family is implied)

- *Generalized source port*:

  -- Could select any subset of data flow in session

  -- Currently select only on UDP/TCP source port and/or protocol id.

  Other cases -- e.g., subflows of hierarchically-encoded signal -- use multiple sessions.

Reservation Functions

- **Receiver initiates request: FD = (Q, F)**
- **Request travels along reverse path to sender(s)**
- **At each node:**

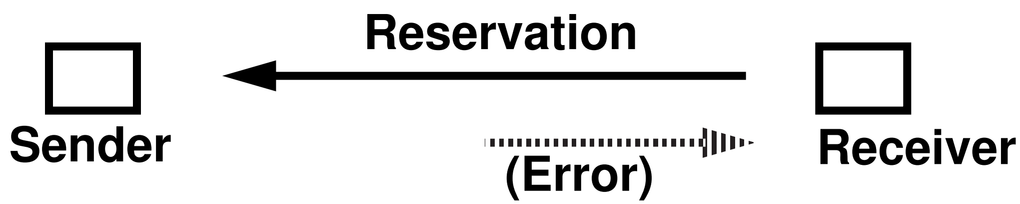
  (1) Local admission control decision on Q.
  (If fails, leave existing reservation in place)

  (2) Make reservation Q on outgoing interface.

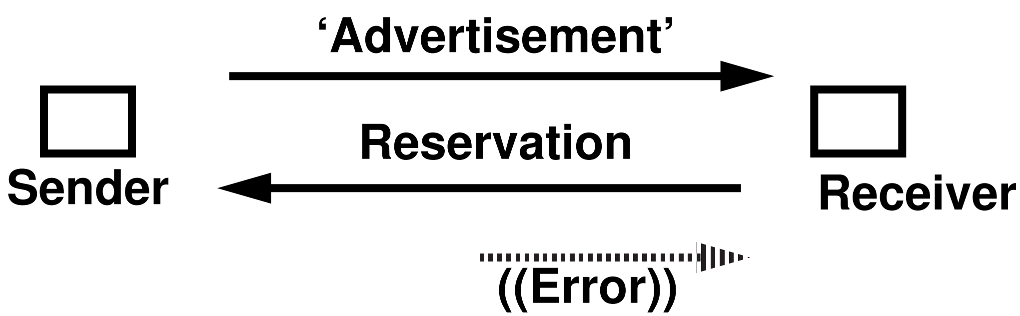
  (3) Forward reservation upstream
  towards set of senders in *scope*.

Reservation Models Supported by RSVP

One-Pass Model



One-Pass-With-Advertising (OPWA) [Shenker]



Reservation Style

Selects particular set of attributes --

- **Reservation for different senders, same session:**
  ***distinct* vs. *shared*?**

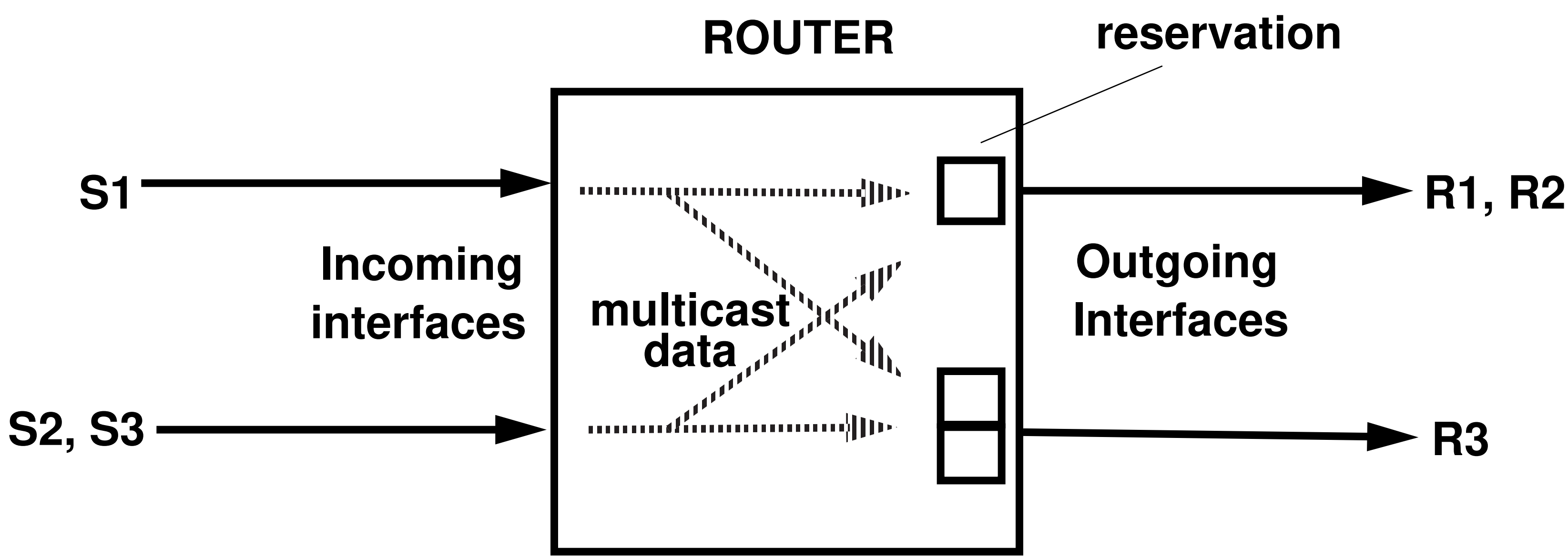
- **Scope:**
  explicit sender* vs. *wildcard?

Current styles:

WF = *shared, wildcard*
FF = *distinct, explicit*
[DF = distinct, wildcard, persistent]

Reservations and Interfaces

(Assume a given session)



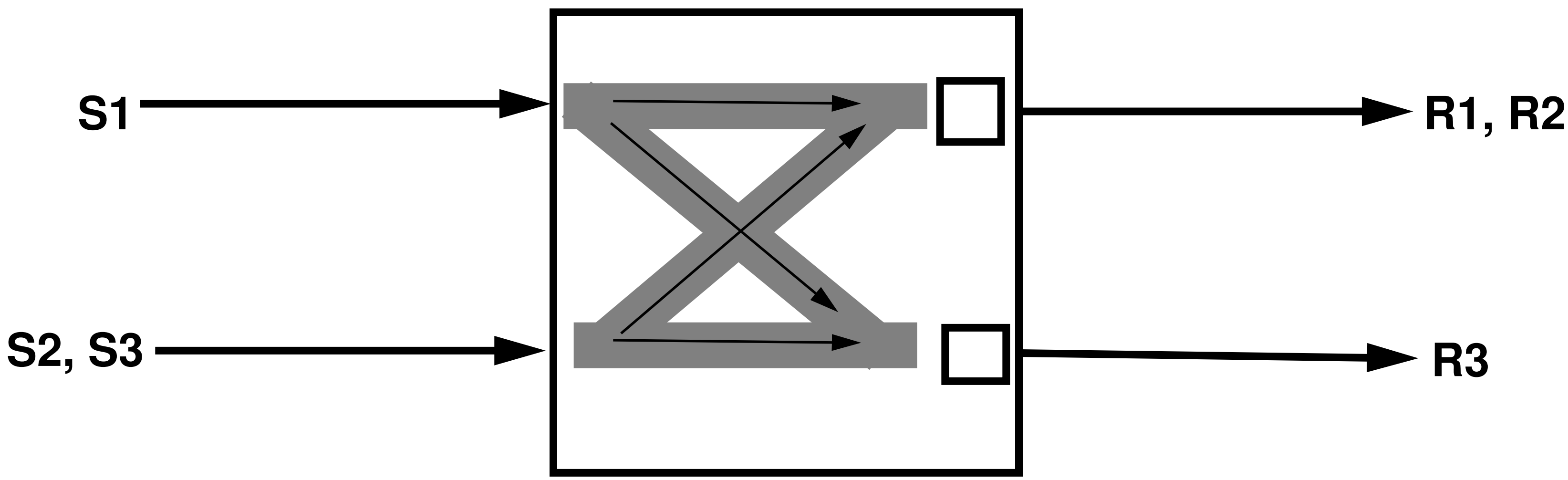
In reality: same physical interface may be both incoming and outgoing interface for RSVP.

Reservation Styles

- **Wildcard Filter (WF) Style**

  ***Shared* reservation, *wildcard* scope.**

  Create single reservation ‘pipe’, *e.g., audio*.

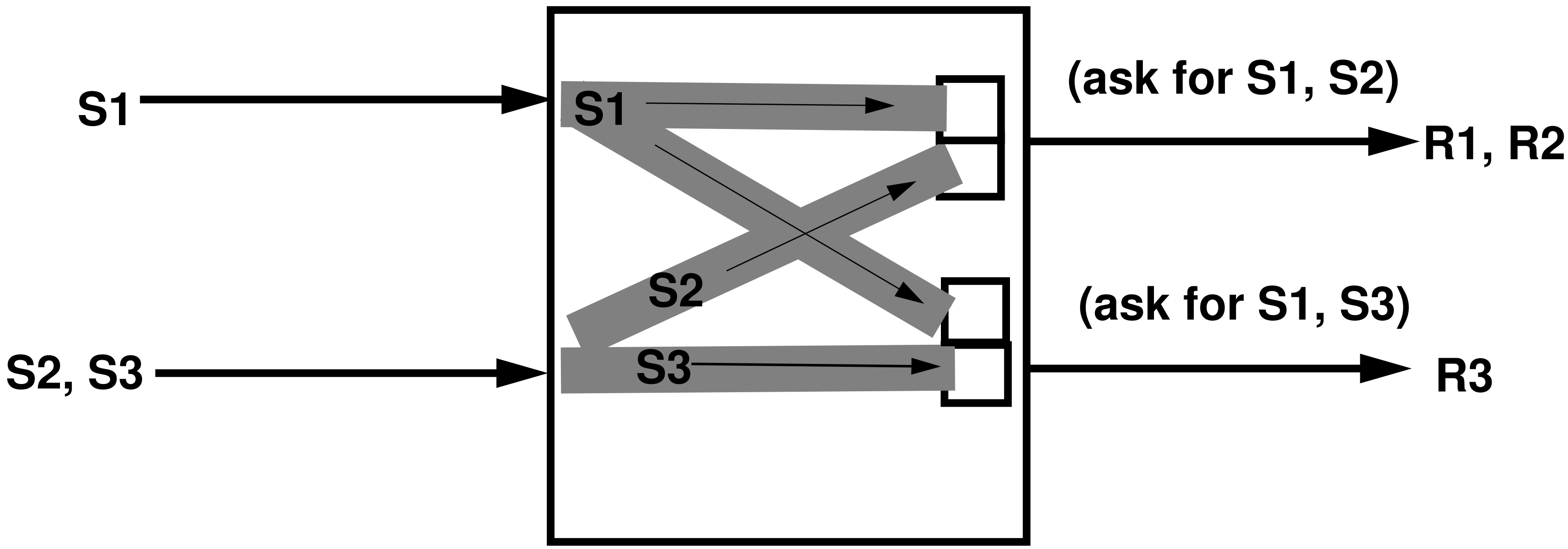


Reservation Styles

- **Filter (FF) Style**

  -- *Distinct* reservation, *explicit* scope

  -- e*.g.*, *video*.



2. PROTOCOL MECHANISMS

2.1 RSVP Messages

2.2 Soft State

2.3 Merging and Packing

2.4 Teardown

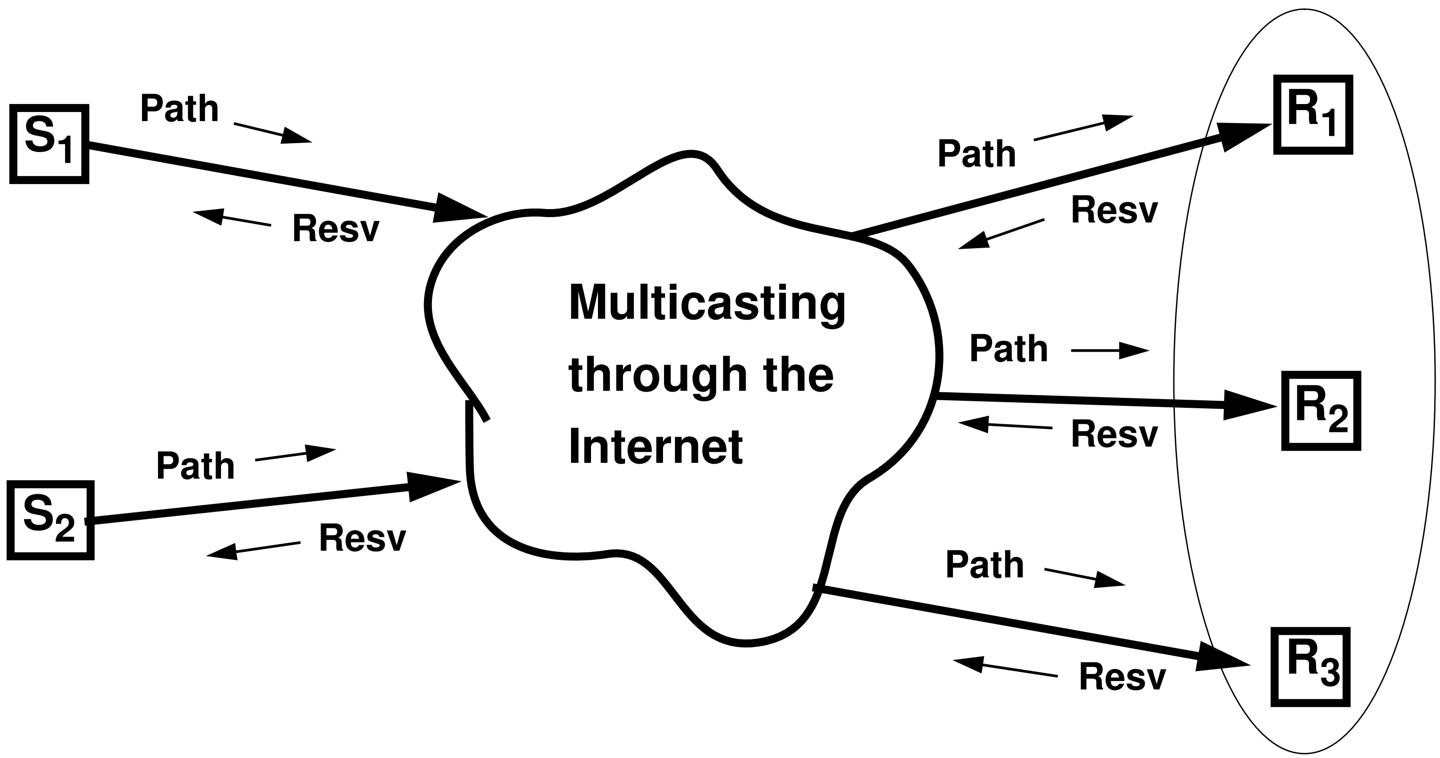
2.5 Security *[new]*

2.6 Automatic RSVP Tunneling

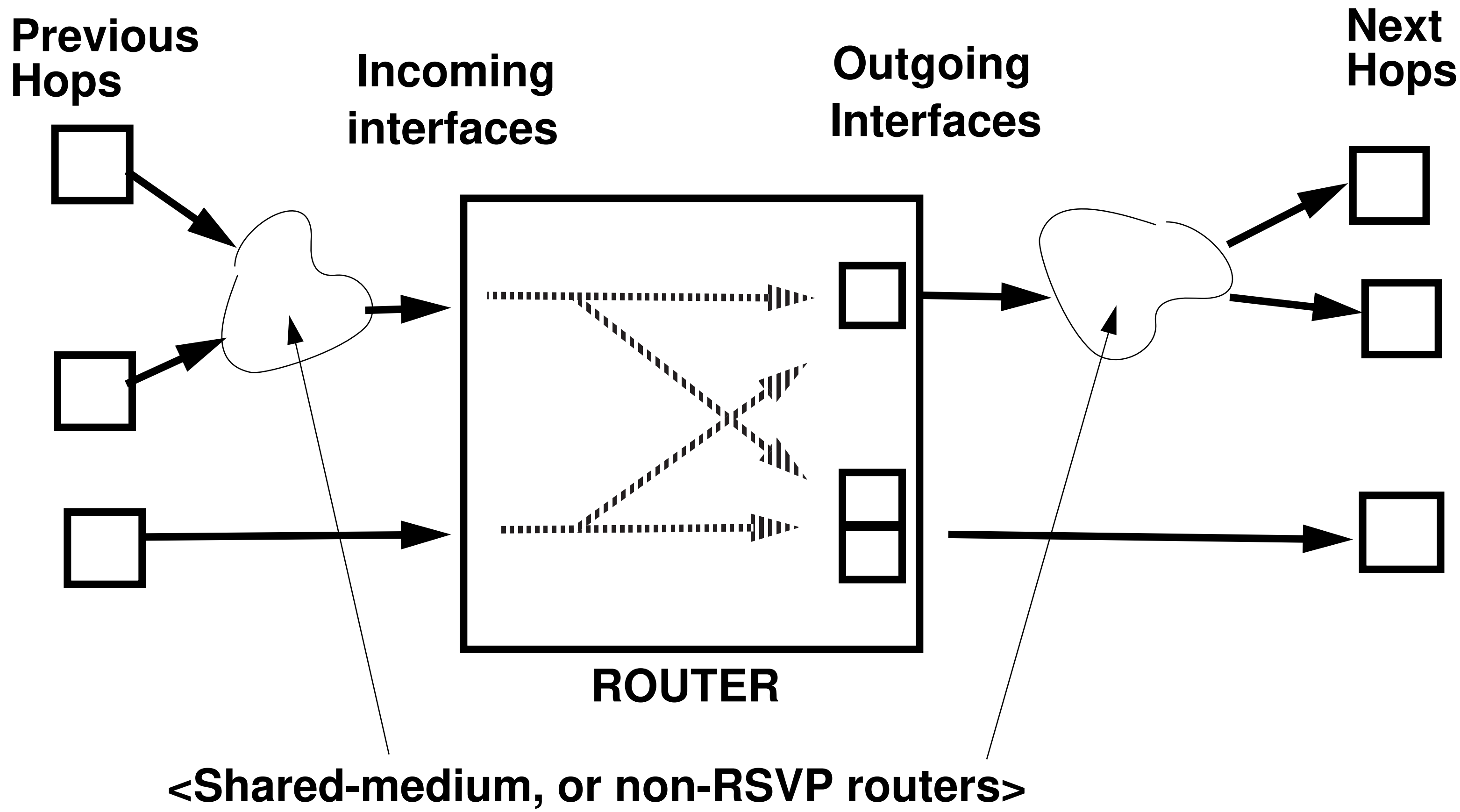
2.7 Session Groups *[new]*

2.8 Host Model

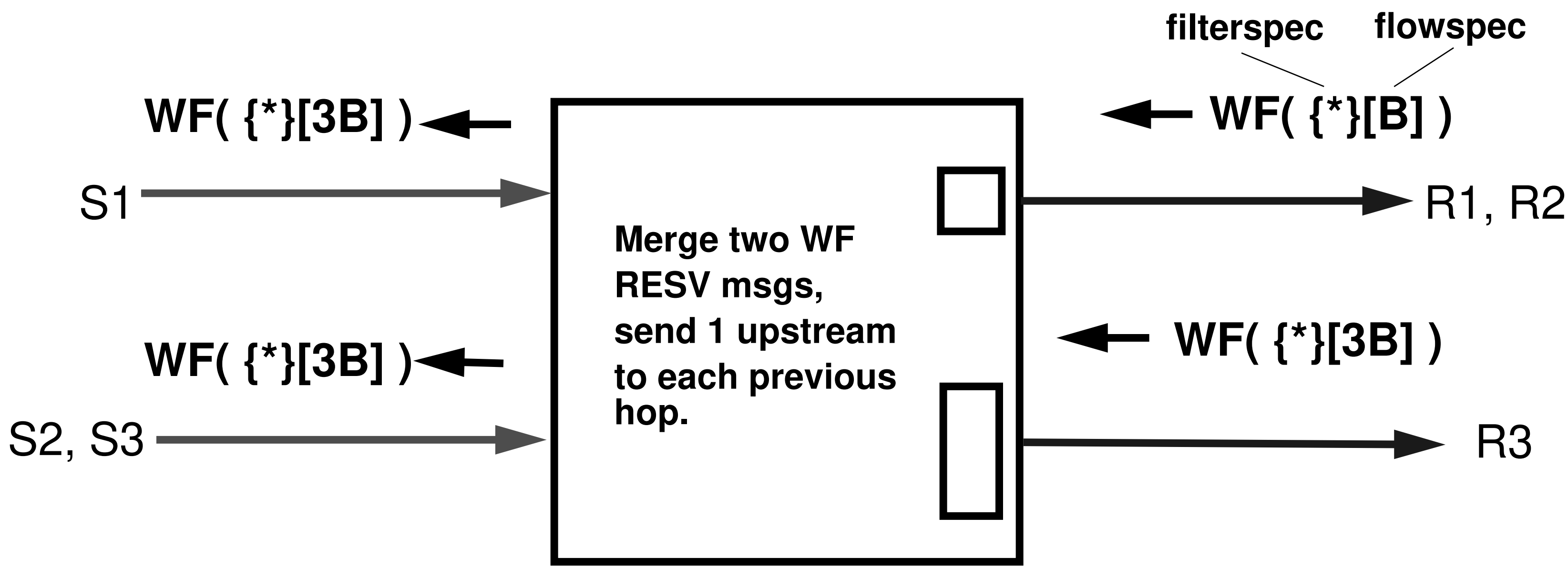
2.1 RSVP Messages



Interfaces vs. Hops



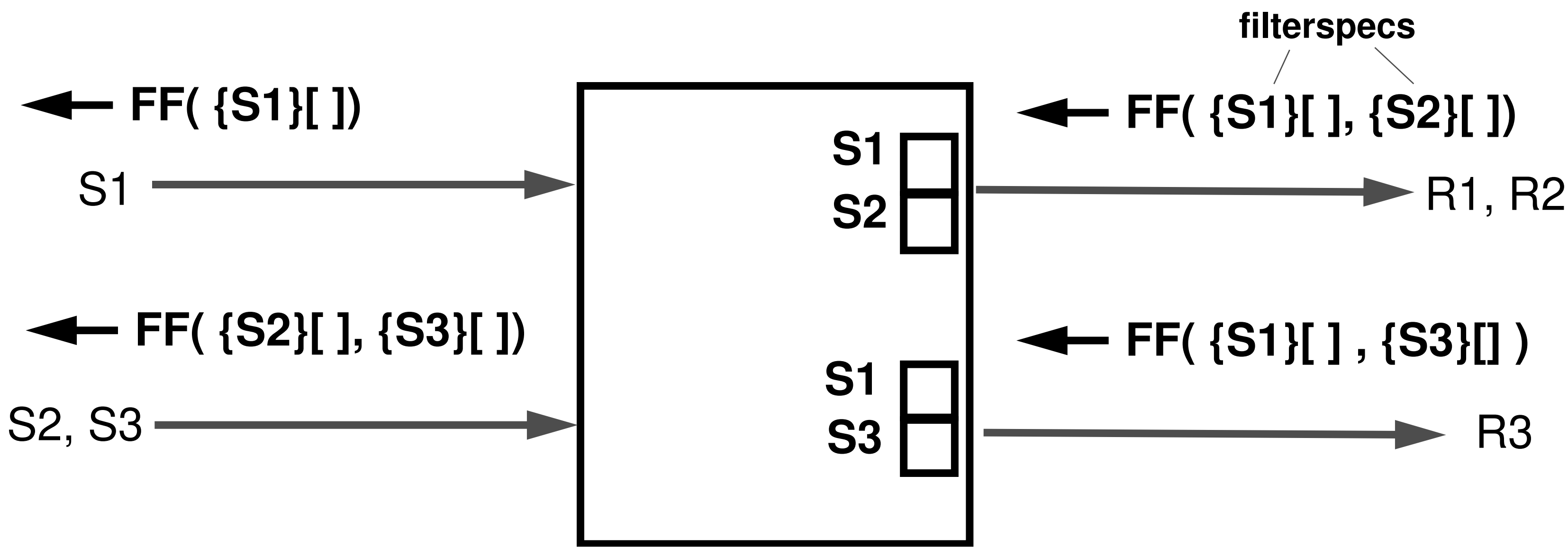
Merging Example: WF Resv



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

Merging Example: FF Reservation

Example: Fixed Filter Style:



2.2 Managing Soft State

- **Changes propagate immediately.**
- **Otherwise, forward message at end of current *refresh period*.**
- **State that is not refreshed times out after *cleanup timeout*.**

Managing Soft State

- If route changes, refreshes establish state along new route.

- If Q changes, simply adjust state as necessary.

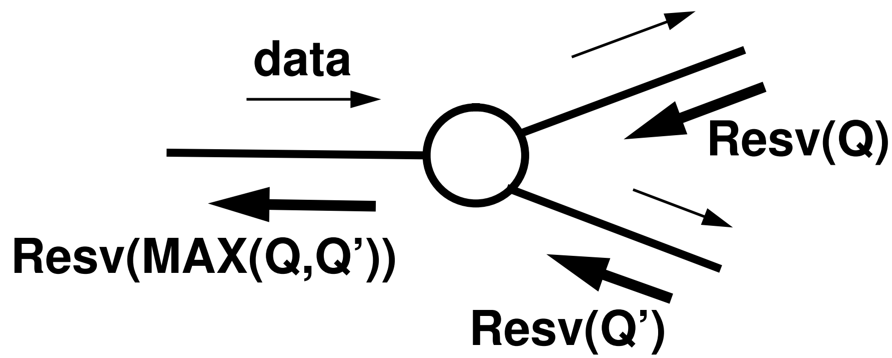
  E.g., change propagates only as far as necessary.

  E.g., a Teardown message may turn into a new Resv message at some point in the path.

- If F changes to F': old state for sender F times out, new state is established for sender F'.

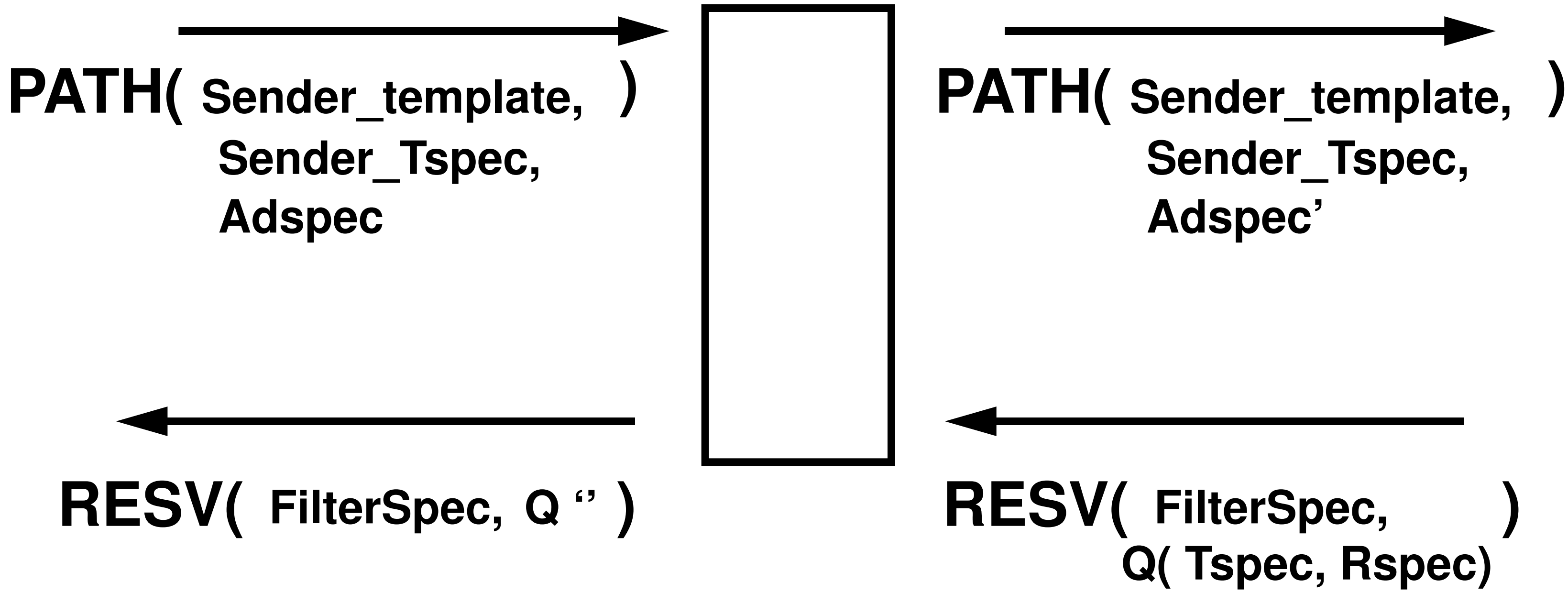
  (could explicitly tear down F state).

2.3 Merging and Packing



- **Merging:**
  - -- Only in Resv messages
  - -- Necessary consequence of multicasting.
- **Packing**
  - -- May be in Resv or Path messages
  - -- Optimization: packaging related info into one packet.

Processing in a Node



Make reservation: Q’ =

Q’ (MIN(Tspec, Sender_Tspec), Rspec)

Merge Q’(s) to compute Q’’

4. FUNCTIONAL SPECIFICATION

4.1 RSVP Message Formats

4.2 Sending RSVP Messages

4.3 Avoiding Message Loops

4.4 Local Repair

4.5 Time Parameters

[4.x Flow IDs]

4.6 RSVP Interfaces

4.1 RSVP Message Format

4.1.1 Common Header

```
      0             1              2             3
+-------------+-------------+-------------+-------------+
| Vers | Type |    Flags    |       Message Length      |
+-------------+-------------+-------------+-------------+
|       RSVP Checksum       |        Object Count       |
+-------------+-------------+-------------+-------------+
```

... plus variable list of variable-length *objects*.

4.1.2 Object Format

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

C-type: Unique within Class

(Class, C-Type) completely determines format of object contents
(hence, implies address family).
See Appendix A for object formats).

Object Classes

0	NULL
1	SESSION
2	SESSION_GROUP
3	RSVP_HOP
4	STYLE
5	FLOWSPEC
6	FILTER_SPEC
7	SENDER_TEMPLATE
8	SENDER_TSPEC
9	ADSPEC
10	TIME_VALUES
11	ERROR_SPEC
12	CREDENTIAL
13	INTEGRITY

Object Types (C-Types)

May define new types for:

- **Extended functions**

- **New address formats (e.g., IP6, or other address family)**

  (Define objects in Appendix A, to make it easier to add new types.)

Spec defines valid object sequences using BNF.

4.1.3 Path Message

<Path Message> ::= <Common Header> <SESSION> <RSVP_HOP>

[<INTEGRITY>] [<TIME_VALUES>]

<sender descriptor list>

<sender descriptor list> ::= <empty > |

<sender descriptor list> <sender descriptor>

<sender descriptor> ::= [<CREDENTIAL>] <SENDER_TEMPLATE>

[<SENDER_TSPEC>] [<ADVERT>]

4.1.4 Resv Message

<Resv Message> ::= <Common Header> <SESSION>

[<SESSION_GROUP>] <RSVP_HOP>

[<INTEGRITY>] [<TIME_VALUES>]

[<CREDENTIAL>] <style-specific tail>

<style-specific-tail> ::=

<Style-WF> [<FILTER_SPEC>] <FLOWSPEC> |

<Style-FF> <flow descriptor list> | ... (etc.)

<flow descriptor list> ::= <empty> |

<flow descriptor list> <FILTER_SPEC> <FLOWSPEC>