RSVP Implementation Status

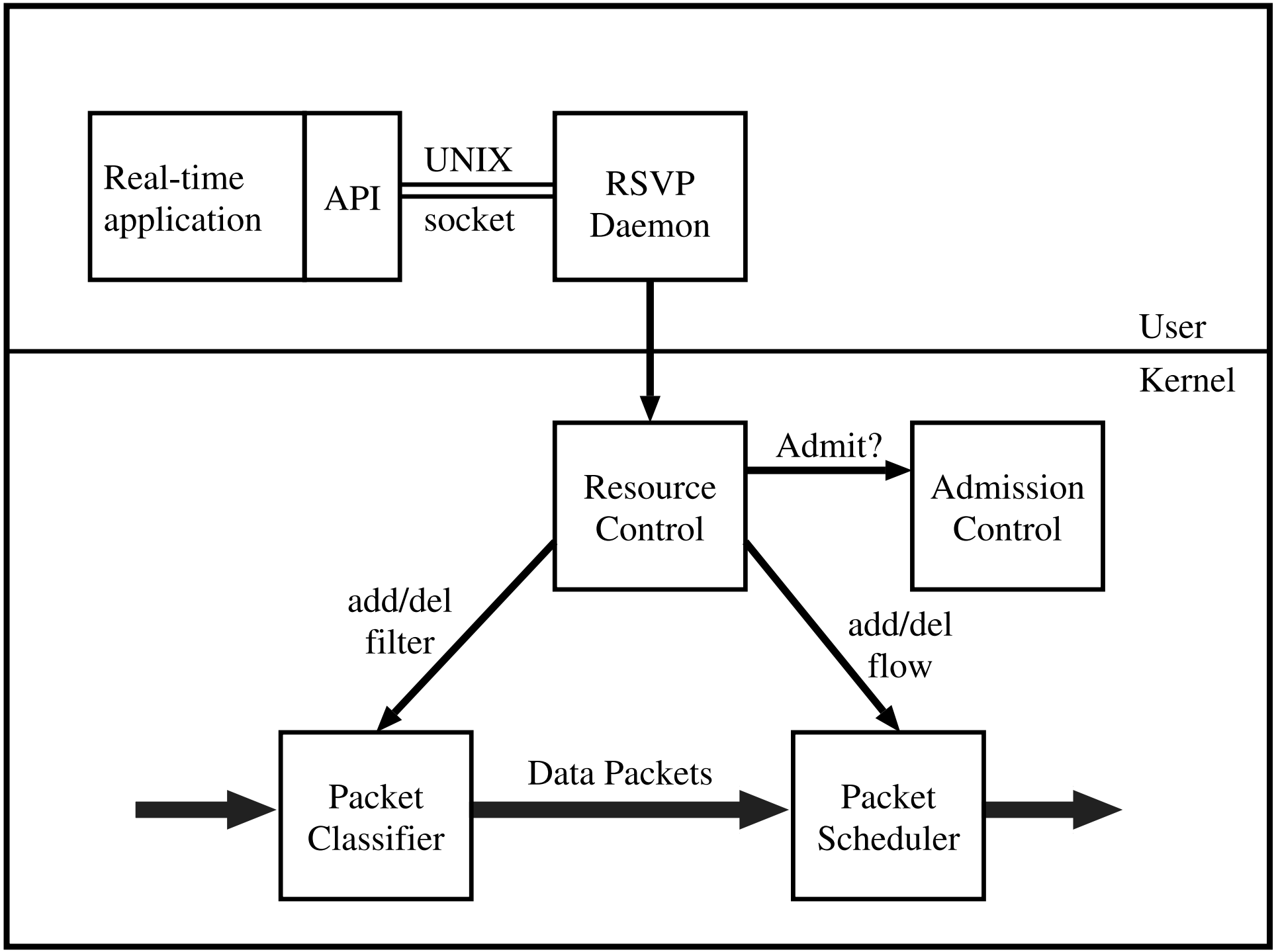
Steven Berson

USC Information Sciences Institute

July, 1994

Overview of Implementation Status

- System Model
- Requirements
- What is implemented
- Testing environment
- What is not (yet) implemented


Real-time application
API
UNIX
socket
RSVP Daemon
User
Kernel
Resource Control
Admit?
Admission Control
add/del filter
add/del flow
Packet Classifier
Data Packets
Packet Scheduler

Requirements

- Requires New IP Multicast (with pruning)
  - +query kernel for list of VIFS
  - +send packets to particular interface, i.e. bypassing routing
  - +pass RSVP protocol messages to daemon
- Requires rsvpd to get routes from mrouted

What is implemented - I

- Working RSVP for SunOS 4.1.x and Solaris
  - + Runs as a user level daemon
  - + Same kernel/daemon runs on routers and hosts
- 3 Reservation Styles DF, FF, and WF
- Merging of Reservations
- Dynamic adaptation to changes in group membership, flowspecs
- Supports tunnels through non-RSVP routers

What is implemented - II

- General flowspecs/filterspecs
  - + Variable length flowspecs and filterspecs
  - + (Mask, Value) for filterspecs
- Teardown messages
  - + Rapid removal of reservation/path state

What is implemented - III

- API
  - + Register a sender
  - + Register a receiver
  - + Make reservations
  - + Synchronous or asynchronous error delivery
- Tools
  - + rtap - tool for testing API
  - + mmcc - multimedia conferencing tool
  - + nv - video display tool
  - + sd/vat - audio tool

DARTnet Testing Environment

- dmap - program to display RSVP state of DARTnet routers
- rsvpdump - tool to dump contents of RSVP packets
- RSVP meta-daemon - uses multicast to send commands to all RSVP sites; uses:
  - +restart a new daemon
  - +adjust the debug level
  - +distribute a new version of daemon

What is not implemented

- Integration with packet-scheduling kernel
  + CBQ
- Support for new ("prune-beta") multicast
- New API (as in new spec)