The Integrated Services Internet: Multicast, Resource Reservations and the Real-Time Transport Protocol

Henning Schulzrinne
GMD Fokus, Berlin
schulzrinne@fokus.gmd.de

Overview

- multicast
- resource reservation protocols
- transport services for real-time data
Broadcast and Multicast

**broadcast**: all hosts on (small, local) network

**directed broadcast**: all hosts on remote network

**multicast**: multiple recipients (group)

Applications for Multicast

- audio-video distribution (1-to-many) and symmetric (all-to-all)
- distributed simulation (war gaming)
- resource discovery
- file distribution (stock market quotes, new software, …)
Connection-Oriented Multicast

- enumerate sources explicitly
- examples:
  - ATM (explicit connection to each end point)
  - ST-II (several end points in setup message)
  - enumeration of end points in packet
- only connection-oriented (packet header size!)
- source needs to know destinations $\leftrightarrow$ resource discovery, dynamic groups

Host Group Model

Deering, 1991:

- senders need not be members;
- groups may have any number of members;
- there are no topological restrictions on group membership;
- membership is dynamic and autonomous;
- host groups may be transient or permanent.
Local Multicast

Some local networks are by nature multi/broadcast: Ethernet, Token Ring, FDDI, …

Ethernet, Tokenring:

- broadcast: all ones
- multicast: 01.xx.xx.xx.xx.xx
- adapter hardware can filter dynamic list of addresses

ATM: point-to-point links ➤ need ATM multicast server

IP Multicast

- host-group model
- network-level; data packets same, only address changes
- need help of routers
- special IP addresses (class D): 224.0.0.0 through 239.255.255.255
- 28 bits ➤ 268 million groups (plus scope)
- 224.0.0.1: all hosts
- some pre-assigned (224.0.1.2: SGI Dogfight)
- others dynamic (224.2.x.x for multimedia conferencing)
- map into Ethernet: 01.00.5E.00.00.00 + lower 23 bits
**IGMP**

Multicast for local (broadcast) networks, between router and hosts

- router listens to all multicast packets on all interfaces
- hosts send report for first process to join group to multicast group
- host does not send report when processes all have left
- router multicasts query to all hosts ≈ once a minute
- host waits and listens for others; if nobody else, send response

**Multicast Forwarding**

1. check incoming interface: discard if not on shortest path to source
2. forward to all outgoing interface except incoming
3. don’t forward if interface has been pruned
4. prunes time out every minute
5. routers may send grafts upstream

routing information: DVMRP
Multicast Forwarding

First packet (truncated broadcast)

**Multicast Forwarding**

With pruning:

- router needs to keep “negative” list for groups
Multicast Programming

UDP, not TCP (obviously...)

```c
struct sockaddr_in name;
struct ip_mreq imr;

sock = socket(AF_INET, SOCK_DGRAM, 0);
imr.imr_multiaddr.s_addr = htonl(groupaddr);
imr.imr_interface.s_addr = htonl(INADDR_ANY);
setsockopt(sock, IPPROTO_IP, IP_ADD_MEMBERSHIP,
           &imr, sizeof(struct ip_mreq));
name.sin_addr.s_addr = htonl(groupaddr);
name.sin_port = htons(groupport);
bind(sock, &name, sizeof(name));
recv(sock, (char *)buf, sizeof(buf), 0);
```

Problems

- “multicast storms”
- state in routers for sparse groups vs. optimal trees
- multicast routing vs. unicast routing (reverse path)
- hierarchical routing
MBONE

- MBONE ≡ multicast backbone
- overlay network over Internet
- needed until deployment of multicast-capable backbone routers
- IP-in-IP encapsulation:

\[
\begin{array}{c|c|c|c|c|c|c}
& IP header & 193.1.1.1 & 224.2.0.1 & 17 & UDP & RTP \\
192.1.2.3 & 128.3.5.6 & (IP) & & UDP & & audio/video data \\
\end{array}
\]

source: 193.1.1.1; group: 224.2.0.1; MBONE tunnel: 192.1.2.3 to 128.3.5.6

- limited capacity, resilience
Multicast: Further Information

- S. Deering, RFC 1112
- D. Comer, *Internetworking with TCP/IP*
- R. Perlman, *Interconnections - Bridges and Routers* ("IP multicast is bad")
- C. Huitema, *Routing in the Internet*

Resource Reservation
Resource Reservation Issues

sender-oriented: sender specifies bandwidth, receivers ➞ ATM, ST-II

receiver-oriented: receivers notify sender ➞ RSVP

hard state: reliable connection set-up (handshake), explicit tear-down ➞ Q.2931, ST-II

soft state: probabilistic set up, time-outs ↔ periodic state refresh

Any protocol should almost never say “no” (unless it is run by a monopoly...)

RSVP

receiver-oriented reservation protocol being standardized by IETF:

- not a routing protocol, but interacts with routing
- transports opaque QOS and policy parameters for sessions
- simplex ➞ setup for unidirectional data flows
- does not prescribe admission or policy control
- sets up packet classifier, but does not handle packets
- independent sessions (can’t tie video and audio session)
- multicast (and unicast)
- either own protocol type or UDP encapsulated
RSVP Objects

Flow descriptor =

**Flowspec:**
- service class
  - Rspec – desired QoS
  - Tspec – describes traffic characteristics

**Filterspec:** which packets get this treatment ➞ sender IP address/port, protocol, other fields ➞ complex (regular expressions? IP options!) ➞ currently, sender IP address and UDP/TCP port ➞ no fragmentation

Reservation Styles

<table>
<thead>
<tr>
<th>sender reservations</th>
<th>selection</th>
<th>distinct for each sender</th>
<th>shared</th>
</tr>
</thead>
<tbody>
<tr>
<td>explicit</td>
<td>fixed filter</td>
<td></td>
<td>shared-explicit</td>
</tr>
<tr>
<td>wildcard (all)</td>
<td>–</td>
<td></td>
<td>wildcard filter</td>
</tr>
</tbody>
</table>

⇒ mutually incompatible

**shared:** only one active data source ➞ e.g., reserve for twice needed for audio

**distinct:** video
RSVP: Basic Operation

- source sends PATH messages to receivers: previous hop to source, Tspec, OPWA
- receivers send RESV messages back to senders
- reservations are merged at each node for same sender (max. flowspec)
- merge point or data sender may send confirmation (if requested)
- reservations \textit{may} get merged between senders (audio!)
- one-pass ➞ receiver doesn’t know final QoS ➞ One Pass With Advertising
- application \textit{should} explicitly tear down reservations

Killer Reservations

1. small reservation in place; another receiver larger reservation ➞ failure? ➞ keep old

2. large reservation fails again and again ➞ blocks new, smaller one
## Mapping RSVP onto ATM

<table>
<thead>
<tr>
<th>Feature</th>
<th>IP, RSVP</th>
<th>ATM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Multicast tree, reservation</td>
<td>sequential</td>
<td>same time</td>
</tr>
<tr>
<td>Origin</td>
<td>receiver</td>
<td>sender (root)</td>
</tr>
<tr>
<td>Change reservations</td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>Routing changes</td>
<td>time-out</td>
<td>re-establish VC</td>
</tr>
<tr>
<td>Flow merging (audio)</td>
<td>yes</td>
<td>no (separate VCs)</td>
</tr>
<tr>
<td>Receiver diversity</td>
<td>not yet</td>
<td>no</td>
</tr>
</tbody>
</table>

Breitband June 21, 1996

## Real-Time Transport Protocol

Breitband June 21, 1996
RTP: The Big Picture

RTP: Real-Time Transport Protocol

- only part of puzzle
- product of Internet Engineering Task Force, AVT WG
- RFC 1889, 1890
- ITU H.323, Netscape
- support for functions, but does not restrict implementation
- compression for low-bandwidth networks under study
RTP: Goals

**lightweight:** specification and implementation

**flexible:** provide mechanism, don’t dictate algorithms

**protocol-neutral:** UDP/IP, ST-II, IPX, ATM-AALx, …

**scalable:** unicast, multicast from 2 to $O(1000)$

**separate control/data:** some functions may be taken over by conference control protocol

**secure:** support for encryption, possibly authentication

RTP: Functions

- segmentation/reassembly done by UDP (or similar)
- resequencing (if needed)
- loss detection for quality estimation, recovery
- intra-media synchronization: remove delay jitter through playout buffer
- intra-media synchronization: drifting sampling clocks
- inter-media synchronization (lip sync)
- quality-of-service feedback and rate adaptation
- source identification
RTP: Mixers, Translators, …

mixer:
- several media streams ➔ one new stream (new encoding)
- mixer: reduced bandwidth networks (dial-up)
- appears as new source

translator:
- single media stream
- may convert encoding
- protocol translation, firewall

Breitband
June 21, 1996
**RTP: Packet format**

<table>
<thead>
<tr>
<th>Field</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>T=2</td>
<td>Padding (for encryption)</td>
</tr>
<tr>
<td>P</td>
<td>Marker bit; indicates frame, talkspurt</td>
</tr>
<tr>
<td>X</td>
<td>Content source count</td>
</tr>
<tr>
<td>CSRC</td>
<td>Synchronization source identifier (SSRC)</td>
</tr>
<tr>
<td>M</td>
<td>Payload type: audio, video encoding method</td>
</tr>
<tr>
<td>sequence</td>
<td></td>
</tr>
<tr>
<td>payload</td>
<td></td>
</tr>
<tr>
<td>type</td>
<td></td>
</tr>
<tr>
<td>timestamp</td>
<td></td>
</tr>
<tr>
<td>content</td>
<td></td>
</tr>
<tr>
<td>source</td>
<td></td>
</tr>
<tr>
<td>identifiers</td>
<td></td>
</tr>
<tr>
<td>(CSRC)</td>
<td></td>
</tr>
</tbody>
</table>

**RTP: Control Protocol – Algorithm**

Goals:

- estimate current number of participants – dynamic
- participant information ➔ talker indication
- quality-of-service feedback ➔ adjust sender rate
- scale to $O(1000)$ participants, small fraction of data bandwidth
- randomized response with rate ↓ as members ↑
- limited by tolerable age of status
- gives active senders more bandwidth
RTP: Control Protocol – Types

stackable packets, similar to data packets

**sender report:** bytes send \(\Rightarrow\) estimate rate;
timestamp \(\Rightarrow\) synchronization

**reception reports:** number of packets sent and expected \(\Rightarrow\) loss,
interarrival jitter, round-trip delay

**source description:** name, email, location, ...

**explicit leave:** in addition to time-out

**extensions:** application-specific

---

RTP: Control Protocol — SR

```
0 8 16 24
T=2 P receiver count payload type = SR packet length

synchronization source identifier (SSRC)

RTP sample-clock timestamp

sender's packet count

sender's byte count

synchronization source identifier of remote source

cumulative number of packets received

cumulative number of packets expected

interarrival jitter

last SR NTP timestamp seen from this source

delay since receiving that SR NTP timestamp
```
RTP Implementations

<table>
<thead>
<tr>
<th>tool</th>
<th>who</th>
<th>media</th>
<th>RSVP</th>
</tr>
</thead>
<tbody>
<tr>
<td>NeVoT</td>
<td>GMD Fokus</td>
<td>audio</td>
<td>soon</td>
</tr>
<tr>
<td>NeViT</td>
<td>GMD Fokus</td>
<td>video</td>
<td>yes</td>
</tr>
<tr>
<td>vic</td>
<td>LBNL</td>
<td>video</td>
<td>no</td>
</tr>
<tr>
<td>vat</td>
<td>LBNL</td>
<td>audio</td>
<td>no</td>
</tr>
</tbody>
</table>

http://www.fokus.gmd.de/step/rtp/

Conclusion

- introduced range of approaches:
  - adaptive applications
  - layered applications
    - resource reservation
    - transport protocols
- authentication
- charging for shared reservations?