Multicast and Real-Time Applications: IGMP, RSVP, RTP

GMD fokus

Overview

- ➤ Internet as “integrated services network”
- ➤ Multicast: why, how
- ➤ The MBONE
- ➤ real-time audio and video
  - requirements
  - network support and resource reservation
  - transport protocols

GMD fokus

“Classic” Internet to “integrated” Internet

“Classic” Internet:

- data services
- email, file transfer, WWW, …
- unicast
- single class of service

“Integrated Services” Internet adds…

- real-time services (audio, video, …)
- resource reservation, differentiated service?
- multicast

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 ➠ explicitly add each end point
  - ST-II ➠ enumerate end points in setup message
  - enumeration of end points in packet
- only connection-oriented (packet header size!)
- source needs to know destinations ↔ 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.

GMD fokus

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

GMD fokus

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.x: local network only
- 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
- ttl value limits distribution: 0=host, 1=network

GMD fokus

IGMP

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

- router listens to all multicast packets on all interfaces
- hosts sends IGMP report for first process to join group to that multicast group (ttl=1)
- host *does not* send report when processes all have left
- router multicasts (group: 0) query to all hosts = 224.2.0.1 $\approx$ once a minute
- host waits and listens for others; if nobody else, send response

GMD fokus

IGMP packet

0–4	4–8	8–16	16–32
version (1)	IGMP type (1-2)	(unused)	16-bit checksum
32-bit group address (class D IP address)			

8 bytes

```
$ netstat -g
Group Memberships
Interface Group                RefCnt
--------- -------------------- ------
lo0       ALL-SYSTEMS.MCAST.NET     1
le0       224.2.127.255             1
le0       ALL-SYSTEMS.MCAST.NET     1
```

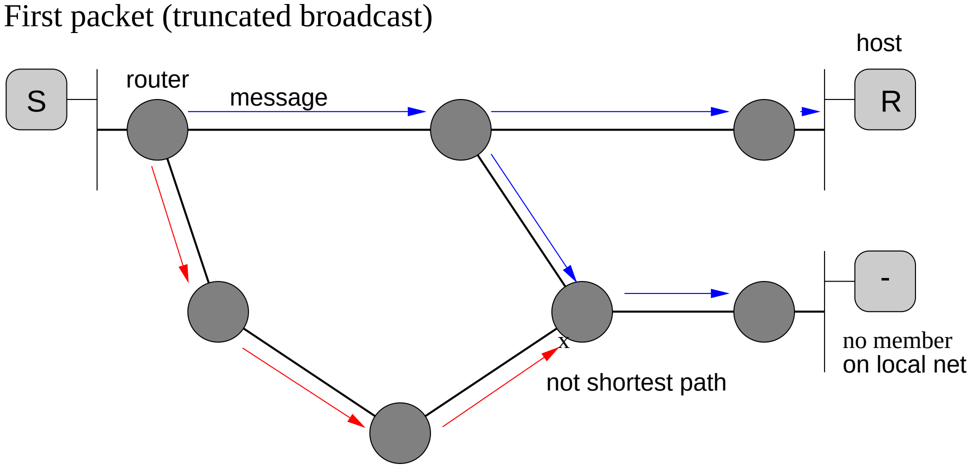
Multicast forwarding

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

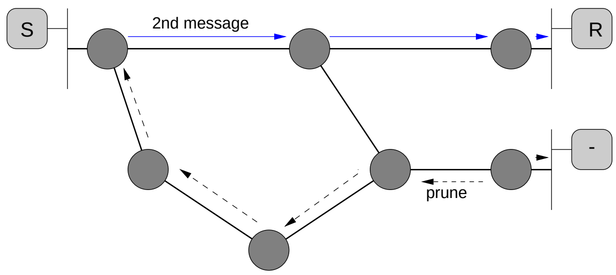
Multicast forwarding

First packet (truncated broadcast)



Multicast forwarding

With pruning:



➠ router needs to keep “negative” list for groups

Multicast programming

UDP, not TCP (obviously…)

```
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);
```

GMD fokus

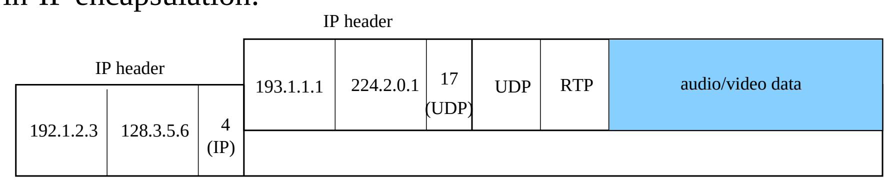
Problems

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

GMD fokus

MBONE

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



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*

Real-time services

Why bother integrating?

- use existing workstations as audio/video terminals
- use existing LAN/WAN infrastructure
- efficiency:
  - true (LAN/WAN) multicast instead of MCUs and bridges – same application scales from two to hundreds of receivers
  - variable-bit rate (VBR) video, but interoperation with H.261 standards
  - audio silence suppression – important for large-scale conferences

Why bother integrating (cont'd)?

- added control functionality:
  - directory services
  - visual speaker indication
  - source selection at receiver
- integration of application-sharing and data applications, WWW

Disadvantages of packet audio/video

- no resource reservation ➠ quality may suffer (but: RSVP)
- may push overloaded networks over the edge
- low frame rates for workstation video codecs
- hands-free speaking in infancy (echo)
- packetization overhead, delay
- operating systems ill-suited for real-time applications ➠ single-user, no background load
- office environment: acoustics, lighting, …

Characteristics of digital audio and video

	audio	video
rate (kb/s)	13…64…1500	200…1500…6000
loss tolerance	$\leq 5\%$	10^{-5}…1%
packet size	small	large
traffic	interrupted CBR	VBR

One-way delay tolerance:

conference audio without echo cancellation: 40 ms

conference audio with echo cancellation: 150 ms

playback audio/video: $\geq$ 500 ms (“VCR response”)

Integration of real-time and data traffic

- TCP (and TP4, …) not suited to carry real-time traffic:
  - flow control ➠ window backoff and slow start
  - retransmission delay
- TCP backoff nice for making room for real-time traffic
- options:
  - low load, rely on data traffic backoff (but NFS doesn’t)
  - priorities at MAC/link level (starvation!)
  - bandwidth-allocating scheduling at link level
- transport protocols can’t help (except with set-up)!

Integration of real-time and data traffic

- single-priority token rings cannot guarantee bounds (token loss), but may be sufficient
- MAC priorities (token ring, FDDI) won't help (much)
- priorities often not even implemented
- ATM: promises, but (most) switches don't implement traffic control

Resource reservation

- only makes sense with differential charging (or administrative controls)
- reserve resources ➠ preferential treatment for packets
- could have many reservation protocols
- sender or receiver oriented

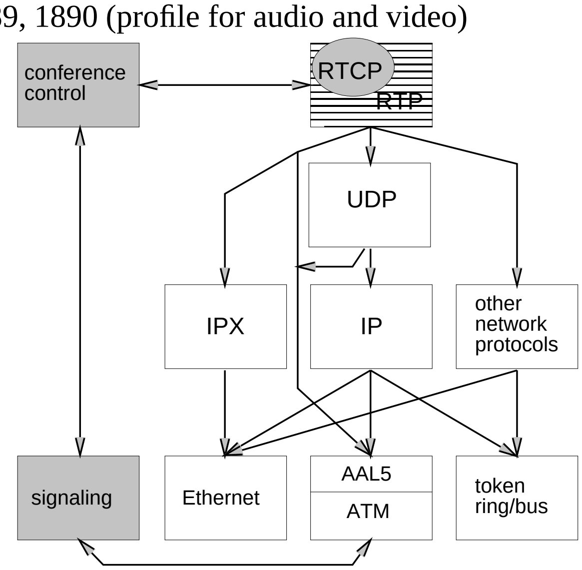
RSVP

receiver-oriented reservation protocol being standardized by IETF:

- multicast (and unicast)
- source sends PATH messages to receivers: path, max. flowspec
- receivers send RESV messages back to senders
- reservations are merged for same sender (max.)
- reservations *may* get merged between senders (audio!)

RTP as part of protocol stack

RTP: RFC 1889, 1890 (profile for audio and video)



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 and translators

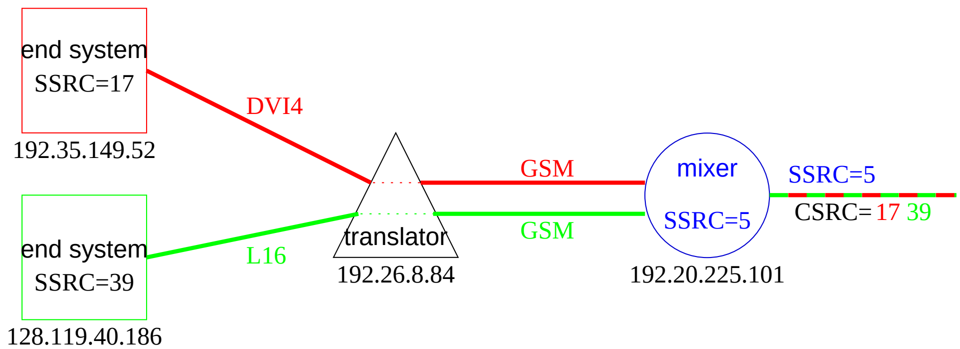
mixer:

- several media stream ➠ one new stream (new encoding)
- reduced bandwidth networks (dial-up)
- appears as new source

translator:

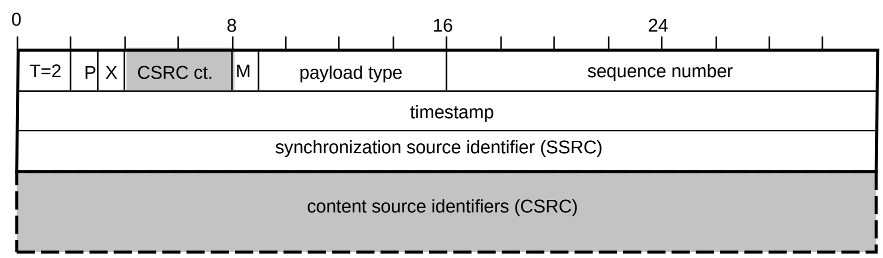
- operates on individual media streams
- *may* convert encoding
- protocol translation, firewall

RTP mixers and translators



fokus

RTP packet format



P: padding (for encryption)

M: marker bit; indicates frame, talkspurt

CC: content source count

Payload type: audio, video encoding method (static and dynamic)

SSRC: synchronization source – random 32-bit identifier

CSRC: list of contributing sources (mixer)

fokus

RTP control protocol – algorithm

Goals:

- estimate current number of participants – dynamic
- participant information ➠ talker indication
- quality-of-service feedback ➠ adjust sender rate
- side effect: connectivity indication
- 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

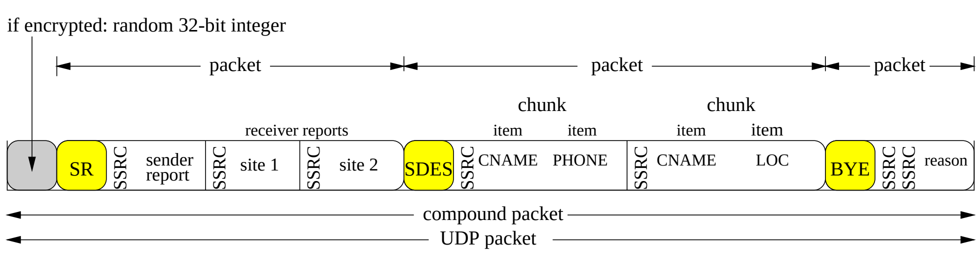
stackable packets, similar to data packets

sender report (SR): bytes send ➠ estimate rate; timestamp ➠ synchronization

reception reports (RR): number of packets sent and expected ➠ loss, interarrival jitter; round-trip delay

source description (SDES): name, email, location, …

explicit leave (BYE): in addition to time-out



Conclusion

- current LANs/WANs can support small number of audio/video connections
- need range of approaches:
  - adaptive applications
  - signaling
  - resource reservation
  - transport protocols
  - switch and router support for QoS
- charging for shared reservations?