# IP multicast and multimedia

TCP/IP class

Jim Binkley

# outline

- introduction
- hw level multicasting
- IGMP - Internet Group Mgmt. Protocol
- multicasting routing
- multicast applications on the MBONE
- even more multicast routing
- multimedia applications/programming
- problems/research

# introduction

- problem 1:  how do we make applications and support net that ship data at regular rate; i.e., **isochronous** (need QOS)?
- apps might include: audio/video/file xfer/distributed parallel, who knows...
- problem 2: how do these applications address each other: 1 to N, N to 1, N to N?
- & 3: what is on channel 10 anyway ; i.e., there is a directory service or rendezvous problem

# circuit-switched or packet switched?

- traditional voice done by reserving pipe end to end - it is isochronous data
- packet switches don’t reserve anything - support bursty (and lossy) use
- in either case, we can make the PIPE FATTER but
- will it be FAT enough for full-screen, full-motion 3-d video?

# atm versus ip multicasting?

- atm - circuit switch heritage and approach
- question is: can atm deal with bursty “datagram”-oriented data ?
- atm consists of virt. circuits in a star topology
- con: can’t broadcast, if we multipoint, how can we find interested parties?
- pro: big(ger) pipe; con: routing unclear

# so why doesn’t ATM take over?

- QOS possible with switch system because buffer space tied down by signaling
- ATM might compete in LAN, network backbone, or WAN **in theory.**
- LAN - ethernet supports broadcast and is cheap!, ATM lost cause on LAN
- same for backbone,  gigabit ethernet cheaper, per port, and per switch
- WAN - ATM has qos, but more overhead than other solutions, in use as pt/pt telco solution

# multicasting., etc

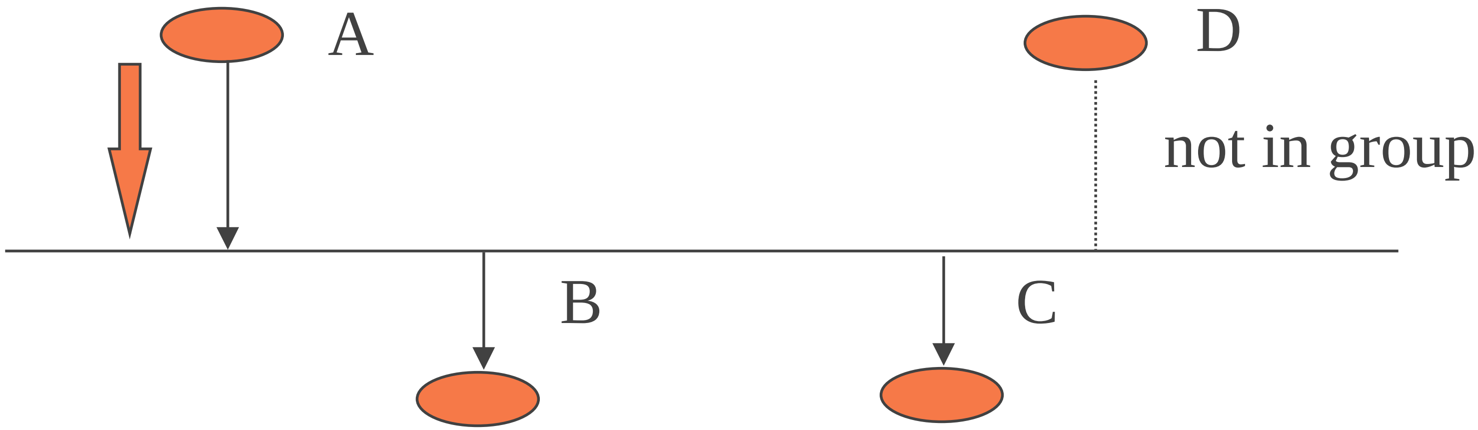
- hardware support (ethernet) on link
- question: how to route across links? in LAN or
- multicasting across **WANS** (MBONE)
- what applications exist on top of MBONE
- where to go from here? research? (yep...) **"whatever it is, it's not done yet..."**

# multicast at hw/link-layer

- MAC address: 01:xx:xx:xx:xx:xx
- IP block: 01:00:5e.00.00.00 - 7f:ff:ff
- 1 write, only interested controllers read
- ethernet controller on recv programmed by app/IP to be interested in multicast IP address
- less inefficient than broadcast
- multicast address is DEST only, not src
- multicast is slowly replacing broadcast in protocol stack; e.g., RIP2 yes, RIP1 no

# multicast write to link group

write to 01:00:5E:01:02:03




read of 01:00:5E:01:02:03

note: A writes, B, C, read,  D ignores

# contrast multipoint (atm) and multicast on link

- multicast is more scalable in terms of N to N fanout between senders/receiver
- plus we write TO an address, not to a set of predetermined (phone/NSAP) numbers
- network must make the notion of writing TO an address working
- this is an interesting proposition for both local LAN and WAN routers (net-level)

# multipoint: N to N connections

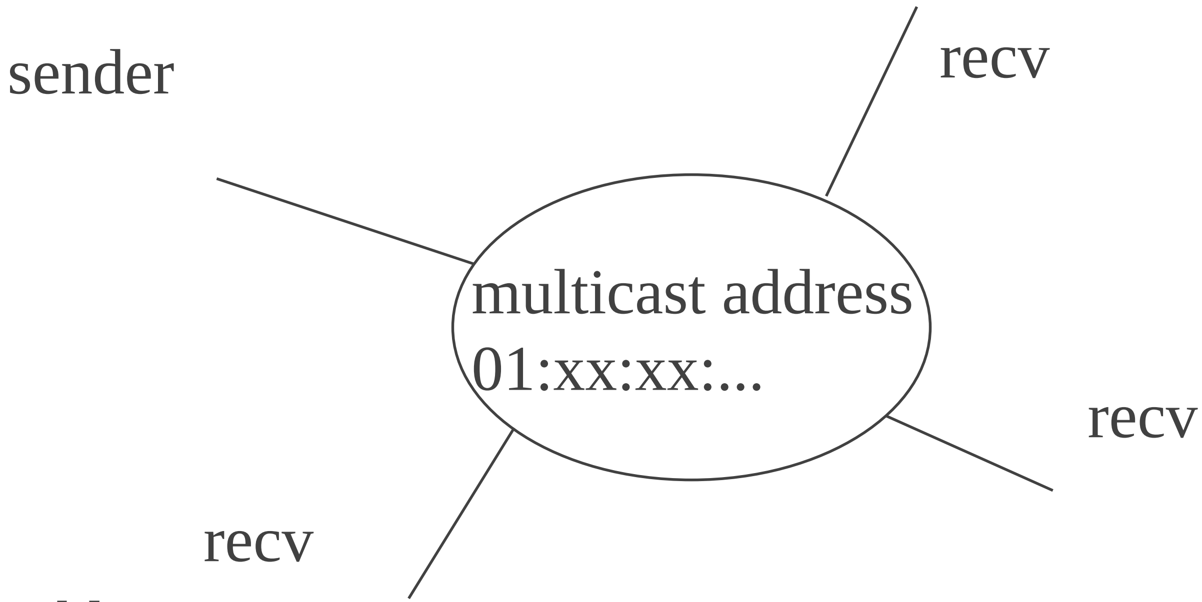
| # of participants | # of connections |
|---|---|
| 2 | 1 |
| 3 | 6 |
| 4 | 12 |
| 5 | 20 |
| 6 | 30 |
| N | (N**2) - N |

# conclusion/s

- multipoint at N**2 not as scalable as multicast
- multipoint - need to know recv addresses a priori
  – multicast - just write to G address
- counter-assumptions include :-...:
  – 1. we know how to do multicast routing?!
  – 2. we know how to do IP/isochronous data?!

# multipoint - multicast

- with multicast everybody just writes to the multicast address - don’t need to know recv addresses beforehand



# IP multicast address mapping

- IETF reserved one block of IEEE multicast addresses, used half of them
- map enet to IP how?
- take 23 bottom bits of IP address, thus 32 ip addresses map to same hw address
- 224.0.0.1 -> 01.00.5E.00.00.01
- x.128.x.y and x.0.x.y will map to the same thing (ip to ethernet)

# some multicast addresses

- 1 address - 1 app or 1 function
- 224.0.0.1 - all systems on this subnet
- 224.0.0.2 - all routers on this subnet
- 224.0.0.5/6 - OSPF
- 224.0.0.9 - used by RIP2
- 224.0.0.0 - .255 - not forwarded (routing)
- 224.0.1.2 - SGI's dogfight application
- see assigned numbers for more

# more multicast addresses/MBONE

- 239.0.0.0 - 239.255.255.255 - administratively scoped
- 239.192.0.0 - 239.195.255.255 - organization local scope
- 239.255.0.0-239.255.255.255 - local scope
- local scoping intended to replace TTL-based scoping as TTL causes pruning problems for DVMRP

# multicast operation on host

- apps (or stack) must notify ip that they are interested in recv a particular ip address on an interface (per interface)
- ip notifies driver
- driver must be MULTICAST capable
  - most modern ethernet controllers are
- host joins MULTICAST GROUP out that interface
- if mcast pkt arrives, must go to all apps that want it - there exists a multicast bind(2) though

# assume linux box

- has eth0 port according to ifconfig -a
- eth0 is programmed by IP to automagically read
  - 224.0.0.1 at boot, why?   then later vat to IP for:
  - 224.1.2.3, because you want to watch the MBONE “i love lucy” rerun at 4:00 p.m
- question: if we wanted to run gated on this linux box,  what are multicast concerns?

# IGMP - multicast control on link

- IGMP - rfc 1112 (Deering, 1989)
- encapsulated like ICMP (transport in IP proto field, but considered at IP layer)
- function is to alert local link **multicast router** that host is interested in IP multicast group
- query (type=1) sent by router
- response (type=2) sent by host
- **group address** - multicast address in question

# IGMP header

| 0 3 | 4 7 | 8 15 | 16 31 |
|---|---|---|---|
| version: 4 bits(1) | type: 4 bits (1,2) | unused | 16-bit checksum |
| 32-bit multicast (group) IP address | | | |

# IGMP protocol - host report

- a process joins a multicast group on a particular interface
- if 1..n procs, one IGMP report (type=2) is sent
- IGMP report, IP ttl = 1, IGMP group address is group address, IP dst is group address,  IP src is host unicast ip
- host sends this report to report when it gets router query too

# IGMP and multicast router

- router must promiscuously hear group reports (ip dest is ANY multicast address)
- hosts don't report leaving. router must query link once it knows that hosts are interested at periodic interval
- IGMP query, IP ttl = 1, IGMP group addr = 0, dest IP = 224.0.0.1, src IP = unicast router ip address
- note: group addr = 0, means ALL G apps please

# some details

- if process wants to send/recv multicast, IP will map IP address to ethernet address and inform device driver
- router only needs to know that one host on that link is interested
- router must *somehow* do multicast routing with other routers to make ideal of writing to multicast IP address work
- ICMP errors are not generated for multicast addrs (traceroute won't work)

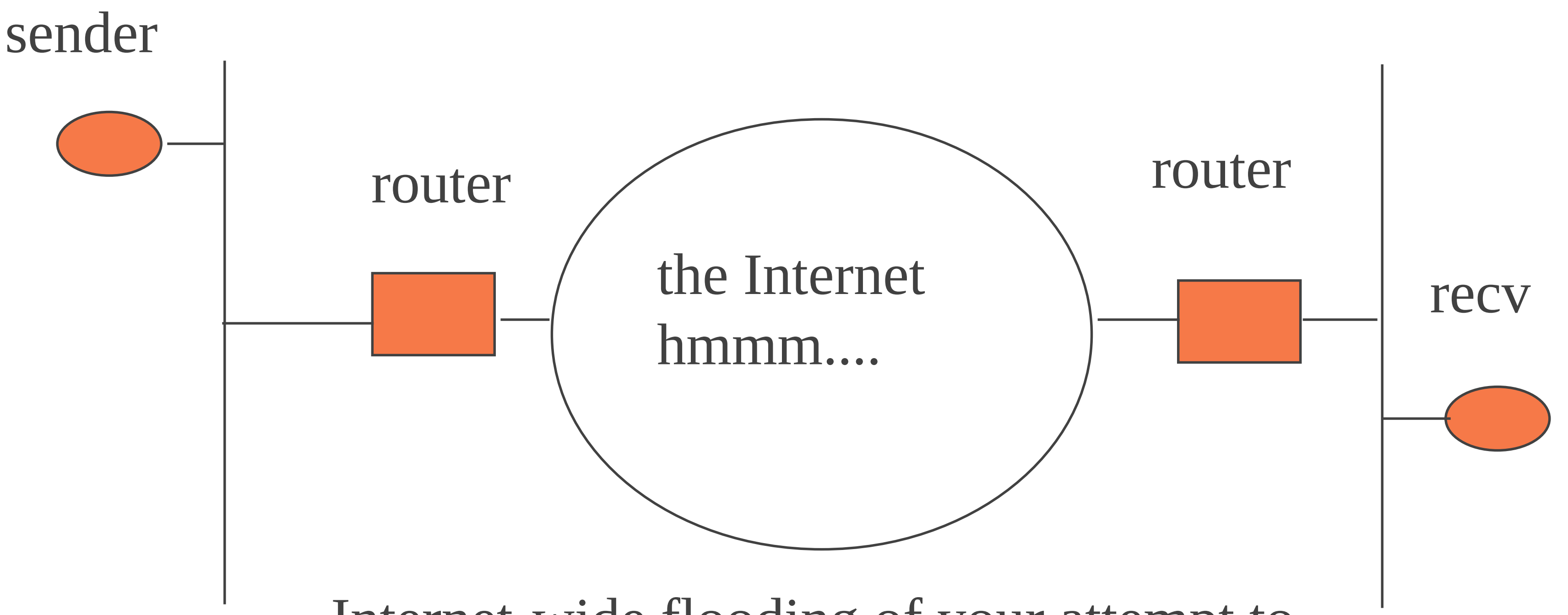
# IGMP(ng)

- defines procedures to elect one multicast router to query end nodes (lower IP wins)
- new **query** message - router can query one group as opposed to all
- **leave** group message - host xmts leave G to 224.0.0.2
- router can turn arund and send new query and discover no members present

# multicast routing on LAN/WAN

- would like model to be simple like hw model
- sender writes to multicast (IP) address
- recvs just tell network they are interested
- sender doesn't know who the recvs are (unless the recvs multicast session info to a group)
- one "small" problem, **how do the routers actually do the multicast routing**?

# multicast LAN/WAN routing




Internet-wide flooding of your attempt to "televise" the Jetsons may not be the best idea...

# DVMRP and mrouted

- multicast routing protocols exist (in their infancy) DVMRP (rfc 1075), distance vector multicast routing protocol,  MOSPF,  PIM
- problem has been that commercial routers didn't support IGMP, DVMRP,
- **mrouted** on workstations was used to construct a multicast virtual backbone - the MBONE on top of the Internet
- secret is IPIP tunnel (proto #=4)

# multicast routing protocols/biblio

- 3com white paper good (used to be):
  - draft-ietf-mboned-intro-multicast-00.txt
  - covered in Huitema, IP Routing
- OSPF book: OSPF - Anatomy of an Internet Routing Protocol, John Moy
  - good intro
- various RFCS/drafts including PIM/CBT/DVMRP, etc.

# tree-based, 2 kinds

- source based (S,G) and dense
  - more like spanning tree from send/to recvs
  - DVMRP, PIM dense mode
    - IGPs, but DVMRP in use as “EGP” anyway
  - MOSPF (but really domain-wide)
- shared-tree of routers (sparse)
  - PIM sparse mode, CBT (core-based trees)
  - send/recv (router surrogates) find trees
  - intended as EGPs, not successful

# more

- DVMRP - distance vector multicast routing protocol
  – combines RIP like unicast routing and
  – multicast dense flooding (**flood & prune**)
- PIM - protocol independent
  – independent of unicast routing (not in the protocol like with DVMRP)
  – needs unicast routing table in implementation, borrowed from unicast routing protocol

# IGP vs EGP

- MBONE on top of DVMRP really overgrown IGP
  - V.D. protocol not good basis for scalability
- MOSPF good but IGP by definition
- shared trees intended to go in that direction but have not
- hierarchical DVMRP proposed by Deering
- research area at present
- suggestions include MBGP/MSDP, BGMP, SIMPLE, EXPRESS, and more

# DVMRP/mrouted

- DVMRP original RFC (out of date) 1075, should be later draft/RFC at this point
- commercial routers originally didn't support MBONE and DVMRP
- mrouted run on Sun workstations - was used to construct virtual MBONE on top of unicast Inet
- secret is IPIP tunnel (IP proto 4)

# DVMRP and IPIP tunnel


cross the Internet via normal routing

multicast IP packet put in unicast external IP frame

packet sent across statically configured tunnel, IP src is m1, IP dest is m2

when M2 gets it, strips the outer IP, “forwards” the normal IP normally

this allows us to “bridge” over arbitrary Inet topology

# IPIP encapsulation

| | | |
|---|---|---|
| outer ip header<br>ip dest = m2,<br>ip src = m1 | inner ip header<br>ip dest = group | data |

tunnel, is point to point virtual link to mrouted
we forward multicast data across it

# DVMRP overview

- source based tree scheme (S,G) - each source/group a different tree in mcast routers; e.g., (S1, G1), (S2, G1), (S2, G2) different
- need multicast and unicast routing table
- infinity = 32
- contains RIP like unicast routing: UPDATE message with (netmask, subnet, metric)
- plus “flood and prune” protocol for multicast DM routing + virtual IPIP interfaces

# overview

- use IPIP tunnels as virtual interfaces to glue together pockets of dense mode DVMRP
- mrouter thus has native dense mode i/fs and tunnel i/fs
- multicast packets are flooded over both kinds of i/fs
- in one port and out the others (roughly)
  - more detail on that RSN

# multicast routing ideas:

- protocol basis is: “**flood and prune**”
  - src packets flooded over entire tree of multicast links to recvs
  - pruned back from leafs (removes sub-trees with no recvs) to constrain flooding
  - periodically flood again in case of bugs or new recvs or for general redundancy
- mcast algorithm: **Reverse Path Forwarding** in mrouters

# 1st: the flood part

- our ideal is a S-based spanning tree, but that
  - is not flooding ... flooding is messier
- we must constrain flooding else use up too much in the way of resources
- 1st assume basic flooding
  - flooding must occur occasionally (constrained by RPF and other mechanisms though)

# why flood?

- 1. links may change (up/down)
  - unicast routing knows, but mcast routing does not
- 2. end recv (actually end mrouter) may not know S -- what does it do?
  - remember mcast data pkt has (S,G) in it, S is IP src, G ip dst
- 3. general redundancy
  - guard against bugs
  - lost multicast packets

# how constrain flooding

- 1. use Reverse Path Forwarding algorithm for multicast routing in MROUTERS
  - do not send packets out unless M pkt comes in on shortest unicast path to S
  - do not send packets to peer/neighbors who we know (from unicast hints) have a shorter path to S
- 2. IGMP driven upstream pruning of tree - removed branches with no Receivers

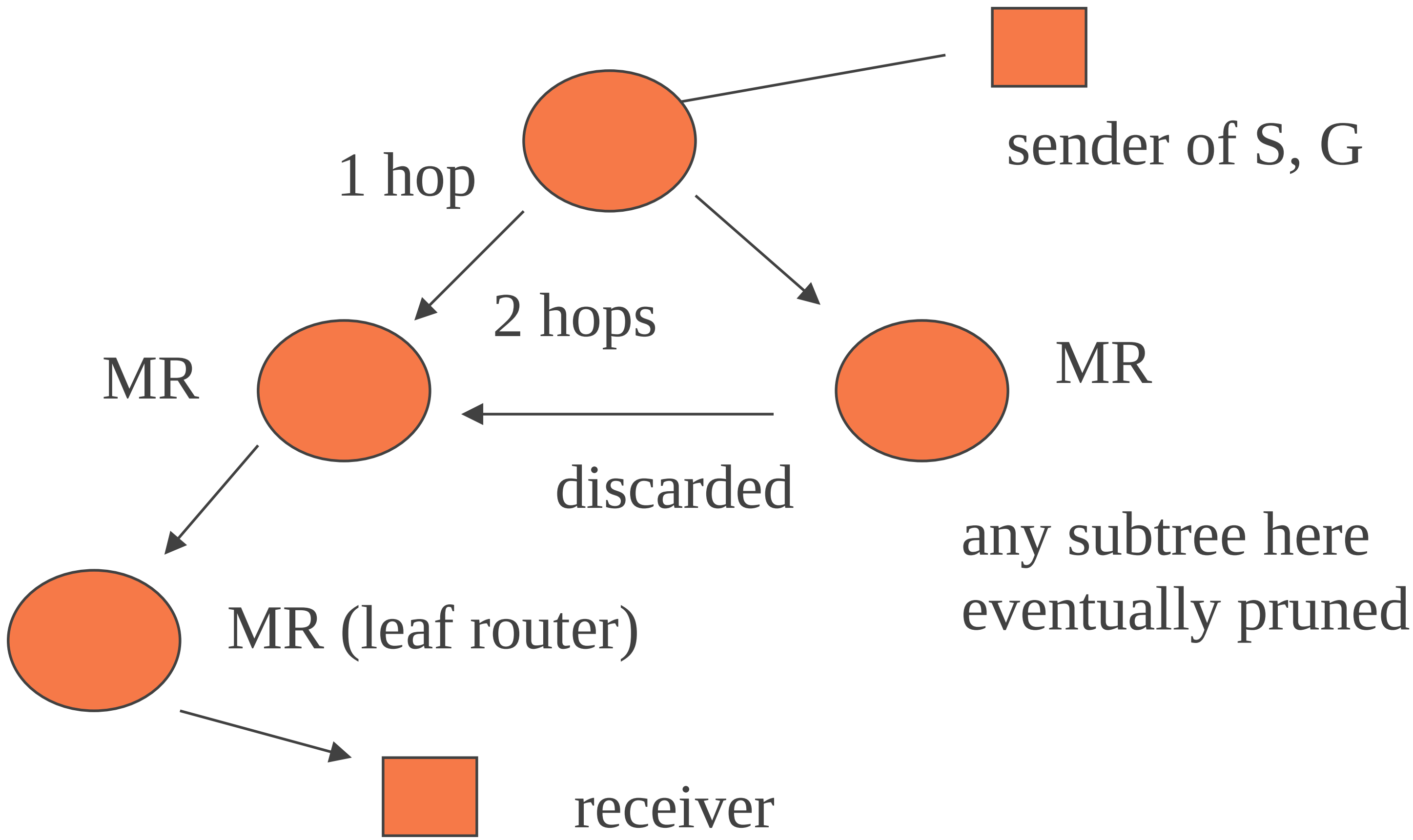
# mcast forwarding algorithm

- decrement IP ttl, if 0 silently discard
- look packet up in mcast table by (S,G)
  – if not there, discard
- if packet acc. to unicast info came in on shortest path to SRC, forward (else toss)
- packet is forwarded out listed i/fs in mcast entry (tunnels or ethernet i/fs)
- if may have TTL threshold: toss if TTL less

# ttl threshold idea

- can set ttl threshold on mcast i/f
- outbound M packets are discarded if their TTL < threshold
- this can give local admin scoping
  - local multicast can't leave ...
- some MBONE apps can be roughly controlled with this (let in X, discard Y)
- causes problems for pruning - it's a hack, jack
- IPv6 multicast scope bit a better idea

# S,G and router tree diagram

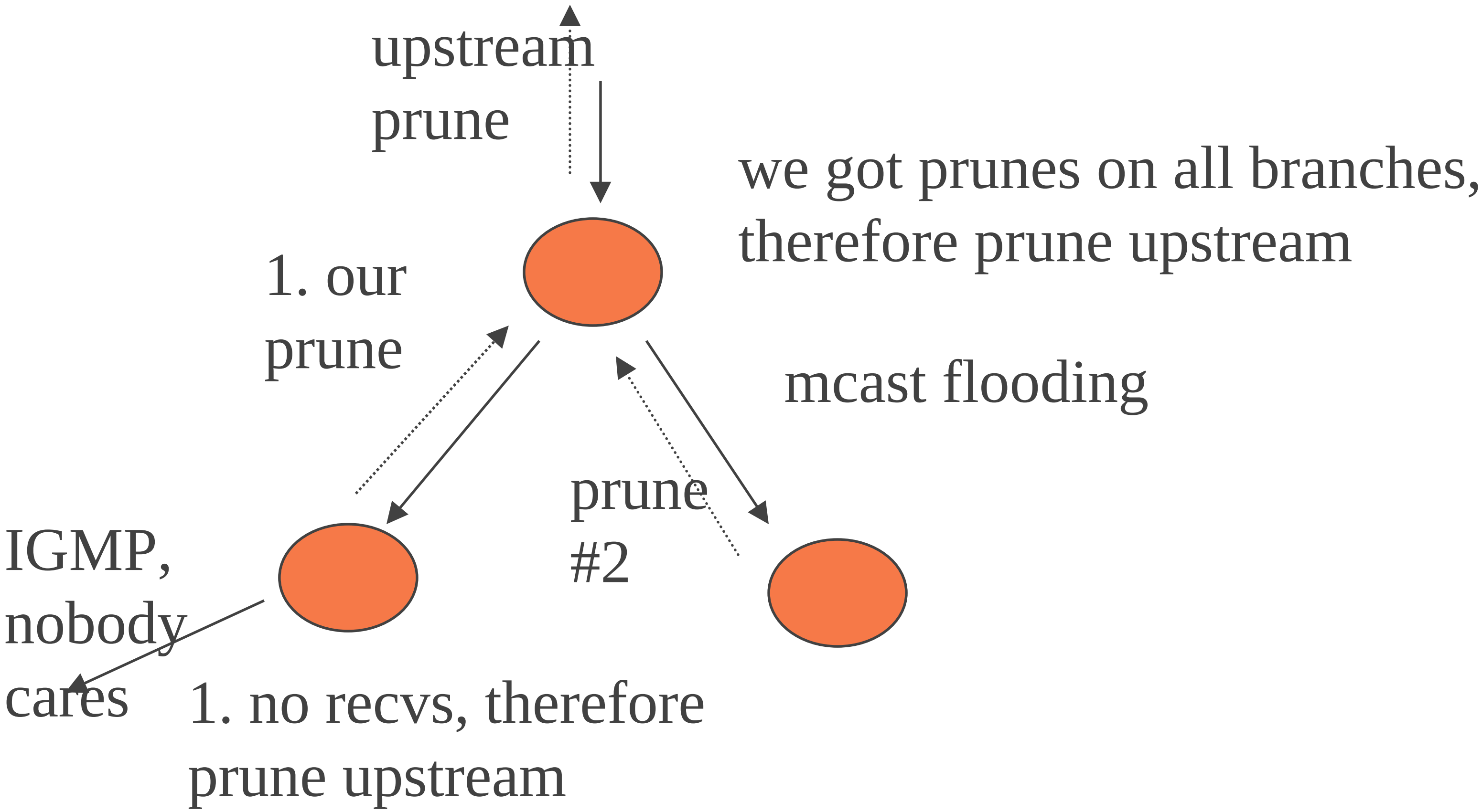



note: RPF pushes towards spanning tree, S to tree of receivers, flood will undo it

# pruning

- IGMP used to learn no group members by leaf routers
- prune messages go upstream
- if upstream router's child interfaces are pruned, it should send a prune back SRC path
- can GRAFT now too - downstream router can send GRAFT upstream
  - grafts are for undoing prunes

# prune prune prune - no jokes about fiber please



# prunes make us

- stateful - we remember this for awhile in order to prevent the flooding
  - must memorize “not that G” for a time period
  - must timeout to guard against mistakes
- if R in pruned tree, must GRAFT to remove the prune
- GRAFTs are reliable (ACK) between mrouters in order to get rid of prune efficiently

# DVMRP protocol

- in IGMP packets, type 0x13, sub-code used
- 1 - probe, for neighbor discovery
- 2 - report, unicast route exchange
  - multiple paths from src are eliminated
- 7 - prune, prune multicast tree
- 8 - graft
- 8 - graft ack (reliable)

# DVMRP unicast routing

- not there to do unicast routing BUT to
- advertise (and determine paths) to multicast sources
  - choose shortest equal-cost path
  - can get rid of equal cost multipath, one more anti-flooding technique
- "source" of RPF unicast information
- split horizon/poison reverse, hold down used

# PIM

- protocol independent in two forms, sparse and dense
- IDMR IETF wg - Inter-Domain Multicast Routing workgroup
- does not rely (unlike DVMRP/MOSPF) on any particular unicast routing table
- but must have a unicast routing table to use
  - therefore implementation dependent
  - call it "DIM" dependent implement ... sorry

# PIM dense mode

- similar to DVMRP except no unicast routing built-in
- must use local unicast routing table
- dense means group members should be many
- RPF and flood and prune is basis
- IP protocol 103

# PIM dense mode protocol

- Hellos
- Join, Prunes, Asserts
- Graft and Graft Ack
- if two routers recv new S,G packet, use PIM Assert to compare metrics
  - only smaller metric/router will forward
  - thus data-driven equal-cost path removal
- leaf removal - if no hellos from link, and no IGMP, you can start pruning

# CBT - core based trees

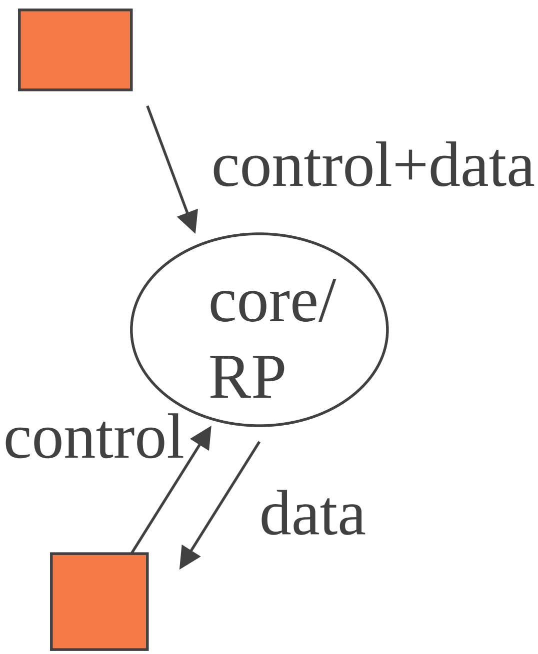
- Ballardie proposed core based tree
- S/R (router surrogates) forward JOIN commands towards center to setup
- center-based path
- multicast info then flows along that path

# CBT/sparse trees pros/cons

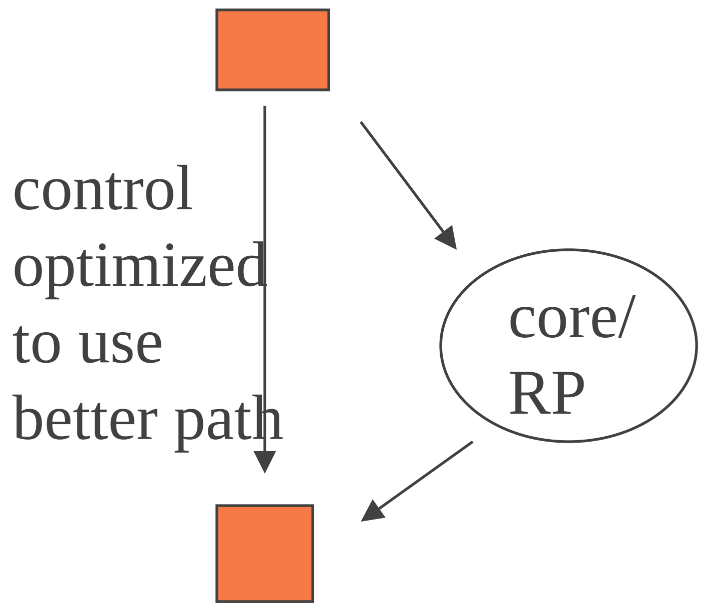
- pros:
  - minimize if not eliminate flooding, if you don't care you don't see multicast, therefore more scalable
  - minimize state in routers, only need G, not S,G
- cons:
  - multicast data path may be sub-optimal
  - may concentrate multicast routing on a few links, not spread it out as flooding does
  - how do we find core, especially inter-domain?
  - single point of failures (e.g., RP in PIM DM)

# unidirectional vs bidirectional trees

1. send to core to join S/R




2. bi-directional




hence whines about SM may be alleviated

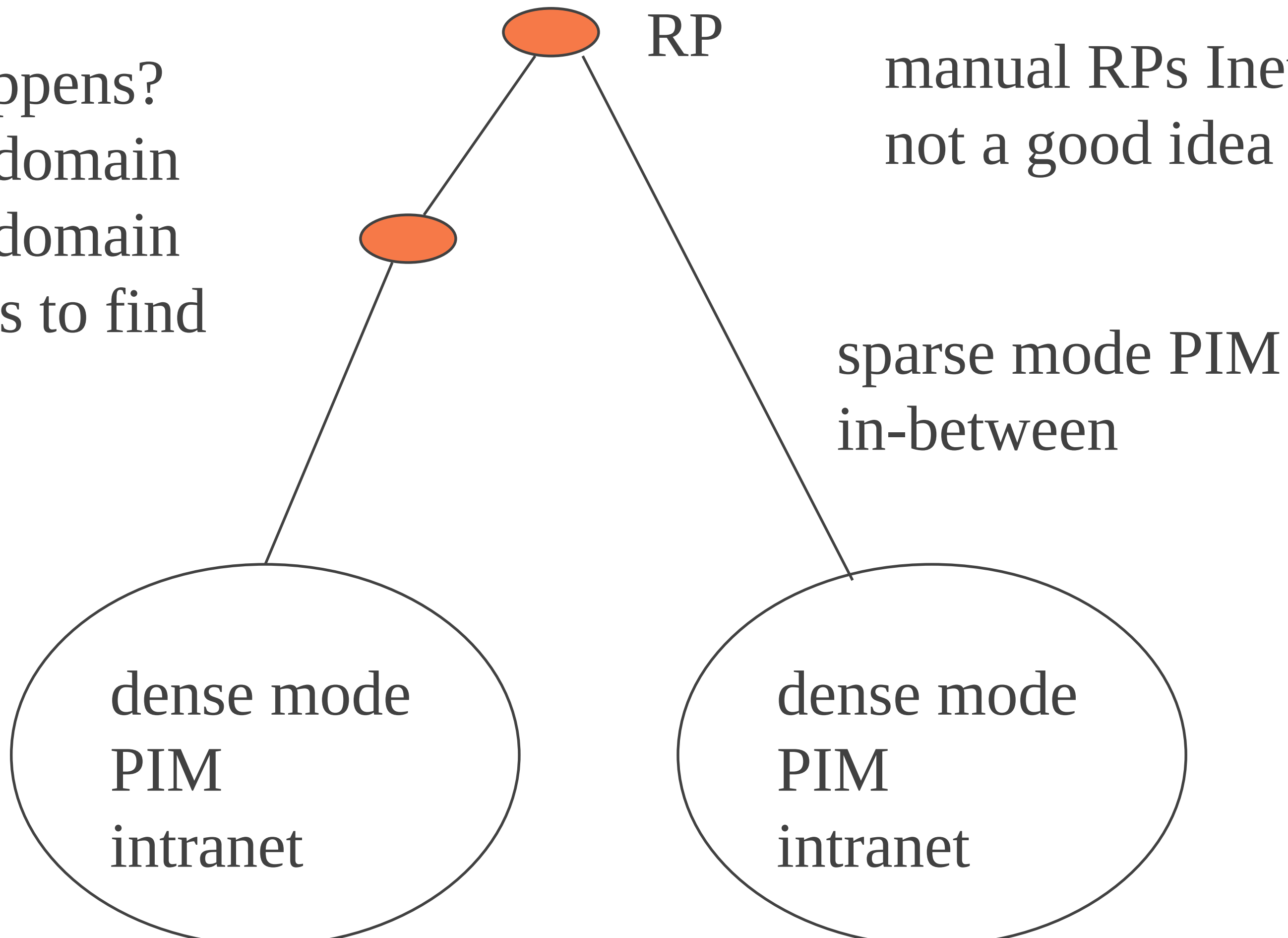
# PIM sparse

- basic idea:  we have RP, rendezvous point
- messages are forwarded to RP to join G,
  - we know RP's unicast address, send to it
- RP manually setup,  may be routing protocol mechanisms to dynamically learn
- hence unidirectional tree mechanism
- can switch from SM to DM when/if decide that group is dense enough

# intra-domain PIM maybe like so?

what happens?
1. inter-domain
2. intra-domain
recv tries to find sender?

manual RPs Inet-wide
not a good idea



# MOSPF

- multicast ospf
- introduces group-membership-lsa
- simply flood G information in OSPF multicast mesh
- tree produced by Dijkstra calculation when multicast packets arrive (data-driven)
- interactions between MOSPF and DVMRP have been defined (else couldn't fit in MBONE)

# mtrace - multicast traceroute

- Bill Fenner/Xerox Parc,  mtrace utility for unix, and elsewhere
- like traceroute, but not same mechanism
- path traced backwards from this host (assume recv.) to src,  uses RPF idea
- collects valuable packet statistics and in general is richer than traceroute in info
- must have mrouters here to there (of course)

# mtrace, cont.

- by default (no params) traces G 224.2.0.1,
  - which is MBONE audio channel
  - host (recv) defaults to you
- # mtrace <src (unicast)> <group>
- uses IGMP packet types
  - 0x1f -traceroute request
  - 0x1e - traceroute response

# how it works (in overview)

- assume RPF like algorithm
- traceroute packet forwarded hop by hop towards source
- each intermediate ROUTER sends data to sender on the way
- stops when we get the ROUTER next to src OR lose the path OR router that doesn’t understand

# MBONE

- experimental MBONE established 1992
  - broadcast IETF sessions (still does)
- not production service
  - MBONE does not (cannot) go last mile
  - scalability problems in routing
  - not universally supported by ISPs or local net
- core uses DVMRP, but MOSPF and PIM used at edges

# problems with MBONE

- MBONE has 1000's of nets, 1000s of routes in DVMRP routing table - wasn't meant to scale to that degree
- need hierarchy - Deering, etc., have proposed how to do hierarchical multicast routing
- reliable data flow is a good question too
- MBONE apps are steady-state flow, and don't back off like TCP (how can a steady-state backoff?)
- inter-domain multicast flow management
  - Big Pipe 1 doesn't want to source Big Pipe 2's receivers for that HDTV multicast session

# MBONE apps

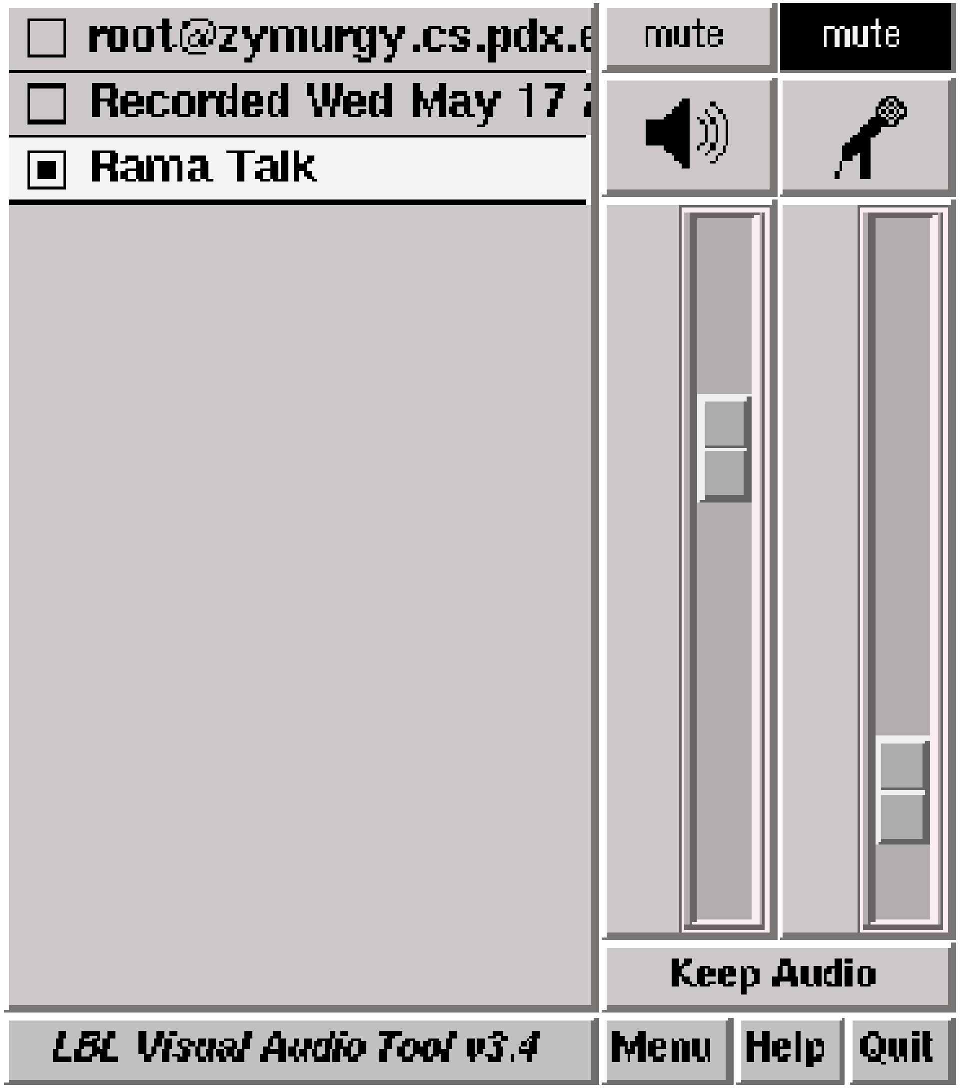
- sd - session directory (now **sdr**)
- audio
  - **vat** (PCM at 78kbps, GSM at 17kbpm)
  - nevot
  - rat also possible
- video
  - nv (video at 128kbps), nv out of service
  - **vic** (nv replacement) - commonly used
- imm - reliable image multicast (disappeared)

# sd - Session Directory

# nv - network video


note: great GUI failure

# vat - Visual Audio Tool



# recent developments in MBONE/multicast protocols

- no EGP, and basically
  - DVMRP/IPIP
  - PIM dense/sparse not scaleable enough
  - also problems of mapping one routing protocol to another (I have MOSPF, you have DVMRP, we want to share)
- recent developments include
- MSDP/MBGP
- BGMP and Perlman's Simple Multicast

# problems again were?

- no real good way to send info across routing domains
  - not as scalable as unicast routing
- consider PIM spare/RPs
  - if only one RP - can have single source of failure
  - how do we find RPs elsewhere? **inter-domain** as well as intra-domain

# MSDP/MBGP

- BGP has been made multi-protocol; i.e.,. can do more than IPv4
  - IPv6/multicast group info come to mind
  - therefore you can get multicast info from a BGP peer
- MSDP - Multicast Source Discovery Protocol
  - inter-domain RP to RP flooding of source “activation” messages using TCP

# with MSDP

- sources must be next hop towards sending RP
  - this is an RPF check, makes us loop free
  - MBGP table used to determine RPF check
    - » (S,G) check (is src best hop else discard info)
  - MSDP/MBGP pairing give us inter-domain info but
  - really limit trees to intra-domain, not inter-domain
    - » PIM cannot function across domains
    - » no way to have bi-directional tree
- thus, this combo viewed as stopgap, not scalable enough
  - of course ...

# BGMP - nextgen inter-domain?

- border gateway multicast protocol
- in draft status
- builds bi-directional trees between routing domains
- operates between border routers
- cross domain, inside domain may use DVMRP, PIM, MOSPF

# BGMP basic idea

- border routers learn there are internal hosts that want to send/recv
- send join messages to “root domain” of multicast group
- given a multicast address,  how do we find “root domain”, eh?  (some kinda DNS?!)
- must invent multicast address allocation scheme to figure that out (MASC)
- multicast address set claim protocol

# another possibility: Perlman - simple multicast

- Perlman/Lee/Ballardie (CBT)/Crowcroft/Wang/Maufer propose
- Simple Multicast (IETF draft)
- propose to NOT have a MASC-like protocol
- rather use address 2-tuple (C,M), where C is unicast address, M multicast
- routers do not have to somehow figure out where M is (use C,M) instead and C is not multicast
- question for R becomes, how do I find C (use the web?)

# multimedia application programming

- apps may be multicast
- send audio/video, use udp
- may be unicast
  - broadly consider voice over IP here
  - streaming media
- control (stop/fast-forward e.g.,) may use TCP

# common app traits

- use UDP and send constant stream
- low bandwidth because T1 has been a bottleneck (now DSL/cable-modem?)
- not reliable
- data needs to be sequenced and there needs to be timing information - RTP protocol provides sequencing/time info/format

# common encapsulation scheme

| | | | |
|---|---|---|---|
| ip | udp | rtp | a/v data |

# some examples

- audio (usually not a fat  bit stream)
  - pcm/telephone “simulation”
  - real audio
- video (typically compressed)
  - streaming media, 3 common formats
    - real/microsoft/apple quicktime
    - real video at 160 kbits - one example
  - H323 ITU spec - voice/video conf
    - netmeeing (usoft)/polycom

# upper end

- has tendency to be MPEG based
  - e.g., “fat” H323 stream < T1 (1.544)
  - Mpeg2 stream might be 2-16mbits
    - » note: compressed by definition, NTSC uncompressed is how big?
  - cisco ip/tv is example
- HDTV is out there
  - UW experiments over Inet2 > 100mbits with some streams
- obviousally compression is important

# application model/s

- on order of
  - 1 stream for voice/UDP
  - 1 stream for video/UDP
  - 1 or more streams for control
    - » use TCP/HTTP ...
- some media formats MAY combine voice/audio (e.g., MPEG)
- note a/v data may/may not be compressed

# error handling

- by definition, we face packet-loss
- by definition, we may not do packet resends
- packets may be out of order  recv sees N+1, N
- recv may need 2-way "realtime" connectivity, hence 2-way delay may be importantf compared to 1-way only
- we may also have **IP jitter** (layer 2 jitter exists, but can be ignored in the face of layer 3 jitter, unless you are phone co)
- jitter -- too much time variation between two packets in an a/v stream
  - could cause "wow/flutter" in playback at recv

# error fixups

- protocol/programming-level only here
- recv has buffer of certain size
  - jitter buffer - reorder packets to introduce fixed delay between them, before playback/decode
  - resequence if possible acc. to timing restraints
  - may have to drop however
- one can recover bits or even packets depending on the level of redundancy used
  - e.g., there exist FEC, forward error correction schemes that can lead to bit fixups

# causes of jitter

- sending host introduces time delays
  - OS scheduling
  - file i/o if file playback
  - heavy network traffic processing
- same for recv host
- router and switching Queues in intermediate systems layer2/layer3
- some researchers think we need multimedia OS in addition to “QOS” in network

# Real Time Protocol/RTP

- RFC 1889, also ITU standard
- basically provides
  - sequence #, 16 bit
  - timestamp/and some kind of code to, 32 bit
  - id AV format
  - note companion protocol RTCP; i.e.,

# RTCP - RTP control protocol

- defined in same RFC as RTP
- control - not data
- if RTP port N used, RTCP is N+1
- senders may send timestamps or user info
- e.g.,Sally Smith is sending this stream
- recv might send stats/error info/jitter state
- info can be app specific

# RTSP

- real time streaming protocol
- RFC 2326
- IETF VCR like control protocol
  - stop/start/ff, etc.
- streaming media oriented

# application security

- if pt. to pt., no different from any other transport protocol
  - encrypt stream has been done in past, but authentication here is a good idea too
  - use IPSEC/ssh/ssl ...
- if multicast
  - open problem, protocol proposals exist
  - suggest consider group size ...
  - a group of 1 million is not secure by definition

# research

- lots of QOS work - open problem! e.g.,
- integrated services - provide some quality of service and soft v.c. in network layer on Inet
  - routers must schedule packets acc. to type
  - RSVP protocol for reserving “circuit”
  - RFC 1633 - Integrated Services Proposal
- mcast inter-domain routing AND management

# other QOS schemes

- not use end to end (RSVP) but simply
- limit QOS over IP to local internet/intranet
- IEEE - 802.1P
  - IPv4 priority-like bits but in MAC header portion
- IETF - diff-serve WG
  - use IPv4 tos bits explicitly
- need queuing schemes in routers/switches

# QOS strategies

- ATM gives harder (less mushy) form of QOS
  - but limited scalability
  - and not likely to go to end system or lan
- RSVP, softer form, end to end
  - but not ultimately scalable in core routers
  - requires host sw mods
- IEEE/diff-serve simplest and softest
- **BIG PIPES remain a good idea**
  - **gigabit ethernet to the doorknob?**

# research

- compression in audio/video (especially) formats - always need to squish video
  - MPEG
- do we reconstruct the entire o.s. and stack for multimedia?
- engineering (open systems ...) of management tools for both unicast/multicast