"Subnetwork Dependent Sublayer"

- We’ve discussed the first piece of the network layer, including addressing and data packet formats.
- Now we’re about to talk about the second piece, which I’ll call neighbor handshaking.

\[
\begin{array}{c|c|c|c|c|c}
E & X & \alpha & \beta & \delta & \varepsilon & Z \\
R1 & R2 & & & & & \\
\text{rest of network} & R3 & \\
\end{array}
\]

**ES-IS**

- Remember, can’t tell from an address what LAN something is on
- Routers discover CLNP address of endnodes because endnodes periodically multicast an ES-HELLO, to “all level 1 ISs”
  - network layer address(es)
  - holding time
  - data link address (from data link header)
- Endnodes discover routers and monitor their aliveness because routers periodically multicast an IS-ES-HELLO to “all ESs”
- Endnodes discover better paths through Redirect messages “To send to network address Q, send to data link address \( \pi \).”

**Endnode Algorithm**

- Keep Router cache based on receipt of IS-ES Hellos
- Keep Destination Cache based on receipt of Redirect messages
- If transmitting to destination \( Z \), if \( Z/\rho \) in Destination Cache, send to DL address \( \rho \)
- Else, send to any router in Router Cache
- Else (no router in router cache, \( Z \) not in Destination Cache), multicast data message to “all ESs”
- If receive data packet to multicast address, then send a Redirect (it’s actually an ES-Hello) to the DL source address in the packet

**IP Equivalent**

- Decide (use own IP address, destination IP address, and mask) whether dest on same LAN
- If on same LAN, need to know DL address. Do that with ARP protocol
- Broadcast ARP Request with IP addr of dest
- Everyone receives it, throws it away if it’s not for them. Destination replies with ARP Reply.
- Keep ARP cache
- If not on same LAN, send to a router. How do you know a router? -- kind of a hole in IP. Various ad hoc methods
  - Configure address of at least one router
  - Assume routers talking RIP, listen for RIP msgs, assume transmitter is a router
  - BOOTP, DHCP, ICMP router discovery
**AppleTalk**

- Same as IP, but don’t need mask because network/node boundary is always in same place
- AppleTalk believes in autoconfiguration
- Find a router by listening to RTMP (AppleTalk equivalent of RIP) transmissions, or do RTMP request and get RTMP reply from router(s).
- Find own network number (range) with ZIP GetNetInfo packet sent to router
- Find own node number by picking one at random and using AppleTalk’s version of ARP (called AARP) to see if anyone else is using it

**DECnet Phase IV**

- Like CLNP, you can’t tell from address what LAN you’re on
- Like ES-IS, endnodes advertise periodically to routers, routers advertise periodically to endnodes
- Key difference: implementors didn’t want to keep a cache of NL/DL, so instead they wanted the 6 byte DL address to be derived from the 2 byte NL address. This is done with a 4 byte constant known as “hiord”.

**Ethernet address**

```
| hiord | DECnet addr |
```

**Phase IV DECnet (cont’d)**

- Long data packet format (the one transmitted on LANs) has on-LAN flag. Packet is originated with on-LAN=true.
- Router that forwards it off a LAN clears on-LAN
- Endnode that receives a packet with on-LAN=True from source S, puts S into its on-LAN cache
- Endnode that receives a packet with on-LAN=false from source S, remembers router it received it from as being best path to S
- If no router, send on LAN (with DL=hiord.NL)

**IPX**

- Like IP and AppleTalk, you can tell from 4 byte network number portion of address what LAN something’s on
- Discover routers through RIP query
- Discover best route by querying before sending something off LAN -- best router will respond
- When something’s on the same LAN — this is the really good part — you use the 6 byte node ID as the data link address. No ARP, no ES-IS.
Comparisons

- ARP schemes: ARP bothers all endnodes. No way to eliminate this traffic.
- ES-IS: probably more packets on wire, but it’s the routers that do the work, not the endnodes. Also, it means you’re not locked into having a different address for each link. IP gets this functionality from bridging!
- DECnet way: overwriting the DL address is a kludge. Certainly wouldn’t work if two different protocols tried to make the same constraint. People get annoyed trying to manage a network when the DL address changes. But nice thing is you don’t need Redirects, since you can derive DL address.
- IPX way: no overhead from ARP or ES/IS. But constrained to have DL address=NL address.

IP Broadcast Storms

- Lots of examples
- Always silly misconfigurations or silly bugs
- Each one easy to fix (perhaps -- sometimes it involves changing code), but these things still persist
- Example: endnode E4 confused and thinks its DL address is all 1’s (broadcast address)

ICMP

- “Internet Control Message Protocol”
- Designed for error messages (destination unreachable, packet too large, bad checksum, TTL expired, source quench, echo(=ping), redirect
- IP header has “protocol” field=1, meaning ICMP
- Some ICMP messages carry “data”, which is the IP hdr+64 bits of original packet

BOOTP/DHCP

- For configuration of IP nodes
- BOOTP didn’t do TLV encoding. Just a bunch of (mostly useless, and sometimes large) fixed length fields, and a field called “vendor specific”
- You say your layer 2 address, you’re told your IP address, your router’s address, how to reach a server with your configuration file, and name of the file
- later tamed “vendor specific” field by specifying TLV encoding with registered “T” values, and a “magic cookie” initial value they hoped didn’t conflict with anything
- DHCP added concept of “leasing” IP address
Specific Routing Protocols

- Interdomain vs intradomain (EGP vs IGP)
- Intradomain:
  - RIP (IP only), IPX-RIP (IPX only)
  - AppleTalk’s RTMP (similar to RIP)
  - OSPF (link state protocol for IP only)
  - IS-IS (link state protocol designed to simultaneously route any set of protocols)
  - NLSP (link state protocol for IPX only)
  - PNNI (link state ATM protocol)
- Interdomain
  - static routing
  - BGP/IDRP (IP only/integrated, i.e. can simultaneously route any set of protocols)
  - IDPR (IP only, good approach, but it died)

RIP Example

dest  hops  port  DL add  age
A    5      1    a     2
B    12     2    c     37
X    3      1    a     15
G    6      1    d     1

- timer tick
  - increment all ages, delete row if age>90
- same info, same neighbor ===> set age to 0
- better info
  - overwrite entire row, set age to 0
- worse info, different neighbor ===> ignore
- worse info, same neighbor
  - SHOULD overwrite row with worse info

More RIP-Like Protocol Info

- IPX has two metrics, ticks (supposedly delay in units of about 1/18 of seconds) and hops. Hops counts to infinity fastest. Ticks is what you route based on. DECnet Phase 3 and 4 use the same trick (two metrics)
- IP RIP encoded for being an integrated protocol. Has “address family/destination” pairs.
- Example:

  dest  hops (ticks)  port  DL add  age
  A    5    7    1    a    2
  B    12   17   2    c    37
  X    3    3    1    a    15
  G    6    7    1    d    1

- Events: timer tick, better info, same info, worse info

IP RIP

- Two values for command: request (1), response (2)

<table>
<thead>
<tr>
<th></th>
<th>command</th>
<th>version</th>
<th>address</th>
<th>metric</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>2</td>
<td>14</td>
<td>4</td>
</tr>
<tr>
<td>2</td>
<td>reserved</td>
<td>reserved</td>
<td>reserved</td>
<td>reserved</td>
</tr>
</tbody>
</table>

use with IP
RIP version 2

- Added subnet masks, authentication, and “route tag” (“internal” or “external”)

  RIP v2
  1 command
  1 version
  2 reserved
  2 authentication=FFFF
  2 authentication type
  16 authentication info
  2 address family=2
  2 tag
  4 IP address
  4 mask
  4 next hop=0 or nbr rtr
  4 metric

IPX RIP

- First comes LAN header, with protocol type or SAP indicating IPX
- Next comes IPX header, with packet type=1 and socket=453Hex
- Next comes RIP info, with up to 50 announcements per packet

  2 bytes   operation   1=req, 2=resp
  4 net #    these three fields repeat up to 50 times
  2 hops
  2 ticks

EGP

- Makes RIP look good
- Like RIP, but don’t pass around metric. Just report what’s reachable
- Doesn’t work except in tree topology. Theoretically only 2 levels
- The first “interdomain” routing protocol. Designer claimed you can’t have a true routing protocol between domains — those guys might buy inferior brand routers or not know how to manage their routers.
- Only used for IP. No masks.

DECnet Phase 3 and 4

- Two metrics, hops and cost
- Distance vector sent reliably on pt-to-pt links, only changes sent
- On LANs, distance vector sent periodically, but not timed out by receivers
- Distance vectors from each neighbor saved, so second best path can be computed
- Path splitting implemented on equal cost paths
- Distance vector from neighbor N discarded when Hello timer expires.
- So instead of periodic RIP messages, there’s one short periodic Hello. However, on pt-to-pt link, Hello is only sent in the absence of data
- Phase IV has LAN support and level 2 (area routing)
### BGP/IDRP

- "path vector", meaning distance vector where entire path is specified. This prevents loops and allows selection of path based on criteria other than metric (policy based routing).
- Example policies: don’t route through domain X unless no other path. Don’t route through domain X when going to domain Y. Don’t ever route through domain X. Don’t tell neighbor R3 about path to domain Y.
- Routing decision based solely on destination. A serious problem with BGP/IDRP if policy based routing is really needed.
- Yes, but’s it’s better than EGP
- Choice of path based not on metric but on examination of path and configured policies (almost everyone uses fewest hops)

### BGP Configuration

- Path preference rules
- Which neighbor to tell about which destinations
- How to “edit” path when telling neighbor N about prefix P (used for adding fake hops to discourage N from using you to get to P)
- It’s possible to have set of policies that don’t converge
- It’s NP-complete (i.e., intractably hard) to look at a set of policies and decide whether they’ll converge
- A “good” set of policies can turn into a bad set if a link goes down
- We’re probably stuck with BGP forever...

### Policies

- Traffic from S1 can’t legally traverse a circle. Traffic from S2 can’t legally traverse a square (example: military, commercial). Let’s say X and D can legally carry both.
- X has choice of two paths to D, K.D or Z.D. If it chooses K.D it will make S1 unhappy. If it chooses Z.D it makes S2 unhappy.
- Really even decision based on source not good enough. What about authorized individuals? What about contents of message?

### Confederations

- Adds more levels of hierarchy. A bunch of domains can be formed into a confederation. From outside, it looks like a single thing.
- Only one hop in route, so routes shorter
- Policies apply to all domains in confederation. Configuration is easier -- if you want different policies for different domains, then hope they don’t form a confederation
- Confederations constrain route since can’t have something twice
**E-GBP vs I-BGP**

- Talking to peer within domain is I-BGP
- Talk to BGP nbr in another domain is E-BGP
- Originally, I-BGP had to be fully connected
- To improve things
  - use confederations to break domain into smaller domains (each fully connected, but looking like single domain outside)
  - use “route reflector”, star topology with BGP router at center, passing routing info to all BGP speakers in domain

**IS-IS/OSPF/NLSP**

- Very similar. OSPF designed for IP. IS-IS designed originally for CLNP (DECnet Phase 5) but since formats so extensible, it was easy to add support for other protocols (like IP). NLSP basically is IS-IS, but only for IPX
- OSPF/IS-IS wars made protocols more similar
  - OSPF LSP distribution didn’t work. So they copied IS-IS method
  - OSPF had an “authentication” field, which was a constant. IS-IS added the field, since that field obviously added so much security
- Not many ways of doing Link State Routing (that work)

**Designated Routers**

- On a LAN, if there’s a lot of information relevant to that LAN (lots of routers, IPX SAP information, AppleTalk zone info, CLNP endnodes) it would be wasteful if every router on that LAN generated an LSP giving all that information
- So instead, one router R is elected Designated Router
- R names the LAN with a name unique to that area (by incorporating R’s name in the name, say R1.25)
- All the routers on the LAN (including R1) claim to be attached to R1.25 — no other info
- R1, claiming to be R1.25, issues another LSP, giving all the LAN info

**LAN LSPs**

```
R1 R2 R3 R4
R1.25 R1.25 R1.25 R1.25
```

```
R1 R2 R3 R4 R5
R1.25 R1.25 R1.25 R1.25 R1.25
```
Designated Router Election

- Election is based on configured priority, with ID breaking ties
- IS-IS was designed to be deterministic (same set of routers up results in same DR)
- OSPF designed to be as stable as possible (once you’re DR you stay DR unless you crash). Don’t “run” unless there’s no DR. Priority only matters when the DR dies and there’s a new election.
- NLSP allows customers to configure things to get either effect. Once you become DR, your priority changes by a constant.

Broadcasting LSP Reliably on LAN

- OSPF way
  - if have LSP to send, don’t multicast it to “all routers”. Instead multicast it to “DR”. It’s multicast rather than unicast so backup DR can hear it too (and keep track of acks)
  - each router that hears it sends ack
  - after 10 seconds, if a router hasn’t ack’d, DR sends individual copy to that router
- IS-IS/NLSP way
  - if have LSP to send, multicast it
  - no acks. Instead, DR periodically (every 10 seconds) sends CSNP, summarizing LSP database
  - if detect discrepancy, transmit or request info

Hierarchy

- IS-IS, OSPF, and NLSP all use the word “area”. LSPs propagate within an area.
- Could have rigid hierarchy (level 1, level 2), or completely flexible. Routers need not know how many levels above or below them. Address summaries are advertised up in hierarchy. Default (or more specific if necessary) can be advertised down.
- OSPF has a rigid 2 level hierarchy. IS-IS has a flexible hierarchy designed for level 2. Level 1 was designed for CLNP routing to ID.

IS-IS/NLSP Packet Types

- Hello: router neighbor discovery
  - pt-pt
  - LAN
- CSNP: summarizes LSP database by just giving (source, sequence number, age, checksum)
- PSNP: acks one or more LSPs
- LSP
  - level 1 (deals with ID portion of CLNP address, deals with address prefix for IP)
  - level 2 (deals with address prefix for CLNP or IP)
  - Note that if not used for CLNP, level 1 and level 2 routing for IS-IS is the same
Hierarchy with IS-IS/NLSP

- Filtering: don’t import/export addresses of the form xyz*, or only import/export specified addresses
- Summarization: report the address summary xyz* (assuming at least one thing matches)
- Exceptions: if something not in summary, should I import/export it?
- Case 1: R runs two instances of IS-IS. What it learns from one area is imported into the other
- Case 2: treat pt-to-point link, R1, and R2 as mini-area

IS-IS Hierarchy, cont’d

- Can have as many levels as you want
- A router isn’t aware of how many levels of hierarchy are below or above it
- Disadvantage: with very complex topology, you get distance vector type count-to-infinity behavior.

Recent IS-IS

- Current WG doesn’t like complete flexibility, because of potential count-to-infinity
- Spec allows, but nobody implemented hierarchy for IS-IS
- WG will redo hierarchy, probably with 8 levels, and one “cross link” allowed (routing info flows up, across at most once, and down)
- Original IS-IS had 6-bit metric
- New metric 4 bytes

OSPF Hierarchy

- Two/three levels of hierarchy
- There are areas, and there’s a single “backbone” area that hooks all the other areas together
- There’s also the assumption of an “interdomain” routing protocol that gives information about IP destinations in other ASs
- So there’s IP destinations in an area, in the set of areas in your AS interconnected through the backbone area, and in other AS’s
- Routing information (LSAs):
  - types 1 and 2 correspond to IS-IS LSPs (1 is an ordinary LSP, 2 is issued by DR on behalf of the LAN)
  - types 3, 4, and 5 are external info
**OSPF Type 3, 4, and 5 LSA**

- area
- AS
- = area border router
- = AS border router
- = IP destination in AS, outside your area
- = IP destination outside your AS

---

**Other OSPF/IS-IS Differences**

- IS-IS puts as many destinations in LSP fragment as fit. OSPF makes a separate (3, 4, or 5) LSA for each destination. OSPF can fit lots of small LSAs into a single packet
  - incremental change smaller with OSPF since only a single destination
  - OSPF database 3 to 6 times as big (depends on how many metrics), because need sequence #, age, etc for each dest. More memory, more bandwidth for periodic info
- LAN LSP propagation: CSNPs vs ACKs
  - CSNPs (IS-IS way) wins if “lots” of routers
  - ACKs (OSPF) wins if “few” routers
  - CSNPs more self-stabilizing

---

**OSPF vs IS-IS (cont’d)**

- Synchronized parameter settings
  - IS-IS allows migration a node at a time
  - OSPF neighbors must agree (Hello and Dead Time, Authentication password, Stub area flag)
- Database Overload
  - IS-IS treats overloaded IS as an ES
  - OSPF doesn’t say
- Partitions
  - IS-IS solves area partition problem with automatic mechanism. OSPF doesn’t.
  - OSPF solves “level 2” partition problem with human intervention, provided all level 1 routers are level 2 routers

---

**Multiprotocol Routing**

- Do we need a separate routing protocol for each shape of packet and size of address?
- Integrated IS-IS has routers compute tree of routers, and then LSPs contain info “these are protocol X destinations I can reach”
- “Ships in Night” approach has separate routing protocols for each NL
  - each protocol duplicating work (like router to router Hellos and computation of router tree)
  - each protocol needs to be separately configured
  - more work for implementers than just sum of protocols, since interactions have to be worked out
- Brouter also acts as bridge (on “other”)
Address Prefix Routing

- Hierarchical routing requires routing to longest matching address prefix
- How do we do that efficiently?
- Assume our forwarding table contains the following destinations.
- Given a packet with a destination address, we need to find the item in our forwarding table that is the longest match to the dest address.
- Two algorithms: modified binary search, and TRIE.
- Items in forwarding table: A, ABC, ABCDEF, ABDQ, AC.

Binary Search

- A, ABC, ABCDEF, ABDQ, AC: Pad each with 0’s and with 1’s. Each is 2 entries.
- ranges: (A (ABC (ABCDEF ABCDEF) (ABDQ ABDQ) (AC AC) A)
- Backpointer to next shorter match after 1’s

```
( A00000000000000
 ( ABC00000000000
 ( ABCDEF11111111
 ( ABC111111111111
 ( ABCDEF00000000
 ( ABC111111111111
 ( ABDQ000000000
 ( ABDQ1111111111
 ( ABDQ000000000
 ( ABDQ1111111111
 ( AC00000000000
 ( AC00000000000
 ( AC111111111111
 ( AC111111111111
 ( A111111111111
 ( A111111111111
 none
```

Binary Search

- Character by character search (A, ABC, ABCDEF, ABDQ, AC).
- Organize data as tree. * means match.
- Remember last time * seen.

```
{ }

A

A*

B

AC*

C

D

ABCD

AB*

ABD

ABDQ

ABCDE

ABCDEF

E

F

G

H

I

J

K

L

M

N

O

P

Q

R

S

T

U

V

W

X

Y

Z

```

TRIE

- Character by character search (A, ABC, ABCDEF, ABDQ, AC).
- Organize data as tree. * means match.
- Remember last time * seen.
Finding Things

• Register in Directory Service. If someone wants to find you, they’ll look there.
• Broadcast a query to find someone, get a response. (call everyone. “Are you Radia?”)
• Advertise your presence. Someone that wants to find you will hear your periodic advertisements. (call everyone saying, “in case you’re ever interested, I’m Radia”)
• AppleTalk and IPX have two interesting mechanisms

AppleTalk

• A zone name is an ASCII string, which is like a directory name.
• It’s assigned, on a per LAN basis.
• Each LAN can have up to 256 different zone names.
• A zone can be on multiple LANs.
• Zones are completely independent of topology
• A name is registered in a zone
• When someone wants to find something, they search a specific zone
• Routers have to know which zones correspond to which LANs so search pkts reach every LAN on which that zone appears
• Legal zone names for a LAN are configured into “seed router”

AppleTalk, cont’d

• To choose a (zone, name) pair, you are presented with a list of zones for your LAN. You choose one, and then choose a name.
• But the name might already be in use.
• In a method similar to choosing an AppleTalk NL address, you choose a name, then “search” for it. If you get a response, you choose a different name.
• To search you send a special (NBP (Name Binding Protocol) Broadcast Request) pkt to your router. Your router (through ZIP protocol) knows mapping of zones to LANs.
• Your router converts your BrReq into n copies of a FwdReq pkt, one for each LAN in the zone

AppleTalk, cont’d

• A router on each target LAN converts the NBP FwdReq into an NBP Lookup pkt which gets multicast on the LAN.
• Every node with a name in that zone name examines the Lookup pkt.
• If it owns the name it sends a response to the original requester
• What DL multicast address is used:
  - AppleTalk has claimed a set of multicast addresses on each type of LAN. For instance, on 802.5, AppleTalk uses all 20 customer definable functional addresses for this purpose!
  - hash zone name. Get integer.
  - If AppleTalk has claimed k addresses on that LAN, take the hash mod k, and use that add.
AppleTalk Zones

- A says “search Z1 for ‘Radia’”
- R1 sends pkt to LANc, LANd, LANe, LANf
- “Radia” responds to A

IPX SAP

- Service Advertisement Protocol
- Like RIP, but “destinations” are service names, up to 48 characters long
- Like RIP, services are advertised every 60 seconds (not just by original service, but by routers)
- This can take a lot of bandwidth
- Filtering helps
- NLSP helps a lot. Rtr continues to listen to SAP and answer SAP queries, but it doesn’t periodically broadcast SAP. Instead one router puts a particular service name into its LSP, and the others find out about it, once, reliably
- Encouraging moving most applications to NDS (directory service), with maybe SAP for finding NDS

Filtering SAP

Suppose you want to filter, but still give authorized users access to everything

- R1 filters all but S1.
- X can log into S1 to find other services

Multihop Multicast

- LAN multicast is a byproduct of the technology
- It’s dangerous on a LAN (IP broadcast storms)
- LAN broadcast storm disables the entire LAN, and is a reason for customers to switch to routers (a router between two LANs blocks the storm — a bridge doing its job merges the LANs into a single LAN
- Notice one application (IP) designed by good people, with years to debug things, and with the entire wisdom of the IP community to help, still suffers from storms
- Is it wise to open WAN multicast to any application? Any application with a bug can bring down, not just a LAN, but your entire network
Multicast Issues

- Does a NL multicast packet look different?
- Does a NL multicast address look different?
- What DL multicast address should it be transmitted on?
- How dynamic is membership? Set up with provider (like SMDS)? Join and leave dynamically? One node controlling membership or can anyone just join?
- What kind of routing will support this?

IP Proposal for Multicast Addressing

- IP Multicast address has top 4 bits 1110
- Algorithmic mapping to 802 LAN multicast address
  - IP address 32 bits, relevant portion 28 bits
  - Would require 16 IEEE blocks if each mapping unique — not feasible
  - IP Multicast addresses given half of IP’s block, (23 bits worth), so 32 IP multicast addresses map to the same 802 address

<table>
<thead>
<tr>
<th>IP Multicast address</th>
<th>1110xxxx</th>
<th>xxxxxxxx</th>
<th>xxxxxxxx</th>
<th>xxxxxxxx</th>
</tr>
</thead>
</table>

| 802 address block    | xxxxxxxx | xxxxxxxx | xxxxxxxx | xxxxxxxx | xxxxxxxx |

IGMP

- Protocol for endnodes on LAN to tell router on LAN which groups they are listening to
- Version 1 router periodically asked, endnodes (waiting random time), responded, to derived layer 2 multicast address. Routers have to promiscuously listen to multicast
- Version 2 added “quit” message followed by group-specific query, to drop out sooner
- Version 3: request (S,G) not just G. Respond to router rather than to derived address (so everyone will respond)

DVMRP

- “Distance Vector Multicast Routing Protocol”
- Uses “reverse path flooding”
  - ignore packet from S unless received on link L, where L is how you forward to S
  - otherwise, it’s just like flooding
- Router knows (via IGMP) what multicast addresses are of interest on the LAN
- Router R complains (sends “prune”) if packet not of interest
- Neighbor N remembers (for a while) not to forward that address to that neighbor
- If all neighbors don’t want the packet, N sends prune when it receives that multicast address
- scaling problems: data goes everywhere, need to remember prune state for all groups’ sources
**MOSPF Method**

- DR puts all multicast groups learned through IGMP into its LSP
- The LSP goes to all routers in area
- Build tree per group per source
- Now, two alternative methods to make sure a potential transmitter in Nepal can transmit to these listeners
  - (At least one) router collects all multicast addresses in area and advertises them to next higher level, so entire universe hears all
  - All multicast traffic sent universally to each area, and can only be dropped there
- Scaling problems: Knowledge of all groups goes everywhere. Too much state and CPU (calculate Dijkstra per source, prune per group)

**Core Based Trees**

- Idea: for each group, one node is “core”. Form tree of shortest paths from the core to each participant.
- Join by sending special pkt towards core
- Routers along the way keep track of the fact that they are now on the path of a multicast. If it reaches a router that already knows about that (core, multicast), the join goes no further

**PIM**

- Start with CBT, then do:
  - Router joins per-source tree “if sufficient volume of traffic from S”. Prunes off from (*,G) with respect to S. Until (S,G) tree unused for “long enough”. Then rejoin.
  - some receivers get pkt from S via shared tree, others via (S,G) tree. To make the tree loop-free, (*,G) tree unidirectional (i.e., tunnel to core)
  - all routers infer core from address. “Core capable routers” advertise. Hash alg from M to {core capable router}
- Scaling problems: Core capable router advertisements, core not likely to be near tree, lots of trees, bursty source problem, complexity

**BGMP/MASC**

- Interdomain, bidirectional tree
- MASC: “somehow” have each domain dynamically acquire multicast address blocks
- Advertise these blocks in BGP
- Core is in domain that owns that block
- Problems: probably not enough addresses to dynamically grab blocks. Loads interdomain routing with passing around multicast address blocks. Not yet designed.
**MSDP**

- “Multicast source distribution protocol”
- Claimed to be “interim” until BGMP/MASC designed
- Involves configuring mesh of tunnels between core-capable routers, and broadcasting info about all active sources for all active groups
- Instead of storing info about all active (S,G) pairs, info is sent periodically.
- Scaling problems: info for all active source for all active groups has to get passed around entire Internet. Configuration-intensive because of tunnels. Many tunnels if for robustness there are many core-capable routers per domain.

**Simple Multicast**

- Two suggestions
  - just do bidirectional tree, using sensible core
  - have group ID be (C,M) and have endnode know both C and M and tell router
- How does endnode know C? Answer: same way it knows M... directory, email, web
- Include C and M in join and data message
- Result
  - Virtually unlimited address space (M only needs to be unique per core)
  - Trivial address allocation (ask C or random)
  - No scaling issues, so single protocol intra and interdomain

**Simple Multicast Variants**

- Express: Only do per-source trees. Group ID is (S,G). If transmitter not S:
  - form new tree
  - tunnel to S
- C and G in join. Data just contains G
  - Less radical idea
  - G’s still have to be unique (easier than aggregatable)
  - What if you receive joins for (C1,G) and (C2,G)?
- C and G in IP option, or new layer 4 header
  - Destination address link-local, like ATM.
    “Join (C,G)”, “Join-ack (C,G), use dest=X”
Quotes

- At first I hoped that such a technically unsound project would collapse but I soon realized it was doomed to success. Almost anything in software can be implemented, sold, and even used given enough determination. There is nothing a mere scientist can say that will stand against the flood of a hundred million dollars. But there is one quality that cannot be purchased in this way—and that is reliability. The price of reliability is the pursuit of the utmost simplicity. It is a price which the very rich find most hard to pay. -- C.A.R. Hoare in The Emperor’s Old Clothes, Turing Award Lecture (27 October 1980)
- With enough thrust, anything can fly...Milo Medin
- There are two ways of constructing a software design: One way is to make it so simple that there are obviously no deficiencies, and the other way is to make it so complicated that there are no obvious deficiencies. -- C. A. R. Hoare