Multilink Subnets

draft-thaler-ipngwg-multilink-subnets-00.txt

Dave Thaler

Christian Huitema

Microsoft

What?

- A “multilink subnet” is a subnet which spans multiple links, but is routed at layer three.
  - Similar to bridging, except hops are visible
  - HopLimit=1 will not reach all nodes in subnet
- An “intra-subnet router (ISR)” is a router with interfaces on multiple links in the subnet.

Why?

- Only need a single /64 for multiple links
  - even works where you can only get a /64
- Simplifies management
  - don’t need to plan subnet numbers
  - useful for adhoc/zeroconf network

Why not…

- Just bridge?
  - Link-scoped multicast hits low BW links
  - Can’t handle heterogeneous link types
  - Can’t put some ifs into promiscuous mode
- Use same bridging protocols at L3?
  - Can’t handle heterogeneous link types
  - Spanning tree leaves unused capacity
  - Algorithms are complex

Design Goals

- No change to end host behavior
- No change to link-scoped behavior (still within a single link)
- Send/receive unicast/anycast at site/global
- Send/receive multicast at subnet scope and larger
- Avoid routing loops
- Fast convergence
- NON-goal: scalability to huge subnets

Four main sub-problems

- Duplicate Address Detection
- Address Resolution
- Multicast
- Router-Router (ISR-ISR) communication

Host's view of the subnet

- Two possible approaches:

"Off-link" approach

  - Prefix in RA has A bit on, L bit OFF
  - Hosts don't send NS, packets go to def router
  - Router redirects if dest on-link

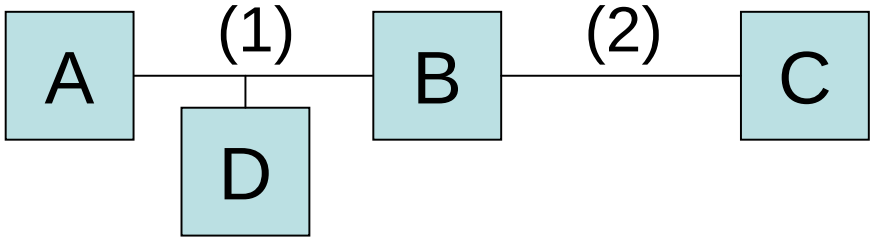
"On-link" approach

  - Prefix in RA has A bit on, L bit ON
  - Hosts send NS
  - Router replies with a proxy NA if dest off-link
    - Override bit clear per RFC 2461

Duplicate Address Detection

- DAD is specified over link-scope multicast, but needs to detect collisions on all links in subnet.
- Per RFC 2461, if have unique link-scope addr, DAD for other addresses with same interface id is optional.
  - Implies even link-scoped addresses must be unique across subnet (or make DAD reqd).
- DAD is otherwise like address resolution.

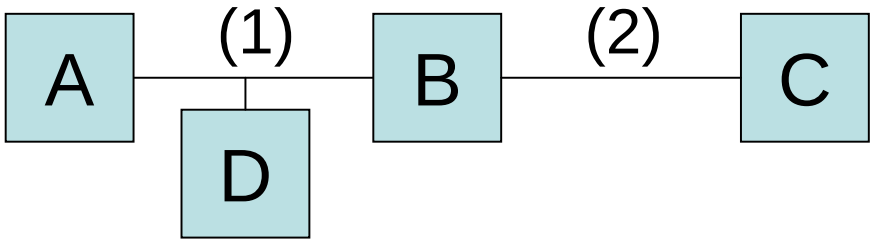
Address Resolution (“Off-link”)



- A sends packet for C to default router (B)
- B finds in nbr cache that C is on link 2
- B forwards packet to C

- A sends packet for D to default router (B)
- B finds in nbr cache that C is on link 1
- B sends redirect back to A

Address Resolution (“On-link”)



- B goes into all-multicast mode (not nec. promiscuous)

- A sends NS for C
- B finds in nbr cache that C is on link 2
- B replies to A with a proxy NA

- A sends NS for D
- B finds in nbr cache that C is on link 1
- D replies to A with a normal NA

Multicast

- Typically based on RPF
  - Router needs to know what direction towards source
- Can’t use subnet route (multiple directions!)
- Need either:
  - a non-RPF-based mechanism inside the subnet (e.g., spanning tree), or
  - host routes/neighbor cache entries for RPF

Router-router communication

- Four possible approaches:
- A) Create a spanning tree
- B) Flood NS's
- C) Proactively populate host routes
- D) Use a subnet server (bad)

- Roughly parallels same set of approaches for intra-domain multicast, with similar tradeoffs.

• Create a spanning tree

- Complex algorithm
- Spanning tree leaves unused capacity
- Solves loop prevention, and maybe multicast
- Still need parts of one of the other solutions for unicast/anycast

- Due to complexity, not fully investigated

B) Flood NS's

- Router stores sources of NS's seen, and floods the NS out all links in subnet
  - Then suppresses them for a short time
- NS propagates across entire subnet
  - ISRs never respond just due to cache hit
- End node responds with NA
- Router sends proxy NA to each NS querier
- Router adds a "Local Distance" option to get shortest path

C) Proactively populate host routes

- Routers listen to DAD attempts
- If host route already exists, router signals a duplicate
- If no host route exists, inject one for the new host
- Multicast works as-is
  - longest match finds host routes
- Problem: 2 hosts doing simultaneous DAD on different links

Comparing NS-flood with host-routes

- NS-flood takes longer to resolve
- NS-flood has bursty-source problem
- NS-flood timers might be tricky when multiple hops exist

- Host-routes uses more state
- Host-routes needs to solve simultaneous DAD problem