IPv6 Investigation (preliminary!)

gdmr, September 2015

Why?

Why might we want to implement IPv6?

- Make ourselves visible to IPv6-only ISPs
- Allow our users to speak to IPv6-only services elsewhere
- Machines are increasingly IPv6-enabled, and if we want to set some policy we have to have the support in the network.

Addressing

The most obvious IPv6 change is to addresses

- 128-bits, written out in 4-hexit chunks separated by colons
- “::” is shorthand for “as many all-zero chunks as are needed here”
  - Only one allowed, to avoid ambiguity
- See RFC 4291 for the full details!

Addressing

- Loopback is ::1/128
- University’s global address block is 2001:630:3c1::/48
- Link-local addresses automatically created from fe00::/10, based on MAC address
- Others...

Getting your IPv6 address

- Hard-wire it in, as we do for IPv4 on managed DICE machines
- DHCPv6
- SLAAC
- Addresses will be validated by DAD
- We can advertise policy per-subnet through the RA packets

How might we assign addresses?

The University's (Sam's) plan is to use the VLAN tag as the subnet identifier

- 129.215.160.0/24 becomes 2001:630:3c1:160::/64
  - Everything's a /64

We need to decide how to allocate the rest!

- Could link it to the IPv4 address somehow
- Might choose to do it completely separately

Linked #1

Could just copy in the “host” part:

- 129.215.160.240/24 might be mapped to 2001:630:3c1:160::240/64
- 129.215.27.244/22 might be mapped to 2001:630:3c1:24::3244/64
  - or perhaps to 2001:630:3c1:24::27:244/64
- 129.215.252.126/25 might be mapped to 2001:630:3c1:252::126/64
- 129.215.252.254/25 might be mapped to 2001:630:3c1:4027::254/64

Linked #2

Alternatively, we might always want to include the (IPv4) network number:

- 129.215.160.240/24 might be mapped to 2001:630:3c1:160::160:240/64
- 129.215.27.244/22 might be mapped to 2001:630:3c1:24::27:244/64
- 129.215.252.126/25 might be mapped to 2001:630:3c1:252::252:126/64
- 129.215.252.254/25 might be mapped to 2001:630:3c1:4027::252:254/64

Linked #3

Or use the numeric value rather than BCD:

- 129.215.160.240/24 might be mapped to 2001:630:3c1:160::f0/64
- 129.215.27.244/22 might be mapped to 2001:630:3c1:24::3f4/64
- 129.215.252.126/25 might be mapped to 2001:630:3c1:252::7e/64
- 129.215.252.254/25 might be mapped to 2001:630:3c1:4027::fe/64

Unlinked

But do we really need to have the IPv4 and IPv6 addresses linked in any way to each other at all?

- Arguably it means that if we know what one is we can tell what the other will be.
- Does it matter?
- Wouldn’t we just look it up anyway?

DNS #1

(If the addresses are linked) we could have makeDNS automatically assign IPv6 addresses for all IPv4 addresses.

- Arguably simplest for us, but ...
- We might well not want all machines to have IPv6 addresses
- If an address exists it is an invitation to use it
  - Timeouts, other breakage

DNS #2

We could add syntax to makeDNS to indicate that we wanted an IPv6 address.

- Per-machine or per-address?
- Getting rather clumsy
- At least under explicit control

DNS #3

Or we could just add a completely separate way to specify IPv6 addresses.

- Totally explicit
- Duplication, and chance of errors?
- If addresses *aren't* linked then much less of an issue
- Could lose hostfile format, giving better clarity and overall simplicity

Assigning addresses, DNS

So, how much do we really care about linking IPv4 and IPv6 addresses?

- If we do, the tools have to be written to make this easy and (relatively) error free
- If we don’t, the tools can be a lot simpler, and some “errors” turn out not really to be errors at all
- There’s lots more address space available. We shouldn’t restrict what we can do unnecessarily
- My vote is for totally unlinked FWIW

Switches

Turning to the network, there are things we would like our switches to do:

- Route IPv6: OSPFv6 and RA
- RA-guard at the edge
- DHCP protection
- MLD-snooping, to control multicasts

Switch support

- Our core switches can do what we want
- Our newer switches (2920, 2910, 2620, 2530) support IPv6
- Our older switches (2610, 2900, 5304) can’t
  - We’re replacing 2610 for data (not phones) this F/Y
  - We hope we won’t be in FH long enough that 2900 there becomes an issue
  - We do need to think about AT3/4/5
  - Some odds and ends, where restricting VLANs carried would probably suffice
  - Should probably replace core3 which connects us to EdLAN at Old College
  - Replace or just get rid of coreA, which gives us backup routing from outside the server room

Routing: core

- IPv4 and IPv6 routing are completely separate
- Router Advertisement
  - Multicast by the core routers
  - Like IPv4 router-discovery, but not quite so flexible
  - Better resilience, particularly for self-managed
  - Also carries policy and prefix information
- OSPFv3 between core and edge
- OSPFv3 (or maybe BGP?) to EdLAN

Routing: edge

- Quagga can speak OSPFv3 (and RA)
  - Some work required to extend lcfg-routing
  - … or replace it entirely!
- BIRD can speak OSPFv3 (and RA)
  - New component would be needed, so perhaps a bit more work
  - Diversity might be a good idea!
- No requirement that we use the same for both IPv4 and IPv6, as long as we’re careful to start only what we need
  - Might complicate component design if we do split up lcfg-routing into lcfg-quagga and something else

Filtering

- Should just be a case of working through the scripts to add IPv6 where needed
- Component already has IPv6 hooks in it
  - Mainly so that we can set a discard-everything policy!

DHCP

- Another project...

OpenVPN

- Support is coming, but isn’t really there yet
- We might not need very much for our use case
- We would need non-VLAN IPv6 address blocks
  - Sam has promised these will be easy to provide

How do we do it all?

As soon as we turn on IPv6 on at least some of our subnets, the machines on them will likely immediately start to use it. So we need to sequence the way we test and enable things:

- Decide on an addressing scheme
- Write a blog article
- Enable IPv6 on S32 and S33 on the Forum core switches, AT1 on the AT core switches, and B (64, transit) on all
  - All the machines on those subnet are managed, so nothing should notice or try to use it
  - With any luck link-local addresses should suffice, at least to start

Then...

- Take and peruse some MIB-walks
- Check and set as necessary the allowed-managers lists
- Decide on OSPFv3 or BGP to speak to EdLAN
  - We don't actually want to quite yet, but what we do will affect the OSPFv3 settings
- Enable OSPFv3 on S33, AT1 and B on the Forum and AT core
  - Routes (intra-area and ASBR-injected "connected") should propagate, and we should see these appear
  - Clone an OSPFv3 version of the netman-scripts tool to show these

… and then?

- Extend our configuration tools so that they know how to deal with IPv6 MIB objects, and can enable and disable things as required
  - In particular, we need to set the RA parameters and policies so that addresses and routes are obtained by Windows and Linux systems in accordance with the way we expect
    - These will vary per VLAN
  - We need to enable ra-guard on the edge switches
  - Ideally we would also want something to control ND

… and then??

- Extend or rewrite makeDNS so that we can assign static IPv6 addresses in a controlled way, in accordance with our addressing scheme decision
- Extend lcfg-network so that machines can pick up and use IPv6 addresses
  - Default routes should appear through RA
  - Until we are routing external traffic any machines which do have an IPv6 address may end up experiencing timeouts and delays as a result of actually trying to use it
  - MPU?

… more ...

- Extend the filtering scripts so that blocks and holes are added as required, as for IPv4
  - Separate project??
- Bring up OSPFv3 on our Linux routers
  - Separate project?
- Enable IPv6 on E42 and E160 aka AT2 and A
- Have IS speak OSPFv3 (or perhaps BGP) to us
  - This should make us globally visible, so anything with an IPv6 address must now be in a position to answer appropriately
  - At this point it becomes reasonable to attempt to use IPv6 for anything other than testing

… and more ...

- Enable IPv6 on remaining "managed" subnets at all sites
  - though note the caveat earlier concerning the AT labs’ switches
- Write another blog article!
- Implement audit tools equivalent to arpwatch
  - We can defer this to here because all addresses will have been assigned statically

… finally?

- Bring DHCP for IPv6 into service
  - Definitely a separate project!
- Enable IPv6 on all remaining subnets, and announce its general availability through another blog article
- Sort out the inevitable teething problems!
- tardis??

Questions

- Answers not guaranteed!