InformationWeek
NETWORKComputing
CONNECTING THE INFRASTRUCTURE COMMUNITY

- 02/03/2015
  8:00 AM

IPv6 Network Testing: Avoid The Top 5 Troubles

--Timothy Carlin is Senior Manager of Software and IP Networking at the UNH-IOL.

Test networks can be tricky, and IPv6 adds another dimension of complexity. Learn the most common issues associated with IPv6 testing and how to fix them.

Network testing is essential but sometimes challenging, and running into bugs and broken devices is a normal part of a test environment. But what happens when there are no bugs and the device isn't broken, but the test still doesn't work? Is it the test that is broken? Is the engineer to blame? Or is the problem something else, perhaps more subtle?

Strange or unanticipated testing issues happen all the time, and running a test network using IPv6 is no stranger to these issues. This handy list will go through some of the most common issues associated with IPv6 testing, as well as their potential resolutions.

1. Hardware: Does it have a link?
Sure, this is an obvious place to start and is not specific to IPv6, but it can be one of the most frustrating issues to try to hunt down, often because you just didn't think the cable was the problem. The truth is, cables slip, switches reboot, and sometimes hardware fails. All of these issues can cause a loss of communication at the most basic of hardware levels, and cause headaches for your testing. Speaking of switching, with the prevalence of wireless, virtualization, and even simple VLANs, hardware failures morph into a sight unseen, and can be even more troublesome to identify.

The best way around this problem is a stable, well labeled, and backed up (as in configuration) test network. A test network that changes rarely and is well known may not prevent failures, but will make it much easier to move through the weeds when problems do arise.

2. Firewalls: They exist!
Firewalls are evolving just as the IPv6 protocols are and are an absolute necessity when it comes to protecting users and software from malice. This means it is important

to be aware of how the firewall is implemented, and what it is configured to monitor or block. Firewalls often block traffic that, while important for protocol test validation, can appear to be a denial-of-service (DoS) or another dangerous attack. An overly cautious firewall can make you wonder if you really are plugged in where you think you are, causing you to hunt down ghosts.

The resolution here is tricky. For testing, a disabled firewall can help narrow down the cause quickly. This doesn't help when it comes to deployment; then you must ensure that your device will behave on the Internet and protect its users. There is no silver bullet here; the best thing to do is to investigate why the firewall was blocking the protocol traffic and seek expert help to determine if this is an anomalous block or a true security vulnerability.

3. Multihoming: And behind this door…

Multihoming is not a new concept with IPv6, but it is one that causes many more issues when it comes to testing and deploying IPv6. This is partly due to increased user-friendliness in the autoconfiguration of IPv6 -- specifically, the autoconfiguration of default routes. A correctly formed ICMPv6 router advertisement (RA) will cause a device to install a default route (or "route of last resort") through the transmitting router.

The problem occurs when more than one router is sending such a packet. You might think that the odds of two routers sending these well-formed packets is slim, but in a test network, it's all too easy for an RA to slip out onto a production network, or a different test network, wreaking havoc. Of course, this is only a problem if your device has more than a single interface -- which turns out to be very common! Phones (WiFi and cellular) and laptops (WiFi and Ethernet) are two very common devices that typically have a minimum of two potential egress interfaces.

Like hardware and firewall issues, a multihoming problem will make it seem as though traffic is being lost, or never sent. The difference here is that it will work -- or rather not work -- seemingly intermittently. This could be due to a difference in timers or lifetimes, allowing the device a "recovery" period, where it appears to operate correctly. This is an

easy one to identify, as two default routes will be clearly present on the system. Of course, resolving it may be difficult, as you now have to track down the sender of the rogue RA. Or, you could try a firewall…

4. Addresses: They're big
And there are a lot of them, and they are hard to type. Testing issues related to something as simple as the IPv6 address abound. This can manifest itself in something as trivial as a mistype of an E for a D, or a 3 for a 6. With so many numbers to type, this happens all the time. Here's a typical IPv6 address:

FC00:41FF:3880:1FE0:5851:75f2:8867:de1b

The best solution here is to copy and paste always. Don't trust yourself to type an IPv6 address under any circumstances!

Other addressing issues are more advanced, and are actually designed to help, rather than hurt networks. But, like with so many other cases, test networks are dramatically different than production networks. Consider duplicate address detection, which is a strategy built into IPv6 for identifying addresses that have been duplicated on a network.

Duplication would be unlikely to happen on a fully autoconfigured network, even in production environments. But in test environments, where tools and devices are configured with the same or similar addresses to streamline testing and configuration, it's all too easy for two devices to end up with the same address. The resolution here is similar to identifying rogue RAs: A simple look at the interface should tell you if the address is duplicated. The hard part is tracking down the thief!

5. Memory: Devices don't forget
IPv6 has a number of components for which, the longer a device is plugged in, the more is discovered about the network and the environment. This includes path MTU, addresses, neighbor state, and more. Again, while learning and tuning a device to the characteristics of a network is perfect for production networks, in a testing environment, it can cause plenty of false negative testing results.

A perfect example of this is the ICMPv6 Packet Too Big message. A device, to be conformant, must remember the contents of this message for a time. The trouble is, this time might be longer than it takes to move on to the next test! So when you see fragmented packets you didn't expect, a reboot is likely in order.

The same applies to neighbor discovery operations. This complex state machine has a number of twists and turns that -- if left in an "unclean" testing state -- can cause lots of problems for an upcoming test. In many cases, a routine to send various cleanup packets can clear out the neighbor cache. However, new standards for IPv6 make this a tougher task, so increasingly the best way to clean up is with a reboot.
