



# Challenges with IPv6 End-to-End Performance

Michael Sinatra, Network Engineer  
ESnet Network Engineering Group

CANS 2011

Kunming, Yunnan, China

August 9, 2011



# Overview



- Why IPv6? Why end-to-end?
- Problem I: OS performance with IPv6 isn't always as good as with IPv4.
- Problem II: IPv4/IPv6 performance differentials in routing equipment.
- Problem III: Lack of IPv6 feature parity leading to end-to-end connectivity issues.
- What can we do?



# Why end-to-end?

- Why are we so concerned about end-to-end performance?
  - As Dave Lambert mentioned, science is becoming increasingly global. It's also becoming increasingly data-intensive.
  - Greater need for scientists to collaborate in real time over large distances and this requires large data transfers.
  - For large data transfers, the end-to-end path needs to be *clean*.
    - No packet loss.
    - No translation mechanisms (e.g. NATs).
    - Other middleboxes should be minimized.

# Why IPv6?



- Some universities in the US have a lot of IPv4 addresses. Do they need to adopt IPv6? Yes.
- With IANA's IPv4 run-out, there is increasing likelihood of IPv6-only networks appearing on the Internet.
- End-to-end performance becomes difficult with IPv4 NAT.
- It also becomes difficult with protocol translation (e.g. NAT-PT or NAT64).
- Being able to communicate with your colleagues is important and some of your colleagues may soon only be able to use IPv6.
- This is generally well-known in Asia, but it is important that IPv6 adoption start to be more universal.



# Problem I: IPv6 OS performance

- ESnet has a number of perfSONAR performance testing nodes throughout its nationwide US-based network.
  - These nodes have 10GE interfaces.
  - Run the FreeBSD operating system, plus a small number of Linux-based hosts.
  - Recently, we configured ESnet's perfSONAR nodes to be dual-stack IPv4 and IPv6
  - Configured scheduled testing with perfSONAR.
  - → Typical results: Chicago to Washington, DC (TCP throughput):
    - IPv4: ~8.2 gbps
    - IPv6: ~2.4 gbps



# Problem I: IPv6 OS performance

- → Typical results: Chicago to Washington, DC (TCP throughput):
  - No significant difference in one-way delay between IPv4 and IPv6.
  - No noticeable packet loss with either protocol.
  - UDP performance very similar between both protocols.
- In examining TCP connections, I noticed a case where SACK holes followed by fast retransmits seemed to be ignored and the entire window segment eventually had to be retransmitted.
- → This turned out to be a different issue.
- Basically, in FreeBSD 8.x, TCP performs extremely well over IPv4 (better than Linux in many cases), but performs much worse over IPv6.



# Problem I: IPv6 OS performance

- Investigation of this problem is still ongoing. Why might it be happening?
  - FreeBSD uses the KAME stack for IPv6. This stack is 13 years old, and much of the code was imported as-is and hasn't seen a lot of maintenance until now.
  - FreeBSD 9.0 will have a revamp of IPv6 kernel code.
  - The KAME stack appears in a number of other OSes.
- We're dealing with some old code that simply hasn't been exercised.



## Problem II: IPv6 routing performance

- Modern routers use a combination of a main CPU for the *control plane* (routing protocols, management, etc.) and specialized hardware, such as ASICs and FPGAs for the actual *forwarding plane*.
- As much of the stuff that needs to happen at wire-speed, such as the forwarding of IPv4 and IPv6 packets, needs to be done “in hardware.”
- IPv4 moved to forwarding in hardware many years ago, although there are still some CPU-based “software” routers in use.
- The result is two-fold: separation of the control plane and forwarding plane, and hardware-based forwarding, both of which improve performance and reliability.



## Problem II: IPv6 routing performance

- When initially implementing IPv6 years ago, router vendors initially implemented IPv6 in software-only (CPU-based) routers. Hardware-based switch/routers, such as the Cisco 6500/7600 series, couldn't even support IPv6 at first.
- Eventually hardware-based routers could support IPv6, but they did all of the forwarding entirely in software!
- Slowly, hardware routers gained the capability of forwarding IPv6 traffic in hardware.
- However, there are still cases where enabling certain features or using certain configurations will cause IPv6 traffic to be forwarded in the main CPU.



# Problem II: IPv6 routing performance

- Example: Cisco 6500 Sup 720:
  - Very popular backbone router among US universities.
  - Some even use it as a border router.
  - Most of the “work” is done in hardware.
  - Has a rather under-powered CPU, since most forwarding done in ASICs.
  - IPv6 now routed in ASICs, not CPU, *but* if you enable unicast reverse path forwarding in IPv6, *all IPv6 traffic gets sent to the CPU!*
  - *This doesn't happen with IPv4!*
  - Unicast RPF is a very important feature for universities, as it is an easy way to prevent IP spoofing.

# Problem II: IPv6 routing performance



- Example: Cisco 6500 Sup 720:
  - This is an example where there is *feature parity* between IPv4 and IPv6, but not *performance parity*.
  - Thanks to Jimmy Kyriannis of New York University for pointing this out to the US IPv6 community.
- Performance parity is still a problem, but so is feature parity, and it has end-to-end implications.



## Problem III: IPv6 routing feature issues

- Feature parity continues to be a problem. Many platforms do not support all of the features in IPv6 that they support in IPv4.
- In some cases, this is because there is no applicable standard for IPv6, e.g. stateful NAT (not that we really want that, but...)
- But in many cases, the feature doesn't exist in IPv6 even if/when there is an applicable standard.



## Problem III: IPv6 routing feature issues

- Two examples that have end-to-end implications: RA guard and IPv6 BFD.
  - RA guard is the equivalent of DHCP snooping in IPv6.
  - “Rogue RAs” announce IPv6 routes and prefixes on wireless nets and even on wired nets.
  - Instead of using the high-speed default router for your LAN, you may end up forwarding all of your traffic through someone’s laptop!
  - This obviously has security implications, but it definitely has end-to-end implications. Imagine trying to transfer a terabyte of data over a 10GE link via someone’s 100mpbs-connected laptop!
  - RA guard standard exists, but vendors haven’t implemented it yet. Worse yet, while it is possible to do the same thing on switches with port-ACLs, only a handful of devices can do IPv6 port-ACLs!



# Problem III: IPv6 routing feature issues

- Two examples that have end-to-end implications: RA guard and IPv6 BFD.
  - Bidirectional Fault Detection is used to determine link-forwarding problems and take actions when they are detected.
  - The IS-IS routing protocol operates directly on top of layer-2. Routing information for address families (IPv4, IPv6) is carried as part of IS-IS messages.
  - It's possible to lose reachability over a specific layer-3 protocol without dropping the IS-IS adjacency. A router may still be trying to forward traffic through a link and may not be able to do so!
  - Solution: Implement BFD to detect loss of reachability over a particular layer-3 protocol. Withdraw route from table when that happens. Traffic can then take an alternate path.



# Problem III: IPv6 routing feature issues

- Unfortunately, some vendors (okay, Juniper) only support BFD for IPv4 over IS-IS. IPv6 is not supported.
- It is possible to black-hole IPv6 traffic if IPv6 transport fails over a link! This is not a good way for a routing protocol to work!
- So far, there doesn't seem to be any effort on Juniper's part to address this issue.

# Commonalities



- In some cases, IPv6 code exists (and it may have existed for a long time), but it has never been exercised.
- IPv6 performance issues haven't been a source of pain for users (who could then create pain for the vendors).
- In other cases, code for features doesn't exist because "nobody has asked for it."



# What to do?

- Many in the US are saying:
  - We have a lot of IPv4 addresses.
  - There are problems with IPv6 (as I have just illustrated).
  - ...so we shouldn't deploy IPv6!
- Actually, these are all reasons we need to deploy IPv6 now! Waiting until we absolutely need it will not fix these problems, it will only give them more serious consequences.
- Not only do we need to deploy IPv6, but we need to exercise it.
  - Big data flows
  - Latency-sensitive applications
  - Multiple concurrent flows



# What to do?

- perfSONAR can help:
  - Supports IPv6.
  - Can reveal differences between IPv4 and IPv6 *if configured correctly*.
  - → We need to do more inter-domain testing with IPv6 perfSONAR to bring in different routing platforms and different operating systems.
- You can help. In the US, there needs to be more universal deployment. There are some institutions that are doing well, but many have not done enough deployment. Time to get moving...
- In China, a lot of IPv6 deployment work has already been done. The world needs your help. Sharing experiences and operating dual-stack perfSONAR nodes can help us solve a lot of problems.