Introduction to Ipv6

Tom Eastep

Hewlett-Packard Company

April 25-26, 2009

LinuxFestNW

Presenation available at:

http://www.shorewall.net/LinuxFestNW-2009.pdf

Outline

- Why Ipv6?
- Ipv4 vs. Ipv6
- Addressing and stateless autoconfiguration
- Setting up a 6to4 Tunnel
- Future of Ipv6
- Q & A

A Little About Me

- 40 Year Veteran of the Computer Industry
  - Burroughs Corporation
  - Tandem Computers
  - Compaq
  - Hewlett-Packard
- Middleware, Operating Systems and File Systems
- Self-taught about Networking
  - Creator/maintainer of Shorewall
- This presentation is my own work and not sponsored or approved by Hewlett-Packard

Why Ipv6?

- Ipv4 address space is becoming depleted
  - Supports 4 Billion addressable nodes
  - Original allocation was inefficient
    - See http://www.iana.org/assignments/ipv4-address-space/
    - Legacy allocations to large US companies use 7% of the available space
    - US government has 3%
    - 6.25% reserved for multicast

Why Ipv6? Continued

- Ipv4 address space is becoming depleted
  - Routing tables in the Internet core routers have become very large
  - Explosion in the number of addressable devices (think cell phones)
  - India and China are increasing pressure
  - Reliance on private addresses and NAT causes a continuous level of pain for network administrators and application developers

Why Ipv6? Continued

- Solution
  - Create a successor with a much wider address space
  - Promote efficient route aggregation
  - Improve the state of the art
    - Auto configuration
    - Mobile IP
    - IP Security

Shorewall and Ipv6

- Shorewall has been Ipv4 only
- Ipv6 is gaining importance in Europe and Asia
- Users have been pushing for Ipv6 support in Shorewall
- I've been reading about Ipv6 for the last two years
- Netfilter Ipv6 support has matured
  - Ipv6 connection tracking in kernel 2.6.20
  - Stable since 2.6.24

Shorewall and Ipv6 Continued

- I took off the month of December 2008 to implement Ipv6 support in Shorewall

Ipv6 vs Ipv4

- IMPORTANT – Ipv6 is **not** just Ipv4 with a wider address space
- But it **does** have a wider address space

Ipv6 vs Ipv4 – Ipv4 Addresses

- Ipv4 address – 32 bits, usually written as *byte1.byte2.byte3.byte4* (e.g., 206.124.146.176)
  - Each byte is written in decimal and leading zeros may be omitted.
- Address is composed of a *network address* and a *host* address.
  - Network in high-order bits; host in low-order bits
  - We write 206.124.146.176/24 to indicate that the network address is 24 bits and the host address is 8 bits.

Ipv6 vs Ipv4 – Ipv6 Addresses

- Ipv6 – 128 bits, usually written as word1:word2:word3:word4:word5:word6:word7:word8 (e.g., 2002:ce7c:92b4:1:21a:24ff:fecb:2bcc)
  - Each 16-bit *word* is written in hex; leading zeros may be omitted.
  - One sequence of words containing zero may be written as "::"
    - 2002:ce7c:92b4:1:0:0:0:1 is the same as 2002:ce7c:92b4:1::1
    - 0:0:0:0:0:0:0:0 is the same as ::

Ipv6 vs Ipv4 – Ipv6 Addresses

- Address Types
  - Unicast – Uniquely identifies an interface on an Ipv6 node
  - Multicast – Identifies a group of Ipv6 interfaces
  - Anycast – Assigned to multiple interfaces on multiple hosts. A packet sent to an anycast address is delivered to exactly one of those hosts. Idea is not fully baked yet.

Ipv6 vs Ipv4 – Ipv6 Addresses

- Multicast addresses are FF00::/10
- Three classes of unicast addresses
  - Link Local (FE80::/10)
  - Site Locall (FEC0::/10)
  - Global (Currently 1/8th of address space)
    - 2000::/3 (2xxx:… and 3xxx:…)

Ipv6 vs Ipv4 – Link Local Addresses

- Link Local Addresses are autoconfigured
  - High-order 64 bits – FF80::
  - Low-order 64 bits – EUI-64 interface address (constructed from MAC address on Ethernet interfaces).
  - Not Routed

Ipv6 vs Ipv4 – Link Local Addresses

$ ip -6 addr ls dev eth0

2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qlen 1000

inet6 **fe80::21b:24ff:fecb:2bcc**/64 scope link

valid_lft forever preferred_lft forever

$ ip link ls dev eth0

2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UNKNOWN qlen 1000

link/ether **00:1b:24:cb:2b:cc** brd ff:ff:ff:ff:ff:ff

$

- Link Local Address is fe80::021b:24ff:fecb:2bcc
- Mac Address is 00:1b:24:cb:2b:cc

Ipv6 vs Ipv4 – Site Local Addresses

- Defined by RFC 3513 which defines the Ipv6 addressing architecture
- Were expected to be used like Ipv4 'private' addresses (RFC 1918)
- Problematic without NAT
- Limited-use senarios
  - Isolated networks
  - Fully proxied networks
- Deprecated by RFC 3879

Ipv6 vs Ipv4 – Global Unicast Addresses

- 2000::/10
- http://www.iana.org/assignments/ipv6-unicast-address-assignments
- Stable for the last three years!

Ipv6 vs Ipv4 – Ipv6-icmp (icmp6)

- Autoconfiguration of link local addresses means that there is no need for an Ipv6 version of the Address Resolution Protocol (ARP).
- Icmp6 is used to perform neighbor discovery (IP->link-level address resolution)
- Icmp6 also facilitates autoconfiguration of global addresses without a stateful server (e.g., without DHCP).

Ipv6 Neighbor Display

$ ip -6 neigh ls dev eth0

fe80::2a0:ccff:fedb:31c4 lladdr 00:a0:cc:db:31:c4 router REACHABLE

$

- Similar to 'arp -na' in Ipv4

Stateless Autoconfiguration

- Assumes Ipv6 network is a /64.
- Client sends *Router Solicitation* Icmp6 to address FF01::2. Source address is the client's auto-configured Link Level Address.
- Router responds with a *Router Advertisement* which includes the public network address.
- Client configures the interface with the address formed by concatinating the network address with the interface's EUI-64 host address.

Stateless Autoconfiguration Continued

- The *Router Advertisement* message's source IP address is the Router's link-local address. That become's the client's gateway.
- To configure a Linux system as an Ipv6 router, install and run **radvd**.
  - Requires /proc/sys/net/ipv6/conf/all/forwarding = 1
- Routers also periodically broadcast Router Advertisements.

Autoconfiguration Example

```
$ ip -6 addr ls dev eth0
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qlen 1000
    inet6 2002:ce7c:92b4:1:21b:24ff:fecb:2bcc/64 scope global dynamic
       valid_lft 2591898sec preferred_lft 604698sec
    inet6 fe80::21b:24ff:fecb:2bcc/64 scope link
       valid_lft forever preferred_lft forever
$
```

- Network address is 2002:ce7c:92b4:1/64
- Host address is 21b:24ff:fecb:2bcc

Autoconfiguration Example Continued

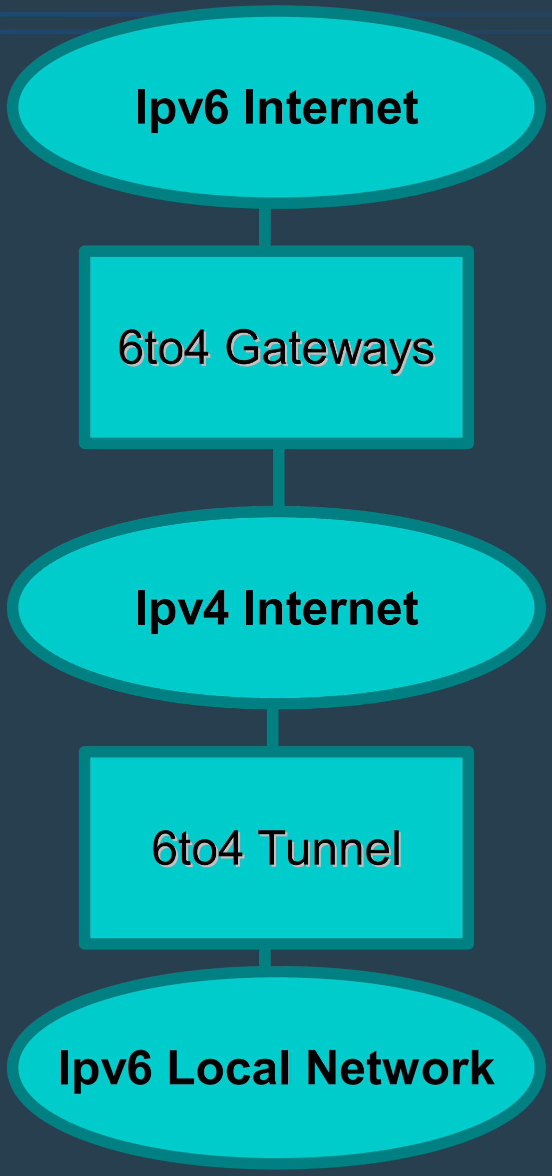
```
$ ip -6 route ls dev eth0
2002:ce7c:92b4:1::/64  proto kernel  metric 256  expires 2591669sec mtu 1500 advmss 1440 hoplimit 4294967295
fe80::/64  proto kernel  metric 256  mtu 1500 advmss 1440 hoplimit 4294967295
default via fe80::2a0:ccff:fedb:31c4  proto kernel  metric 1024  expires 1307sec mtu 1500 advmss 1440 hoplimit 64
$
```

- Default Gateway is fe80::2a0:ccff:fedb:31c4

Autoconfiguration Continued

- On Windows Vista, two public addresses are configured:
  - EUI-64
  - UUID
- Both raise privacy concerns
- Linux supports privacy protection by allowing an option for using a random number rather than the EUI-64
    - /proc/sys/net/ipv6/conf/all/use_tempaddr

6to4


Ipv6 Internet
6to4 Gateways
Ipv4 Internet
6to4 Tunnel
Ipv6 Local Network

6to4

- 6to4 allows you to experiment with Ipv6 even if your ISP doesn't offer native Ipv6 support.
- The global unicast network 2002/16 is reserved for 6to4
- An Ipv4 host with a static IP address can configure a 6to4 *tunnel* which allows access to the Ipv6 internet.
- Instructions at http://www.shorewall.net/6to4.htm

6to4 Example

```
$ ip -6 addr ls dev sit1
13: sit1@NONE: <NOARP,UP,LOWER_UP> mtu 1480
    inet6 ::206.124.146.180/128 scope global
       valid_lft forever preferred_lft forever
    inet6 2002:ce7c:92b4::1/128 scope global
       valid_lft forever preferred_lft forever
$
```

- Note that the device has two Ipv6 addresses:
  - ::206.124.146.180 – Obviously formed from an Ipv4 Address
  - 2002:ce7c:92b4::1 – Less obviously formed from the same Ipv4 Address

6to4 Example Continued

- 2002:ce7c:92b4 is formed by concatenating '2002' and 'ce7c:92b4'
- 'ce7c:92b4' is just 206.124.146.180 in hex!
- So for every public Ipv4 address, you can have a /48 Ipv6 network.

6to4 Example

```
$ ip -6 route ls dev sit1
::/96 via ::  metric 256  expires 19607541sec mtu 1480
advmss 1420 hoplimit 4294967295
2002:ce7c:92b4::1  metric 256  expires 15455183sec
mtu 1480 advmss 1420 hoplimit 4294967295
fe80::/64  metric 256  expires 19607541sec mtu 1480
advmss 1420 hoplimit 4294967295
default via ::192.88.99.1  metric 1  expires 19607541sec
mtu 1480 advmss 1420 hoplimit 4294967295
$
```

6to4 Gateways

- 6to4 *gateways* are routers that interface to both the Ipv4 and Ipv6 internet.
- 192.88.99.1 is an Ipv4 *Anycast* address. Each 6to4 gateway advertises a route to 192.88.99.0/24 on the Ipv4 internet.
- Each 6to4 gateway advertises a route to 2002/16 on the Ipv6 internet.
- BGP propagates these routes so that from any point on either the Ipv4 or Ipv6 internet, traffic sent to the other internet will be routed through the nearest 6to4 gateway

DNS and Ipv6

- AAAA records are used for name->Ipv6-address translation
- New reverse lookup domain for ipv6-address->name translation
  - ipv6.int.in
- Radvd supports specifying name server information (RDNSS)
- Rdnssd may run on clients to handle RDNSS information from router
- Coordinating /etc/resolv.conf is an issue

DNS and Ipv6 Continued

- My 6to4 link has high latency so using Ipv6 for DNS resolution is pretty painful

Adoption of Ipv6 in the US is Slower

- NetworkWorld 3/20/2009
  - "Business incentives are completely lacking today for upgrading to IPv6, the next generation Internet protocol, according to a survey of network operators conducted by the Internet Society (ISOC). In a new report, ISOC says that ISPs, enterprises and network equipment vendors report that there are 'no concrete business drivers for IPv6.' However, survey respondents said customer demand for IPv6 is on the rise and that they are planning or deploying IPv6 because they feel it is the next major development in the evolution of the Internet."

Q & A

Ipv6 vs Ipv4 – Ipv6 Addresses

- Address Format
  - N => 16
  - M => 48

Global Routing Prefix n bits	Subnet ID m bits	Host ID 128 – n – m bits