

---

# IPv6 — An introduction

---

Owen DeLong  
[owend@he.net](mailto:owend@he.net)

LoL Kitteh sez:



More IPv4 NAT

Are you fscking kidding me?

# IPv6 Transition -- How ready are we?

## ■ Things that are ready

- Backbones
- CMTS Systems (DOCSIS 3)
- MacOS (10.4+)
- Linux (2.6 Kernels)
- Windows (7, 2008, XP (limited))

- WiMax (specification, head end equipment)
- LTE (some)
- CPE (very limited)
- Early Adopters and some industry experts
- Hurricane Electric
- Me

# IPv6 Transition -- How ready are we?


- Things that are NOT ready


-  PON Systems

-  DSL Systems

-  CMTS Systems (DOCSIS 2)

-  WDS/EVDO/HSPA

-  WIMAX (handsets, providers)


-  Older Windows (XP and earlier)

-  Embedded systems

-  Printers

-  Home entertainment devices

-  CPE (most)

-  Most IT staff and management

# An Important Decision

# Which Approach will you take?

IPv4/IPv6 Dual Stack Now



My dual stack network is running great!

IPv4 is just fine.  
We just need MOAR NAT!!



# What we'll cover

- Basics of IPv6
- IPv6 Addressing Methods
  - SLAAC
  - DHCP
  - Static
  - Privacy
- Linux Configuration for Native Dual Stack
- IPv6 without a native backbone available
- Free IPv6?



---

# Some additional topics

- Routing
- Firewalls
- DNS
- Reverse DNS
- Troubleshooting
- Staff Training



# Basics: IPv4 vs. IPv6

| Property   | IPv4 Address   | IPv6 Address   |
|--|--|--|
| Bits   | 32   | 128  |
| Total address space  | 3,758,096,384 unicast<br>268,435,456 multicast<br>268,435,456 Experimental/other (Class E, F, G) | 42+ Undecillion assignable<br>297+ Undecillion IANA reserved               |
| Most prevalent network size  | /24 (254 usable hosts)   | /64 (18,446,744,073,709,551,616 host addresses)                            |
| Notation   | Dotted Decimal Octets<br>(192.0.2.239)   | Hexidecimal Quads<br>(2001:db8:1234:9fef::1)                               |
| Shortening   | Suppress leading zeroes per octet  | Suppress leading zeroes per quad, longest group of zeroes replaced with :: |
| <sup>1</sup> 42,535,295,865,117,307,932,921,825,928,971,026,432 assignable unicast (1/8th of total)<br><sup>2</sup> 297,747,071,055,821,155,530,452,781,502,797,185,024 IANA reserved (7/8th of total) |  |  |



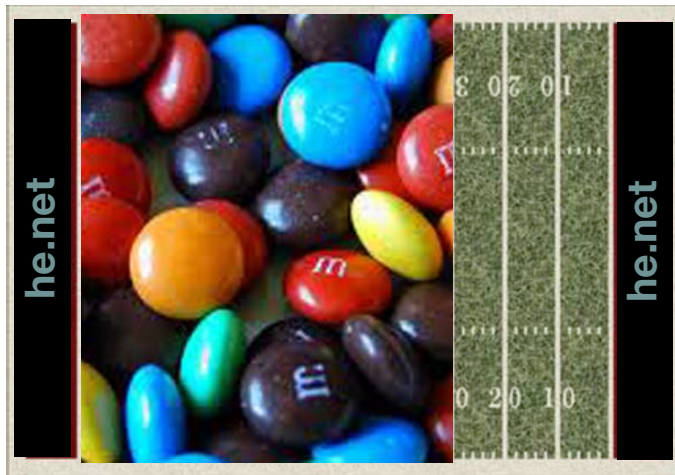


# Network Size and Number of networks (The tasty version)



One IPv4 /24 -- 254 M&Ms

One IPv6 /64 -- Enough M&Ms to fill all 5 of the great lakes.



Full Address Space, One M&M per /24 covers 70% of a football field



Full Address Space, One M&M per /64 fills all 5 great lakes.

Comparison based on Almond M&Ms, not plain. Caution! Do not attempt to eat a /64 worth of any style of M&Ms.

# Basics: IPv4 vs. IPv6 thinking

| Thought                   | IPv4 dogma                                     | IPv6 dogma  |
|---------------------------|--|---|
| Assignment Unit           | Address (/32)                                  | Network (/64)   |
| Address Optimization      | Tradeoff -- Aggregation, Scarcity              | Aggregation (At least for this first 1/8th of the address space)                                  |
| Address Issue Methodology | Sequential, Slow Start, frequent fragmentation | Bisection (minimize fragmentation), issue large, minimal requests for more, aggregate expansions. |
| NAT                       | Necessary for address conservation             | Not supported, Not needed -- Breaks more than it solves (other than possible NAT64)               |
| Address Configuration     | Static, DHCP                                   | Stateless Autoconf, Static, some DHCP (needs work), DHCP-PD (NEW!!)                               |



This is the Internet:



This is what NAT does to your packets:



This is what's left of your audit trail:



Any quesitons?

# Basics Address Scopes

- Link Local -- fe80::- Site Local (deprecated) -- Only valid within site, use ULA or global as substitute.
- Unique Local Addresses (ULA) -- Essentially replaces IPv4 RFC-1918, but, more theoretical uniqueness.
- Global -- Pretty much any other address, currently issued from 2000::/3, globally unique and valid in global routing tables.



---

# Basics: Stateless Autoconfiguration

- Easiest configuration
- No host configuration required
- Provides only Prefix and Router information, limited ability to provide services addresses (DNS, NTP, etc.) (on some implementations)
- Assumes that all advertising routers are created equal, rogue RA can be pretty transparent to user (RA guard required on switches to avoid)



# Stateless Autoconfiguration Process

- Host uses MAC address to produce Link Local Address. If MAC is EUI-48, convert to EUI-64 per IEEE process: invert 0x02 bit of first octet, insert 0xFFFE between first 24 bits and last 24 bits fe80::
- IPv6 shutdown on interface if duplicate detected.
- ICMP6 Router Solicitation sent to All Routers Multicast Group



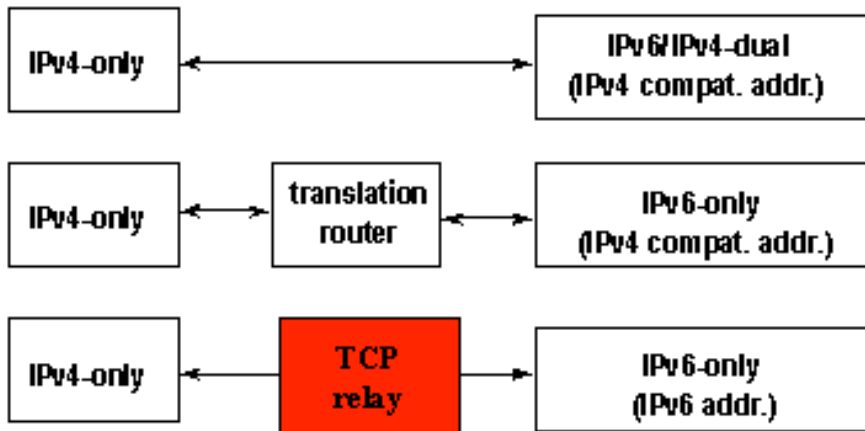
# Stateless Autoconfiguration Process (cont.)

- Routers send ICMP6 Router Advertisement to link local unicast in response. Also sent to All Hosts Multicast group at regular intervals.
- Router Advertisement includes Prefix(es), Preference, Desired Lifetime, Valid Lifetime.
- Host resets applicable Lifetime counters each time valid RA received.
- Address no longer used for new connections after Desired lifetime expires.
- Address removed from interface at end of Valid lifetime.
- Prefix(es)+EUI-64 = Host EUI-64 Global Address, netmask always /64 for SLAAC.



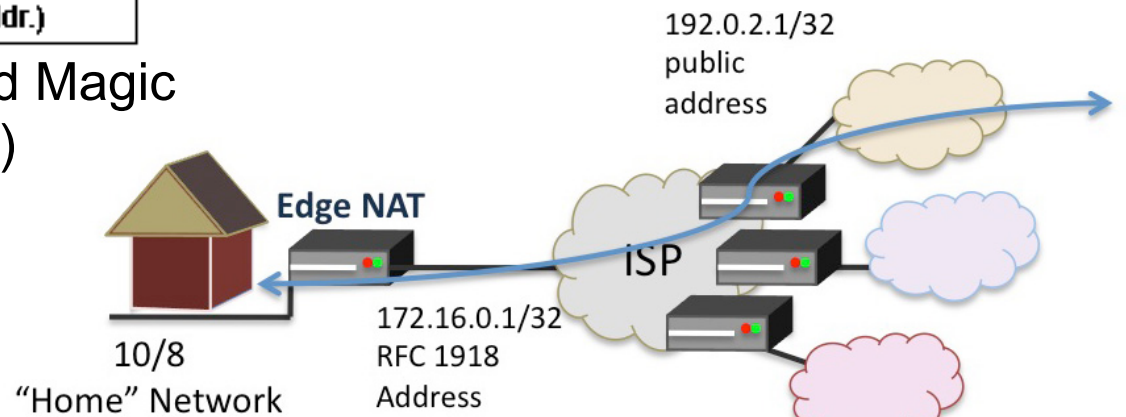
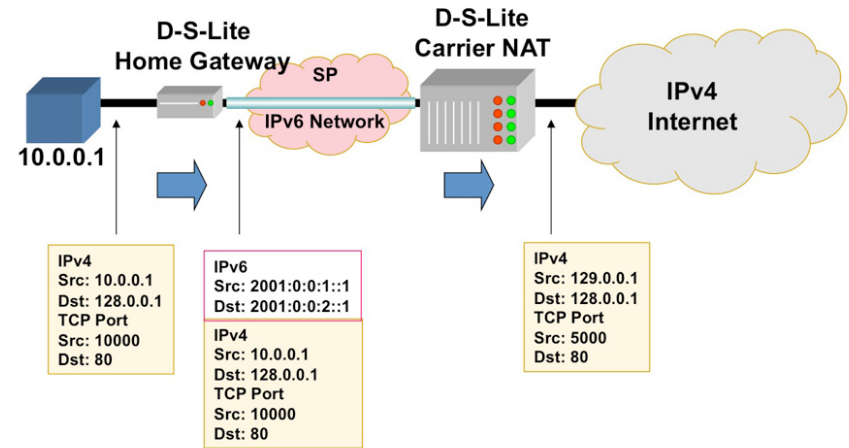
# If you think IPv6 is hard, wait until you try any of these.

Communication between IPv4 nodes and IPv6 nodes



As yet undefined/unimplemented Magic (TCP relay could be SSH tunnel)

## Dual Stack Lite (ISC)



## Multiple Layer NAT (Carrier Grade NAT)



# DHCPv6

- Can assign prefixes other than /64 --  
Theoretically to routers which then delegate various networks automatically downstream, a few limited implementations of this feature.
- Can assign addresses to hosts, cannot provide default router information.
- Can provide additional information about servers (DNS, Bootfile, NTP, etc.)
- Not much vendor support (yet)



# Static Addressing

- IPv6 can be assigned statically, same as IPv4
- Common to use one of two techniques for IPv4 overlay networks:
  - Prefix::`<addr>` (first 12 bits of 64 bit `<addr>` must be 0)
  - Either `<addr>` is IPv4 last octet(s) expressed as BCD, or `<addr>` is IPv4 last octet(s) converted to hex.
  - e.g. 192.0.2.154/24 -> 2001:db8:cafe:beef::`154/64` (BCD) or 2001:db8:cafe:beef::`9a/64` (Hex)
  - These mappings won't conflict with autoconfigured addresses since autoconfigured addresses will never be `000x:xxxx:xxxx:xxxx`.



# Privacy Addresses

- Essentially a special form of Stateless Address Autoconfiguration which uses a new suffix for each flow and obfuscates the MAC address.
- RFC-3041
- Uses MD5 Hash with random component to generate temporary address
- Preferred and Valid lifetimes derived from SLAAC address



# Multiple addresses per interface

- IPv4 has some support for this in most implementations.
- IPv6 has full support for this in all implementations.
- IPv4, multiple addresses/interface are exception.
- IPv6, single address on an interface nearly impossible in useful implementation (link local required, global optional)



# IPSEC

- In IPv4, IPSEC is add-on software.
- In IPv6, IPSEC is a required part of any IPv6 implementation
- IPv6 does NOT require IPSEC utilization
- IPSEC is considerably easier to configure in IPv6.
- IPSEC automation may be possible in future IPv6 implementations.



# Configuring IPv6 Native on Linux

- Interface Configuration depends on your distro.
- Debian based distros (Debian, Ubuntu, etc.) use `/etc/interfaces`
- Red Hat based distros (RHEL, Fedora, CentOS) use `/etc/sysconfig/network-scripts/ifcfg-<int>`



# /etc/network/interfaces

```
iface eth0 inet static
    address 192.0.2.127
    netmask 255.255.255.0
    gateway 192.0.2.1
```

IPv4 (Static)

```
iface eth0 inet6 static
    address 2001:db8:c0:0002::7f
    netmask 64
    gateway 2001:db8:c0:0002::1
```

IPv6 (Static)

```
iface eth1 inet6 auto
```

IPv6 (Autoconf)



# /etc/sysconfig/network-scripts/ ifcfg-`<int>`

DEVICE=eth0

ONBOOT=yes

IPADDR=192.159.10.2

IPv4 (Static)

NETMASK=255.255.255.0

GATEWAY=192.159.10.254

IPV6INIT=yes

IPV6ADDR=2620:0:930::0200:1/64

IPv6 (Static)

IPV6\_DEFAULTGW=2620:0:930::dead:beef

IPV6\_AUTOCONF=no

IPV6ADDR\_SECONDARIES="\

2001:470:1f00:3142::0200:1/64 \

2001:470:1f00:3142::0200:2/64"

IPV6INIT=yes

IPv6 (Autoconf)

IPV6\_AUTOCONF=yes





# IPv6 without a native connection

- Three options (In order of preference)
  - 6in4 -- Tunnel your IPv6 in an IPv4 GRE Tunnel
  - 6to4 -- Tunnel your IPv6 in an auto-tunnel using an any-casted IPv6 mapping service
  - Teredo -- Tunnel your IPv6 in an auto-tunnel using a multi-server auto-configured process defined by Microsoft.



# Why 6in4

- GRE is well understood by most networkers
- Simple and deterministic
- No anycast magic -- Simplifies debugging
- Controlled by two endpoint administrators -- Greatly simplifies debugging
- Disadvantage: Manual config, but, not hard.



# Why 6to4

- Automatic configuration
- When it works, it's pretty clean and relatively self-optimizing.
- May be good option for mobile devices (laptop, cellphone, etc.)
- Hard to troubleshoot when it doesn't work.
- Disadvantage: Anycast == Non-deterministic debugging process.



# Why Teredo?

- Autoconfiguration
- May bypass more firewalls than 6to4
- Enabled by default in Windows (whether you want it or not)
- Meredo available for Linux (client and server)
- Disadvantage: Complicated and tricky to debug if problems occur.



# Configuring a 6in4 tunnel on Linux

- Not as straightforward as you would hope.
- Help available at <http://tunnelbroker.net>
- Example (route2, most 2.6+ kernels):

```
modprobe ipv6
ip tunnel add he-ipv6 mode sit remote 64.71.128.82 local 192.159.10.254
ttl 255
ip link set he-ipv6 up
ip addr add 2001:470:1F02:BE2::2/64 dev he-ipv6
ip route add ::/0 dev he-ipv6
ip -f inet6 addr
```

- Recent Debian (and derivatives) have straightforward configuration in `.../interfaces` file.



# Configuring 6in4 continued

- Example Net Tools (most 2.4 kernels, some 2.6)

```
ifconfig sit0 up
ifconfig sit0 inet6 tunnel ::64.71.128.82
ifconfig sit1 up
ifconfig sit1 inet6 add 2001:470:1F02:BE2::2/64
route -A inet6 add ::/0 dev sit1
```

- Also not supported in configuration files



# Fedora 12 Configuration Files

## ■ Example:

### □ /etc/sysconfig/network-scripts/ifcfg-sit1

```
DEVICE=sit1
BOOTPROTO=none
ONBOOT=yes
IPV6INIT=yes
IPV6TUNNELIPV4=64.71.128.82
IPV6TUNNELIPV4LOCAL=192.159.10.2
IPV6ADDR=2001:470:1f02:BE2::2/64
```

### □ /etc/sysconfig/network

```
NETWORKING=yes
NETWORKING_IPV6=yes
HOSTNAME=myhost.example.com
IPV6_ROUTER=yes
IPV6FORWARDING=yes
```

# Fedora 12 Configuration Files

- Example:

- /etc/sysconfig/static-routes-ipv6

```
sit1    ::/0
```

- /etc/sysconfig/network-scripts/route6-sit1

```
2001:470:1f00:3142::/64
```





---

# IPv6 For Free? YES!!

- Several tunnel brokers offer free IPv6.
  - My favorite is the HE Tunnelbroker at [www.tunnelbroker.net](http://www.tunnelbroker.net)
- If you or your organization has a presence at an exchange point with Hurricane Electric, we currently offer free IPv6 Transit.



# Routing

- Usual suspects
  - OSPF (OSPFv3)
  - BGP (BGP4 Address Family inet6)
  - RA and RADVD
  - Support in Quagga and others



# Firewalls

- ip6tables much like iptables
  - Excerpt from my ip6tables configuration

```
-A RH-Firewall-1-INPUT -d 2620:0:930::200:2/128 -m state --state NEW -m tcp -p tcp
--dport 3784 -j ACCEPT
-A RH-Firewall-1-INPUT -d 2620:0:930::200:1/128 -m state --state NEW -m udp -p udp
--dport 53 -j ACCEPT
-A RH-Firewall-1-INPUT -d 2001:470:1f00:3142::200:1/128 -m state --state NEW -m udp
-p udp --dport 53 -j ACCEPT
-A RH-Firewall-1-INPUT -d 2620:0:930::200:2/128 -m state --state NEW -m udp
-p udp --dport 53 -j ACCEPT
```



# DNS

- Forward DNS

- Instant IPv6 -- Just add AAAA

- Reverse DNS

- Slightly more complicated
- ip6.arpa
- 2620:0:930::200:2 ->  
2620:0000:0930:0000:0000:0000:0200:0002
- 2620:0000:0930:0000:0000:0000:0200:0002 ->  
2000:0020:0000:0000:0000:0390:0000:0262
- 2000:0020:0000:0000:0000:0390:0000:0262 ->  
2.0.0.0.0.2.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.3.9.0.0.0.0.0.0.2.6.2.ip6.arpa



# DNS -- BIND Configuration

- Current BIND versions ship with IPv6 template zones (hints, rfc1912, etc.)
- IPv6 addresses valid in ACLs just like IPv4, same rules
- Zone configuration identical except reverse zones for IPv6 ranges called "ip6.arpa":

```
zone "0.3.9.0.0.0.0.0.0.2.6.2.ip6.arpa" IN {  
    type master;  
    file "named.2620:0:930::-48.rev";  
};
```



# DNS -- BIND Configuration

- In IPv6 Reverse Zone files, \$ORIGIN is your friend!
- Forward Zones A for IPv4, AAAA for IPv6, basically what you're used to:

```
mailhost          IN      A       192.159.10.2
                  IN      AAAA    2620:0:930::200:2
```

- Reverse Zones PTR records, as described above:

```
$ORIGIN 0.0.2.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.3.9.0.0.0.0.0.0.2.6.2.ip6.arpa.
1.0.0.0          IN      PTR     ns.delong.sj.ca.us.
2.0.0.0          IN      PTR     owen.delong.sj.ca.us.
4.0.0.0          IN      PTR     irkutsk.delong.sj.ca.us.
```





# DNS -- Common Reverse DNS mistakes

- Not enough zeroes -- 2620:0:930::200:2 is much easier to type, but, remember for reverse DNS you have to expand all those suppressed zeroes before you reverse the address.
- Missing dots (.) -- Every nibble gets one.
  - ❑ 2.0.0.0.0.0.2.0.0.0.0.0.0.0.0.0.0.0.0.3.9.0.0.0.0.0.0.0.2.6.2
  - ❑ Do you see the error in the previous line?
- Reversing first then expanding
  - ❑ 0.0.0.2.0.2.0.0.0.0.0.0.0.0.0.0.0.0.0.3.9.0.0.0.0.0.0.2.6.2.0





# Troubleshooting

- Mostly like troubleshooting IPv4
- Mostly the same kinds of things go wrong
- Just like IPv4, start at L1 and work up the stack until it all works.
- If you are using IPv4 and IPv6 together, may be easier (due to familiarity) to troubleshoot L1-2 on IPv4.



# A wee bit about Neighbor Discovery and other tools

- No broadcasts, no ARP
- This is one of the key differences with IPv6.
- Instead a solicited node multicast address is used.
- IPv4: `arp 192.0.2.123`
- IPv6: `ip -f inet6 neigh show 2620:0:930::200:2`
- `ping` -> `ping6`
- `tracert` -> `tracert6`
- `telnet`, `ssh`, `wget`, etc. just work



# Cool SSH trick

- Special for those that made it through the whole presentation:
- If you have a dual stack host you can SSH to in between an IPv4 only and an IPv6 only host that need to talk TCP, then, you can do this from the client:
- `ssh user@dshost -L <lport>:server:<dport>`
- Then, from the client, connect to localhost:lport and the SSH tunnel will actually protocol translate the session.



# SSH trick example

- myhost -- IPv6-only host 2620:0:930::200:f9
- dshost -- IPv4/v6 dual stack host: 192.159.10.2 and 2620:0:930::200:2
- desthost -- IPv4-only host 192.159.10.100
- On myhost I type:
  - `ssh owen@2620:0:930::200:2 -L 8000:192.159.10.100:80`
  - Then, I can browse to `http://[::1]:8000`
- My browser will connect to the ssh tunnel via IPv6, and, the SSH daemon at dshost will pass the contents along via IPv4.



# Staff Training

- Hopefully this presentation works towards that.
- You'll need more.
- Plan for it.
- Budget for it.
- Allocate time for it.
- If possible, have the staff being trained leave their pagers/blackberries/iPhones/etc. in the car during training.



# Q&A



## Contact:

Owen DeLong  
IPv6 Evangelist  
Hurricane Electric  
760 Mission Court  
Fremont, CA 94539, USA  
<http://he.net/>

owend at he dot net  
+1 (408) 890 7992

