Linux Home
Automation IPv6
(Last updated: May 07, 2010)

Home networking with IPv6

First let me state this is a work in progress so it may have sentences that end abruptly, sections that make no sense at all, logic errors, formatting errors and links that go nowhere. Sorry I'll get to them. In the meantime here's what I've got. Remember these are more or less my engineering notes. Don't expect to come here and learn IPv6. I can only recommend that you pick up a good book to get you started and then start reading the RFCs.

A little background

I am a network engineer. I work on managed networks. Most of my work is done on the WAN portion of the network (the part that connects the LANs together). I'm always interested in learning and I'm definitely interested in IPv6.

I, of course, have a home network which is atypical even for some of the more hardware geeks among us (see below). I consider it my test bed and yes, I really have all those devices.

My notes

This document has grown out of my frustration with finding out about how upgrading to IPv6 is going to affect my current IPv4 home network. In June 2009, Comcast (my current ISP) announced that they will embrace IPv6 (info on a Comcast proposed RFC called Dual stack lite). While it's still in trials, it most likely means that we will see IPv6 networking become a reality in the next few years. In late 2009, the IPv4 network started using network 1.0.0.0/8. With less than 10% of the IPv4 addresses available to a growing world it seems more evident that IPv6 will happen and a lot sooner than some people would like.

What I'd really like to know is what size subnet mask. excuse me - prefix length (see subnetting/prefix length) can I expect for my CPE and how can I locally administer my IP addresses so I can reach my private servers. I'm sure I won't be the only person with this kind of a problem. I know that some devices support UPnP and DLNA but that doesn't always work and is not univerally supported. Also I am curious as to what migration plans are in place. There will always be IPv4 devices that need to reach IPv4 sites so something will need to be in place to allow them to continue to do that in the future.

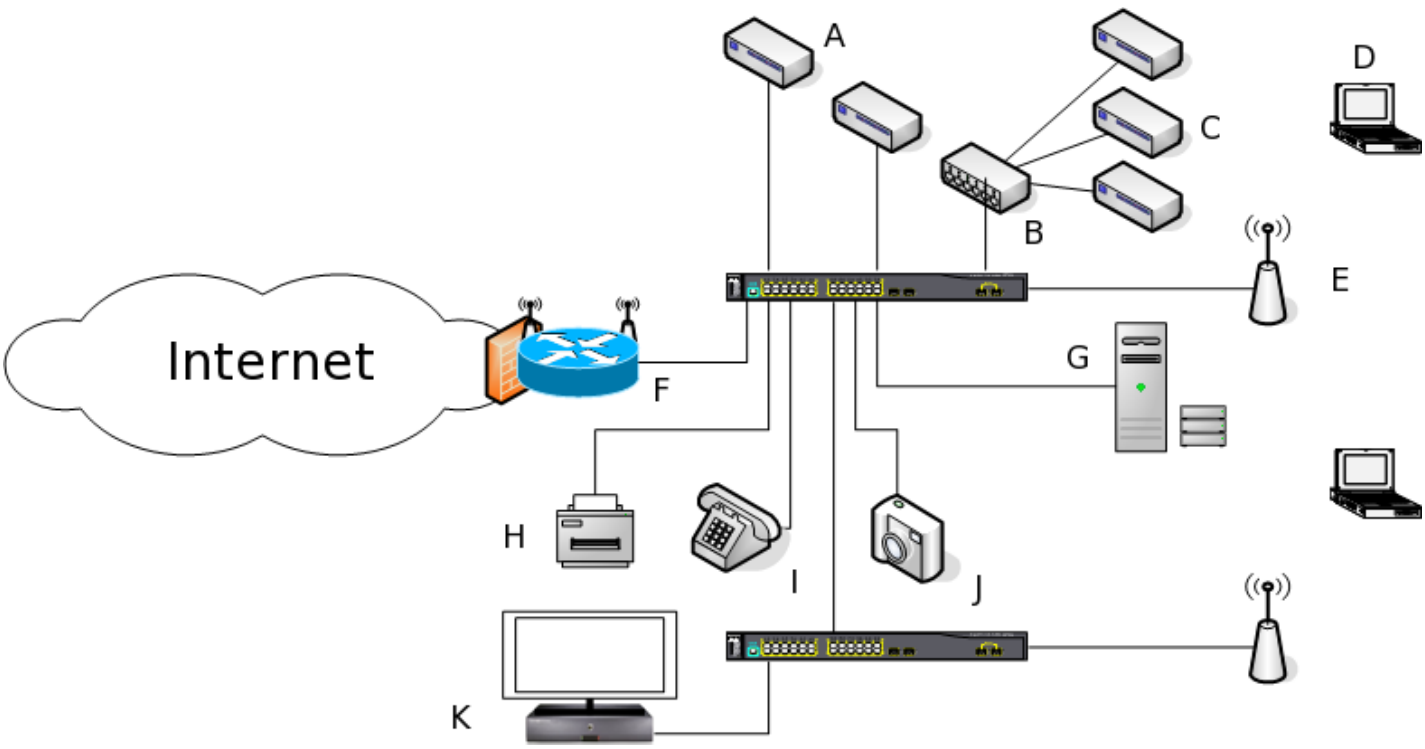
Update: Feb. 2010 - I now have a working tunnel between my WGT634U router and Hurricane

Electric! I didn't leave it up as I need to work on the IPv6 portion of the Firewall but this is a huge jump forward. I think I've also found an lead on the radvd/dhcp/local renumbering issue (see below). I've also added the frame work to explain subnetting a little better.

Update: Jan. 2010 - Comcast has asked for volunteer customers to trial their IPv6. I've signed up but not received any word yet (to be expected). It appears they'll have 4 phases. I could participate in 3 of the trial phases. And the last link, more information from the Comcast Blog. I'm very excited to see this and hope it goes well (though I do expect several bumps, IPv4 to IPv6 is a major transition and some devices are not going to behave).

Unlike most folks, I have a rather large home network. Many of these devices are very old and have no chance of being upgraded to the latest IPv6 standards. Examples of this include an ancient Cisco 16 port terminal server, a couple of older Tivo *DVRs*, IP cameras, a relatively new, modern and efficient Laser Printer and many other devices. What this means is that I need to have both IPv4 and IPv6 protocol suites mixed in my network. Because at layer 2 the protocol is Ethernet (or 802.3) there is no problem with running IPv4 or IPv6 over the LAN switches (i.e. no need to upgrade the existing layer 2 switches). It's at layer 3 where IPv4 and IPv6 differ. On the other side of the coin, I have several Linux and Windows boxes that are already running IPv6 and IPv4 (dual stack) and I have WAPs that run both IPv4 and IPv6 with one WAP running only IPv6.

Getting IPv6 working with Linux is actually pretty easy. Deciding which way to set it up is a little interesting (lots of choices). Getting my local DNS and DHCP properly setup for IPv6 support is going to be a nightmare. The first thing I need to do is to get a router/firewall which will support IPv4 and IPv6 (my OpenWRT page). In the future I may chose to build my own with a ALIX2D2 and my own Linux build using BuildRoot. So far Buildroot has been a dream to use, though I am having a little trouble compiling the microperl package. Actually I've backed off a bit on Buildroot and started plugging in OpenWRT boxes for the time being. Sorry just got busy with a lot of stuff.

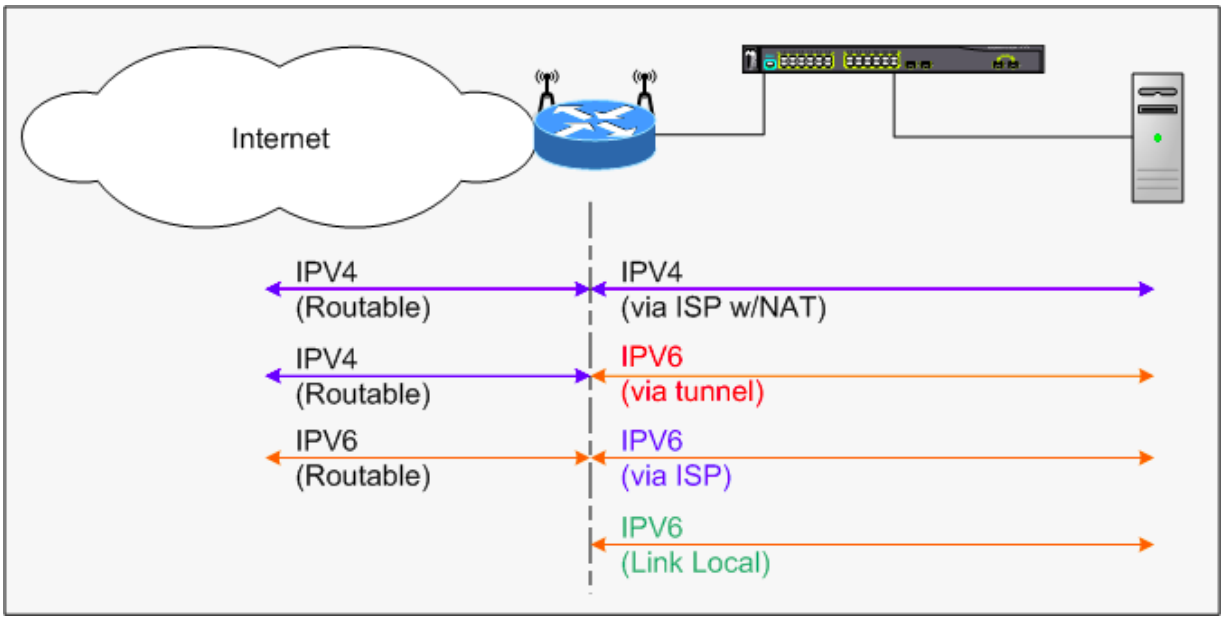


A complex IPv4/v6 network (not totally atypical)

Pretty isn't it? Yes, I actually have these devices (and more). In the upper right are my embedded systems that I develop with. To the left are a few development boards with network

interfaces. The rest are typical devices found in many home networks. The addition of IPv6 might not be typical but this is my network. :-) And, of course, the cloud, which is the Internet.

A. Ethernet Embedded system (some with Linux, some without, depending on the CPU size)
B. Terminal server
C. Serial Embedded system (RS232 or RS485)
D. Ordinary laptop, netbook, or wireless touch screen
E. Wireless Access point (bridging and no firewall, OpenWRT)
F. Firewall/Router (OpenWRT)
G. Central server (DHCP, DNS, SMTP, HA, Samba, etc)
H. IP printer
I. VoIP phone
J. IP Camera
K. Tivo, Roku/Netflix box or other Streaming media player (video and music)
L. ... and more



IPv4 and IPv6 networks (i.e. dual stacks)

I think this picture does a pretty good job of describing a 'dual stack' setup. The dual stack method is the current 'preferred method' for the migration of IPv4 networks to IPv6 networks. Though the current consensus seems to be that IPv4 will be around for a long time to come. The first portion of the network is the typical IPv4 home setup using NAT. NAT provides a little extra security because the typical RFC1918 addresses are not routed to the internet.

The next is the IPv6 tunnel. I'm using (but it's not up yet) a tunnel broker (Hurricane Electric or Go6) to get IPv6 over the existing IPv4 network. Note that the IPv6 addresses are fully reachable from the IPv6 network. Without the correct firewall rules my IPv6 machines would be sitting naked on the internet. The nice thing about the tunnel service is that it allows me to experiment with IPv6 before my ISP begins delivering IPv6. The same mechanisms would behave the same on the home network. The reason for the delay in getting the tunnel service working is that my IPv6 routers have limited RAM and flash. Things are a little tight and I have to make sure that the iptables are correct before I open up my network to the entire world. Baby step for right now.

The last two IPv6 portions of the network are actually part of the IPv6 protocol. The difference from the tunnel service is that the ISP will be delivering the IPv6 protocol over it's IPv6

network. To the rest of the home network nothing has changed. In fact the tunnel and the ISP delivered IPv6 can coexist side by side.

While researching IPv6 I found out that there is support of IPv6 *NAT* (IPv6 - IPv6). I've also heard about *NAT-PT* (IPv6 - IPv4) and application gateways. The last two are quite complex because there is no 1:1 translation between the two protocols. I hope to find some working examples.

What I expect

One of the frustrations with such migrations is that I have no point of reference to use to plan with. Will Comcast give me one IPv6 address? Will they give me a network range? (Looks like it will be a range, probably with a /48 or /56 prefix length. Although I'm no longer so certain of that.) or maybe give me only one network (a /64 prefix length). How often can I expect this to change? How do I manage and administer my local devices so that those who need to get out can and those that don't stay put? IPv6 has a vast range of networks for special purposes. There's the link-local, unique local unicast (replaces site-local), global unicast, and various trasitional addressing. The good news is that you can have more than one IPv6 address on an interface and it will help 'do the right thing' (I hope). What protocols will be used, looks like DHCPv6 is on the way, what about Comcast's proposed DS Lite?

In addition to the number of IPv6 addresses I'll receive I also would like to figure out how I can get the addresses and dynamically redistribute them to my machine (I think radvd comes into play for this and maybe DHCPv6). Next I need a way to update the local DNS (not the global one) with the correct information. I have way too many devices to be trying to remember IP addresses. Never mind that the IPv6 address is difficult to even type than remember (2001:db8:::ea7:dead:fe0d:beef/64). For now I expect that locally I'll keep using the IPv4 address which my local DNS can handle. Actually my local DNS can also deliver IPv6 addresses but manual entry is quite a pain (need to work on that web gui now).

- Link-local addresses are akin to the private, non-routable addresses in IPv4 (10.0.0.0/8, 172.16.0.0/12, 192.168.0.0/16). They are not meant to be routed, but confined to a single network segment. Link-local addresses mean you can easily throw together a temporary LAN, such as for conferences or meetings, or set up a permanent small LAN the easy way. (Does a link-local address use ND? I think it does. Need to look this up and read further.)
- global unicast address
- unique local unicast address
- Anycast addresses
- Multicast addresses

IPv6 Prefix	Allocation
0000::/8	Reserved by IETF
::/128	Unspecified address
::1/128	Link local address - the loopback address
::ffff:0:0/96	IPv4 transition - this prefix is used for IPv4 mapped addresses (see Transition mechanisms).
2000::/32	Global Unicast
2001::/32	IPv4 transition - Used for Teredo tunneling.
2001:db8::/32	Documentation - this prefix is used in documentation (RFC 3849).
2002::/16	IPv4 transition - this prefix is used for 6to4 addressing.

FC00::/7	Unique Local Unicast - unique local addresses (ULA) are routable only within a set of cooperating sites (replacement for site-local addresses - RFC4193).
FE80::/10	Link Local Unicast - analogous to the Autoconfiguration IP addresses 169.254.0.0/16 in IPv4
FEC0::/10	Site local addressing, DEPRECATED (don't use)
FF00::/8	Multicast - Similar to the 224.0.0.0 multicast network

IPv6 Subnets

```
2001:0DB8:0400:000e:0000:0000:0000:402b
      ||| |||| |||| |||| |||| |||| ||||
      ||| |||| |||| |||| |||| |||| |||128
      ||| |||| |||| |||| |||| |||| ||124
      ||| |||| |||| |||| |||| |||| |120
      ||| |||| |||| |||| |||| |||| 116
      ||| |||| |||| |||| |||| |||112
      ||| |||| |||| |||| |||| ||108
      ||| |||| |||| |||| |||| |104
      ||| |||| |||| |||| |||| 100
      ||| |||| |||| |||| |||96
      ||| |||| |||| |||| ||92
      ||| |||| |||| |||| |88
      ||| |||| |||| |||| 84
      ||| |||| |||| |||80
      ||| |||| |||| ||76
      ||| |||| |||| |72
      ||| |||| |||| 68
      ||| |||| |||64
      ||| |||| ||60
      ||| |||| |56
      ||| |||| 52
      ||| |||48
      ||| ||44
      ||| |40
      ||| 36
      ||32
      |28
      24
```

Know your prefix:

Prefix Length	Number of IPv6 IPs	*Network/Host* Space
/128	0	*xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx*
/127	2*	*xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx*
/124	8*	*xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx*
/120	256	*xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx*
/64	18,446,744,073,709,551,616	*xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx*
/48	1,208,925,819,614,629,174,706,176	*xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx*
/32	79,228,162,514,264,337,593,543,950,336	*xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx*

Note: a single 'x' represents 4 bits

Allocation means the address space available for subnets and more IPs. If you have a /127 address you can only connect one box to the IPv6 interface, with a /48 you can connect "all the devices in the world" using the allocation.

If your prefix is *2001:DB8:400::/48* then you are assigned all space beginning with 2001:DB8:400. You can use the space *2001:DB8:400:xxxx:xxxx:xxxx:xxxx:xxxx*.

To organize yourself *2001:DB8:400::/48* can be divided into the subnets:

- *2001:DB8:400:xxxx:xxxx:xxxx:xxxx:xxxx/48* - 1 network, 0 subnets, 1,208,925,819,614,629,174,706,176 hosts
- *2001:DB8:400:000E:xxxx:xxxx:xxxx:xxxx/64* - 1 network, 64K subnets, 16+E hosts (???)
- *2001:DB8:400:000F:xxxx:xxxx:xxxx:xxxx/64* - 1 network, 64K subnets, 16+E hosts (???)

Yes I know I'm using 1024 instead of 1000, I just like orders of 2 better and I've added the '+' to cover my butt. What's a couple of *really big number* between friends.

Link local, site local, and private address space

Argh! I need to get this cleared up (not just the loopback interface bit above). Seems the site local addressing is deprecated and now there's private address space (like RFC1918). More on this later (hmm, maybe I have the above address table correct ...).

Subnetting/Prefix length

As I said above, my main interest in what Comcast doing with IPv6 is that I'd like to see the size of their subnet and the protocol standards they'll support. As a guess I'd suspect that they won't go any larger than a /112 prefix length. This would give you access to more than 64k devices in the home. A /108 would support more than 1M devices and a /104 would support 16M devices. More than likely we'll see a /64 for the host as many protocols depend on that. If we'll see anything larger (so we can subnet) remains to be seen. I'm not even sure the average end user would know what to do with subnets but some services might want the ability (such as your utilities for new green technologies).

While on Broadband/DSL Reports I participated in an interesting discussion on IPv6. The user RARPSL posted a nice link to IPv6 Address space - Wikipedia. He corrected my notion of how the addressing might be handed down to the end user. It might look something like this: RIR - /23 -> ISP - /32 -> Customer - /48 -> Host /64, addressed using EUI-64. I think that RARPSL is correct, the end user will probably see something like a /48. He also suggested that we could see a /56 instead which would still give the end user some 256 networks. The host will probablly never see anything smaller than a /64 as this is the default prefix length for neighbor discovery (RFC4861, obsoletes RFC2461).

I just read this: RFC3267 - Use of /127 Prefix Length Between Routers Considered Harmful. Basically the author recommends using a /64 prefix length (Well there goes all those addresses that IPv6 was to provide!). Sorry I know that IPv6 has a gazillion addresses and can afford a 'little' waste but it's that kind of thinking that ends up wasting huge amounts of space. It's the '*640K ought to be enough for anybody*' type of thinking and before you know it half the address space is wasted and the network engineers are left to clean up the mess. The good news is the author also states the the ping-pong problem with /127 does not exist with /126. Also note there is a problem with Mobile IPv6 Home Agent Discovery (if you use /126) and I'll have to do further reading to get a better understanding of the issue.

U/G - bits 70 & 71

... One should note that [ADDRARCH] specifies universal/local bits (u/g), which are the 70th and 71st bits in any address from non-000/3 range. When assigning prefixes longer than 64 bits, these should be taken into consideration; in almost every case, u should be 0, as the last 64 bits of a long prefix is very rarely unique. 'G' is still unspecified, but defaults to zero. Thus, all prefixes with u or g=1 should be avoided. ... [5. Other Problems with Long Prefixes]

One of my co-workers (thanks S.G.) pointed out that the solicited-node multicast addresses are used for address resolution and DAD.

xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx/128

xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx/64
*xxxx:xxxx:xxxx:xxxx:xxxx:xx**FF:FE**xx:xxxx/64* <-- Mapping IEEE 802 addresses to IPv6 interface identifiers (EUI-64)
xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx/64

Examples

Need to put examples here using the 2001:db8::/32 Documentation address space.

ARIN recommendations

Following the links on the RFCs led me to a reference to ARIN's IPv6 address recommendations for ISPs to use /64 prefix as the largest prefix length (including things like point to point circuits). So I'm guessing that most end users (i.e. consumers) will probably see /64 for their home network. Why would anyone give out such a large network? Maybe because the service providers don't want huge routng tables. I'm not refering to the Intenet backbone routers but rather the IPS's core routers. The larger the prefix mask the end user has the higher the probability that the routing will contain more unsummerized routes at the customer edges. This doesn't preclude the end user from further subnetting but it does mean that only a summary route of the assigned prefix will be routed to that location. Of course I'm not sure there are many users out there who need to have subnets (yes they are useful). Of course this may mess with RFC4861 - Neighbor Discovery for IP IPv6 (which obsoletes RFC2461).

Ex: Let's say that an ISP gets a /48, then they give the end user a /64 - that's a possible 64K route entries in the route table. Give the end user a /32 and suddenly it could be 4G route entries. Those core routers would require a lot of memory to deal with that many routes. In reality the network engineers have to come to some kind of comprimise, hence the /64 recommendation. Of course proper route summerization will lessen the problem and this doesn't mean that the end user can't further subnet the /64. Also any good engineer would use sumerization technics to minimize the network entries resource needs.

Naming a New Node on the Network

A string hostname will be resolved to IPv6 numeric addresses by using `/etc/hosts` (hosts(5)), and/or DNS. This is much like the same as IPv4 case. yp(8) may be used, however, the author has no experience using yp(8) with IPv6 addresses.

1. `/etc/hosts` (hosts(5)): For small networks of a few nodes, the hostname/IP maps can be manually duplicated in the /etc/hosts files of each node.

   `/etc/hosts`:

   ```
   2001:db8:1234::a:b:c:d  host2.example.net host2
   ```

2. DNS: The hostname maps can be centralized into zone-files which are accessed by the name-server, named(8). (there are many documents at www.dns.net which deal with setting up and maintaining DNS files).

   Forward zone file entry

   ```
   host2  IN AAAA     2001:db8:1234::a:b:c:d
   ```

   Reverse zone file entry

   ```
   d.0.0.0.c.0.0.0.b.0.0.0.a.0.0.0.0.0.0.0.4.3.2.1.8.b.d.0.1.0.0.2.ip6.arpa.
           IN PTR   host2.example.net
   ```

> Note: If you have heard about A6 records and bitstring labels for DNS – A6 and bitstrings are now in experimental status, so you don't need to worry about them. AAAA is to be used for production service and works fine.
>
> Also note that the reverse DNS table is moving from `ip6.int` to `ip6.arpa`; 6Bone (3ffe) addresses are at the time of this writing only known in ip6.int, production addresses should be announced under ip6.arpa, as ip6.int will go away.

Please note that the following addresses MUST NOT appear in the global DNS cloud:

link-local addresses (matches fe80::/10, like fe80::1)

> This one is not globally reachable.

unique-local addresses (matches fc00::/7, like fd00::1)

> This one is not globally reachable either. This one somewhat like a private address, so you can put it into your intranet DNS cloud.

v4 mapped addresses (matches ::ffff:0.0.0.0/96, like ::ffff:10.1.1.1)

> This address is only for internal use in a node. Do not put this into DNS database directly.

It is not recommended to put these addresses onto DNS cloud:

1. multicast addresses (matches ff00::/8, like ff05::1)

For the NetBSD/KAME IPv6 code, putting the following address does not make sense due to lack of support:

v4 compatible address (matches ::0.0.0.0/96, like ::10.1.1.1)

> NetBSD/KAME does not support RFC2893 auto tunnel.

IPv6 unicast address allocation

There are couple of differences between IPv6 and IPv4 address allocation.

- The prefix length for an IPv6 subnet will always be /64; no more, no less. It allows you to place as many IPv6 devices as the underlying network medium allows. (This is not true, it appears that using a /64 prefix length is a best recommendation it does not mean that you will not see longer prefixes).

  With IPv4, prefix length varies between subnets to subnets, and it caused painful costs when renumbering subnets (for example, imagine when you renumber an IPv4 subnet from /28 to /29 or vice versa).

- An ordinary leaf site will always get /48 of address space. It will make site renumbering easier, and allows you to switch ISP more easily.

  With IPv4, the allocation varies by the size of the site, and made it very painful when you migrated from one ISP to another, for example.

Okay, forgive me, I no longer know where I found the above! It sounds ridiculous because of the words must and always. I'll probably remove or reword this in the future.

Broadcast/Multicast

IPv6 doesn't support broadcast addresses but it deos replace this with multicast replacements. (More to come)

Solicited-Node Multicast address = IPV4 ARP (FF02:0:0:0:0:1:FF00:0000/104) concat'd with a 24 low-order bits of a corresponding IPv6 Unicast or anycast address)

IPv6 Links

- Wikipedia on IPv6
- Wikipedia on IPv6 Addresses
- IPv6 Addressing Plans
- A Flexible Method for Managing the Assignment of Bits of an IPv6 Address Block
- On the Assignment of Subnet Numbers
- go6 - The IPv6 Portal - Tunnel Broker
- Hurricane Electric - Tunnel broker and free certification (knowledge testing).
- How to build an IPv6 router
- Network Address Translation/Protocol Translation
- Dual-stack lite broadband deployments post IPv4 exhaustion - draft-durand-dual-stack-lite-00
- LORHA
- IPv6 intelligence
- Everything you need to know about IPv6
- IPv6 using IP
- Linux IPv6
- Linux IPv6 Howto
- Free IPv4 to IPv6 Tunnel Brokers - Good link, just saved me from trying SixXS (not a very nice rep).
- Why you want IPv6 - A really nice collection of IPv6 information for Linux (and others really).
- Private IPv6 address range - This site will build you a unique private IPv6 address range generated just for you
- What's my IP?
- IPv6 over Low-Power Wireless Personal Area Networks
- IPv6
- DD-WRT LAN IPv6
- Comcast6.net IPv6 Information Center
- IPv6 Address space - Wikipedia
- 6-surprising-facts-about-ipv6 - 6 surprising facts about IPv6, well not really
- Using address table object in firewall builder
- HomeLANSecurity
- IP6Wall
- Managing IPv6 Part 1
- Managing IPv6 - Part 2

Commands

A few of the commands I needed to get IPv6 up and running:

```
# insmod /lib/modules/2.4.36/ipv6.o
# echo 1 > /proc/sys/net/ipv6/conf/all/forwarding

# ip a a 2001:468:181:f100::1 dev eth1
# ip r a 2001:468:181:f100::/64 dev eth1

# cat /tmp/radvd.conf
interface br0 {
   AdvSendAdvert on;
```

```
   prefix 2001:db8:0:1::/64 {
    AdvOnLink on;
    AdvAutonomous on;
   };
};

#
```

- radvd.conf manpage
- http://www.linuxhq.com/IPv6/radvd.html

Here's what Hurricane Electric (IPv6 tunnel broker) provided me:

```
# LOCAL_IPv4='192.168.138.25'
# REMOTE_IPv4='192.168.138.24'
# LOCAL_IPv6='2001:db8::f101:04fe:ff00:0a02/64'
# modprobe ipv6

# echo 1 > /proc/sys/net/ipv6/conf/all/forwarding

# ip tunnel add he-ipv6 mode sit remote ${LOCAL_IPv4} local ${REMOTE_IPv4} ttl 255

# ip link set he-ipv6 up
# ip addr add ${LOCAL_IPv6} dev he-ipv6
# ip route add ::/0 dev he-ipv6
# ip -f inet6 addr
```

From the OpenWRT IPV6 web page (needs reformatting)

IPv6 on the LAN

At this point I suppose that you have a working ipv6 connection on the wrt, that you can *ping6 www.kame.net* without error.

Note: When using a SixXS tunnel (and probably others), only ::1 (the PoP) and ::2 (your endpoint) can be used as the rest is not routed! Therefore, you need to request a subnet and enable it before you are able to utilize radvd.

Using our mythical

```
2001:db8:0:f101::/64
```

network, we would put in /etc/radvd.conf the following lines:

```
# For more examples, see the radvd documentation.
interface br0
{
        AdvSendAdvert on;
        prefix 2001:db8:0:f101::/64
        {
                AdvOnLink on;
                AdvAutonomous on;
        };
};
```

Now we add

```
2001:db8:0:f101::1
```

to br0 using the command below. To keep the changes at boot add it to the */etc/init.d /S40network* script. If the release is kamikaze, ip6addr theaddress/theprefixlength is an interface option in /etc/config/network. Forwarding of our delegated /64 subnet to br0 is done automatically in *S51radvd*

```
ip -6 addr add 2001:db8:0:f101::1/64 dev br0
```

In the /etc/init.d/S51radvd we have to add an route in case the aiccu is used:

```
ip -6 route add 2001:db8:0:f101::1/64 dev br0
```

After all this you can start the daemon:

```
/etc/init.d/S51radvd start
```

You can listen to its advertisments via the *radvdump* program.

====== Example for debugging purposes ======

Interface configuration:

```
root@OpenWrt:~# ip addr show
1: lo:  mtu 16436 qdisc noqueue
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
    inet6 ::1/128 scope host
2: eth0:  mtu 1500 qdisc pfifo_fast qlen 1000
    link/ether 00:0f:66:56:ee:6f brd ff:ff:ff:ff:ff:ff
    inet6 fe80::20f:66ff:fe56:ee6f/64 scope link
3: eth1:  mtu 1500 qdisc pfifo_fast qlen 1000
    link/ether 00:0f:66:56:ee:71 brd ff:ff:ff:ff:ff:ff
    inet6 fe80::20f:66ff:fe56:ee71/64 scope link
4: sit0@NONE:  mtu 1480 qdisc noop
    link/sit 0.0.0.0 brd 0.0.0.0
5: br0:  mtu 1500 qdisc noqueue
    link/ether 00:0f:66:56:ee:6f brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.1/24 brd 192.168.1.255 scope global br0
    inet6 2001:6f8:309:1::1/64 scope global
    inet6 fe80::20f:66ff:fe56:ee6f/64 scope link
6: vlan0:  mtu 1500 qdisc noqueue
    link/ether 00:0f:66:56:ee:6f brd ff:ff:ff:ff:ff:ff
    inet6 fe80::20f:66ff:fe56:ee6f/64 scope link
7: vlan1:  mtu 1500 qdisc noqueue
    link/ether 00:0f:66:56:ee:70 brd ff:ff:ff:ff:ff:ff
    inet 212.68.233.114/24 brd 212.68.233.255 scope global vlan1
    inet6 fe80::20f:66ff:fe56:ee70/64 scope link
8: sixxs@NONE:  mtu 1280 qdisc noqueue
    link/sit 212.68.233.114 peer 212.100.184.146
    inet6 2001:6f8:202:e::2/64 scope global
    inet6 fe80::d444:e972/64 scope link
    inet6 fe80::c0a8:101/64 scope link
```

Routing table:

```
root@OpenWrt:~# ip route show
192.168.1.0/24 dev br0  proto kernel  scope link  src 192.168.1.1
212.68.233.0/24 dev vlan1  proto kernel  scope link  src 212.68.233.114
default via 212.68.233.1 dev vlan1
root@openwrt:~# ip -6 route show
2001:6f8:202:e::/64 via :: dev sixxs  metric 256  mtu 1280 advmss 1220
2001:6f8:309:1::/64 dev br0  metric 256  mtu 1500 advmss 1220
fe80::/64 dev eth0  metric 256  mtu 1500 advmss 1220
fe80::/64 dev vlan0  metric 256  mtu 1500 advmss 1220
fe80::/64 dev eth1  metric 256  mtu 1500 advmss 1220
fe80::/64 dev br0  metric 256  mtu 1500 advmss 1220
fe80::/64 dev vlan1  metric 256  mtu 1500 advmss 1220
fe80::/64 via :: dev sixxs  metric 256  mtu 1280 advmss 1220
ff00::/8 dev eth0  metric 256  mtu 1500 advmss 1220
ff00::/8 dev vlan0  metric 256  mtu 1500 advmss 1220
ff00::/8 dev eth1  metric 256  mtu 1500 advmss 1220
ff00::/8 dev br0  metric 256  mtu 1500 advmss 1220
ff00::/8 dev vlan1  metric 256  mtu 1500 advmss 1220
```

```
ff00::/8 dev sixxs  metric 256  mtu 1280 advmss 1220
default via 2001:6f8:202:e::1 dev sixxs  metric 1024  mtu 1280 advmss 1220
```

Interface configuration of a client machine:

```
~$ ip addr show
1: lo:  mtu 16436 qdisc noqueue
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: sit0:  mtu 1480 qdisc noop
    link/sit 0.0.0.0 brd 0.0.0.0
3: eth0:  mtu 1500 qdisc pfifo_fast qlen 1000
    link/ether 00:11:2f:1e:bf:65 brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.42/24 brd 192.168.1.255 scope global eth0
    inet6 2001:6f8:309:1:211:2fff:fe1e:bf65/64 scope global dynamic
       valid_lft 2591812sec preferred_lft 604612sec
    inet6 fe80::211:2fff:fe1e:bf65/64 scope link
       valid_lft forever preferred_lft forever
```

====== Using IPv6 by default with Windows XP ======

Now you have 6to4 installed on your OpenWrt router with a radvd server, you can enable IPv6 on your Windows box by typing

```
netsh interface ipv6 install
```

at the command prompt. This will install IPv6 and you will get a 6to4 address. However Windows will only use it to communicate with other 6to4 addresses or other IPv6 only hosts by default (it will prefer IPv4 otherwise). To force IPv6 with dual stack non-6to4 hosts, use this:

```
C:\> netsh
netsh> interface ipv6
netsh interface ipv6> show prefixpolicy
Querying active state ...
Precedence  Label  Prefix
----------  -----  ------------------------
         5      5  3ffe:831f::/32
        10      4  ::ffff:0:0/96
        20      3  ::/96
        30      2  2002::/16
        40      1  ::/0
        50      0  ::1/128

netsh interface ipv6> set prefixpolicy
One or more essential parameters were not entered.
Verify the required parameters, and reenter them.
The syntax supplied for this command is not valid. Check help for the correct syntax.
Usage: set prefixpolicy [prefix=]/ [precedence=]
             [label=] persistent]
Parameters:
       Tag              Value
       prefix         - Prefix for which to add a policy.
       precedence     - Precedence value for ordering.
       label          - Label value for matching.
       store          - One of the following values:
                        active: Change only lasts until next boot.
                        persistent: Change is persistent (default).
Remarks: Modifies a source and destination address selection policy
         for a given prefix.
Example:
       set prefixpolicy ::/96 3 4
netsh interface ipv6>set prefixpolicy ::1/128 50 0
Ok.
netsh interface ipv6>set prefixpolicy ::/0 40 1
Ok.
netsh interface ipv6>set prefixpolicy 2002::/16 30 1
Ok.
```

```
netsh interface ipv6>set prefixpolicy ::/96 20 3
Ok.
netsh interface ipv6>set prefixpolicy ::ffff:0:0/96 10 4
Ok.
netsh interface ipv6>set prefixpolicy 3ffe:831f::/32 5 5
Ok.
netsh interface ipv6>show prefixpolicy
Querying active state...
Precedence  Label  Prefix
----------  -----  --------------------------------
         5      5  3ffe:831f::/32
        10      4  ::ffff:0:0/96
        20      3  ::/96
        30      1  2002::/16
        40      1  ::/0
        50      0  ::1/128
netsh interface ipv6>exit
C:\>
```

Notice how the same label is used for both 6to4 (2002::/16) and normal IPv6 (::/0) telling Windows they can be used together at each end of a communication link. Now if you go to an IPv6 enabled website (e.g. www.kame.net) you will connect to it using IPv6 instead of IPv4.

====== Using IPv6 by default with Windows Vista ======

Even though Vista comes pre-configured with IPv6 support it still only uses the stack to communicate with other 6to4 addresses or other IPv6 only hosts by default (it will prefer IPv4 otherwise). To force IPv6 with dual stack non-6to4 hosts the instructions are the same as above in the Windows XP howto with a couple minor differences. The "set prefixpolicy" command only works once, wiping out all other policies in the process. To recreate these policies you have to use the "add prefixpolicy" command. Also to show the current policies, you need to use the "show prefixpolicies" command instead of "show prefixpolicy".

Short method to editing policies:

```
C:\> netsh
netsh> interface ipv6
netsh interface ipv6> show prefixpolicies
Querying active state...
Precedence  Label  Prefix
----------  -----  --------------------------------
        50      0  ::1/128
        40      1  ::/0
        30      2  2002::/16
        20      3  ::/96
        10      4  ::ffff:0:0/96
         5      5  2001::/32

netsh interface ipv6> delete prefixpolicy 2001::/32
Ok.
netsh interface ipv6> add prefixpolicy 3ffe:831f::/32 5 5
Ok.
netsh interface ipv6> show prefixpolicies
Querying active state...
Precedence  Label  Prefix
----------  -----  --------------------------------
        50      0  ::1/128
        40      1  ::/0
        30      2  2002::/16
        20      3  ::/96
        10      4  ::ffff:0:0/96
         5      5  3ffe:831f::/32
netsh interface ipv6> exit
C:\>
```

Long method:

```
C:\> netsh
netsh> interface ipv6
netsh interface ipv6> show prefixpolicies
Querying active state...
Precedence  Label  Prefix
----------  -----  --------------------------------
        50      0  ::1/128
        40      1  ::/0
        30      2  2002::/16
        20      3  ::/96
        10      4  ::ffff:0:0/96
         5      5  2001::/32

netsh interface ipv6> set prefixpolicy ::1/128 50 0
Ok.
netsh interface ipv6> add prefixpolicy ::/0 40 1
Ok.
netsh interface ipv6> add prefixpolicy 2002::/16 30 1
Ok.
netsh interface ipv6> add prefixpolicy ::/96 20 3
Ok.
netsh interface ipv6> add prefixpolicy ::ffff:0:0/96 10 4
Ok.
netsh interface ipv6> add prefixpolicy 3ffe:831f::/32 5 5
Ok.
netsh interface ipv6> show prefixpolicies
Querying active state...
Precedence  Label  Prefix
----------  -----  --------------------------------
        50      0  ::1/128
        40      1  ::/0
        30      2  2002::/16
        20      3  ::/96
        10      4  ::ffff:0:0/96
         5      5  3ffe:831f::/32
netsh interface ipv6> exit
C:\>
```

====== Links ======

- IPv6 on OpenWrt with Hurricane Electric
- JOIN IPv6 Test Page (ping, traceroute, tracepath)
- Route Advertising Daemon Homepage
- Peter Bieringer's IPv6 HOWTO

====== ToDo ======

- list of IPv6 ready application available in OpenWrt
- start/stop radvd when connection goes up/down
- add firewall rules for incoming IPv6 connections

====== Questions ======

Q: How would I go about setting up radvd to announce an v6 address (6to4), derived from an DHCP assigned v4 address (it changes every few weeks)?

A: Change the prefix in the radvd.conf (first 3 segments for a /48) to 0, so `2001:db8:0:f101::/64` becomes `0:0:0:f101::/64`, and add "*Base6to4Interface ppp0;*" (where *ppp0* is your wan interface) to the section, and set *AdvValidLifetime* and *AdvPreferredLifetime* to a low number, so if the v4 address changes, the v6 routing info will be updated quickly, so the finished section would look something like this:

Announced prefix	2001:0db8:0000:0000:0000:0000:0000:0000/48
Subnet	2001:0db8:0000:f101:0000:0000:0000:0000/64
EUI-64 MAC	0010:08ff:fe40:1234/64
Configured address	2001:0db8:0000:f101:0010:08ff:fe40:1234/64

And yes I know the address is not in zero-compressed format. For looking at subnetting it's easier to look at the whole IPv6 address.

```
prefix 0:0:0:f101::/64
{
        AdvOnLink on;
        AdvAutonomous on;
        Base6to4Interface ppp0;
        # Very short lifetimes for dynamic addresses
        AdvValidLifetime 300;
        AdvPreferredLifetime 120;
};
```

That assumes ppp0 is your wan interface, and that you have a /48 address (according to http://ezine.daemonnews.org/200101/6to4.html you do get one with 6to4)