1 Introduction

FIXME write a better introduction This document will show you how to setup a little ipv6 network and how to connect to the 6bone using a static tunnel. While there are other ways to connect to the ipv6 world they are (not yet) discussed here.

2 Getting Ready

2.1 Ipv6 basics

An ipv6 address is 128 bit long and is represented by 32 hexadecimal characters. Because humans aren't very compatible with large numbers like this, a semicolon is insert every four characters. So a ipv6 could for example look like this:

```
3ffe:80ee:046d:0000:0000:96ff:fe2a:0001
```

to compress the address a little bit you allowed to leave out the leading zero's. So our address becomes:

```
3ffe:80ee:46d:0:0:96ff:fe2a:1
```

You can write this address even more compact because a sequense of zero's may be replace by :: . When doing that our adress can be written like this:

```
3ffe:80ee:46d::96ff:fe2a:1
```

The ipv6 localhost ip is

```
0000:0000:0000:0000:0000:0000:0000:0001
```

and can thus be written as

```
::1
```

Just like with ipv4 every every network has a netmask. And just as with ipv4 it can be represented with the slash notation. Thus for example your localhost address has can look like this `::1/128 . The current reserved address space for global unicast addressed is 2000::/3 and thus includes all ipv6 address from 2000:: untill 3fff:: .`

2.2 Needed Tools

`The following tools are used in this manual`

Command	Debian package	description
ping6	iputils-ping	ipv6 variant of ping
ip	iproute	Network configuration tool (ifconfig++)
ipv6calc	ipv6calc	a small utility which formats and calculates IPv6 addresses
radvd	radvd	Router Advertisment deamon (routers only)

2.3 Kernel Configuration

`The needed kernel options for various kernel. The standard packaged debian kernels should already have these options enabled. Please not that these are only the extra options you need for ipv6, you still need your normal kernel options for ipv4 networking.`

2.3.1 Kernel 2.2.x

- CONFIG_NETLINK
- CONFIG_NETLINK_DEV
- CONFIG_IPV6
- CONFIG_IPV6_EUI64
- CONFIG_IPV6_NO_PB

2.3.2 Kernel 2.4.x

- CONFIG_NETLINK_DEV
- CONFIG_IPV6

2.4 Getting a ipv6 prefix

If your lan/provider doesn't have native ipv6 support, you need to get your own little bit of ipv6 address space and a way to hookup to the rest of the ipv6 world. Tunnelbrokers provide exactly this type of service. They will give you a ipv6 prefix and will setup a tunnel for you to route your traffic through. It's important that the machine that's providing the tunnel for you is a close as possible to networkwise as possible. A few tunnelbrokers include

xs26: xs26 provides it's own ipv6 backbone which interconnects with the 6bone. There various PoP's around the world, so there is probably always one close to you. Provides a /48 zone deligation, static tunnels and DNS delegations. Website: http://xs26.net

freenet6: An canada based tunnelbroker wich uses a specialised client to setup tunnels to your ipv6 prefix dynamicly. The client is in the freenet6 debian package Website: http://xs26.net

Various others: These can be found on http://hs247.com/

3 Host setup

You should be running an ipv6 capable kernel by now. So your interfaces already have at least one ipv6 address. so the command

```
$ ip addr show
```

would include a line like

```
inet6 fe80::201:2ff:fee1:4c19/10 scope link
```

This is the link local address of that interface. It can be used only used between machines on the same link and provides a sort of ad-hoc plug and play networking (yeah buzzwords). To ping all the ipv6 capable nodes on the same link as one of your interfaces you can use

```
$ ping6 -I <interface> ff02::1
```

which results in an output like

```
$ ping6 -I eth0 ff02::1
PING ff02::1(ff02::1) from fe80::201:2ff:fee1:4c19 eth0: 56 data bytes
64 bytes from ::1: icmp_seq=1 ttl=64 time=0.088 ms
64 bytes from fe80::220:afff:fe35:e3c7: icmp_seq=1 ttl=64 time=0.948 ms (DUP!)
64 bytes from fe80::280:5aff:fe12:a00d: icmp_seq=1 ttl=64 time=1.29 ms (DUP!)
64 bytes from ::1: icmp_seq=2 ttl=64 time=0.090 ms
```

In case your lan is already ipv6 enabled and uses autoconfiguration, you should have also gotten an global ipv6 address. Do another $ ip addr show and look for output like

```
inet6 3ffe:80ee:46d:0:201:2ff:fee1:4c19/64 scope global dynamic
```

This means that your machine already had an autoconfigured ipv6 address. So you can try to ping6 www.kame.net and you've just set your first steps in the ipv6 world.

4 Router setup

If you want to your own lan to support ipv6 you will need to setup an ipv6 router. And probably some autoconfiguration

4.1 Interface setup

The routers network interfaces shouldn't be autoconfigurated so you will need to set some kernel parameters to do this. And ofcourse your router should be forwarding packets. Just setup you /etc/sysctl.conf to include the following:

```
net.ipv6.conf.all.autoconf = 0
net.ipv6.conf.all.accept_ra = 0
net.ipv6.conf.all.accept_redirects = 0
net.ipv6.conf.all.forwarding = 1
net.ipv6.conf.all.router_solicitations = 0
```

and run # sysctl -p.

A ipv6 ethernet subnet always has a 64 bit netmask. So if your tunnelbroker provides you with al zone like this 3ffe:80ee:46d::/48. You will need to pick out one 64 bit subnet out of that. For example 3ffe:80ee:46d::/64. Now you should pick a ipv6 address for your router out of that subnet. It's shouldn't be a problem if you pick an address that can be autoconfigured, but it's not very nice. Fortunaly ipv6 generates the addresses out of the network cards mac address in a predictable way. I don't want to discuss the details, you just need to know that byte 14 is 0xff. So just don't choose a *:*:*:*:*:*ff:*:* address your ok. So for example we could choose 3ffe:80ee:46d::/128 for our router. Configuring the eth0 interface with ip goes like this:

```
# ip link set eth0 up
# ip addr add 3ffe:80ee:46d::/128  eth0
# ip addr show eth0
2: eth0: <BROADCAST,MULTICAST,UP> mtu 1500 qdisc pfifo_fast qlen 100
    link/ether 00:20:af:35:e3:c7 brd ff:ff:ff:ff:ff:ff
    inet 192.168.0.1/24 brd 192.168.0.255 scope global eth0
    inet6 fe80::220:afff:fe35:e3c7/10 scope link
    inet6 3ffe:80ee:46d::/64 scope global
```

Or just put the following in /etc/network/interface

```
iface eth0 inet6 static
        address 3ffe:80ee:046d::/64
        netmask 64
```

4.2 Router advertisment deamon

Radvd is a daemon advertises rrouters/networks. Any autoconfiguring ipv6 host wich joins the network, will sendout such an router solicitation and radvd will answer with a router advertisement. This will tell the host on which network prefix it's located so it can generated it's ipv6 address and what the default router is for this network. To setup radvd on our example network we should put the following in our radvd.conf

```
interface eth0
{
     AdvSendAdvert on;

    prefix 3ffe:80ee:046d::/64
    {
      AdvOnLink on;
      AdvAutonomous on;
    };
};
```

And start the daemon with # /etc/init.d/radvd start . So now if you put another ipv6 host on your lan, it will get an ipv6 addres automagically

4.3 Tunnel configuration

To lay a tunnel to your tunnel broker you should have gotten the ipv4 address of the tunnel endpoint you should use. If your tunnel broker hasn't given you an ipv6 address for your tunnel endpoint. You should pick one yourself out of your prefix. In our example prefix that could be 3ffe:80ee:046d:ffff:ffff:ffff:ffff:0/128. In this example setup my tunnel endpoint will be a fictional ipv4 addres namely 10.0.0.1 Setting up this tunnel goes as follows

```
# ip tunnel add mytunnel mode sit remote 10.0.0.1
# ip link set mytunnel up
# ip addr add 3ffe:80ee:046d:ffff:ffff:ffff:ffff:0/128 dev mytunnel
# ip route add 2000::/3 dev mytunnel
```

Ofcourse you can also use /etc/network/interfaces. Just put the following in:

```
auto mytunnel
iface mytunnel inet6 v4tunnel
  address 3ffe:80ee:046d:ffff:ffff:ffff:ffff:0
  netmask 128
  endpoint 10.0.0.1
  up ip route add 2000::/3 dev mytunnel
```

If all went ok, you should be able to ping6 some ipv6 enabled machines for example www.kame.net.

5 DNS setup

5.1 Local zone lookup

5.2 Forward zone lookup

5.3 Reverse zone lookup