

Connecting to the IPv6 Internet

Ibrahim Haddad

In a previous article published on LinuxDevCenter.com, we introduced the IPv6 protocol and demonstrated how to support IPv6 on your Linux machine. This article goes a bit further into demonstrating how to connect your Linux machine to the IPv6 Internet (also known as the 6bone) using the Freenet6 Tunnel Server Protocol (TSP).

The IPv6 Internet

The IPv6 Internet was established in 1996 as a test network for IPv6. At the beginning, most of the backbone was established using tunnels over the current IPv4 Internet, making it appear as a virtual IPv6 network. Currently, the IPv6 Internet is made of both IPv6 native links and tunneled links over the IPv4 Internet.

The initial goal of the IPv6 Internet was to provide an experimental worldwide network for testing standards and IPv6 implementations. Years after its creation, the IPv6 Internet has changed its focus in a new direction: testing of transition and operational procedures. For many years to come, IPv4 and IPv6 will co-exist before a complete migration to IPv6 takes place. For this reason, the IETF has developed a number of co-existence and transition techniques that should be applied to help adopters migrate toward a full IPv6 deployment. In this context, the IPv6 Internet is playing a major role as a test IPv6 backbone for transitional mechanisms and operational procedures.

Connecting to the IPv6 Internet

To connect to the IPv6 Internet, you need a provider that offers the service. Quite a few ISPs offer IPv6 connectivity depending on where you live. If you cannot find one directly or if your current provider does not offer the service, then the easiest and cheapest solution to connect to the IPv6 Internet is to create a tunnel to a provider or a site that is willing to offer you the transit service (sometimes for free). Hexago (a spin-off of Viagénie) started the Freenet6 initiative to help people experiment and deploy IPv6. Freenet6 offers a free and automated tunnel service that can connect any individual or organization to the IPv6 Internet.

For the purpose of clarity, we will first cover the concept of tunneling, discuss which Freenet6 connection model is best for you, and follow a step-by-step tutorial on how to set up the connection.

Tunneling

The IETF has standardized tunneling as the transitional method to deploy IPv6, in coexistence with IPv4. A tunnel encapsulates IPv6 packets over IPv4. As a result, IPv6 hosts will be able to establish a link to the IPv6 Internet through an IPv4 connection. An IPv6-over-IPv4 tunnel is established with both endpoints configuring the IPv4 and the IPv6 address of the other endpoint. When one of the endpoints changes its IPv4 address, both endpoints of the tunnel need to change their configuration accordingly. This is especially cumbersome when the IPv4 node has a dialup connection or if it changes addresses often.

Freenet6 Tunnel Server Protocol provides an IPv6-over-IPv4 tunneling implementation that overcomes this problem: each time the tunnel client changes its IPv4 address, for instance at boot time if the host is configured for a DHCP service, the TSP client sends updated and authenticated information to the server, so the tunnel remains active without any reconfiguration.

Freenet6 Tunnel Server Protocol (TSP)

As previously mentioned, the Freenet6 service was the first public tunnel server service, and the most used in the world to delegate automatically one single IPv6 address to any host already connected to an IPv4 network over configured tunnels.

Connecting to the IPv6 Internet

Ibrahim Haddad

Freenet6's TSP is based on a client/server approach. It uses a protocol where a client requests one single IPv6 address or a full IPv6 prefix from a tunnel server. TSP is modeled after the tunnel broker (RFC 3053) where an IPv6-over-IPv4 tunnel is established between a node and the tunnel broker. However, Freenet6 is an enhanced version, where the node is using a tunnel-setup protocol to negotiate the establishment of the tunnel with the server. The client node, in this case, may be a host or a router.

The TSP server of Freenet6 provides not only tunnels but also a large address space to any user of the service. The address space provided is a /48 network, giving 2^{16} subnets, each of which may have up to 264 nodes -- more addresses than the entire current IPv4 Internet address space. This address space is assigned to the user of the service to enable any user, university, or organization to have the freedom of an abundance of addresses for servers and services that were not easy to do with NAT in IPv4.

How the Tunnel Is Established

Here are the steps that take place when establishing a tunnel session using the Freenet6 TSP:

- The IPv6 host, which has a connection to the Internet, initiates a request to the tunnel server by starting the TSP client program. This client will request a tunnel according to the specifications inside the TSP configuration file on the host machine.
- The TSP server processes the request and assigns either a single IPv6 address or a full IPv6 prefix to the requester (depending on the type of request).
- The TSP server will then establish the IPv6-over-IPv4 tunnel according to the information sent in the request.
- The client will receive the tunnel configuration information from the tunnel server and will configure its tunnel interface as well as its default IPv6 routes.
- The IPv6 host (the client machine) now has full IPv6 connectivity.

Freenet6 has a detailed description on how TSP works.

Freenet6 TSP Connection Models

TSP supports two connection models.

In the Single Host Connection Model, as shown in Figure 1, only a single host is connected to the IPv6 Internet using the Freenet6 service.

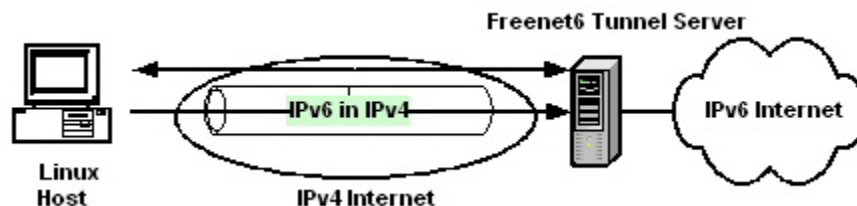


Figure 1. Connecting a Single Machine.

In the Multiple Hosts Connection Model, as shown in Figure 2, you can connect a full network to the IPv6 Internet using the Freenet6 service. What differs from the first scenario is the need for a machine

Connecting to the IPv6 Internet

Ibrahim Haddad

to act as an IPv6 router, which will be providing router advertisement for the hosts on your network to auto-configure themselves for IPv6.

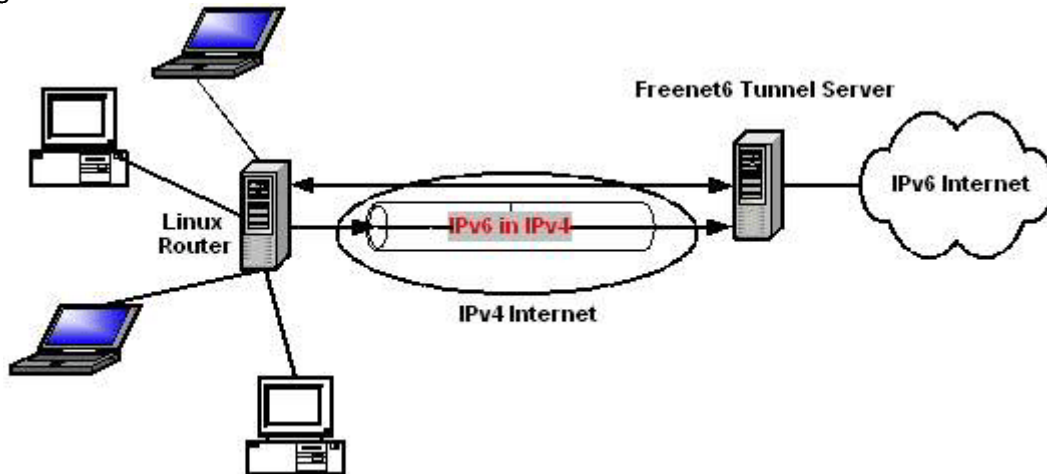


Figure 2. Connecting a Full Network

Freenet6 TSP Requirements

Your Linux host should meet some basic requirements to be able to use the Freenet6 TSP service:

- Support for IPv6. The host must support IPv6. We covered this topic in the previous article.
- A Public IPv4 Address. Tunnel servers do not accept private addresses. Your Linux host must have a public IPv4 address.
- Root Access. You need to have root access to install and configure the Freenet6 TSP client program.
- Enabled TSP Ports on Firewall/Router. If your Linux host is behind a firewall, firewalls and routers on the client side must allow protocol number 41 and TCP port 4343 between Freenet6 and the end-user network to allow IPv6 connectivity from Freenet6.
- A NAT-Free Environment. If an end-user is behind a NAT gateway, it is not possible to get IPv6-over-IPv4 traffic from any tunnel server, except when the NAT gateway handles static NAT addressing. The network administrator could map one Internet unicast globally unique IP address to the end-user Linux host behind the NAT. This means the local network administrator will control and authorize this special configuration for end-users. Please note that Freenet6 is working on a NAT traversal technique that will be available in the near future. It will enable the establishment of a tunnel over NAT without any modification of the NAT gateway. This will be of benefit for those of us who would like to connect to the IPv6 Internet but are inside a NAT environment.

Setting the Tunnel and Connecting a Single Host

We will demonstrate how to set up your Linux host to connect to the IPv6 Internet using Freenet6 TSP. We assume that you already meet the requirements previously presented. My own test installation used Fedora Core.

To proceed with the installation and configuration please follow these steps:

1. Enable IPv6 support.

Connecting to the IPv6 Internet

Ibrahim Haddad

In my case, I am loading the IPv6 module.

```
[root@fedora-core bin]# insmod ipv6
Using /lib/modules/2.4.22-1.2115.nptl/kernel/net/ipv6/ipv6.o
```

When I check the status on my network device, I can see that IPv6 support is enabled.

```
[root@fedora-core bin]# ifconfig
eth0      Link encap:Ethernet  HWaddr 00:B0:D0:A4:A9:FA
          inet6 addr: fe80::2b0:d0ff:fea4:a9fa/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:1547 errors:0 dropped:0 overruns:0 frame:0
          TX packets:1424 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:1319368 (1.2 Mb)  TX bytes:181558 (177.3 Kb)
          Interrupt:10 Base address:0xfc00

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:2348 errors:0 dropped:0 overruns:0 frame:0
          TX packets:2348 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:1672742 (1.5 Mb)  TX bytes:1672742 (1.5 Mb)

ppp0     Link encap:Point-to-Point Protocol
          inet addr:67.69.185.115  P-t-P:64.230.254.136  Mask:255.255.255.255
          UP POINTOPOINT RUNNING NOARP MULTICAST  MTU:1492  Metric:1
          RX packets:1478 errors:0 dropped:0 overruns:0 frame:0
          TX packets:1349 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:3
          RX bytes:1282448 (1.2 Mb)  TX bytes:147037 (143.5 Kb)
```

2. Register a Freenet6 user account.

Registering a user account is mandatory when you need to receive a permanent IPv6 address for your Linux host that can keep the IPv6 address, although the IPv4 address changes. To register a user account, go to Freenet6's registration page and provide a user name and your email address. Freenet6 will generate a password for you and email it to the address you provided.

Once you fill out the form and submit it, you will receive a confirmation similar to the one below:

```
User Account Creation Succeed
Userid (account) created : 'username'.
A new password generated randomly by the system has been sent to
username@example.com
```

The email you receive few seconds later from Freenet6 will look like this:

```
<Your account has been created on the Migration Broker>
User id: username
Password: XXXXXXXXXXXX
Email address registred: username@example.com
```

Connecting to the IPv6 Internet

Ibrahim Haddad

Note: Use these values with TSP client to get your IPv6 connectivity.
</Your account has been created on the Migration Broker>

```
<CUT AND PASTE TO TSPC.CONF>
#
userid=username
passwd=XXXXXXXXX
#
</CUT AND PASTE TO TSPC.CONF>
```

3. Download and install the TSP client.

Download the latest TSP package that corresponds to your Linux distribution. I will be using freenet6-0.9.8.tgz. However, you can download either the binary package or the RPM package if you like. For the remainder of the tutorial I will be using freenet6-0.9.8.tgz to demonstrate the procedure. Please replace this with the package name you downloaded for your specific system. After you download the package (say, into /tmp), you need to install it:

```
[root@fedora-core tmp]# tar -xzf freenet6-0.9.8.tgz
```

This will unpack the source package for the Freenet6 TSP. Next, you need to switch to that directory and compile and build the binaries.

```
[root@fedora-core tmp]# cd freenet6-0.9.8
[root@fedora-core freenet6-0.9.8]# make install target=linux
installdir=/usr/local/tsp
```

This command will start the compilation process on a Linux machine, specified by the target=linux directive, and will automatically install the binaries in /usr/local/tsp, the destination directory. You can change this to a directory of your choice.

4. Edit the TSP client configuration file.

tspc.conf is located under the installation directory, /usr/local/tsp/bin by default. It controls the configuration of the TSP client. You need to edit it and add your registered user ID and password, as you received them from Freenet6 by email.

```
<CUT AND PASTE TO TSPC.CONF>
#
userid=username
passwd=XXXXXXXXX
#
</CUT AND PASTE TO TSPC.CONF>
```

5. Start the tspc client.

You are now ready to start using the TSP client to create an IPv6-over-IPv4 tunnel to the IPv6 Internet.

```
[root@fedora-core root]# cd /usr/local/tsp/
[root@fedora-core bin]# ./tspc -vf tspc.conf
tspc - Tunnel Server Protocol Client
Loading configuration file
```

Connecting to the IPv6 Internet

Ibrahim Haddad

```
Connecting to server
Using [67.69.185.115] as source IPv4 address.
Send request
Process response from server
TSP_HOST_TYPE          host
TSP_TUNNEL_INTERFACE  sit1
TSP_HOME_INTERFACE
TSP_CLIENT_ADDRESS_IPV4  67.69.185.115
TSP_CLIENT_ADDRESS_IPV6  3ffe:0bc0:8000:0000:0000:0000:1bf9
TSP_SERVER_ADDRESS_IPV4  206.123.31.115
TSP_SERVER_ADDRESS_IPV6  3ffe:0bc0:8000:0000:0000:0000:1bf8
TSP_TUNNEL_PREFIXLEN    128
TSP_VERBOSE            1
TSP_HOME_DIR           /usr/local/tsp
```

```
--- Start of configuration script. ---
Script: linux.sh
sit1 setup
Setting up link to 206.123.31.115
This host is: 3ffe:0bc0:8000:0000:0000:0000:1bf9/128
Adding default route
--- End of configuration script. ---
```

```
Exiting with return code : 0 (0 = no error)
[root@fedora-core bin]#
```

The -v flag in the command line indicates the verbose mode, which lets you see exactly what is happening during the process. Let's examine the network interfaces after successfully establishing the tunnel:

```
[root@fedora-core bin]# ifconfig
eth0      Link encap:Ethernet  HWaddr 00:B0:D0:A4:A9:FA
          inet6 addr: fe80::2b0:d0ff:fea4:a9fa/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:1635 errors:0 dropped:0 overruns:0 frame:0
          TX packets:1516 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:1327322 (1.2 Mb)  TX bytes:189687 (185.2 Kb)
          Interrupt:10 Base address:0xfc00

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:2348 errors:0 dropped:0 overruns:0 frame:0
          TX packets:2348 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:1672742 (1.5 Mb)  TX bytes:1672742 (1.5 Mb)

ppp0     Link encap:Point-to-Point Protocol
          inet addr:67.69.185.115  P-t-P:64.230.254.136  Mask:255.255.255.255
          UP POINTOPOINT RUNNING NOARP MULTICAST  MTU:1492  Metric:1
          RX packets:1554 errors:0 dropped:0 overruns:0 frame:0
          TX packets:1429 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:3
          RX bytes:1288010 (1.2 Mb)  TX bytes:152686 (149.1 Kb)
```

Connecting to the IPv6 Internet

Ibrahim Haddad

```
sit1    Link encap:IPv6-in-IPv4
        inet6 addr: fe80::4345:b973/64 Scope:Link
        inet6 addr: 3ffe:bc0:8000::1bf9/128 Scope:Global
        UP POINTOPOINT RUNNING NOARP  MTU:1472  Metric:1
        RX packets:1 errors:0 dropped:0 overruns:0 frame:0
        TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
        collisions:0 txqueuelen:0
        RX bytes:72 (72.0 b)  TX bytes:0 (0.0 b)
```

```
[root@fedora-core bin]#
```

The tunnel interface is sit1. The global IPv6 address assigned to this interface is 3ffe:bc0:8000::1bf9 and the local scope IPv6 address is fe80::4345:b973. The ppp0 interface is my xDSL connection providing me with an IPv4 address and a connection to the Internet.

The easiest way to test your connection is to ping6 some web sites that support IPv6.

```
[root@fedora-core bin]# ping6 www.6bone.net
PING www.6bone.net(www.6bone.net) 56 data bytes
64 bytes from www.6bone.net: icmp_seq=0 ttl=62 time=60.1 ms
64 bytes from www.6bone.net: icmp_seq=1 ttl=62 time=59.8 ms
64 bytes from www.6bone.net: icmp_seq=2 ttl=62 time=48.1 ms
64 bytes from www.6bone.net: icmp_seq=3 ttl=62 time=49.1 ms
64 bytes from www.6bone.net: icmp_seq=4 ttl=62 time=75.3 ms

--- www.6bone.net ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 80932ms
rtt min/avg/max/mdev = 48.185/58.530/75.303/9.807 ms, pipe 2
[root@fedora-core bin]#
```

```
[root@fedora-core bin]# ping6 www.kame.net
PING www.6bone.net(www.6bone.net) 56 data bytes
64 bytes from www.6bone.net: icmp_seq=0 ttl=62 time=63.7 ms
64 bytes from www.6bone.net: icmp_seq=1 ttl=62 time=60.5 ms
64 bytes from www.6bone.net: icmp_seq=2 ttl=62 time=57.4 ms
64 bytes from www.6bone.net: icmp_seq=3 ttl=62 time=47.2 ms
64 bytes from www.6bone.net: icmp_seq=4 ttl=62 time=50.8 ms
--- www.6bone.net ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 80980ms
rtt min/avg/max/mdev = 47.265/55.972/63.749/6.092 ms, pipe 2

[root@fedora-core bin]#
```

Conclusion

In this article, we demonstrated how to connect your Linux machine to the IPv6 Internet using the Freenet6 service. In an upcoming article, we'll show you how to connect a full network to the IPv6 Internet using a Linux machine as a router. Stay tuned!