A basic introduction to IPv6 networking and security tools

version 1.0 by xxradar
mailto:xxradar@radarhack.com

1 Introduction

Reading some articles stating that Ipv6 projects are starting off in Japan, China, etc... and the advent of UMTS projects using IPv6 backbones, it inspired me to see if I could setup an IPv6 network myself and if indeed applications are available.
After a few searches on Google, I learned that most current releases of Linux are already equipped with an IPv6 ready kernel and IP stack.
There is even very good support WinXPsp1 en Windows2003.
There is also a driver available for W2k (sp2 only).

Following systems were used in examples and demos:
RedHat9.0, Suse8.2, Knoppix-STD and WinXP SP1.

2 Checking if IPv6 is loaded.

First of all, let us make sure IPv6 is running on Linux.
Do a “lsmod |grep ipv6”, and see what it says.
If no module is loaded, run “modprobe ipv6” to load the IPv6 module.

```
linux:~ # lsmod |grep ipv6
ipv6                  134388  -1  (autoclean)
linux:~
```

To get Ipv6 on windows, you need to install ‘Advanced Networking Pack for Windows XP KB817778’. Then type ‘ipv6 install’ on the command line.

3 Check for available IPv6 addresses

You can easily figure out your IPv6 addresses by doing an “ifconfig –a”.

```
eth1  Link encap:Ethernet  HWaddr 00:04:E2:00:D0:17
      inet addr:192.168.10.33  Bcast:192.168.10.255  Mask:255.255.255.0
      inet6 addr: fe80::204:e2ff:fe00:d017/64 Scope:Link
      UP BROADCAST NOTRAILERS RUNNING MULTICAST  MTU:1500  Metric:1
      RX packets:3122 errors:0 dropped:0 overruns:0 frame:0
      TX packets:2443 errors:0 dropped:0 overruns:0 carrier:0
      collisions:0
      RX bytes:3328381 (3.1 Mb)  TX bytes:271625 (265.2 Kb)

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:43 errors:0 dropped:0 overruns:0 frame:0
      TX packets:43 errors:0 dropped:0 overruns:0 carrier:0
      collisions:0
      RX bytes:3308 (3.2 Kb)  TX bytes:3308 (3.2 Kb)
```

```
sit0    Link encap:IPv6-in-IPv4
        NOARP  MTU:1480  Metric:1
        RX packets:0 errors:0 dropped:0 overruns:0 frame:0
        TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
        collisions:0
        RX bytes:0 (0.0 b)  TX bytes:0 (0.0 b)
```

There are few addresses of interest for this tutorial. The link local address and localhost. We will add addresses later.
Link local addresses are addresses assigned automatically to each interface, which are not routable and based on the MAC address of the interface (supposed to be unique). Where does this address come from?

HWaddr	***00:04:E2:00:D0:17***
inet6 addr:	***fe80:0000:0000:0000:0204:e2ff:fe00:d017***
	fe80::204:e2ff:fe00:d017

-The second bit of the last byte of the MAC address is inverted
-fe80:0000:0000:0000 indicates the scope, link local (aka. Reserved network /64)
-ff:fe is used to convert the MAC address into a 64 unique IPv6 host address

Check out RFC3515 for more details on IPv6 addresses
The localhost address ::1 can be compared with the well know IPv4 127.0.0.1 address.

<u>4 Doing an IPv6 ping</u>

Pinging in IPv6 is supported by the command line utility *ping6.*

```
linux:~ # ping6 -I eth1 ::1
PING ::1(::1) from ::1 eth1: 56 data bytes
64 bytes from ::1: icmp_seq=1 ttl=64 time=0.123 ms
64 bytes from ::1: icmp_seq=2 ttl=64 time=0.087 ms
64 bytes from ::1: icmp_seq=3 ttl=64 time=0.077 ms

--- ::1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 1998ms
rtt min/avg/max/mdev = 0.077/0.095/0.123/0.022 ms
linux:~ #

linux:~ # ping6 -I eth1 fe80::24f:4eff:fe07:e1bb
PING fe80::24f:4eff:fe07:e1bb(fe80::24f:4eff:fe07:e1bb) from
fe80::204:e2ff:fe00:d017 eth1: 56 data bytes
64 bytes from fe80::24f:4eff:fe07:e1bb: icmp_seq=1 ttl=64 time=0.250 ms
64 bytes from fe80::24f:4eff:fe07:e1bb: icmp_seq=2 ttl=64 time=0.225 ms
64 bytes from fe80::24f:4eff:fe07:e1bb: icmp_seq=3 ttl=64 time=0.235 ms
```

```
64 bytes from fe80::24f:4eff:fe07:e1bb: icmp_seq=4 ttl=64 time=0.218 ms

--- fe80::24f:4eff:fe07:e1bb ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3011ms
rtt min/avg/max/mdev = 0.218/0.232/0.250/0.012 ms
linux:~ #
```

To make it easy, make a reference in the /etc/hosts file!
It simplifies things a lot!

```
linux:~ # cat /etc/hosts
#
# IP-Address  Full-Qualified-Hostname  Short-Hostname
#
127.0.0.1       localhost
# special IPv6 addresses
::1             localhost ipv6-localhost ipv6-loopback
fe00::0                         ipv6-localnet
ff00::0                         ipv6-mcastprefix
ff02::1                         ipv6-allnodes
ff02::2                         ipv6-allrouters
ff02::3                         ipv6-allhosts

127.0.0.2                       linux.local     linux
fe80::24f:4eff:fe07:e1bb        host6rh90
fe80::204:e2ff:fe00:d017        host6su82
linux:~ #


linux:~ # ping6 -I eth1 host6rh90
PING host6rh90(host6rh90) from fe80::204:e2ff:fe00:d017 eth1: 56 data
bytes
64 bytes from host6rh90: icmp_seq=1 ttl=64 time=0.238 ms
64 bytes from host6rh90: icmp_seq=2 ttl=64 time=0.214 ms
64 bytes from host6rh90: icmp_seq=3 ttl=64 time=0.219 ms

--- host6rh90 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 1998ms
rtt min/avg/max/mdev = 0.214/0.223/0.238/0.020 ms
linux:~ #
```

5 Showing the IPv6 address and layer 2 address mapping

```
linux:~ # ip -6 neigh show dev eth1
fe80::24f:4eff:fe07:e1bb lladdr 00:4f:4e:07:e1:bb nud stale
linux:~ #
```

This is in fact the new ARP table (arp -a on most systems).
The ARP protocol is not used anymore and replaced with an
IPv6 ICMP alternative.

The ARP request is replaced by an ICMPv6 'Neighbor Solicitation' message. An ICMPv6 'Neighbor advertisement' provides the answer.
You can capture and decode IPv6 traffic by using tcpdump or ethereal.

```
[root@localhost root]# tcpdump -i eth1
09:34:27.618717 fe80::204:e2ff:fe00:d017 > ff02::1:ff07:e1bb: icmp6:
neighbor sol: who has fe80::24f:4eff:fe07:e1bb
09:34:27.618834 fe80::24f:4eff:fe07:e1bb > fe80::204:e2ff:fe00:d017:
icmp6: neighbor adv: tgt is fe80::24f:4eff:fe07:e1bb
09:34:27.618966 fe80::204:e2ff:fe00:d017 > fe80::24f:4eff:fe07:e1bb:
icmp6: echo request
09:34:27.619016 fe80::24f:4eff:fe07:e1bb > fe80::204:e2ff:fe00:d017:
icmp6: echo reply
09:34:28.617666 fe80::204:e2ff:fe00:d017 > fe80::24f:4eff:fe07:e1bb:
icmp6: echo request
09:34:28.617741 fe80::24f:4eff:fe07:e1bb > fe80::204:e2ff:fe00:d017:
icmp6: echo reply
09:34:29.625752 fe80::204:e2ff:fe00:d017 > fe80::24f:4eff:fe07:e1bb:
icmp6: echo request
09:34:29.625832 fe80::24f:4eff:fe07:e1bb > fe80::204:e2ff:fe00:d017:
icmp6: echo reply
09:34:30.625722 fe80::204:e2ff:fe00:d017 > fe80::24f:4eff:fe07:e1bb:
icmp6: echo request
09:34:30.625799 fe80::24f:4eff:fe07:e1bb > fe80::204:e2ff:fe00:d017:
icmp6: echo reply
[root@localhost root]#
```

Find a more detailed trace of the 'Neighbor Solicitation" message.

```
9:34:27.618717 0:4:e2:0:d0:17 33:33:ff:7:e1:bb ip6 86:
fe80::204:e2ff:fe00:d017 >  ff02::1:ff07:e1bb : icmp6: neighbor sol: who
has fe80::24f:4eff:fe07:e1bb
```

The **solicited-node multicast** address consists of the prefix FF02::1:FF00:0/104 and the last 24-bits of the IPv6 address that is being resolved fe80::24f:4eff:fe07:e1bb, so f02::1:ff07:e1bb. This is mapped to the multicast MAC address 33:33:ff:7:e1:bb, instead of a L2 broadcast address. This L2 multicast address was set to be received when you loaded the IPv6 stack on the remote machine.

```
root@localhost root]# cat /proc/net/dev_mcast
2    eth0            1     0     3333ff07e1bb
2    eth0            1     0     333300000001
2    eth0            1     0     01005e000001
[root@localhost root]#
```

5 Pinging the ipv6 all nodes address.

```
linux:~ # ping6 -I eth1 ff02::1
PING ff02::1(ff02::1) from fe80::204:e2ff:fe00:d017 eth1: 56 data bytes
64 bytes from ::1: icmp_seq=1 ttl=64 time=0.118 ms
64 bytes from fe80::24f:4eff:fe07:e1bb: icmp_seq=1 ttl=64 time=0.347 ms (DUP!)
64 bytes from ::1: icmp_seq=2 ttl=64 time=0.094 ms
64 bytes from fe80::24f:4eff:fe07:e1bb: icmp_seq=2 ttl=64 time=0.276 ms (DUP!)

--- ff02::1 ping statistics ---
2 packets transmitted, 2 received, +2 duplicates, 0% packet loss, time 1005ms
rtt min/avg/max/mdev = 0.094/0.208/0.347/0.107 ms
linux:~ #
```

6 Taking it a step further.

Let us assign our own Ipv6 addresses.
To make things easier, choose easy (but real) Ipv6 addresses in the network. You need to do this, because most 'ipv6' aware applications do not use Link Local addresses.

I hooked up an WinXpSP1 machine and gave it the address 3ffe::3, by typing 'ipv6 adu 4/3ffe::3'. The 4/ indicates the interface index, which can be found when you type 'ipconfig /All'

We can now use the XP telnet client, ping and IE as client software.

I installed apache 2.0.47 for win32, as well as the Ipv6 patch. This yielded me some Ipv6 server software, such as a web server and the Ipv6 aware 'simple TCP/UDP services' included in WinXp.

The syntax for Linux for adding additional addresses is '/sbin/ip –6 addr add 3ffe::1/64 dev eth0'.

There is some server/client Ipv6 software available for Linux, but I used the latest NMAP release, netcat6 for Linux, as well as SOCAT for the testing purposes.

To check everything, do a 'ping6' to see if everything is working.

Analysing UDP/TCP on top of Ipv6.

```
[root@localhost root]# nc6 -6 3ffe::3 19
ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_`abcdefghijklmnopqrstuvwxyz{|} !"#$%&'()*
BCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_`abcdefghijklmnopqrstuvwxyz{|} !"#$%&'()*+
CDEFGHIJKLMNOPQRSTUVWXYZ[\]^_`abcdefghijklmnopqrstuvwxyz{|} !"#$%&'()*+,
DEFGHIJKLMNOPQRSTUVWXYZ[\]^_`abcdefghijklmnopqrstuvwxyz{|} !"#$%&'()*+,-
EFGHIJKLMNOPQRSTUVWXYZ[\]^_`abcdefghijklmnopqrstuvwxyz{|} !"#$%&'()*+,-.
FGHIJKLMNOPQRSTUVWXYZ[\]^_`abcdefghijklmnopqrstuvwxyz{|} !"#$%&'()*+,-./
GHIJKLMNOPQRSTUVWXYZ[\]^_`abcdefghijklmnopqrstuvwxyz{|} !"#$%&'()*+,-./0
HIJKLMNOPQRSTUVWXYZ[\]^_`abcdefghijklmnopqrstuvwxyz{|} !"#$%&'()*+,-./01
IJKLMNOPQRSTUVWXYZ[\]^_`abcdefghijklmnopqrstuvwxyz{|} !"#$%&'()*+,-./012
JKLMNOPQRSTUVWXYZ[\]^_`abcdefghijklmnopqrstuvwxyz{|} !"#$%&'()*+,-./0123
KLMNOPQRSTUVWXYZ[\]^_`abcdefghijklmnopqrstuvwxyz{|} !"#$%&'()*+,-./01234

20:12:11.984560 3ffe::1.32906 > 3ffe::3.chargen: . ack 765161 win 4320
20:12:11.989833 3ffe::3.chargen > 3ffe::1.32906: . 765161:766601(1440) ack 1 win 17280
20:12:11.991661 3ffe::3.chargen > 3ffe::1.32906: . 766601:768041(1440) ack 1 win 17280
20:12:11.993539 3ffe::3.chargen > 3ffe::1.32906: . 768041:769481(1440) ack 1 win 17280
20:12:12.024556 3ffe::1.32906 > 3ffe::3.chargen: . ack 769481 win 1440
20:12:12.029932 3ffe::3.chargen > 3ffe::1.32906: . 769481:770921(1440) ack 1 win 17280
20:12:12.104547 3ffe::1.32906 > 3ffe::3.chargen: . ack 770921 win 0
```

```
[root@localhost root]# nc6 -6 -u -n 3ffe::3 7
this
is
a
test
[root@localhost root]#
```

```
20:10:38.851097 3ffe::1.32770 > 3ffe::3.echo: udp 4
20:10:40.377862 3ffe::1.32770 > 3ffe::3.echo: udp 3
20:10:41.581220 3ffe::1.32770 > 3ffe::3.echo: udp 5
```

Connecting to the web server via netcat6.

```
[root@localhost root]# nc6 -6 -n 3ffe::3 80
HEAD / HTTP/1.0

HTTP/1.1 200 OK
Date: Tue, 04 Nov 2003 19:16:44 GMT
Server: Apache/2.0.47 (Win32)
Content-Location: index.html.en
Vary: negotiate,accept-language,accept-charset
TCN: choice
Last-Modified: Thu, 03 May 2001 16:01:18 GMT
ETag: "fbb3-5d6-8ba74f80;fbcf-9dc-d0c58f00"
Accept-Ranges: bytes
Content-Length: 1494
Connection: close
Content-Type: text/html; charset=ISO-8859-1
Content-Language: en
Expires: Tue, 04 Nov 2003 19:16:44 GMT

[root@localhost root]#
```

Connecting an Ipv4 webbrowser on Linux via Socat to Ipv6

```
[root@localhost root]# socat TCP4-LISTEN:80 TCP6:3ffe::3:80
```

Point your browser to http://127.0.0.1 (or the address SOCAT is running) and it will be tunneled via Ipv6 to an Ipv6 capable webserver.

```
[root@localhost root]# tcpdump -i eth0 not port 22
tcpdump: listening on eth0
20:29:57.955313 3ffe::1 > ff02::1:ff00:3: icmp6: neighbor sol: who has 3ffe::3
20:29:57.957990 3ffe::3 > 3ffe::1: icmp6: neighbor adv: tgt is 3ffe::3
20:29:57.958062 3ffe::1.33138 > 3ffe::3.http: S 144853817:144853817(0) win 5760 <mss
1440,sackOK,timestamp 230750 0,nop,wscale 0>
20:29:57.960737 3ffe::3.http > 3ffe::1.33138: S 1298524274:1298524274(0) ack 144853818 win
17280 <mss 1440>
20:29:57.960835 3ffe::1.33138 > 3ffe::3.http: . ack 1 win 5760
20:29:57.965388 3ffe::1.33138 > 3ffe::3.http: P 1:447(446) ack 1 win 5760
20:29:57.974593 3ffe::3.http > 3ffe::1.33138: . 1:1441(1440) ack 447 win 16834
20:29:57.974706 3ffe::1.33138 > 3ffe::3.http: . ack 1441 win 8640
20:29:57.975003 3ffe::3.http > 3ffe::1.33138: P 1441:1935(494) ack 447 win 16834
...
```

Scanning with NMAP on an Ipv6 address.

```
[root@localhost root]# nmap -6 -sT 3ffe::3

Starting nmap 3.48 ( http://www.insecure.org/nmap/ ) at 2003-11-04 20:32
CET
Interesting ports on 3ffe::3:
(The 1646 ports scanned but not shown below are in state: closed)
PORT     STATE SERVICE
7/tcp    open  echo
9/tcp    open  discard
13/tcp   open  daytime
17/tcp   open  qotd
19/tcp   open  chargen
80/tcp   open  http
135/tcp  open  msrpc
1025/tcp open  NFS-or-IIS
1026/tcp open  LSA-or-nterm
1032/tcp open  iad3
2105/tcp open  eklogin

Nmap run completed -- 1 IP address (1 host up) scanned in 3.779 seconds
[root@localhost root]#
```

Conclusion.

This small paper is a description of my first steps in Ipv6. It started with the intention of setting up a small Ipv6 network, but quickly learned that it is not so simple. A lot of stuff has changed (protocols, command line, ….). After I initially got most of the things working, I stumbled over available 'security' tools like netcat, socat and nmap. This made me realize that security will be an even bigger issue, since most applications are in fact

modified Ipv4 applications, and things do not change much at the application layer. Configuring Ipv6 is more difficult, certainly in the beginning, and will be error prone.
Another interesting topic is, that is possible to tunnel existing traffic in Ipv6 traffic (and vice versa) on existing LANS, fooling some IDS system and other security measures.
Things to be discussed in next papers are a design with routers and a connecting to the Ipv6 backbone with 6to4 tunneling. If you feel that some information is incorrect, or you have any comments, please let me know.

Appendix A

Advanced Networking Pack for Windows XP KB817778
http://public.planetmirror.com/pub/microsoft/patches/winxp/sp1-anp/

Apache 2.0.47 for win32 Ipv6 patch
http://win6.jp/Apache2/

Nmap
http://www.insecure.org/nmap)

Netcat6
http://ftp.deepspace6.net/pub/sources/nc6/nc6-0.5.tar.gz

SOCAT
http://www.dest-unreach.org/socat/

Check out this link, pointing to Ipv6 aware software.
http://www.hs247.com/modules.php?name=Web_Links