Giving Your NET+OS Device an IPv6 Address

1 Document History

Date	Version	Change Description	
8/20/2014	V1.0	Initial Entry	
8/26/2014	V1.1	Continued entry	
8/26/2014	V1.2	Clean-up	

2 Table of Contents

3 Introduction

This paper describes a method for generating both routable and non-routable IPv6 network addresses for your NET+OS device. Though written with NET+OS in mind, the addresses created and the method discussed is applicable to any application needing IPv6 addresses.

3.1 Problem Solved

In this paper we will generate a site-local (non-routing) and a static address (routing) and an aggregatable global Unicast Address. We will base these addresses on the MAC address of the device. You will be able to use this method to give your devices IPv6 addresses.

3.2 Audience

This paper is intended for developers that have been working with IPv4 network addresses but have a need to dip a toe into the IPv6 waters.

3.3 Assumptions

This paper assumes that the addresses you are creating are for testing purposes only. Generally, in the field, your device will get its IP address using DHCP.

3.4 Scope

This paper takes the extremely narrow scope of explaining the creation of one or more IPv6 addresses for your device. This paper does not discuss any other topics including using or developing within NET+OS, computer programming, or any other topic not listed above.

4 Creating IPv6 addresses

We will start our exploration of creating IPv6 addresses by creating the EUI-64 identifier. You will see that all of the addresses we create in this paper are based on a EUI-64 identifier. We will then create addresses using this EUI-64 identifier.

4.1 Creating the EUI-64 Identifier

A EUI-64 (64-bit Extended Unique Identifier) identifier is an IEEE defined interface unique identifier. This allows a device to assign itself a unique identifier for use in an IPv6 address without using either manual or DHCP configuration methods (see http://packetlife.net/blog/2008/aug/4/eui-64-ipv6/, a blog entry by Jeremy Stretch). The process for creating a EUI-64 identifier is defined in RFC 2373 (http://www.ietf.org/rfc/rfc2373.txt).

The network piece of our addresses will be made up of the MAC address, turned into a EUI-64 identifier. So our first step is to turn our MAC address into a EUI-64 identifier. We will be working with MAC address 00:40:9D:28:08:59.

4.2 Add in 0xFF-FE

The first step we need to follow is to place 0xFF-FE between the third and fourth bytes of the MAC address. Doing this we'd end up with the following:

00:40:9D:FF:FE:28:08:59.

What is so special about 0xFF:FE? In Jeremy Stretch's blog entry mentioned above he references the IEEE's Guidelines for 64-bit Global Identifier (http://standards.ieee.org/develop/regauth/tut/eui64.pdf) "this is a reserved value which equipment manufacturers cannot include in "real" EUI-64 address assignments". Further, Jeremy states that the inclusion of the FF:FE in the EUI-64 identifies the EUI-64 as having been generated from a EUI-48 or MAC address.

4.3 Complement the universal/local bit.

The second to last bit of the most significant byte of the MAC address is considered the universal/local bit. According to an article in Wikipedia on MAC address, if this bit is 0, the address is administered universally. If the bit is a 1, the address is administered locally. We want it designated as locally administrated, so we must complement the bit. Since the most significant byte is 0x00, in binary this is 0000 0000. Complementing the second to last bit we get (in binary) 0000 0010. Thus, we have changed the EUI-64 identifier as follows: 02:40:9D:FF:FE:28:08:59.

With this done we have our EUI-64 identifier. Now we can start creating IPv6 addresses.

4.4 Link local VS. Site local addresses

To quote from the boot IPv6 Essentials, "A link-local address is for use on a single link and should never be routed. It can be used for autoconfiguration mechanisms, for neighbor discovery, and on networks with no routers, so it is useful for creating temporary networks". Further, "site-local addresses contain subnet information within the address. They can be routed within a site, but routers should not forward then outside the site. The format of these two addresses types can be used without a global prefix.

4.4.1 Link-local address

Link-local addresses start with the prefix 0xFE80. In addition, the address must end up containing 128 bits. So taking this all into account, we end up with the following link-local address:
FE80:0000:0000:0000:0240:9DFF:FE28:0859 (notice we have added 0000:0000:0000 to fill the address out to 128 bits). The IPv6 compressed form of this address might be the following:
FE80::0240:9DFF:FE28:0859.

4.4.2 Pinging the link-local address

An interesting thing happens when I try to ping the link-local address, as follows:

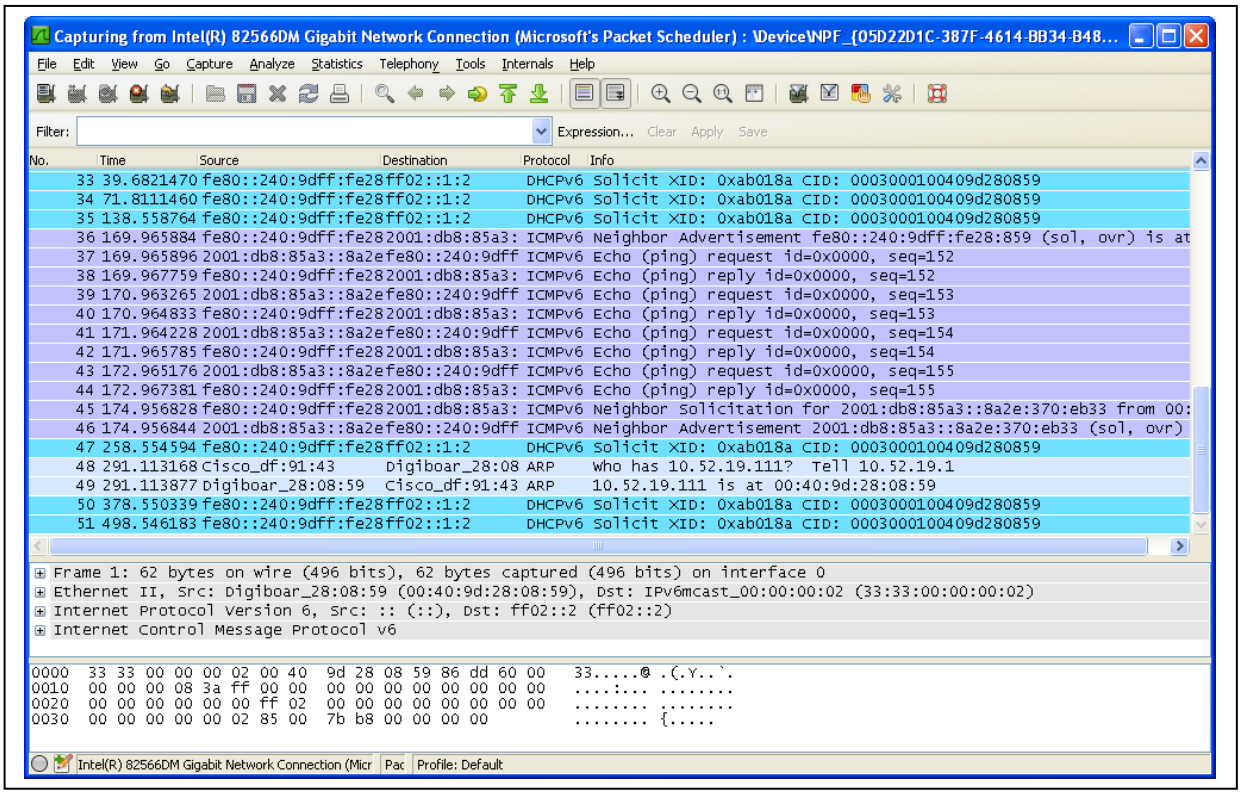
```
C:\ >ping6 FE80:0000:0000:0000:0240:9DFF:FE28:0859

Pinging fe80::240:9dff:fe28:859 with 32 bytes of data:

No route to destination.
  Specify correct scope-id or use -s to specify source address.
No route to destination.
  Specify correct scope-id or use -s to specify source address.
No route to destination.
  Specify correct scope-id or use -s to specify source address.
No route to destination.
  Specify correct scope-id or use -s to specify source address.

Ping statistics for fe80::240:9dff:fe28:859:
    Packets: Sent = 4, Received = 0, Lost = 4 (100% loss),
```

Our PC cannot find the device. This is probably because the address has no subnet information. What we have to do is include the source address of the PC in the ping6 command as follows (the IPv6 address used as the value to the attribute –s is the IPv6 address of my PC. I got it by typing in "ipconfig –all" at the command line):

```
C:\>ping6 FE80:0000:0000:0000:0240:9DFF:FE28:0859 -s 2001:db
8:85A3::8A2E:370:EB33

Pinging fe80::240:9dff:fe28:859
from 2001:db8:85a3::8a2e:370:eb33 with 32 bytes of data:

Reply from fe80::240:9dff:fe28:859%8: bytes=32 time=5ms
Reply from fe80::240:9dff:fe28:859%8: bytes=32 time=1ms
Reply from fe80::240:9dff:fe28:859%8: bytes=32 time=1ms
Reply from fe80::240:9dff:fe28:859%8: bytes=32 time=2ms

Ping statistics for fe80::240:9dff:fe28:859:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 1ms, Maximum = 5ms, Average = 2ms
```

By including the source address of the PC using the –s argument, we are able to successfully ping the device using its link-local address.

The screen shot below shows a wireshark trace of the successful ping6 session.



4.4.3 Site-local address

Site-local addresses start with the prefix 0xFECO. In addition, the address must end up containing 128 bits. So taking this into account, we end up with the following site-local address:
FECO:0000:0000:0000:0240:9DFF:FE28:0859 (notice we have added 0000:0000:0000 to fill the address out to 128 bits). The IPv6 compressed form of this address might be the following:
FECO::0240:9DFF:FE28:0859.

4.4.4 Pinging the site-local address

An interesting thing happens when I try to ping the site-local address, as follows:

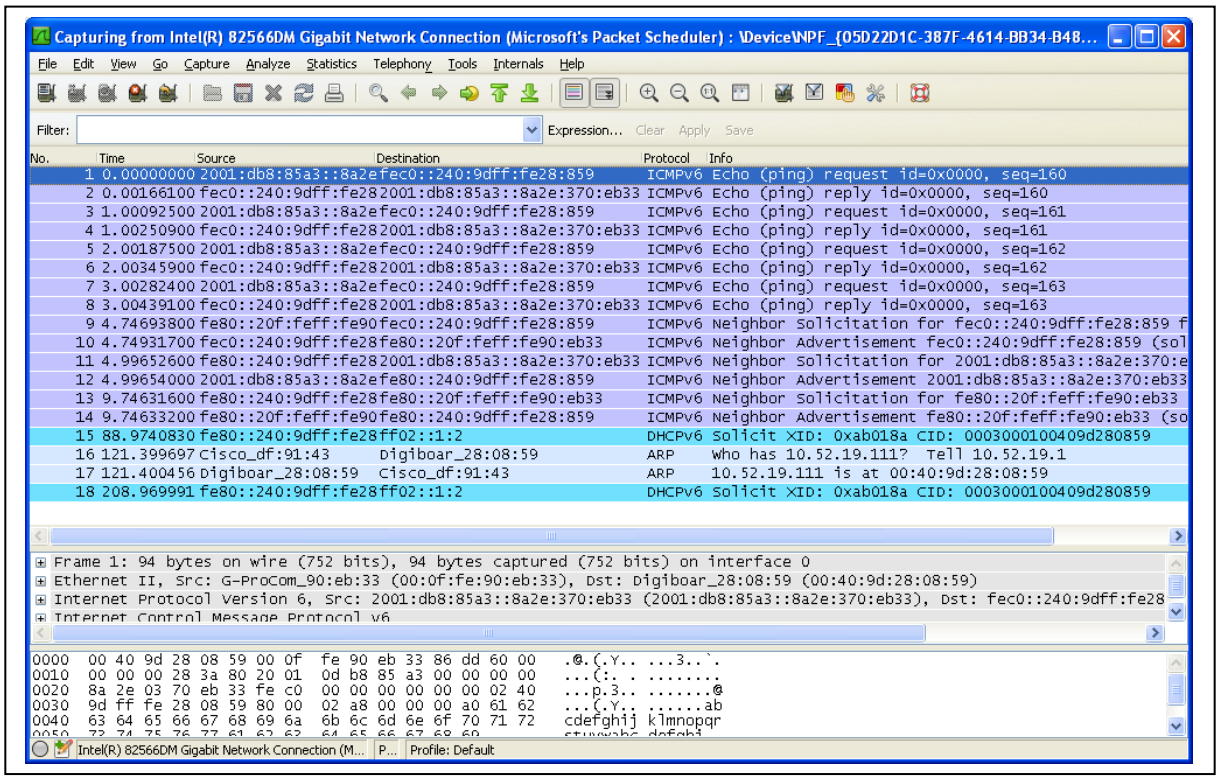
```
C:\>ping6 FEC0:0000:0000:0000:0240:9DFF:FE28:0859

Pinging fec0::240:9dff:fe28:859 with 32 bytes of data:

No route to destination.
  Specify correct scope-id or use -s to specify source address.
No route to destination.
  Specify correct scope-id or use -s to specify source address.
No route to destination.
  Specify correct scope-id or use -s to specify source address.
No route to destination.
  Specify correct scope-id or use -s to specify source address.

Ping statistics for fec0::240:9dff:fe28:859:
    Packets: Sent = 4, Received = 0, Lost = 4 (100% loss),
```

Our PC cannot find the device. This is probably because the address has no subnet information. What we have to do is include the source address of the PC in the ping6 command as follows (the IPv6 address used as the value to the attribute –s is the IPv6 address of my PC. I got it by typing in “ipconfig –all” at the command line):

```
C:\>ping6 FEC0:0000:0000:0000:0240:9DFF:FE28:0859 -s 2001:db
8:85A3::8A2E:370:EB33

Pinging fec0::240:9dff:fe28:859
from 2001:db8:85a3::8a2e:370:eb33 with 32 bytes of data:

Reply from fec0::240:9dff:fe28:859%2: bytes=32 time=5ms
Reply from fec0::240:9dff:fe28:859%2: bytes=32 time=1ms
Reply from fec0::240:9dff:fe28:859%2: bytes=32 time=1ms
Reply from fec0::240:9dff:fe28:859%2: bytes=32 time=1ms
```

Ping statistics for fec0::240:9dff:fe28:859:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 1ms, Maximum = 5ms, Average = 2ms

The screen shot below shows a wireshark trace of the successful ping6 session.



4.5 Aggregatable Global Unicast Address

At this stage I'd like to create an IPv6 address with which I do not have to use the –s option when using ping6 to ping my device. To do this I'll need some local subnet and routing information. To get that the easiest way is to look at the address assigned to my PC via DHCP6 (using ipconfig/all) as follows:

Ethernet adapter Local Area Connection:

```
Connection-specific DNS Suffix  . : digi.com
Description . . . . . . . . . . . : Intel(R) 82566DM Gigabit Network Connection
Physical Address. . . . . . . . . : 00-0F-FE-90-EB-33
Dhcp Enabled. . . . . . . . . . . : Yes
Autoconfiguration Enabled . . . . : Yes
IP Address. . . . . . . . . . . . : 10.52.19.133
Subnet Mask . . . . . . . . . . . : 255.255.255.0
IP Address. . . . . . . . . . . . : 2001:db8:85a3::8a2e:370:eb33
IP Address. . . . . . . . . . . . : fe80::20f:feff:fe90:eb33%8
Default Gateway . . . . . . . . . : 10.52.19.1
DHCP Server . . . . . . . . . . . : 10.10.8.15
DNS Servers . . . . . . . . . . . : 10.10.8.62
                                   10.10.8.64
                                   fec0:0:0:ffff::1%2
                                   fec0:0:0:ffff::2%2
                                   fec0:0:0:ffff::3%2
Primary WINS Server . . . . . . . : 10.10.8.64
Lease Obtained. . . . . . . . . . : Wednesday, August 20, 2014 9:34:43 AM
Lease Expires . . . . . . . . . . : Thursday, August 21, 2014 9:34:43 AM
```

You can see that my PC has a link local address of fe80::20f:feff:fe90:eb33. In addition it has a aggregatable global unicast address of 2001:db8:85a3::8a2e:370:eb33. What we want to do is "steal" as much of the official IPv6 addressing information as possible and graft onto that the EUI-64 identifier we used above for creating our link-local and site-local addresses. So take 2001:db8:85a3 and concatenate it to 0240:9DFF:FE28:0859. If you do this you'll notice that we are short 4 hex bytes. Thus we need a filler of 4 hex bytes of 0. What we end up with is the following:
2001:0db8:85a3:0000:0240:9DFF:FE28:0859.

4.5.1 Pinging the aggregatable global unicast address

Now when we ping our device using ping6 and pinging the aggregatable global unicast address, I see the following:

C:\Python27\Lib\idlelib>ping6 2001:0db8:85A3:0000:0240:9DFF:FE28:0859

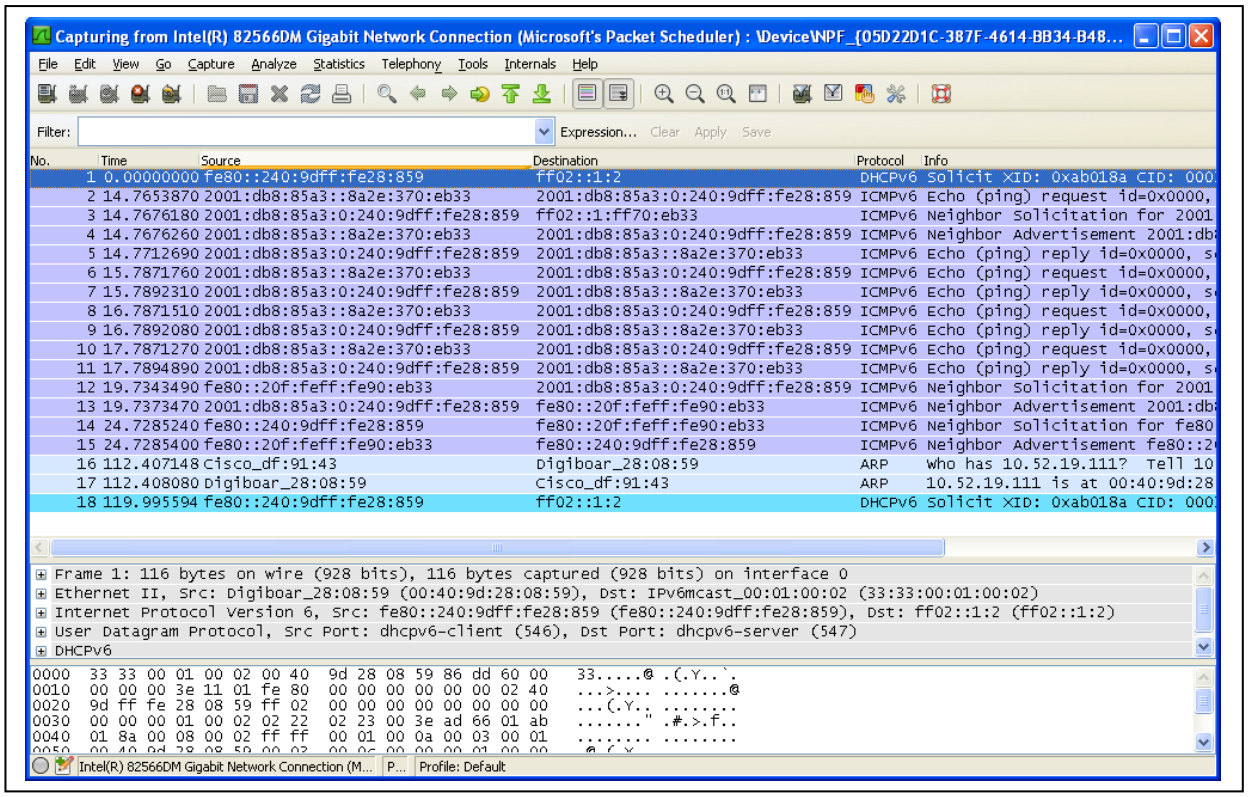
Pinging 2001:db8:85a3:0:240:9dff:fe28:859
from 2001:db8:85a3::8a2e:370:eb33 with 32 bytes of data:

Reply from 2001:db8:85a3:0:240:9dff:fe28:859: bytes=32 time=5ms
Reply from 2001:db8:85a3:0:240:9dff:fe28:859: bytes=32 time=2ms
Reply from 2001:db8:85a3:0:240:9dff:fe28:859: bytes=32 time=2ms
Reply from 2001:db8:85a3:0:240:9dff:fe28:859: bytes=32 time=2ms

Ping statistics for 2001:db8:85a3:0:240:9dff:fe28:859:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 2ms, Maximum = 5ms, Average = 2ms

Notice that we successfully pinged the device without including the source address option in the ping6 command.

The following screen shot shows a wireshark trace of the successful ping session.



5 Conclusion

IPv6 addresses are a little more intimidating than are IPv4 addresses. Additionally since you have probably been using IPv4 addresses for a long time you are comfortable creating them. Hopefully using the information included above, you'll be able to generate and use IPv6 addresses with the same ease that you have been using IPv4 addresses.

6 Appendix

6.1 Citations

MAC Address. Wikipedia. Wikimedia Foundation, Inc.
http://en.wikipedia.org/wiki/MAC_address

Hagen, Silvia. IPv6 Essentials. O'ReillyMedia, Inc. 2002

Stretch, Jeremy. Packetlife.net 2008.
http://packetlife.net/blog/2008/aug/4/eui-64-ipv6/

Guidelines for 64-bit Global Identifier. IEEE.
http://standards.ieee.org/develop/regauth/tut/eui64.pdf