Subject: IPv6 auto-configuration issue with virtual Ethernet
Posted by lars.bailey on Wed, 15 Dec 2010 05:40:54 GMT
View Forum Message <> Reply to Message

I thought I would share an issue that I ran across,in using auto-config IPv6 virtual Ethernet containers,alongside IPv4 virtual Ethernet containers.
In this scenario,the IPv4 containers will pull an auto-configured IPv6 address.(this issue will arise whether you
bridge a container or not)
The obvious part of my remarks,and the "issue" at hand,is in router advertisements.
Depending on what your end result is to be,this may not be a bad thing.(dual homed hosts)
It does not make much sense,segregating router advertisements on a per interface level.
You want to segregate the IPv4 virtual Ethernet containers,from the auto-configured IPv6 virtual Ethernet containers.
To do that,you are going to have to perform this one simple task,on any IPv4 virtual Ethernet containers,that you do not want to be dual-homed.
Edit the IPv4 container's "sysctl.conf" file,and add;

net.ipv6.conf.eth0.accept_ra = 0

Restart the container's networking,and examine the container's IPv6 routing.
The only routable IPv6 address shown,will be the container' link-local.

Subject: Re: IPv6 auto-configuration issue with virtual Ethernet
Posted by lars.bailey on Thu, 16 Dec 2010 00:34:15 GMT
View Forum Message <> Reply to Message

Since my thread seems to be generating some interest,I decided to expanded it a little.
Actually,this is probably what I should have posted from the beginning.
To start,we are currently in the process of transitioning from IPv4 to IPv6.
A testbed network was setup that consists of a IPv4/IPv6 edge router(PC),and one OVZ Node server.(dual-stack)
The edge router,is configured to drop all outbound IPv6 packets,and accept IPv6 ICMP packets from the test network.
Two private "/64" IPv6 sub-nets,was used for the testbed.
One for internal IPv6 networking,and one for auto-configured IPv6.
I configured one Ethernet bridge on the Node server,which serves as the IPv4-IPv6 gateway,and proper routing added.

VZBR0 routing

192.168.200.0/24 scope link
192.168.254.0/24 proto kernel scope link src 192.168.254.4

fd60:1014:9458:4b60::/64 metric 1 mtu 1500 advmss 1440 hoplimit 4294967295
fd98:f0bd:b577:3c8b::/64 proto kernel metric 256 mtu 1500 advmss 1440 hoplimit 4294967295
fe80::/64 proto kernel metric 256 mtu 1500 advmss 1440 hoplimit 4294967295

RADVD was configured to advertise the selected "/64",on the Ethernet bridge.

Advertised prefix fd60:1014:9458:4b60::/64

In creating and configuring a OVZ test container,we used a in-house application,that builds a VE automatically,based on
the configuration input and the OS template selected.
All OS specific network configuration files,are removed during build,then a new one is created and proper,via the script.
(i.e "/etc/network,/etc/hosts,you get the picture)
So,when a VE container is configured for IPv4 only,it is built to support IPv4 only.
For IPv6,the same principle is applied.(static,dual-homed,auto-configure)
Five IPv4 containers was created,configured,and added to the bridge,and based on Fedora 13.
Five IPv6 auto-configured containers,was also created and added to the bridge,based on fedora 13.
In checking the network setup(s) for VE 4101,you get this;

/etc/sysconfig/network-scripts/ifcfg-eth0

```
DEVICE=eth0
TYPE=Ethernet
ONBOOT=yes
IPADDR=192.168.100.101
NETMASK=255.255.255.0
GATEWAY=192.168.100.101
```

/etc/sysconfig/network

```
HOSTNAME="moe.lan04.intranet"
GATEWAYDEV=eth0
```

This is the current network ouput for 4101;

```
[root@stooge network-scripts]# vzctl enter 4101
entered into VE 4101
[root@moe /]# ifconfig eth0
eth0      Link encap:Ethernet  HWaddr xx:xx:xx:xx:xx:xx
          inet addr:192.168.100.101  Bcast:192.168.100.255  Mask:255.255.255.0
          inet6 addr: fd60:1014:9458:4b60:218:51ff:fe80:6450/64 Scope:Global
          inet6 addr: fe80::218:xxxx:fe80:6450/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:5837 errors:0 dropped:0 overruns:0 frame:0
          TX packets:11 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:560200 (547.0 KiB)  TX bytes:560 (560.0 b)
```

VE4101 IPv4 routing

[root@moe /]# ip -4 ro show dev eth0
192.168.100.0/24 proto kernel scope link src 192.168.100.101
169.254.0.0/16 scope link
default via 192.168.100.101

VE4101 IPv6 routing

[root@moe /]# ip -6 ro show dev eth0
fd60:1014:9458:4b60::/64 proto kernel metric 256 expires 2147154sec mtu 1500 advmss 1440 hoplimit 4294967295
fe80::/64 proto kernel metric 256 mtu 1500 advmss 1440 hoplimit 4294967295
default via fe80::214:xxxx:fe5e:513f proto kernel metric 1024 expires 27sec mtu 1500 advmss 1440 hoplimit 64
[root@moe /]#

Here,VE4101 shows global IPv4 connectivity.

[root@moe /]# ping -c 3 www.openvz.org
PING www.openvz.org (64.131.90.7) 56(84) bytes of data.
64 bytes from openvz.org (64.131.90.7): icmp_seq=1 ttl=47 time=789 ms
64 bytes from openvz.org (64.131.90.7): icmp_seq=2 ttl=47 time=38.4 ms
64 bytes from openvz.org (64.131.90.7): icmp_seq=3 ttl=47 time=38.7 ms

--- www.openvz.org ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2042ms
rtt min/avg/max/mdev = 38.485/289.027/789.809/354.106 ms
[root@moe /]#

As you can tell from the IPv6 routing table,VE4101 has a default link-local route to the Ethernet bridge.
Yet,VE 4101 does not have any reference to any IPv6 specific configuration.
Hmmm.
The question that comes to my mind is whether or not,this same issue will arise on a physically network segment.
I guess,this will require additional testing on my part.
In conclusion,I really do not see at this time,this is an OVZ specific issue,and I really do not see a practical use of auto-configured IPv6 for containers.
The fix,was simply segregating IPv4 from IPv6,using an additional bridged interface.

Subject: Re: IPv6 auto-configuration issue with virtual Ethernet
Posted by lars.bailey on Fri, 17 Dec 2010 04:53:18 GMT
View Forum Message <> Reply to Message

I am going to share my final conclusions,based on late night testing,and close the thread.
This is going to be a specific issue.
To refresh the test setup;

Node server - Fedora 13 Node
Test containers - Fedora 13

I downloaded two additional OS template caches from the OpenVZ wiki,and configured each for IPV4.

debian-5.0-x86
suse-11.1-x86

Both containers was bound to the Ethernet bridge,that sends RA's.
The issue of IPv4 configured containers,pulling auto-configured IPv6 addresses,does not seem to effect containers built with a Debian,or OpenSUSE template cache.(based on their specific type of networking setup)

Here is the Debian container's configuration output.

```
moe:/# ifconfig eth0
eth0      Link encap:Ethernet  HWaddr 00:18:51:03:78:64
          inet addr:192.168.100.101  Bcast:192.168.100.255  Mask:255.255.255.0
          inet6 addr: fe80::218:XXXX:fe03:7864/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:456 errors:0 dropped:0 overruns:0 frame:0
          TX packets:55 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:45322 (44.2 KiB)  TX bytes:3704 (3.6 KiB)

moe:/# ip -4 ro show dev eth0
192.168.100.0/24  proto kernel  scope link  src 192.168.100.101
default via 192.168.100.101  scope link
moe:/# ip -6 ro show dev eth0
fe80::/64  proto kernel  metric 256  mtu 1500 advmss 1440 hoplimit 4294967295
moe:/#
```

Here is the OpenSUSE container's configuration output.

```
shemp:/ # ifconfig eth0
eth0      Link encap:Ethernet  HWaddr 00:18:51:55:4B:36
          inet addr:192.168.100.103  Bcast:192.168.100.255  Mask:255.255.255.0
          inet6 addr: fe80::218:51ff:fe55:4b36/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:401 errors:0 dropped:0 overruns:0 frame:0
```

```
        TX packets:27 errors:0 dropped:0 overruns:0 carrier:0
        collisions:0 txqueuelen:0
        RX bytes:38955 (38.0 Kb)  TX bytes:1405 (1.3 Kb)

shemp:/ # ip -4 ro show dev eth0
192.168.100.0/24  proto kernel  scope link  src 192.168.100.103
169.254.0.0/16  scope link
default via 192.168.100.103
shemp:/ # ip -6 ro show dev eth0
fe80::/64  proto kernel  metric 256  mtu 1500 advmss 1440 hoplimit 4294967295
shemp:/ #
```

And of course,the Fedora 13 container.

```
[root@curly /]# ifconfig eth0
eth0      Link encap:Ethernet  HWaddr 00:18:51:78:64:52
        inet addr:192.168.100.102  Bcast:192.168.100.255  Mask:255.255.255.0
        inet6 addr: fd60:1014:9458:4b60:218:51ff:fe78:6452/64 Scope:Global
        inet6 addr: fe80::218:51ff:fe78:6452/64 Scope:Link
        UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
        RX packets:473 errors:0 dropped:0 overruns:0 frame:0
        TX packets:45 errors:0 dropped:0 overruns:0 carrier:0
        collisions:0 txqueuelen:0
        RX bytes:46552 (45.4 KiB)  TX bytes:2920 (2.8 KiB)

[root@curly /]# ip -4 ro show dev eth0
192.168.100.0/24  proto kernel  scope link  src 192.168.100.102
169.254.0.0/16  scope link
default via 192.168.100.102
[root@curly /]# ip -6 ro show dev eth0
fd60:1014:9458:4b60::/64  proto kernel  metric 256  expires 2147151sec mtu 1500 advmss 1440
hoplimit 4294967295
fe80::/64  proto kernel  metric 256  mtu 1500 advmss 1440 hoplimit 4294967295
default via fe80::214:bfff:fe5e:513f  proto kernel  metric 1024  expires 24sec mtu 1500 advmss
1440 hoplimit 64
[root@curly /]#
```

Now the question is,"If you use other types operating systems for a Node server,will this same issue arise in using containers,based on the Node operating system type?
My opinion,is inclined to say "yes".
On a personal note,we have no intentions of using auto-configuration on any Node server,and I have no intentions
of performing further auto-configuration testing.
I just wanted to experiment with auto-config IPv6,as we do have a few trivial clients,lying around on the network.
As far as the issue effecting our networking world,it won't.
But I thought I would share the info,as I have not found any docs on the Net/Wiki,pertaining to the issue at hand.

Subject: Re: IPv6 auto-configuration issue with virtual Ethernet
Posted by lars.bailey on Mon, 27 Dec 2010 03:16:50 GMT
View Forum Message <> Reply to Message

After performing other IPv6 configurations on a test OVZ IPv6 network,my final solution for IPv4 RedHat containers pulling auto-config IPv6 addresses,did not sit well with me,as I knew this issue could be fixed without much effort.
Then I realized,the solution lied on the Node server.
All Node servers,use this configuration now,in "/etc/sysconfig/network" file.

NETWORKING=yes
HOSTNAME=stooge.lan.intranet
NETWORKING_IPV6=yes
IPV6FORWARDING=yes
IPV6_AUTOCONF=no
IPV6_AUTOTUNNEL=no

By adding this to an IPv4 RedHat container;

NETWORKING_IPV6=yes
IPV6_AUTOCONF=no

it no longer pulls auto-configured IPv6 addresses.
This of course,if you bridge both IPv4 and IPv6 containers,and you are advertising a "/64" on the bridge interface.
Why it effects RedHat containers only,without proper IPv6 configurations,I can only conclude it has to do with using
bridged Ethernet.
Since,the IPv4 VETH interface is bound to the bridge,and a container has a link-local address,this seems to be the only
correct answer.
If anybody has any other ideas,post it,as I'm still a little
curious as to why.