Subject: Re: [patch 1/1][NETNS][IPV6] protect addrconf from loopback registration
Posted by ebiederm on Mon, 12 Nov 2007 19:50:53 GMT
View Forum Message <> Reply to Message

Daniel Lezcano <dlezcano@fr.ibm.com> writes:

> Eric W. Biederman wrote:
>> "Denis V. Lunev" <den@sw.ru> writes:
>>
>>>> Index: linux-2.6-netns/net/ipv6/addrconf.c
>>>> ===
>>>> --- linux-2.6-netns.orig/net/ipv6/addrconf.c
>>>> +++ linux-2.6-netns/net/ipv6/addrconf.c
>>>> @@ -2272,7 +2272,8 @@ static int addrconf_notify(struct notifi
>>>> switch(event) {
>>>> case NETDEV_REGISTER:
>>>> - if (!idev && dev->mtu >= IPV6_MIN_MTU) {
>>>> + if (!(dev->flags & IFF_LOOPBACK) &&
>>>> + !idev && dev->mtu >= IPV6_MIN_MTU) {
>>
>> It is idev being true here for the loopback device that would
>> prevent things not missing the REGISTER event.
>>
>> Hmm. But we do call ipv6_add_dev on loopback and now the loopback
>> device is practically guaranteed to be the first device so we can
>> probably just remove the special case in addrconf_init.
>>
>> Anyway Daniels patch makes increasingly less sense the more I look
>> at it.
>
> Let me try to clarify:
>
> * when the init network namespace is created, the loopback is created first,
> before ipv6, and the notifier call chain for ipv6 is not setup, so the protocol
> does not receive the REGISTER event
>
> * when the init network namespace is destroyed during shutdown, the loopback is
> not unregistered, so there is no UNREGISTER event

* When addrconf_init calls register_netdevice_notifier we receive
  NETDEV_REGISTER and NETDEV_UP for all network devices that are in
  the system including the loopback device.

> * when we create a new network namespace, a new instance of the loopback is
> created and a NETDEV_REGISTER is sent to ipv6 because the notifier call chain
> has been setup by the init netns (while ipv6 protocol is not yet configured for
> the namespace which is being created)

Possibly there may be some ordering issues here.

> * when the network namespace exits, the loopback is unregistered after the ipv6
> protocol but the NETDEV_UNREGISTER is sent to addrconf_notify while the ipv6
> protocol has been destroyed.
>
>
> The objective of the patch is to discard these events because they were never
> taken into account and they are not expected to be receive by ipv6 protocol.

My opinion is that both your analysis is slightly off (as to the cause
of your problems) and that your approach to fix your problem is wrong
because you don't untangle the knot you keep it.

...
I have register_pernet_subsys and register_per_net_device to ensure
that when we create a new network namespace all of the subsystems are
initialized before the network devices are initialize. So ipv6 should
be ready before we initialize the new loopback device comes into
existence.

The preservation of the order of the network namespace callbacks
ensures that the loopback device will be the first network device
registered, and if it helps we can take advantage of that in reference
to the weirdness from the comment below.

```
 /* The addrconf netdev notifier requires that loopback_dev
  * has it's ipv6 private information allocated and setup
  * before it can bring up and give link-local addresses
  * to other devices which are up.
  *
  * Unfortunately, loopback_dev is not necessarily the first
  * entry in the global dev_base list of net devices.  In fact,
  * it is likely to be the very last entry on that list.
  * So this causes the notifier registry below to try and
  * give link-local addresses to all devices besides loopback_dev
  * first, then loopback_dev, which cases all the non-loopback_dev
  * devices to fail to get a link-local address.
  *
  * So, as a temporary fix, allocate the ipv6 structure for
  * loopback_dev first by hand.
  * Longer term, all of the dependencies ipv6 has upon the loopback
  * device and it being up should be removed.
  */
```

We can just special case registration of the loopback device to

do:
 ip6_null_entry.u.dst.dev = init_net.loopback_dev;
 ip6_null_entry.rt6i_idev = in6_dev_get(init_net.loopback_dev);
#ifdef CONFIG_IPV6_MULTIPLE_TABLES
 ip6_prohibit_entry.u.dst.dev = init_net.loopback_dev;
 ip6_prohibit_entry.rt6i_idev = in6_dev_get(init_net.loopback_dev);
 ip6_blk_hole_entry.u.dst.dev = init_net.loopback_dev;
 ip6_blk_hole_entry.rt6i_idev = in6_dev_get(init_net.loopback_dev);
#endif

Which would remove the special case from addrconf_init.

Eric

_____
Containers mailing list
Containers@lists.linux-foundation.org
https://lists.linux-foundation.org/mailman/listinfo/containers