Subject: Re: [patch 1/1][NETNS][IPV6] protect addrconf from loopback registration
Posted by Daniel Lezcano on Mon, 12 Nov 2007 16:11:29 GMT
View Forum Message <> Reply to Message

Denis V. Lunev wrote:
> Daniel Lezcano wrote:
>> The loopback is now dynamically allocated. The ipv6 code was written
>> considering the loopback is allocated before the ipv6 protocol
>> initialization. This is still the case when we don't use multiple
>> network namespaces.
>>
>> In the case of the network namespaces, ipv6 notification handler is
>> already setup and active (done by the initial network namespace),
>> so when a network namespace is created, a new instance of the
>> loopback device, via dynamic allocation, will trigger a REGISTER event
>> to addrconf_notify and this one will try to setup the network device
>> while the ipv6 protocol is not yet initialized for the network namespace.
>>
>> Because the ipv6 is relying on the fact that the loopback device will
>> not trigger REGISTER/UNREGISTER events, I just protect the addrconf_notify
>> function when the loopback register event is triggered.
>>
>> In the case of multiple network namespaces, the usual ipv6 protocol
>> initialization will be done after the loopback initialization with
>> the subsystem registration mechanism.
>>
>> Signed-off-by: Daniel Lezcano <dlezcano@fr.ibm.com>
>> Signed-off-by: Benjamin Thery <benjamin.thery@bull.net>
>> ---
>> net/ipv6/addrconf.c | 9 +++++++--
>> 1 file changed, 7 insertions(+), 2 deletions(-)
>>
>> 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) {
>>   idev = ipv6_add_dev(dev);
>>   if (!idev)
>>   return notifier_from_errno(-ENOMEM);
>> @@ -2366,11 +2367,15 @@ static int addrconf_notify(struct notifi
>>   /* MTU falled under IPV6_MIN_MTU. Stop IPv6 on this interface. */

>>
>>   case NETDEV_DOWN:
>> + addrconf_ifdown(dev, 0);
>> + break;
>> +
>>   case NETDEV_UNREGISTER:
>>   /*
>>   * Remove all addresses from this interface.
>>   */
>> - addrconf_ifdown(dev, event != NETDEV_DOWN);
>> + if (!(dev->flags & IFF_LOOPBACK))
>> +   addrconf_ifdown(dev, 1);
>>   break;
>>
>>   case NETDEV_CHANGENAME:
>>
>
> why should we care on down? we are destroying the device. It should
> gone. All references to it should also gone. So, we should perform the
> cleaning and remove all IPv6 addresses, so notifier should also work.

We need to take care of netdev down, someone can put the loopback down
if he wants.

> The code relies on the "persistent" loopback and this is a _bad_ thing.
> This is longstanding bug in the code, that the dst_entry should have a
> valid reference to a device. This is the only purpose for a loopback
> persistence. Though, at the namespace death no such entries must be and
> this will be checked during unregister process. This patch definitely
> breaks this assumption :(
>
> Namespaces are good to catch leakage using standard codepaths, so they
> should be preserved as much as possible. So, _all_ normal down code
> should be called for a loopback device in other than init_net context.

I agree with you, this is a bug in ipv6 and the loopback; when playing
with ipv6 we found that the loopback is still referenced 9 times when
the system is shutdown.

The purpose of this patch is to protect the __actual__ code from the new
loopback behavior. We are looking at a more generic approach with the
namespace for ipv6, as you mentioned, namespaces are good for network
leakage detection as we create several instances of the network stack.