Subject: [patch 0/3][netns] several fixes/enhancements
Posted by Daniel Lezcano on Mon, 24 Sep 2007 12:21:12 GMT
View Forum Message <> Reply to Message

The two first patches enable the arp and rtnetlink events.
They were working in previous version, so I don't know if
they were disabled for a particular reason but I re-enable
them again just in case ...

The third patch allows to consolidate the netlink attributes
when we try to show a network device linked with a network device
assigned to another network namespace

--

Subject: [patch 1/3][netns] net: Activate inetdev event for IPV4
Posted by Daniel Lezcano on Mon, 24 Sep 2007 12:21:13 GMT
View Forum Message <> Reply to Message

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

The event notification is disabled when the current
network namespace is not the init one. The route are
not created when setting up a new IP address.

I activate the notification to fix these problems.

Signed-off-by: Daniel Lezcano <dlezcano@fr.ibm.com>
---
 net/core/fib_rules.c | 3 ---
 net/core/rtnetlink.c | 3 ---
 net/ipv4/fib_frontend.c | 3 ---
 3 files changed, 9 deletions(-)

Index: linux-2.6-netns/net/core/fib_rules.c
===
--- linux-2.6-netns.orig/net/core/fib_rules.c
+++ linux-2.6-netns/net/core/fib_rules.c
@@ -605,9 +605,6 @@ static int fib_rules_event(struct notifi
  struct net *net = dev->nd_net;
  struct fib_rules_ops *ops;

- if (dev->nd_net != &init_net)

```
-  return NOTIFY_DONE;
-
  ASSERT_RTNL();
  rcu_read_lock();

Index: linux-2.6-netns/net/core/rtnetlink.c
===================================================================
--- linux-2.6-netns.orig/net/core/rtnetlink.c
+++ linux-2.6-netns/net/core/rtnetlink.c
@@ -1348,9 +1348,6 @@ static int rtnetlink_event(struct notifi
 {
  struct net_device *dev = ptr;

- if (dev->nd_net != &init_net)
-  return NOTIFY_DONE;
-
  switch (event) {
  case NETDEV_UNREGISTER:
   rtmsg_ifinfo(RTM_DELLINK, dev, ~0U);
Index: linux-2.6-netns/net/ipv4/fib_frontend.c
===================================================================
--- linux-2.6-netns.orig/net/ipv4/fib_frontend.c
+++ linux-2.6-netns/net/ipv4/fib_frontend.c
@@ -902,9 +902,6 @@ static int fib_netdev_event(struct notif
  struct net_device *dev = ptr;
  struct in_device *in_dev = __in_dev_get_rtnl(dev);

- if (dev->nd_net != &init_net)
-  return NOTIFY_DONE;
-
  if (event == NETDEV_UNREGISTER) {
   fib_disable_ip(dev, 2);
   return NOTIFY_DONE;

--

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

Subject: [patch 2/3][netns] net: Activate arp for non init netns
Posted by Daniel Lezcano on Mon, 24 Sep 2007 12:21:14 GMT
View Forum Message <> Reply to Message

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

Arp is disabled for network namespace different from the init_net.

This patch enables it again.

Signed-off-by: Daniel Lezcano <dlezcano@fr.ibm.com>
---
 net/ipv4/arp.c | 6 ------
 1 file changed, 6 deletions(-)

Index: linux-2.6-netns/net/ipv4/arp.c
===
--- linux-2.6-netns.orig/net/ipv4/arp.c
+++ linux-2.6-netns/net/ipv4/arp.c
@@ -939,9 +939,6 @@ static int arp_rcv(struct sk_buff *skb,
 {
  struct arphdr *arp;

- if (dev->nd_net != &init_net)
- goto freeskb;
-
  /* ARP header, plus 2 device addresses, plus 2 IP addresses. */
  if (!pskb_may_pull(skb, (sizeof(struct arphdr) +
      (2 * dev->addr_len) +
@@ -1216,9 +1213,6 @@ static int arp_netdev_event(struct notif
 {
  struct net_device *dev = ptr;

- if (dev->nd_net != &init_net)
- return NOTIFY_DONE;
-
  switch (event) {
  case NETDEV_CHANGEADDR:
   neigh_changeaddr(&arp_tbl, dev);

--

_____

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

---

Subject: [patch 3/3][netns] net: hide master/linked interface from netlink
Posted by Daniel Lezcano on Mon, 24 Sep 2007 12:21:15 GMT
View Forum Message <> Reply to Message

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

Actually when a network device is linked to another, the name appears
to be @<link>. For example, if a macvlan0 is created on top of eth0,
the ip link show is:

```
6: macvlan0@eth0: <BROADCAST,MULTICAST> mtu 1500 qdisc noop
    link/ether 6a:d4:10:0d:a8:55 brd ff:ff:ff:ff:ff:ff
```

But if we move macvlan0 to a network namespace, eth0 does no longer exist inside it and the result will be:

```
6: macvlan0@if2: <BROADCAST,MULTICAST> mtu 1500 qdisc noop
    link/ether 6a:d4:10:0d:a8:55 brd ff:ff:ff:ff:ff:ff
```

if2 is, I guess, some random value. That can do invalid memory access or inconsistent data showing.

The patchset will avoid such case, it checks if the linked device exist into the current network namespace and if it doesn't the result will be:

```
6: macvlan0@NONE: <BROADCAST,MULTICAST> mtu 1500 qdisc noop
    link/ether 6a:d4:10:0d:a8:55 brd ff:ff:ff:ff:ff:ff
```

Signed-off-by: Daniel Lezcano <dlezcano@fr.ibm.com>

```
---
 net/core/rtnetlink.c |   14 ++++++++++---
 1 file changed, 11 insertions(+), 3 deletions(-)

Index: linux-2.6-netns/net/core/rtnetlink.c
===================================================================
--- linux-2.6-netns.orig/net/core/rtnetlink.c
+++ linux-2.6-netns/net/core/rtnetlink.c
@@ -636,6 +636,8 @@ static int rtnl_fill_ifinfo(struct sk_bu
 {
  struct ifinfomsg *ifm;
  struct nlmsghdr *nlh;
+ int ifindex = 0;
+ struct net_device *d;

  nlh = nlmsg_put(skb, pid, seq, type, sizeof(*ifm), flags);
  if (nlh == NULL)
@@ -656,11 +658,17 @@ static int rtnl_fill_ifinfo(struct sk_bu
  NLA_PUT_U8(skb, IFLA_LINKMODE, dev->link_mode);
  NLA_PUT_U32(skb, IFLA_MTU, dev->mtu);

- if (dev->ifindex != dev->iflink)
-  NLA_PUT_U32(skb, IFLA_LINK, dev->iflink);
+ if (dev->ifindex != dev->iflink) {
+  d = dev_get_by_index(dev->nd_net, dev->iflink);
+  ifindex = d?dev->iflink:0;
+  NLA_PUT_U32(skb, IFLA_LINK, ifindex);
```

```
+ }

- if (dev->master)
+ if (dev->master) {
+  d = dev->master;
+  ifindex = dev->nd_net == d->nd_net?dev->master->ifindex:0;
   NLA_PUT_U32(skb, IFLA_MASTER, dev->master->ifindex);
+ }

  if (dev->qdisc_sleeping)
   NLA_PUT_STRING(skb, IFLA_QDISC, dev->qdisc_sleeping->ops->id);

--
```

_____

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

---

Subject: Re: [patch 0/3][netns] several fixes/enhancements
Posted by ebiederm on Wed, 26 Sep 2007 19:14:47 GMT
View Forum Message <> Reply to Message

dlezcano@fr.ibm.com writes:

> The two first patches enable the arp and rtnetlink events.
> They were working in previous version, so I don't know if
> they were disabled for a particular reason but I re-enable
> them again just in case ...
>
> The third patch allows to consolidate the netlink attributes
> when we try to show a network device linked with a network device
> assigned to another network namespace

At a first pass these patches look sane.

Eric

_____

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

---

Subject: Re: [patch 3/3][netns] net: hide master/linked interface from netlink
Posted by ebiederm on Thu, 27 Sep 2007 08:53:45 GMT
View Forum Message <> Reply to Message

dlezcano@fr.ibm.com writes:

> From: Daniel Lezcano <dlezcano@fr.ibm.com>
>
> Actually when a network device is linked to another, the name appears
> to be @<link>. For example, if a macvlan0 is created on top of eth0,
> the ip link show is:
>
> 6: macvlan0@eth0: <BROADCAST,MULTICAST> mtu 1500 qdisc noop
>   link/ether 6a:d4:10:0d:a8:55 brd ff:ff:ff:ff:ff:ff
>
> But if we move macvlan0 to a network namespace, eth0 does no longer
> exist inside it and the result will be:
>
> 6: macvlan0@if2: <BROADCAST,MULTICAST> mtu 1500 qdisc noop
>   link/ether 6a:d4:10:0d:a8:55 brd ff:ff:ff:ff:ff:ff
>
> if2 is, I guess, some random value. That can do invalid memory
> access or inconsistent data showing.
>
> The patchset will avoid such case, it checks if the linked device exist
> into the current network namespace and if it doesn't the result will
> be:
>
> 6: macvlan0@NONE: <BROADCAST,MULTICAST> mtu 1500 qdisc noop
>   link/ether 6a:d4:10:0d:a8:55 brd ff:ff:ff:ff:ff:ff

Hmm. Currently the ifindex space is global. Something that ultimately
needs to be fixed to handle migration so we can preserve the ifindex
on a device when we migrate it.

So the @if2 piece is harmless, as it is just reporting the ifindex
number because it can't figure out the name that corresponds to
that network device.

So the worst we get is hints that some extra data exists somewhere.
I suspect the correct fix is to not even write the additional attribute
in this case instead of setting the value to zero.

I am not yet convinced we need to handle this case yet, at most this
is a cosmetic issue.

Eric

Subject: Re: [patch 3/3][netns] net: hide master/linked interface from netlink
Posted by Daniel Lezcano on Thu, 27 Sep 2007 09:15:45 GMT
View Forum Message <> Reply to Message

Eric W. Biederman wrote:
> dlezcano@fr.ibm.com writes:
>
>> From: Daniel Lezcano <dlezcano@fr.ibm.com>
>>
>> Actually when a network device is linked to another, the name appears
>> to be @<link>. For example, if a macvlan0 is created on top of eth0,
>> the ip link show is:
>>
>> 6: macvlan0@eth0: <BROADCAST,MULTICAST> mtu 1500 qdisc noop
>>   link/ether 6a:d4:10:0d:a8:55 brd ff:ff:ff:ff:ff:ff
>>
>> But if we move macvlan0 to a network namespace, eth0 does no longer
>> exist inside it and the result will be:
>>
>> 6: macvlan0@if2: <BROADCAST,MULTICAST> mtu 1500 qdisc noop
>>   link/ether 6a:d4:10:0d:a8:55 brd ff:ff:ff:ff:ff:ff
>>
>> if2 is, I guess, some random value. That can do invalid memory
>> access or inconsistent data showing.
>>
>> The patchset will avoid such case, it checks if the linked device exist
>> into the current network namespace and if it doesn't the result will
>> be:
>>
>> 6: macvlan0@NONE: <BROADCAST,MULTICAST> mtu 1500 qdisc noop
>>   link/ether 6a:d4:10:0d:a8:55 brd ff:ff:ff:ff:ff:ff
>
> Hmm. Currently the ifindex space is global. Something that ultimately
> needs to be fixed to handle migration so we can preserve the ifindex
> on a device when we migrate it.

One interesting thing with the global ifindex, is we can keep track to the network device. For example, from outside the namespace we create macvlan0, push it to the netns, inside the netns, we rename it eth0 (eg. to fit os template), the netns finishes and the netdev is pop out to the initial network namespace. The names will conflict and the eth0 (former macvlan0) is renamed to dev0. If we want to automate the destruction of such interface, we should be able to identify them when the namespace exits. The global ifindex allow that.

If the ifindex is changed to be local to the namespace, the ifindex will be changed each time we change namespaces. So in this case, it will be hard to track these interfaces.

This point is out of migration consideration and having a ifindex per namespace makes sense.

Should we consider an identifier for the network devices ? so we can setup this identifier each time we create/migrate an interface and find them by this identifier.

> So the @if2 piece is harmless, as it is just reporting the ifindex
> number because it can't figure out the name that corresponds to
> that network device.
>
> So the worst we get is hints that some extra data exists somewhere.
> I suspect the correct fix is to not even write the additional attribute
> in this case instead of setting the value to zero.
>
> I am not yet convinced we need to handle this case yet, at most this
> is a cosmetic issue.

_____