Subject: [PATCH 3/5] net: Make the netlink methods in rtnetlink handle multiple network namespaces
Posted by ebiederm on Sat, 29 Sep 2007 01:04:35 GMT
View Forum Message <> Reply to Message

After the previous prep work this just consists of removing checks
limiting the code to work in the initial network namespace, and
updating rtmsg_ifinfo so we can generate events for devices in
something other then the initial network namespace.

Referring to network other network devices like the IFLA_LINK
and IFLA_MASTER attributes do, gets interesting if those network
devices happen to be in other network namespaces. Currently
ifindex numbers are allocated globally so I have taken the path
of least resistance and not still report the information even
though the devices they are talking about are invisible.

If applications start getting confused or when ifindex
numbers become local to the network namespace we may need
to do something different in the future.

Signed-off-by: Eric W. Biederman <ebiederm@xmission.com>
---
 net/core/rtnetlink.c | 27 +++------------------------
 1 files changed, 3 insertions(+), 24 deletions(-)

diff --git a/net/core/rtnetlink.c b/net/core/rtnetlink.c
index fc49104..809a9fb 100644
--- a/net/core/rtnetlink.c
+++ b/net/core/rtnetlink.c
@@ -741,9 +741,6 @@ static int rtnl_dump_ifinfo(struct sk_buff *skb, struct netlink_callback *cb)
  int s_idx = cb->args[0];
  struct net_device *dev;

- if (net != &init_net)
- return 0;
-
  idx = 0;
  for_each_netdev(net, dev) {
   if (idx < s_idx)
@@ -946,9 +943,6 @@ static int rtnl_setlink(struct sk_buff *skb, struct nlmsghdr *nlh, void *arg)
  struct nlattr *tb[IFLA_MAX+1];
  char ifname[IFNAMSIZ];

- if (net != &init_net)
- return -EINVAL;
-
  err = nlmsg_parse(nlh, sizeof(*ifm), tb, IFLA_MAX, ifla_policy);

```
  if (err < 0)
   goto errout;
@@ -997,9 +991,6 @@ static int rtnl_dellink(struct sk_buff *skb, struct nlmsghdr *nlh, void *arg)
  struct nlattr *tb[IFLA_MAX+1];
  int err;

- if (net != &init_net)
-  return -EINVAL;
-
  err = nlmsg_parse(nlh, sizeof(*ifm), tb, IFLA_MAX, ifla_policy);
  if (err < 0)
   return err;
@@ -1081,9 +1072,6 @@ static int rtnl_newlink(struct sk_buff *skb, struct nlmsghdr *nlh, void
*arg)
  struct nlattr *linkinfo[IFLA_INFO_MAX+1];
  int err;

- if (net != &init_net)
-  return -EINVAL;
-
 #ifdef CONFIG_KMOD
 replay:
 #endif
@@ -1210,9 +1198,6 @@ static int rtnl_getlink(struct sk_buff *skb, struct nlmsghdr* nlh, void
*arg)
  struct sk_buff *nskb;
  int err;

- if (net != &init_net)
-  return -EINVAL;
-
  err = nlmsg_parse(nlh, sizeof(*ifm), tb, IFLA_MAX, ifla_policy);
  if (err < 0)
   return err;
@@ -1248,13 +1233,9 @@ errout:

 static int rtnl_dump_all(struct sk_buff *skb, struct netlink_callback *cb)
 {
- struct net *net = skb->sk->sk_net;
  int idx;
  int s_idx = cb->family;

- if (net != &init_net)
-  return 0;
-
  if (s_idx == 0)
   s_idx = 1;
  for (idx=1; idx<NPROTO; idx++) {
```

```
@@ -1276,6 +1257,7 @@ static int rtnl_dump_all(struct sk_buff *skb, struct netlink_callback *cb)

 void rtmsg_ifinfo(int type, struct net_device *dev, unsigned change)
 {
+ struct net *net = dev->nd_net;
  struct sk_buff *skb;
  int err = -ENOBUFS;

@@ -1290,10 +1272,10 @@ void rtmsg_ifinfo(int type, struct net_device *dev, unsigned change)
   kfree_skb(skb);
   goto errout;
  }
- err = rtnl_notify(skb, &init_net, 0, RTNLGRP_LINK, NULL, GFP_KERNEL);
+ err = rtnl_notify(skb, net, 0, RTNLGRP_LINK, NULL, GFP_KERNEL);
 errout:
  if (err < 0)
-  rtnl_set_sk_err(&init_net, RTNLGRP_LINK, err);
+  rtnl_set_sk_err(net, RTNLGRP_LINK, err);
 }

 /* Protected by RTNL sempahore.  */
@@ -1392,9 +1374,6 @@ static int rtnetlink_event(struct notifier_block *this, unsigned long
event, voi
 {
  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);
--
1.5.3.rc6.17.g1911
```