Subject: [PATCH RFC 24/31] net: Make rtnetlink network namespace aware
Posted by ebiederm on Thu, 25 Jan 2007 19:00:26 GMT
View Forum Message <> Reply to Message

From: Eric W. Biederman <ebiederm@xmission.com> - unquoted

After this patch none of the netlink callback support anything
except the initial network namespace but the rtnetlink infrastructure
now handles multiple network namespaces.

Signed-off-by: Eric W. Biederman <ebiederm@xmission.com>
---
 include/linux/rtnetlink.h | 8 ++--
 net/bridge/br_netlink.c | 4 +-
 net/core/fib_rules.c | 4 +-
 net/core/neighbour.c | 4 +-
 net/core/rtnetlink.c | 74 ++++++++++++++++++++++++++++++++++++---------
 net/core/wireless.c | 5 ++-
 net/decnet/dn_dev.c | 4 +-
 net/decnet/dn_route.c | 2 +-
 net/decnet/dn_table.c | 4 +-
 net/ipv4/devinet.c | 4 +-
 net/ipv4/fib_semantics.c | 4 +-
 net/ipv4/ipmr.c | 4 +-
 net/ipv4/route.c | 2 +-
 net/ipv6/addrconf.c | 14 ++++++----
 net/ipv6/route.c | 6 ++--
 net/sched/cls_api.c | 2 +-
 net/sched/sch_api.c | 4 +-
 17 files changed, 98 insertions(+), 51 deletions(-)

diff --git a/include/linux/rtnetlink.h b/include/linux/rtnetlink.h
index 4a629ea..6c8281d 100644
--- a/include/linux/rtnetlink.h
+++ b/include/linux/rtnetlink.h
@@ -581,11 +581,11 @@ struct rtnetlink_link
 };

 extern struct rtnetlink_link * rtnetlink_links[NPROTO];
-extern int rtnetlink_send(struct sk_buff *skb, u32 pid, u32 group, int echo);
-extern int rtnl_unicast(struct sk_buff *skb, u32 pid);
-extern int rtnl_notify(struct sk_buff *skb, u32 pid, u32 group,
+extern int rtnetlink_send(struct sk_buff *skb, net_t net, u32 pid, u32 group, int echo);
+extern int rtnl_unicast(struct sk_buff *skb, net_t net, u32 pid);
+extern int rtnl_notify(struct sk_buff *skb, net_t net, u32 pid, u32 group,
         struct nlmsghdr *nlh, gfp_t flags);
-extern void rtnl_set_sk_err(u32 group, int error);
+extern void rtnl_set_sk_err(net_t net, u32 group, int error);

```
 extern int rtnetlink_put_metrics(struct sk_buff *skb, u32 *metrics);
 extern int rtnl_put_cacheinfo(struct sk_buff *skb, struct dst_entry *dst,
         u32 id, u32 ts, u32 tsage, long expires,
diff --git a/net/bridge/br_netlink.c b/net/bridge/br_netlink.c
index 85165a1..372fb18 100644
--- a/net/bridge/br_netlink.c
+++ b/net/bridge/br_netlink.c
@@ -94,10 +94,10 @@ void br_ifinfo_notify(int event, struct net_bridge_port *port)
  /* failure implies BUG in br_nlmsg_size() */
  BUG_ON(err < 0);

- err = rtnl_notify(skb, 0, RTNLGRP_LINK, NULL, GFP_ATOMIC);
+ err = rtnl_notify(skb, init_net(), 0, RTNLGRP_LINK, NULL, GFP_ATOMIC);
 errout:
  if (err < 0)
-  rtnl_set_sk_err(RTNLGRP_LINK, err);
+  rtnl_set_sk_err(init_net(), RTNLGRP_LINK, err);
 }

 /*
diff --git a/net/core/fib_rules.c b/net/core/fib_rules.c
index 00b4148..5f65973 100644
--- a/net/core/fib_rules.c
+++ b/net/core/fib_rules.c
@@ -418,10 +418,10 @@ static void notify_rule_change(int event, struct fib_rule *rule,
  /* failure implies BUG in fib_rule_nlmsg_size() */
  BUG_ON(err < 0);

- err = rtnl_notify(skb, pid, ops->nlgroup, nlh, GFP_KERNEL);
+ err = rtnl_notify(skb, init_net(), pid, ops->nlgroup, nlh, GFP_KERNEL);
 errout:
  if (err < 0)
-  rtnl_set_sk_err(ops->nlgroup, err);
+  rtnl_set_sk_err(init_net(), ops->nlgroup, err);
 }

 static void attach_rules(struct list_head *rules, struct net_device *dev)
diff --git a/net/core/neighbour.c b/net/core/neighbour.c
index d89c6fe..6f61207 100644
--- a/net/core/neighbour.c
+++ b/net/core/neighbour.c
@@ -2453,10 +2453,10 @@ static void __neigh_notify(struct neighbour *n, int type, int flags)
  /* failure implies BUG in neigh_nlmsg_size() */
  BUG_ON(err < 0);

- err = rtnl_notify(skb, 0, RTNLGRP_NEIGH, NULL, GFP_ATOMIC);
+ err = rtnl_notify(skb, init_net(), 0, RTNLGRP_NEIGH, NULL, GFP_ATOMIC);
 errout:
```

```
  if (err < 0)
-  rtnl_set_sk_err(RTNLGRP_NEIGH, err);
+  rtnl_set_sk_err(init_net(), RTNLGRP_NEIGH, err);
 }

 void neigh_app_ns(struct neighbour *n)
diff --git a/net/core/rtnetlink.c b/net/core/rtnetlink.c
index 9be586c..29a81bf 100644
--- a/net/core/rtnetlink.c
+++ b/net/core/rtnetlink.c
@@ -58,7 +58,7 @@
 #endif /* CONFIG_NET_WIRELESS_RTNETLINK */

 static DEFINE_MUTEX(rtnl_mutex);
-static struct sock *rtnl;
+static DEFINE_PER_NET(struct sock *, rtnl);

 void rtnl_lock(void)
 {
@@ -72,9 +72,17 @@ void __rtnl_unlock(void)

 void rtnl_unlock(void)
 {
+ net_t net;
  mutex_unlock(&rtnl_mutex);
- if (rtnl && rtnl->sk_receive_queue.qlen)
-  rtnl->sk_data_ready(rtnl, 0);
+
+ net_lock();
+ for_each_net(net) {
+  struct sock *rtnl = per_net(rtnl, net);
+  if (rtnl && rtnl->sk_receive_queue.qlen)
+   rtnl->sk_data_ready(rtnl, 0);
+ }
+ net_unlock();
+
  netdev_run_todo();
 }

@@ -151,8 +159,9 @@ size_t rtattr_strlcpy(char *dest, const struct rtattr *rta, size_t size)
  return ret;
 }

-int rtnetlink_send(struct sk_buff *skb, u32 pid, unsigned group, int echo)
+int rtnetlink_send(struct sk_buff *skb, net_t net, u32 pid, unsigned group, int echo)
 {
+ struct sock *rtnl = per_net(rtnl, net);
  int err = 0;
```

```
  NETLINK_CB(skb).dst_group = group;
@@ -164,14 +173,17 @@ int rtnetlink_send(struct sk_buff *skb, u32 pid, unsigned group, int
echo)
  return err;
 }

-int rtnl_unicast(struct sk_buff *skb, u32 pid)
+int rtnl_unicast(struct sk_buff *skb, net_t net, u32 pid)
 {
+ struct sock *rtnl = per_net(rtnl, net);
+
  return nlmsg_unicast(rtnl, skb, pid);
 }

-int rtnl_notify(struct sk_buff *skb, u32 pid, u32 group,
+int rtnl_notify(struct sk_buff *skb, net_t net, u32 pid, u32 group,
    struct nlmsghdr *nlh, gfp_t flags)
 {
+ struct sock *rtnl = per_net(rtnl, net);
  int report = 0;

  if (nlh)
@@ -180,8 +192,10 @@ int rtnl_notify(struct sk_buff *skb, u32 pid, u32 group,
  return nlmsg_notify(rtnl, skb, pid, group, report, flags);
 }

-void rtnl_set_sk_err(u32 group, int error)
+void rtnl_set_sk_err(net_t net, u32 group, int error)
 {
+ struct sock *rtnl = per_net(rtnl, net);
+
  netlink_set_err(rtnl, 0, group, error);
 }

@@ -649,7 +663,7 @@ static int rtnl_getlink(struct sk_buff *skb, struct nlmsghdr* nlh, void *arg)
  /* failure impilies BUG in if_nlmsg_size or wireless_rtnetlink_get */
  BUG_ON(err < 0);

- err = rtnl_unicast(nskb, NETLINK_CB(skb).pid);
+ err = rtnl_unicast(nskb, net, NETLINK_CB(skb).pid);
 errout:
  kfree(iw_buf);
  dev_put(dev);
@@ -698,10 +712,10 @@ void rtmsg_ifinfo(int type, struct net_device *dev, unsigned change)
  /* failure implies BUG in if_nlmsg_size() */
  BUG_ON(err < 0);
```

```
- err = rtnl_notify(skb, 0, RTNLGRP_LINK, NULL, GFP_KERNEL);
+ err = rtnl_notify(skb, init_net(), 0, RTNLGRP_LINK, NULL, GFP_KERNEL);
 errout:
  if (err < 0)
-  rtnl_set_sk_err(RTNLGRP_LINK, err);
+  rtnl_set_sk_err(init_net(), RTNLGRP_LINK, err);
 }

 /* Protected by RTNL sempahore.  */
@@ -713,6 +727,7 @@ static int rtattr_max;
 static __inline__ int
 rtnetlink_rcv_msg(struct sk_buff *skb, struct nlmsghdr *nlh, int *errp)
 {
+ net_t net = skb->sk->sk_net;
  struct rtnetlink_link *link;
  struct rtnetlink_link *link_tab;
  int sz_idx, kind;
@@ -767,7 +782,7 @@ rtnetlink_rcv_msg(struct sk_buff *skb, struct nlmsghdr *nlh, int *errp)
   if (link->dumpit == NULL)
    goto err_inval;

-  if ((*errp = netlink_dump_start(rtnl, skb, nlh,
+  if ((*errp = netlink_dump_start(per_net(rtnl, net), skb, nlh,
       link->dumpit, NULL)) != 0) {
    return -1;
   }
@@ -875,6 +890,36 @@ static struct notifier_block rtnetlink_dev_notifier = {
  .notifier_call = rtnetlink_event,
 };

+
+static int rtnetlink_net_init(net_t net)
+{
+ struct sock *sk;
+ sk = netlink_kernel_create(net, NETLINK_ROUTE, RTNLGRP_MAX,
+       rtnetlink_rcv, THIS_MODULE);
+ if (!sk)
+  return -ENOMEM;
+
+ /* Don't hold an extra reference on the namespace */
+ put_net(sk->sk_net);
+ per_net(rtnl, net) = sk;
+ return 0;
+}
+
+static void rtnetlink_net_exit(net_t net)
+{
+ /* At the last minute lie and say this is a socket for the
```

```
+  * initial network namespace.  So the socket will be safe to
+  * free.
+  */
+ per_net(rtnl, net)->sk_net = get_net(init_net());
+ sock_put(per_net(rtnl, net));
+}
+
+static struct pernet_operations rtnetlink_net_ops = {
+ .init = rtnetlink_net_init,
+ .exit = rtnetlink_net_exit,
+};
+
 void __init rtnetlink_init(void)
 {
  int i;
@@ -887,10 +932,9 @@ void __init rtnetlink_init(void)
  if (!rta_buf)
   panic("rtnetlink_init: cannot allocate rta_buf\n");

- rtnl = netlink_kernel_create(init_net(), NETLINK_ROUTE, RTNLGRP_MAX,
--         rtnetlink_rcv, THIS_MODULE);
- if (rtnl == NULL)
--  panic("rtnetlink_init: cannot initialize rtnetlink\n");
+ if (register_pernet_subsys(&rtnetlink_net_ops))
+    panic("rtnetlink_init: cannot initialize rtnetlink\n");
+
  netlink_set_nonroot(NETLINK_ROUTE, NL_NONROOT_RECV);
  register_netdevice_notifier(&rtnetlink_dev_notifier);
  rtnetlink_links[PF_UNSPEC] = link_rtnetlink_table;
diff --git a/net/core/wireless.c b/net/core/wireless.c
index d1418bf..9036359 100644
--- a/net/core/wireless.c
+++ b/net/core/wireless.c
@@ -1935,7 +1935,7 @@ static void wireless_nlevent_process(unsigned long data)
  struct sk_buff *skb;

  while ((skb = skb_dequeue(&wireless_nlevent_queue)))
-  rtnl_notify(skb, 0, RTNLGRP_LINK, NULL, GFP_ATOMIC);
+  rtnl_notify(skb, init_net(), 0, RTNLGRP_LINK, NULL, GFP_ATOMIC);
 }

 static DECLARE_TASKLET(wireless_nlevent_tasklet, wireless_nlevent_process, 0);
@@ -1992,6 +1992,9 @@ static inline void rtmsg_iwinfo(struct net_device * dev,
  struct sk_buff *skb;
  int size = NLMSG_GOODSIZE;

+ if (!net_eq(dev->nd_net, init_net()))
+  return;
```

```
+
  skb = alloc_skb(size, GFP_ATOMIC);
  if (!skb)
   return;
diff --git a/net/decnet/dn_dev.c b/net/decnet/dn_dev.c
index a09275b..bad972d 100644
--- a/net/decnet/dn_dev.c
+++ b/net/decnet/dn_dev.c
@@ -788,10 +788,10 @@ static void dn_ifaddr_notify(int event, struct dn_ifaddr *ifa)
  /* failure implies BUG in dn_ifaddr_nlmsg_size() */
  BUG_ON(err < 0);

- err = rtnl_notify(skb, 0, RTNLGRP_DECnet_IFADDR, NULL, GFP_KERNEL);
+ err = rtnl_notify(skb, init_net(), 0, RTNLGRP_DECnet_IFADDR, NULL, GFP_KERNEL);
 errout:
  if (err < 0)
-  rtnl_set_sk_err(RTNLGRP_DECnet_IFADDR, err);
+  rtnl_set_sk_err(init_net(), RTNLGRP_DECnet_IFADDR, err);
 }

 static int dn_nl_dump_ifaddr(struct sk_buff *skb, struct netlink_callback *cb)
diff --git a/net/decnet/dn_route.c b/net/decnet/dn_route.c
index d942ea0..4b353d4 100644
--- a/net/decnet/dn_route.c
+++ b/net/decnet/dn_route.c
@@ -1604,7 +1604,7 @@ int dn_cache_getroute(struct sk_buff *in_skb, struct nlmsghdr *nlh,
void *arg)
   goto out_free;
  }

- return rtnl_unicast(skb, NETLINK_CB(in_skb).pid);
+ return rtnl_unicast(skb, init_net(), NETLINK_CB(in_skb).pid);

 out_free:
  kfree_skb(skb);
diff --git a/net/decnet/dn_table.c b/net/decnet/dn_table.c
index 3ff151c..4090ab5 100644
--- a/net/decnet/dn_table.c
+++ b/net/decnet/dn_table.c
@@ -371,10 +371,10 @@ static void dn_rtmsg_fib(int event, struct dn_fib_node *f, int z, u32
tb_id,
  /* failure implies BUG in dn_fib_nlmsg_size() */
  BUG_ON(err < 0);

- err = rtnl_notify(skb, pid, RTNLGRP_DECnet_ROUTE, nlh, GFP_KERNEL);
+ err = rtnl_notify(skb, init_net(), pid, RTNLGRP_DECnet_ROUTE, nlh, GFP_KERNEL);
 errout:
  if (err < 0)
```

```
-  rtnl_set_sk_err(RTNLGRP_DECnet_ROUTE, err);
+  rtnl_set_sk_err(init_net(), RTNLGRP_DECnet_ROUTE, err);
 }

 static __inline__ int dn_hash_dump_bucket(struct sk_buff *skb,
diff --git a/net/ipv4/devinet.c b/net/ipv4/devinet.c
index 7769b1c..59acce2 100644
--- a/net/ipv4/devinet.c
+++ b/net/ipv4/devinet.c
@@ -1241,10 +1241,10 @@ static void rtmsg_ifa(int event, struct in_ifaddr* ifa, struct nlmsghdr
*nlh,
  /* failure implies BUG in inet_nlmsg_size() */
  BUG_ON(err < 0);

- err = rtnl_notify(skb, pid, RTNLGRP_IPV4_IFADDR, nlh, GFP_KERNEL);
+ err = rtnl_notify(skb, init_net(), pid, RTNLGRP_IPV4_IFADDR, nlh, GFP_KERNEL);
 errout:
  if (err < 0)
-  rtnl_set_sk_err(RTNLGRP_IPV4_IFADDR, err);
+  rtnl_set_sk_err(init_net(), RTNLGRP_IPV4_IFADDR, err);
 }

 static struct rtnetlink_link inet_rtnetlink_table[RTM_NR_MSGTYPES] = {
diff --git a/net/ipv4/fib_semantics.c b/net/ipv4/fib_semantics.c
index 76218e5..8c64334 100644
--- a/net/ipv4/fib_semantics.c
+++ b/net/ipv4/fib_semantics.c
@@ -317,11 +317,11 @@ void rtmsg_fib(int event, __be32 key, struct fib_alias *fa,
  /* failure implies BUG in fib_nlmsg_size() */
  BUG_ON(err < 0);

- err = rtnl_notify(skb, info->pid, RTNLGRP_IPV4_ROUTE,
+ err = rtnl_notify(skb, init_net(), info->pid, RTNLGRP_IPV4_ROUTE,
      info->nlh, GFP_KERNEL);
 errout:
  if (err < 0)
-  rtnl_set_sk_err(RTNLGRP_IPV4_ROUTE, err);
+  rtnl_set_sk_err(init_net(), RTNLGRP_IPV4_ROUTE, err);
 }

 /* Return the first fib alias matching TOS with
diff --git a/net/ipv4/ipmr.c b/net/ipv4/ipmr.c
index d2e7e55..15e0eb4 100644
--- a/net/ipv4/ipmr.c
+++ b/net/ipv4/ipmr.c
@@ -314,7 +314,7 @@ static void ipmr_destroy_unres(struct mfc_cache *c)
    e->error = -ETIMEDOUT;
    memset(&e->msg, 0, sizeof(e->msg));
```

```
-   rtnl_unicast(skb, NETLINK_CB(skb).pid);
+   rtnl_unicast(skb, init_net(), NETLINK_CB(skb).pid);
    } else
     kfree_skb(skb);
   }
@@ -527,7 +527,7 @@ static void ipmr_cache_resolve(struct mfc_cache *uc, struct mfc_cache
*c)
      memset(&e->msg, 0, sizeof(e->msg));
     }

-   rtnl_unicast(skb, NETLINK_CB(skb).pid);
+   rtnl_unicast(skb, init_net(), NETLINK_CB(skb).pid);
    } else
     ip_mr_forward(skb, c, 0);
   }
diff --git a/net/ipv4/route.c b/net/ipv4/route.c
index 509bfb1..5f8592e 100644
--- a/net/ipv4/route.c
+++ b/net/ipv4/route.c
@@ -2802,7 +2802,7 @@ int inet_rtm_getroute(struct sk_buff *in_skb, struct nlmsghdr* nlh, void
*arg)
   if (err <= 0)
    goto errout_free;

- err = rtnl_unicast(skb, NETLINK_CB(in_skb).pid);
+ err = rtnl_unicast(skb, init_net(), NETLINK_CB(in_skb).pid);
 errout:
   return err;

diff --git a/net/ipv6/addrconf.c b/net/ipv6/addrconf.c
index 83b7312..597bc10 100644
--- a/net/ipv6/addrconf.c
+++ b/net/ipv6/addrconf.c
@@ -3362,7 +3362,7 @@ static int inet6_rtm_getaddr(struct sk_buff *in_skb, struct nlmsghdr*
nlh,
   /* failure implies BUG in inet6_ifaddr_msgsize() */
   BUG_ON(err < 0);

- err = rtnl_unicast(skb, NETLINK_CB(in_skb).pid);
+ err = rtnl_unicast(skb, init_net(), NETLINK_CB(in_skb).pid);
 errout_ifa:
   in6_ifa_put(ifa);
 errout:
@@ -3382,10 +3382,10 @@ static void inet6_ifa_notify(int event, struct inet6_ifaddr *ifa)
   /* failure implies BUG in inet6_ifaddr_msgsize() */
   BUG_ON(err < 0);
```

```
- err = rtnl_notify(skb, 0, RTNLGRP_IPV6_IFADDR, NULL, GFP_ATOMIC);
+ err = rtnl_notify(skb, init_net(), 0, RTNLGRP_IPV6_IFADDR, NULL, GFP_ATOMIC);
 errout:
  if (err < 0)
-  rtnl_set_sk_err(RTNLGRP_IPV6_IFADDR, err);
+  rtnl_set_sk_err(init_net(), RTNLGRP_IPV6_IFADDR, err);
 }

 static void inline ipv6_store_devconf(struct ipv6_devconf *cnf,
@@ -3539,10 +3539,10 @@ void inet6_ifinfo_notify(int event, struct inet6_dev *idev)
  /* failure implies BUG in inet6_if_nlmsg_size() */
  BUG_ON(err < 0);

- err = rtnl_notify(skb, 0, RTNLGRP_IPV6_IFADDR, NULL, GFP_ATOMIC);
+ err = rtnl_notify(skb, init_net(), 0, RTNLGRP_IPV6_IFADDR, NULL, GFP_ATOMIC);
 errout:
  if (err < 0)
-  rtnl_set_sk_err(RTNLGRP_IPV6_IFADDR, err);
+  rtnl_set_sk_err(init_net(), RTNLGRP_IPV6_IFADDR, err);
 }

 static inline size_t inet6_prefix_nlmsg_size(void)
@@ -3604,10 +3604,10 @@ static void inet6_prefix_notify(int event, struct inet6_dev *idev,
  /* failure implies BUG in inet6_prefix_nlmsg_size() */
  BUG_ON(err < 0);

- err = rtnl_notify(skb, 0, RTNLGRP_IPV6_PREFIX, NULL, GFP_ATOMIC);
+ err = rtnl_notify(skb, init_net(), 0, RTNLGRP_IPV6_PREFIX, NULL, GFP_ATOMIC);
 errout:
  if (err < 0)
-  rtnl_set_sk_err(RTNLGRP_IPV6_PREFIX, err);
+  rtnl_set_sk_err(init_net(), RTNLGRP_IPV6_PREFIX, err);
 }

 static struct rtnetlink_link inet6_rtnetlink_table[RTM_NR_MSGTYPES] = {
diff --git a/net/ipv6/route.c b/net/ipv6/route.c
index 02fd8ae..cf568f6 100644
--- a/net/ipv6/route.c
+++ b/net/ipv6/route.c
@@ -2210,7 +2210,7 @@ int inet6_rtm_getroute(struct sk_buff *in_skb, struct nlmsghdr* nlh,
void *arg)
   goto errout;
  }

- err = rtnl_unicast(skb, NETLINK_CB(in_skb).pid);
+ err = rtnl_unicast(skb, init_net(), NETLINK_CB(in_skb).pid);
 errout:
  return err;
```

```
 }
@@ -2237,10 +2237,10 @@ void inet6_rt_notify(int event, struct rt6_info *rt, struct nl_info *info)
  /* failure implies BUG in rt6_nlmsg_size() */
  BUG_ON(err < 0);

- err = rtnl_notify(skb, pid, RTNLGRP_IPV6_ROUTE, nlh, gfp_any());
+ err = rtnl_notify(skb, init_net(), pid, RTNLGRP_IPV6_ROUTE, nlh, gfp_any());
 errout:
  if (err < 0)
-  rtnl_set_sk_err(RTNLGRP_IPV6_ROUTE, err);
+  rtnl_set_sk_err(init_net(), RTNLGRP_IPV6_ROUTE, err);
 }

 /*
diff --git a/net/sched/cls_api.c b/net/sched/cls_api.c
index 09a3ec8..c69b4fc 100644
--- a/net/sched/cls_api.c
+++ b/net/sched/cls_api.c
@@ -369,7 +369,7 @@ static int tfilter_notify(struct sk_buff *oskb, struct nlmsghdr *n,
   return -EINVAL;
  }

- return rtnetlink_send(skb, pid, RTNLGRP_TC, n->nlmsg_flags&NLM_F_ECHO);
+ return rtnetlink_send(skb, init_net(), pid, RTNLGRP_TC, n->nlmsg_flags&NLM_F_ECHO);
 }

 struct tcf_dump_args
diff --git a/net/sched/sch_api.c b/net/sched/sch_api.c
index 7e33f73..ae55988 100644
--- a/net/sched/sch_api.c
+++ b/net/sched/sch_api.c
@@ -850,7 +850,7 @@ static int qdisc_notify(struct sk_buff *oskb, struct nlmsghdr *n,
  }

  if (skb->len)
-  return rtnetlink_send(skb, pid, RTNLGRP_TC, n->nlmsg_flags&NLM_F_ECHO);
+  return rtnetlink_send(skb, init_net(), pid, RTNLGRP_TC, n->nlmsg_flags&NLM_F_ECHO);

 err_out:
  kfree_skb(skb);
@@ -1082,7 +1082,7 @@ static int tclass_notify(struct sk_buff *oskb, struct nlmsghdr *n,
   return -EINVAL;
  }

- return rtnetlink_send(skb, pid, RTNLGRP_TC, n->nlmsg_flags&NLM_F_ECHO);
+ return rtnetlink_send(skb, init_net(), pid, RTNLGRP_TC, n->nlmsg_flags&NLM_F_ECHO);
 }
```

struct qdisc_dump_args
--
1.4.4.1.g278f

_____

Containers mailing list
Containers@lists.osdl.org
https://lists.osdl.org/mailman/listinfo/containers