Subject: [PATCH 2.6.25 1/6] net: Modify all rtnetlink methods to only work in the initial namespace (v2)
Posted by den on Thu, 15 Nov 2007 15:56:48 GMT
View Forum Message <> Reply to Message

Before I can enable rtnetlink to work in all network namespaces
I need to be certain that something won't break. So this
patch deliberately disables all of the rtnletlink methods in everything
except the initial network namespace. After the methods have been
audited this extra check can be disabled.

Changes from v1:
- added IPv6 addrlabel protection

Signed-off-by: Denis V. Lunev <den@openvz.org>
Signed-off-by: Eric W. Biederman <ebiederm@xmission.com>
---
net/bridge/br_netlink.c | 9 +++++++++
net/core/fib_rules.c | 11 +++++++++++
net/core/neighbour.c | 18 ++++++++++++++++++
net/core/rtnetlink.c | 19 +++++++++++++++++++
net/decnet/dn_dev.c | 12 ++++++++++++
net/decnet/dn_fib.c | 8 ++++++++
net/decnet/dn_route.c | 8 ++++++++
net/decnet/dn_table.c | 4 ++++
net/ipv4/devinet.c | 12 ++++++++++++
net/ipv4/fib_frontend.c | 12 ++++++++++++
net/ipv4/route.c | 4 ++++
net/ipv6/addrconf.c | 31 +++++++++++++++++++++++++++++++
net/ipv6/addrlabel.c | 12 ++++++++++++
net/ipv6/ip6_fib.c | 4 ++++
net/ipv6/route.c | 12 ++++++++++++
net/sched/act_api.c | 10 ++++++++++
net/sched/cls_api.c | 10 ++++++++++
net/sched/sch_api.c | 21 +++++++++++++++++++++
18 files changed, 217 insertions(+), 0 deletions(-)

diff --git a/net/bridge/br_netlink.c b/net/bridge/br_netlink.c
index 53ab8e0..a4ffa2b 100644
--- a/net/bridge/br_netlink.c
+++ b/net/bridge/br_netlink.c
@@ -13,6 +13,7 @@
#include <linux/kernel.h>
#include <net/rtnetlink.h>
#include <net/net_namespace.h>
+#include <net/sock.h>
#include "br_private.h"

```
 static inline size_t br_nlmsg_size(void)
@@ -107,9 +108,13 @@ errout:
  */
 static int br_dump_ifinfo(struct sk_buff *skb, struct netlink_callback *cb)
 {
+ struct net *net = skb->sk->sk_net;
  struct net_device *dev;
  int idx;

+ if (net != &init_net)
+  return 0;
+
  idx = 0;
  for_each_netdev(&init_net, dev) {
   /* not a bridge port */
@@ -135,12 +140,16 @@ skip:
  */
 static int br_rtm_setlink(struct sk_buff *skb,  struct nlmsghdr *nlh, void *arg)
 {
+ struct net *net = skb->sk->sk_net;
  struct ifinfomsg *ifm;
  struct nlattr *protinfo;
  struct net_device *dev;
  struct net_bridge_port *p;
  u8 new_state;

+ if (net != &init_net)
+  return -EINVAL;
+
  if (nlmsg_len(nlh) < sizeof(*ifm))
   return -EINVAL;

diff --git a/net/core/fib_rules.c b/net/core/fib_rules.c
index 848132b..3b20b6f 100644
--- a/net/core/fib_rules.c
+++ b/net/core/fib_rules.c
@@ -228,6 +228,9 @@ static int fib_nl_newrule(struct sk_buff *skb, struct nlmsghdr* nlh, void
*arg)
  struct nlattr *tb[FRA_MAX+1];
  int err = -EINVAL, unresolved = 0;

+ if (net != &init_net)
+  return -EINVAL;
+
  if (nlh->nlmsg_len < nlmsg_msg_size(sizeof(*frh)))
   goto errout;

@@ -358,12 +361,16 @@ errout:
```

```
 static int fib_nl_delrule(struct sk_buff *skb, struct nlmsghdr* nlh, void *arg)
 {
+ struct net *net = skb->sk->sk_net;
  struct fib_rule_hdr *frh = nlmsg_data(nlh);
  struct fib_rules_ops *ops = NULL;
  struct fib_rule *rule, *tmp;
  struct nlattr *tb[FRA_MAX+1];
  int err = -EINVAL;

+ if (net != &init_net)
+  return -EINVAL;
+
  if (nlh->nlmsg_len < nlmsg_msg_size(sizeof(*frh)))
   goto errout;

@@ -539,9 +546,13 @@ skip:

 static int fib_nl_dumprule(struct sk_buff *skb, struct netlink_callback *cb)
 {
+ struct net *net = skb->sk->sk_net;
  struct fib_rules_ops *ops;
  int idx = 0, family;

+ if (net != &init_net)
+  return -EINVAL;
+
  family = rtnl_msg_family(cb->nlh);
  if (family != AF_UNSPEC) {
   /* Protocol specific dump request */
diff --git a/net/core/neighbour.c b/net/core/neighbour.c
index 175bbc0..29f0a4d 100644
--- a/net/core/neighbour.c
+++ b/net/core/neighbour.c
@@ -1449,6 +1449,9 @@ static int neigh_delete(struct sk_buff *skb, struct nlmsghdr *nlh, void
*arg)
  struct net_device *dev = NULL;
  int err = -EINVAL;

+ if (net != &init_net)
+  return -EINVAL;
+
  if (nlmsg_len(nlh) < sizeof(*ndm))
   goto out;

@@ -1515,6 +1518,9 @@ static int neigh_add(struct sk_buff *skb, struct nlmsghdr *nlh, void
*arg)
  struct net_device *dev = NULL;
```

```
  int err;

+ if (net != &init_net)
+  return -EINVAL;
+
  err = nlmsg_parse(nlh, sizeof(*ndm), tb, NDA_MAX, NULL);
  if (err < 0)
   goto out;
@@ -1789,11 +1795,15 @@ static const struct nla_policy nl_ntbl_parm_policy[NDTPA_MAX+1]
= {

 static int neightbl_set(struct sk_buff *skb, struct nlmsghdr *nlh, void *arg)
 {
+ struct net *net = skb->sk->sk_net;
  struct neigh_table *tbl;
  struct ndtmsg *ndtmsg;
  struct nlattr *tb[NDTA_MAX+1];
  int err;

+ if (net != &init_net)
+  return -EINVAL;
+
  err = nlmsg_parse(nlh, sizeof(*ndtmsg), tb, NDTA_MAX,
      nl_neightbl_policy);
  if (err < 0)
@@ -1913,11 +1923,15 @@ errout:

 static int neightbl_dump_info(struct sk_buff *skb, struct netlink_callback *cb)
 {
+ struct net *net = skb->sk->sk_net;
  int family, tidx, nidx = 0;
  int tbl_skip = cb->args[0];
  int neigh_skip = cb->args[1];
  struct neigh_table *tbl;

+ if (net != &init_net)
+  return 0;
+
  family = ((struct rtgenmsg *) nlmsg_data(cb->nlh))->rtgen_family;

  read_lock(&neigh_tbl_lock);
@@ -2042,9 +2056,13 @@ out:

 static int neigh_dump_info(struct sk_buff *skb, struct netlink_callback *cb)
 {
+ struct net *net = skb->sk->sk_net;
  struct neigh_table *tbl;
  int t, family, s_t;
```

```
+ if (net != &init_net)
+  return 0;
+
  read_lock(&neigh_tbl_lock);
  family = ((struct rtgenmsg *) nlmsg_data(cb->nlh))->rtgen_family;
  s_t = cb->args[0];
diff --git a/net/core/rtnetlink.c b/net/core/rtnetlink.c
index e1ba26f..219bb31 100644
--- a/net/core/rtnetlink.c
+++ b/net/core/rtnetlink.c
@@ -700,6 +700,9 @@ 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)
@@ -902,6 +905,9 @@ 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;
@@ -950,6 +956,9 @@ 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;
@@ -1031,6 +1040,9 @@ 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
@@ -1157,6 +1169,9 @@ 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;
@@ -1192,9 +1207,13 @@ 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++) {
diff --git a/net/decnet/dn_dev.c b/net/decnet/dn_dev.c
index 66e266f..a92f3f9 100644
--- a/net/decnet/dn_dev.c
+++ b/net/decnet/dn_dev.c
@@ -647,12 +647,16 @@ static const struct nla_policy dn_ifa_policy[IFA_MAX+1] = {

 static int dn_nl_deladdr(struct sk_buff *skb, struct nlmsghdr *nlh, void *arg)
 {
+ struct net *net = skb->sk->sk_net;
  struct nlattr *tb[IFA_MAX+1];
  struct dn_dev *dn_db;
  struct ifaddrmsg *ifm;
  struct dn_ifaddr *ifa, **ifap;
  int err = -EADDRNOTAVAIL;

+ if (net != &init_net)
+  goto errout;
+
  err = nlmsg_parse(nlh, sizeof(*ifm), tb, IFA_MAX, dn_ifa_policy);
```

```
  if (err < 0)
   goto errout;
@@ -679,6 +683,7 @@ errout:

 static int dn_nl_newaddr(struct sk_buff *skb, struct nlmsghdr *nlh, void *arg)
 {
+ struct net *net = skb->sk->sk_net;
  struct nlattr *tb[IFA_MAX+1];
  struct net_device *dev;
  struct dn_dev *dn_db;
@@ -686,6 +691,9 @@ static int dn_nl_newaddr(struct sk_buff *skb, struct nlmsghdr *nlh, void
*arg)
  struct dn_ifaddr *ifa;
  int err;

+ if (net != &init_net)
+  return -EINVAL;
+
  err = nlmsg_parse(nlh, sizeof(*ifm), tb, IFA_MAX, dn_ifa_policy);
  if (err < 0)
   return err;
@@ -791,11 +799,15 @@ errout:

 static int dn_nl_dump_ifaddr(struct sk_buff *skb, struct netlink_callback *cb)
 {
+ struct net *net = skb->sk->sk_net;
  int idx, dn_idx = 0, skip_ndevs, skip_naddr;
  struct net_device *dev;
  struct dn_dev *dn_db;
  struct dn_ifaddr *ifa;

+ if (net != &init_net)
+  return 0;
+
  skip_ndevs = cb->args[0];
  skip_naddr = cb->args[1];

diff --git a/net/decnet/dn_fib.c b/net/decnet/dn_fib.c
index 3760a20..5413e1b 100644
--- a/net/decnet/dn_fib.c
+++ b/net/decnet/dn_fib.c
@@ -506,10 +506,14 @@ static int dn_fib_check_attr(struct rtmsg *r, struct rtattr **rta)

 static int dn_fib_rtm_delroute(struct sk_buff *skb, struct nlmsghdr *nlh, void *arg)
 {
+ struct net *net = skb->sk->sk_net;
  struct dn_fib_table *tb;
  struct rtattr **rta = arg;
```

```
  struct rtmsg *r = NLMSG_DATA(nlh);

+ if (net != &init_net)
+  return -EINVAL;
+
  if (dn_fib_check_attr(r, rta))
   return -EINVAL;

@@ -522,10 +526,14 @@ static int dn_fib_rtm_delroute(struct sk_buff *skb, struct nlmsghdr *nlh,
void *

 static int dn_fib_rtm_newroute(struct sk_buff *skb, struct nlmsghdr *nlh, void *arg)
 {
+ struct net *net = skb->sk->sk_net;
  struct dn_fib_table *tb;
  struct rtattr **rta = arg;
  struct rtmsg *r = NLMSG_DATA(nlh);

+ if (net != &init_net)
+  return -EINVAL;
+
  if (dn_fib_check_attr(r, rta))
   return -EINVAL;

diff --git a/net/decnet/dn_route.c b/net/decnet/dn_route.c
index 6ee7846..7cf97cf 100644
--- a/net/decnet/dn_route.c
+++ b/net/decnet/dn_route.c
@@ -1511,6 +1511,7 @@ rtattr_failure:
  */
 static int dn_cache_getroute(struct sk_buff *in_skb, struct nlmsghdr *nlh, void *arg)
 {
+ struct net *net = in_skb->sk->sk_net;
  struct rtattr **rta = arg;
  struct rtmsg *rtm = NLMSG_DATA(nlh);
  struct dn_route *rt = NULL;
@@ -1519,6 +1520,9 @@ static int dn_cache_getroute(struct sk_buff *in_skb, struct nlmsghdr
*nlh, void
  struct sk_buff *skb;
  struct flowi fl;

+ if (net != &init_net)
+  return -EINVAL;
+
  memset(&fl, 0, sizeof(fl));
  fl.proto = DNPROTO_NSP;

@@ -1596,10 +1600,14 @@ out_free:
```

```
  */
 int dn_cache_dump(struct sk_buff *skb, struct netlink_callback *cb)
 {
+ struct net *net = skb->sk->sk_net;
  struct dn_route *rt;
  int h, s_h;
  int idx, s_idx;

+ if (net != &init_net)
+  return 0;
+
  if (NLMSG_PAYLOAD(cb->nlh, 0) < sizeof(struct rtmsg))
   return -EINVAL;
  if (!(((struct rtmsg *)NLMSG_DATA(cb->nlh))->rtm_flags&RTM_F_CLONED))
diff --git a/net/decnet/dn_table.c b/net/decnet/dn_table.c
index fda0772..a3bdb8d 100644
--- a/net/decnet/dn_table.c
+++ b/net/decnet/dn_table.c
@@ -463,12 +463,16 @@ static int dn_fib_table_dump(struct dn_fib_table *tb, struct sk_buff
*skb,

 int dn_fib_dump(struct sk_buff *skb, struct netlink_callback *cb)
 {
+ struct net *net = skb->sk->sk_net;
  unsigned int h, s_h;
  unsigned int e = 0, s_e;
  struct dn_fib_table *tb;
  struct hlist_node *node;
  int dumped = 0;

+ if (net != &init_net)
+  return 0;
+
  if (NLMSG_PAYLOAD(cb->nlh, 0) >= sizeof(struct rtmsg) &&
   ((struct rtmsg *)NLMSG_DATA(cb->nlh))->rtm_flags&RTM_F_CLONED)
    return dn_cache_dump(skb, cb);
diff --git a/net/ipv4/devinet.c b/net/ipv4/devinet.c
index 55d199e..0ff2ef6 100644
--- a/net/ipv4/devinet.c
+++ b/net/ipv4/devinet.c
@@ -441,6 +441,7 @@ struct in_ifaddr *inet_ifa_byprefix(struct in_device *in_dev, __be32 prefix,

 static int inet_rtm_deladdr(struct sk_buff *skb, struct nlmsghdr *nlh, void *arg)
 {
+ struct net *net = skb->sk->sk_net;
  struct nlattr *tb[IFA_MAX+1];
  struct in_device *in_dev;
  struct ifaddrmsg *ifm;
```

```
@@ -449,6 +450,9 @@ static int inet_rtm_deladdr(struct sk_buff *skb, struct nlmsghdr *nlh, void
*arg

  ASSERT_RTNL();

+ if (net != &init_net)
+  return -EINVAL;
+
  err = nlmsg_parse(nlh, sizeof(*ifm), tb, IFA_MAX, ifa_ipv4_policy);
  if (err < 0)
   goto errout;
@@ -561,10 +565,14 @@ errout:

 static int inet_rtm_newaddr(struct sk_buff *skb, struct nlmsghdr *nlh, void *arg)
 {
+ struct net *net = skb->sk->sk_net;
  struct in_ifaddr *ifa;

  ASSERT_RTNL();

+ if (net != &init_net)
+  return -EINVAL;
+
  ifa = rtm_to_ifaddr(nlh);
  if (IS_ERR(ifa))
   return PTR_ERR(ifa);
@@ -1175,12 +1183,16 @@ nla_put_failure:

 static int inet_dump_ifaddr(struct sk_buff *skb, struct netlink_callback *cb)
 {
+ struct net *net = skb->sk->sk_net;
  int idx, ip_idx;
  struct net_device *dev;
  struct in_device *in_dev;
  struct in_ifaddr *ifa;
  int s_ip_idx, s_idx = cb->args[0];

+ if (net != &init_net)
+  return 0;
+
  s_ip_idx = ip_idx = cb->args[1];
  idx = 0;
  for_each_netdev(&init_net, dev) {
diff --git a/net/ipv4/fib_frontend.c b/net/ipv4/fib_frontend.c
index 732d8f0..c211887 100644
--- a/net/ipv4/fib_frontend.c
+++ b/net/ipv4/fib_frontend.c
@@ -538,10 +538,14 @@ errout:
```

```
 static int inet_rtm_delroute(struct sk_buff *skb, struct nlmsghdr* nlh, void *arg)
 {
+ struct net *net = skb->sk->sk_net;
  struct fib_config cfg;
  struct fib_table *tb;
  int err;

+ if (net != &init_net)
+  return -EINVAL;
+
  err = rtm_to_fib_config(skb, nlh, &cfg);
  if (err < 0)
   goto errout;
@@ -559,10 +563,14 @@ errout:

 static int inet_rtm_newroute(struct sk_buff *skb, struct nlmsghdr* nlh, void *arg)
 {
+ struct net *net = skb->sk->sk_net;
  struct fib_config cfg;
  struct fib_table *tb;
  int err;

+ if (net != &init_net)
+  return -EINVAL;
+
  err = rtm_to_fib_config(skb, nlh, &cfg);
  if (err < 0)
   goto errout;
@@ -580,12 +588,16 @@ errout:

 static int inet_dump_fib(struct sk_buff *skb, struct netlink_callback *cb)
 {
+ struct net *net = skb->sk->sk_net;
  unsigned int h, s_h;
  unsigned int e = 0, s_e;
  struct fib_table *tb;
  struct hlist_node *node;
  int dumped = 0;

+ if (net != &init_net)
+  return 0;
+
  if (nlmsg_len(cb->nlh) >= sizeof(struct rtmsg) &&
      ((struct rtmsg *) nlmsg_data(cb->nlh))->rtm_flags & RTM_F_CLONED)
   return ip_rt_dump(skb, cb);
diff --git a/net/ipv4/route.c b/net/ipv4/route.c
index f0b28f9..4bb2db1 100644
```

```
--- a/net/ipv4/route.c
+++ b/net/ipv4/route.c
@@ -2524,6 +2524,7 @@ nla_put_failure:

 static int inet_rtm_getroute(struct sk_buff *in_skb, struct nlmsghdr* nlh, void *arg)
 {
+ struct net *net = in_skb->sk->sk_net;
  struct rtmsg *rtm;
  struct nlattr *tb[RTA_MAX+1];
  struct rtable *rt = NULL;
@@ -2533,6 +2534,9 @@ static int inet_rtm_getroute(struct sk_buff *in_skb, struct nlmsghdr*
nlh, void
  int err;
  struct sk_buff *skb;

+ if (net != &init_net)
+  return -EINVAL;
+
  err = nlmsg_parse(nlh, sizeof(*rtm), tb, RTA_MAX, rtm_ipv4_policy);
  if (err < 0)
   goto errout;
diff --git a/net/ipv6/addrconf.c b/net/ipv6/addrconf.c
index 80795e1..88ad440 100644
--- a/net/ipv6/addrconf.c
+++ b/net/ipv6/addrconf.c
@@ -2962,11 +2962,15 @@ static const struct nla_policy ifa_ipv6_policy[IFA_MAX+1] = {
 static int
 inet6_rtm_deladdr(struct sk_buff *skb, struct nlmsghdr *nlh, void *arg)
 {
+ struct net *net = skb->sk->sk_net;
  struct ifaddrmsg *ifm;
  struct nlattr *tb[IFA_MAX+1];
  struct in6_addr *pfx;
  int err;

+ if (net != &init_net)
+  return -EINVAL;
+
  err = nlmsg_parse(nlh, sizeof(*ifm), tb, IFA_MAX, ifa_ipv6_policy);
  if (err < 0)
   return err;
@@ -3019,6 +3023,7 @@ static int inet6_addr_modify(struct inet6_ifaddr *ifp, u8 ifa_flags,
 static int
 inet6_rtm_newaddr(struct sk_buff *skb, struct nlmsghdr *nlh, void *arg)
 {
+ struct net *net = skb->sk->sk_net;
  struct ifaddrmsg *ifm;
  struct nlattr *tb[IFA_MAX+1];
```

```
  struct in6_addr *pfx;
@@ -3028,6 +3033,9 @@ inet6_rtm_newaddr(struct sk_buff *skb, struct nlmsghdr *nlh, void
*arg)
  u8 ifa_flags;
  int err;

+ if (net != &init_net)
+  return -EINVAL;
+
  err = nlmsg_parse(nlh, sizeof(*ifm), tb, IFA_MAX, ifa_ipv6_policy);
  if (err < 0)
   return err;
@@ -3301,26 +3309,42 @@ done:

 static int inet6_dump_ifaddr(struct sk_buff *skb, struct netlink_callback *cb)
 {
+ struct net *net = skb->sk->sk_net;
  enum addr_type_t type = UNICAST_ADDR;
+
+ if (net != &init_net)
+  return 0;
+
  return inet6_dump_addr(skb, cb, type);
 }

 static int inet6_dump_ifmcaddr(struct sk_buff *skb, struct netlink_callback *cb)
 {
+ struct net *net = skb->sk->sk_net;
  enum addr_type_t type = MULTICAST_ADDR;
+
+ if (net != &init_net)
+  return 0;
+
  return inet6_dump_addr(skb, cb, type);
 }


 static int inet6_dump_ifacaddr(struct sk_buff *skb, struct netlink_callback *cb)
 {
+ struct net *net = skb->sk->sk_net;
  enum addr_type_t type = ANYCAST_ADDR;
+
+ if (net != &init_net)
+  return 0;
+
  return inet6_dump_addr(skb, cb, type);
 }
```

```
 static int inet6_rtm_getaddr(struct sk_buff *in_skb, struct nlmsghdr* nlh,
         void *arg)
 {
+ struct net *net = in_skb->sk->sk_net;
  struct ifaddrmsg *ifm;
  struct nlattr *tb[IFA_MAX+1];
  struct in6_addr *addr = NULL;
@@ -3329,6 +3353,9 @@ static int inet6_rtm_getaddr(struct sk_buff *in_skb, struct nlmsghdr*
nlh,
  struct sk_buff *skb;
  int err;

+ if (net != &init_net)
+  return -EINVAL;
+
  err = nlmsg_parse(nlh, sizeof(*ifm), tb, IFA_MAX, ifa_ipv6_policy);
  if (err < 0)
   goto errout;
@@ -3546,11 +3573,15 @@ nla_put_failure:

 static int inet6_dump_ifinfo(struct sk_buff *skb, struct netlink_callback *cb)
 {
+ struct net *net = skb->sk->sk_net;
  int idx, err;
  int s_idx = cb->args[0];
  struct net_device *dev;
  struct inet6_dev *idev;

+ if (net != &init_net)
+  return 0;
+
  read_lock(&dev_base_lock);
  idx = 0;
  for_each_netdev(&init_net, dev) {
diff --git a/net/ipv6/addrlabel.c b/net/ipv6/addrlabel.c
index 204d4d6..b9b5d57 100644
--- a/net/ipv6/addrlabel.c
+++ b/net/ipv6/addrlabel.c
@@ -361,12 +361,16 @@ static const struct nla_policy ifal_policy[IFAL_MAX+1] = {
 static int ip6addrlbl_newdel(struct sk_buff *skb, struct nlmsghdr *nlh,
         void *arg)
 {
+ struct net *net = skb->sk->sk_net;
  struct ifaddrlblmsg *ifal;
  struct nlattr *tb[IFAL_MAX+1];
  struct in6_addr *pfx;
  u32 label;
  int err = 0;
```

```
+ if (net != &init_net)
+  return 0;
+
  err = nlmsg_parse(nlh, sizeof(*ifal), tb, IFAL_MAX, ifal_policy);
  if (err < 0)
   return err;
@@ -445,11 +449,15 @@ static int ip6addrlbl_fill(struct sk_buff *skb,

 static int ip6addrlbl_dump(struct sk_buff *skb, struct netlink_callback *cb)
 {
+ struct net *net = skb->sk->sk_net;
  struct ip6addrlbl_entry *p;
  struct hlist_node *pos;
  int idx = 0, s_idx = cb->args[0];
  int err;

+ if (net != &init_net)
+  return 0;
+
  rcu_read_lock();
  hlist_for_each_entry_rcu(p, pos, &ip6addrlbl_table.head, list) {
   if (idx >= s_idx) {
@@ -479,6 +487,7 @@ static inline int ip6addrlbl_msgsize(void)
 static int ip6addrlbl_get(struct sk_buff *in_skb, struct nlmsghdr* nlh,
      void *arg)
 {
+ struct net *net = in_skb->sk->sk_net;
  struct ifaddrlblmsg *ifal;
  struct nlattr *tb[IFAL_MAX+1];
  struct in6_addr *addr;
@@ -487,6 +496,9 @@ static int ip6addrlbl_get(struct sk_buff *in_skb, struct nlmsghdr* nlh,
  struct ip6addrlbl_entry *p;
  struct sk_buff *skb;

+ if (net != &init_net)
+  return 0;
+
  err = nlmsg_parse(nlh, sizeof(*ifal), tb, IFAL_MAX, ifal_policy);
  if (err < 0)
   return err;
diff --git a/net/ipv6/ip6_fib.c b/net/ipv6/ip6_fib.c
index 946cf38..31b60a0 100644
--- a/net/ipv6/ip6_fib.c
+++ b/net/ipv6/ip6_fib.c
@@ -361,6 +361,7 @@ end:

 static int inet6_dump_fib(struct sk_buff *skb, struct netlink_callback *cb)
```

```
 {
+ struct net *net = skb->sk->sk_net;
  unsigned int h, s_h;
  unsigned int e = 0, s_e;
  struct rt6_rtnl_dump_arg arg;
@@ -369,6 +370,9 @@ static int inet6_dump_fib(struct sk_buff *skb, struct netlink_callback *cb)
  struct hlist_node *node;
  int res = 0;

+ if (net != &init_net)
+  return 0;
+
  s_h = cb->args[0];
  s_e = cb->args[1];

diff --git a/net/ipv6/route.c b/net/ipv6/route.c
index feff707..6a2ca79 100644
--- a/net/ipv6/route.c
+++ b/net/ipv6/route.c
@@ -1998,9 +1998,13 @@ errout:

 static int inet6_rtm_delroute(struct sk_buff *skb, struct nlmsghdr* nlh, void *arg)
 {
+ struct net *net = skb->sk->sk_net;
  struct fib6_config cfg;
  int err;

+ if (net != &init_net)
+  return -EINVAL;
+
  err = rtm_to_fib6_config(skb, nlh, &cfg);
  if (err < 0)
   return err;
@@ -2010,9 +2014,13 @@ static int inet6_rtm_delroute(struct sk_buff *skb, struct nlmsghdr* nlh,
void *a

 static int inet6_rtm_newroute(struct sk_buff *skb, struct nlmsghdr* nlh, void *arg)
 {
+ struct net *net = skb->sk->sk_net;
  struct fib6_config cfg;
  int err;

+ if (net != &init_net)
+  return -EINVAL;
+
  err = rtm_to_fib6_config(skb, nlh, &cfg);
  if (err < 0)
   return err;
```

```
@@ -2147,6 +2155,7 @@ int rt6_dump_route(struct rt6_info *rt, void *p_arg)

 static int inet6_rtm_getroute(struct sk_buff *in_skb, struct nlmsghdr* nlh, void *arg)
 {
+ struct net *net = in_skb->sk->sk_net;
  struct nlattr *tb[RTA_MAX+1];
  struct rt6_info *rt;
  struct sk_buff *skb;
@@ -2154,6 +2163,9 @@ static int inet6_rtm_getroute(struct sk_buff *in_skb, struct nlmsghdr*
nlh, void
  struct flowi fl;
  int err, iif = 0;

+ if (net != &init_net)
+  return -EINVAL;
+
  err = nlmsg_parse(nlh, sizeof(*rtm), tb, RTA_MAX, rtm_ipv6_policy);
  if (err < 0)
   goto errout;
diff --git a/net/sched/act_api.c b/net/sched/act_api.c
index 72cdb0f..8528291 100644
--- a/net/sched/act_api.c
+++ b/net/sched/act_api.c
@@ -18,6 +18,8 @@
 #include <linux/skbuff.h>
 #include <linux/init.h>
 #include <linux/kmod.h>
+#include <net/net_namespace.h>
+#include <net/sock.h>
 #include <net/sch_generic.h>
 #include <net/act_api.h>
 #include <net/netlink.h>
@@ -924,10 +926,14 @@ done:

 static int tc_ctl_action(struct sk_buff *skb, struct nlmsghdr *n, void *arg)
 {
+ struct net *net = skb->sk->sk_net;
  struct rtattr **tca = arg;
  u32 pid = skb ? NETLINK_CB(skb).pid : 0;
  int ret = 0, ovr = 0;

+ if (net != &init_net)
+  return -EINVAL;
+
  if (tca[TCA_ACT_TAB-1] == NULL) {
   printk("tc_ctl_action: received NO action attribs\n");
   return -EINVAL;
@@ -997,6 +1003,7 @@ find_dump_kind(struct nlmsghdr *n)
```

```
 static int
 tc_dump_action(struct sk_buff *skb, struct netlink_callback *cb)
 {
+ struct net *net = skb->sk->sk_net;
  struct nlmsghdr *nlh;
  unsigned char *b = skb_tail_pointer(skb);
  struct rtattr *x;
@@ -1006,6 +1013,9 @@ tc_dump_action(struct sk_buff *skb, struct netlink_callback *cb)
  struct tcamsg *t = (struct tcamsg *) NLMSG_DATA(cb->nlh);
  struct rtattr *kind = find_dump_kind(cb->nlh);

+ if (net != &init_net)
+  return 0;
+
  if (kind == NULL) {
   printk("tc_dump_action: action bad kind\n");
   return 0;
diff --git a/net/sched/cls_api.c b/net/sched/cls_api.c
index bb98045..fdab6a5 100644
--- a/net/sched/cls_api.c
+++ b/net/sched/cls_api.c
@@ -23,6 +23,8 @@
 #include <linux/init.h>
 #include <linux/kmod.h>
 #include <linux/netlink.h>
+#include <net/net_namespace.h>
+#include <net/sock.h>
 #include <net/netlink.h>
 #include <net/pkt_sched.h>
 #include <net/pkt_cls.h>
@@ -119,6 +121,7 @@ static __inline__ u32 tcf_auto_prio(struct tcf_proto *tp)

 static int tc_ctl_tfilter(struct sk_buff *skb, struct nlmsghdr *n, void *arg)
 {
+ struct net *net = skb->sk->sk_net;
  struct rtattr **tca;
  struct tcmsg *t;
  u32 protocol;
@@ -135,6 +138,9 @@ static int tc_ctl_tfilter(struct sk_buff *skb, struct nlmsghdr *n, void *arg)
  unsigned long fh;
  int err;

+ if (net != &init_net)
+  return -EINVAL;
+
 replay:
  tca = arg;
  t = NLMSG_DATA(n);
```

```
@@ -375,6 +381,7 @@ static int tcf_node_dump(struct tcf_proto *tp, unsigned long n, struct
tcf_walke

 static int tc_dump_tfilter(struct sk_buff *skb, struct netlink_callback *cb)
 {
+ struct net *net = skb->sk->sk_net;
  int t;
  int s_t;
  struct net_device *dev;
@@ -385,6 +392,9 @@ static int tc_dump_tfilter(struct sk_buff *skb, struct netlink_callback *cb)
  const struct Qdisc_class_ops *cops;
  struct tcf_dump_args arg;

+ if (net != &init_net)
+  return 0;
+
  if (cb->nlh->nlmsg_len < NLMSG_LENGTH(sizeof(*tcm)))
   return skb->len;
  if ((dev = dev_get_by_index(&init_net, tcm->tcm_ifindex)) == NULL)
diff --git a/net/sched/sch_api.c b/net/sched/sch_api.c
index 259321b..f30e3f7 100644
--- a/net/sched/sch_api.c
+++ b/net/sched/sch_api.c
@@ -29,6 +29,7 @@
 #include <linux/hrtimer.h>

 #include <net/net_namespace.h>
+#include <net/sock.h>
 #include <net/netlink.h>
 #include <net/pkt_sched.h>

@@ -599,6 +600,7 @@ check_loop_fn(struct Qdisc *q, unsigned long cl, struct qdisc_walker *w)

 static int tc_get_qdisc(struct sk_buff *skb, struct nlmsghdr *n, void *arg)
 {
+ struct net *net = skb->sk->sk_net;
  struct tcmsg *tcm = NLMSG_DATA(n);
  struct rtattr **tca = arg;
  struct net_device *dev;
@@ -607,6 +609,9 @@ static int tc_get_qdisc(struct sk_buff *skb, struct nlmsghdr *n, void *arg)
  struct Qdisc *p = NULL;
  int err;

+ if (net != &init_net)
+  return -EINVAL;
+
  if ((dev = __dev_get_by_index(&init_net, tcm->tcm_ifindex)) == NULL)
   return -ENODEV;
```

```
@@ -660,6 +665,7 @@ static int tc_get_qdisc(struct sk_buff *skb, struct nlmsghdr *n, void *arg)

 static int tc_modify_qdisc(struct sk_buff *skb, struct nlmsghdr *n, void *arg)
 {
+ struct net *net = skb->sk->sk_net;
  struct tcmsg *tcm;
  struct rtattr **tca;
  struct net_device *dev;
@@ -667,6 +673,9 @@ static int tc_modify_qdisc(struct sk_buff *skb, struct nlmsghdr *n, void
*arg)
  struct Qdisc *q, *p;
  int err;

+ if (net != &init_net)
+  return -EINVAL;
+
 replay:
  /* Reinit, just in case something touches this. */
  tcm = NLMSG_DATA(n);
@@ -872,11 +881,15 @@ err_out:

 static int tc_dump_qdisc(struct sk_buff *skb, struct netlink_callback *cb)
 {
+ struct net *net = skb->sk->sk_net;
  int idx, q_idx;
  int s_idx, s_q_idx;
  struct net_device *dev;
  struct Qdisc *q;

+ if (net != &init_net)
+  return 0;
+
  s_idx = cb->args[0];
  s_q_idx = q_idx = cb->args[1];
  read_lock(&dev_base_lock);
@@ -920,6 +933,7 @@ done:

 static int tc_ctl_tclass(struct sk_buff *skb, struct nlmsghdr *n, void *arg)
 {
+ struct net *net = skb->sk->sk_net;
  struct tcmsg *tcm = NLMSG_DATA(n);
  struct rtattr **tca = arg;
  struct net_device *dev;
@@ -932,6 +946,9 @@ static int tc_ctl_tclass(struct sk_buff *skb, struct nlmsghdr *n, void *arg)
  u32 qid = TC_H_MAJ(clid);
  int err;
```

```
+ if (net != &init_net)
+  return -EINVAL;
+
  if ((dev = __dev_get_by_index(&init_net, tcm->tcm_ifindex)) == NULL)
   return -ENODEV;

@@ -1106,6 +1123,7 @@ static int qdisc_class_dump(struct Qdisc *q, unsigned long cl, struct
qdisc_walk

 static int tc_dump_tclass(struct sk_buff *skb, struct netlink_callback *cb)
 {
+ struct net *net = skb->sk->sk_net;
  int t;
  int s_t;
  struct net_device *dev;
@@ -1113,6 +1131,9 @@ static int tc_dump_tclass(struct sk_buff *skb, struct netlink_callback
*cb)
  struct tcmsg *tcm = (struct tcmsg*)NLMSG_DATA(cb->nlh);
  struct qdisc_dump_args arg;

+ if (net != &init_net)
+  return 0;
+
  if (cb->nlh->nlmsg_len < NLMSG_LENGTH(sizeof(*tcm)))
   return 0;
  if ((dev = dev_get_by_index(&init_net, tcm->tcm_ifindex)) == NULL)
--
1.5.3.rc5
```

Subject: Re: [PATCH 2.6.25 1/6] net: Modify all rtnetlink methods to only work in the initial namespace (v2)

Posted by davem on Tue, 20 Nov 2007 06:33:32 GMT

View Forum Message <> Reply to Message

From: "Denis V. Lunev" <den@openvz.org>
Date: Thu, 15 Nov 2007 18:56:48 +0300

> Before I can enable rtnetlink to work in all network namespaces
> I need to be certain that something won't break. So this
> patch deliberately disables all of the rtnletlink methods in everything
> except the initial network namespace. After the methods have been
> audited this extra check can be disabled.

>
> Changes from v1:
> - added IPv6 addrlabel protection
>
> Signed-off-by: Denis V. Lunev <den@openvz.org>
> Signed-off-by: Eric W. Biederman <ebiederm@xmission.com>

Applied, thanks Denis.

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