Subject: [PATCH 0/17] Finish IPv4 infrastructure namespacing.
Posted by den on Tue, 19 Feb 2008 07:28:36 GMT
View Forum Message <> Reply to Message

This set finally allows to manipulate with network devices inside a
namespace and allows to configure them [via netlink]. 'route' is not yet
supported (but prepared to) as it requires a socket.

Additionally, better routing cache support is added.

Signed-off-by: Denis V. Lunev <den@openvz.org>

Subject: [PATCH 1/17 net-2.6.26] [IPV4]: Remove ifa != NULL check.
Posted by den on Tue, 19 Feb 2008 07:31:21 GMT
View Forum Message <> Reply to Message

This is a callback registered to inet address notifier chain.
The check is useless as:
- ifa is always != NULL
- similar checks are abscent in all other notifiers.

Signed-off-by: Denis V. Lunev <den@openvz.org>
---
 drivers/net/via-velocity.c | 22 ++++++++++++----------
 1 files changed, 10 insertions(+), 12 deletions(-)

diff --git a/drivers/net/via-velocity.c b/drivers/net/via-velocity.c
index cc0addb..c50fdee 100644
--- a/drivers/net/via-velocity.c
+++ b/drivers/net/via-velocity.c
@@ -3460,21 +3460,19 @@ static int velocity_resume(struct pci_dev *pdev)
 static int velocity_netdev_event(struct notifier_block *nb, unsigned long notification, void *ptr)
 {
  struct in_ifaddr *ifa = (struct in_ifaddr *) ptr;
+ struct net_device *dev = ifa->ifa_dev->dev;
+ struct velocity_info *vptr;
+ unsigned long flags;

- if (ifa) {
- struct net_device *dev = ifa->ifa_dev->dev;
- struct velocity_info *vptr;
- unsigned long flags;
-
- spin_lock_irqsave(&velocity_dev_list_lock, flags);
- list_for_each_entry(vptr, &velocity_dev_list, list) {
-   if (vptr->dev == dev) {
-    velocity_get_ip(vptr);

```
-    break;
-   }
+ spin_lock_irqsave(&velocity_dev_list_lock, flags);
+ list_for_each_entry(vptr, &velocity_dev_list, list) {
+  if (vptr->dev == dev) {
+   velocity_get_ip(vptr);
+   break;
    }
-  spin_unlock_irqrestore(&velocity_dev_list_lock, flags);
   }
+ spin_unlock_irqrestore(&velocity_dev_list_lock, flags);
+
   return NOTIFY_DONE;
  }

--
1.5.3.rc5
```

Subject: [PATCH 2/17 net-2.6.26] [IPV4]: Remove check for ifa->ifa_dev != NULL.
Posted by den on Tue, 19 Feb 2008 07:31:22 GMT

View Forum Message <> Reply to Message

This is a callback registered to inet address notifier chain.
The check is useless as:
- ifa->ifa_dev is always != NULL
- similar checks are abscent in all other notifiers.

Signed-off-by: Denis V. Lunev <den@openvz.org>

```
---
 net/atm/clip.c |    4 ----
 1 files changed, 0 insertions(+), 4 deletions(-)

diff --git a/net/atm/clip.c b/net/atm/clip.c
index 86b885e..dd96440 100644
--- a/net/atm/clip.c
+++ b/net/atm/clip.c
@@ -648,10 +648,6 @@ static int clip_inet_event(struct notifier_block *this, unsigned long event,
   struct in_device *in_dev;

   in_dev = ((struct in_ifaddr *)ifa)->ifa_dev;
- if (!in_dev || !in_dev->dev) {
-  printk(KERN_WARNING "clip_inet_event: no device\n");
-  return NOTIFY_DONE;
- }
   /*
    * Transitions are of the down-change-up type, so it's sufficient to
    * handle the change on up.
```

--
1.5.3.rc5

Subject: [PATCH 3/17 net-2.6.26] [NETFILTER]: Consolidate masq_inet_event and masq_device_event.
Posted by den on Tue, 19 Feb 2008 07:31:23 GMT
View Forum Message <> Reply to Message

They do exactly the same job.

Signed-off-by: Denis V. Lunev <den@openvz.org>
---
 net/ipv4/netfilter/ipt_MASQUERADE.c | 14 ++----------
 1 files changed, 2 insertions(+), 12 deletions(-)

diff --git a/net/ipv4/netfilter/ipt_MASQUERADE.c b/net/ipv4/netfilter/ipt_MASQUERADE.c
index d80fee8..313b3fc 100644
--- a/net/ipv4/netfilter/ipt_MASQUERADE.c
+++ b/net/ipv4/netfilter/ipt_MASQUERADE.c
@@ -139,18 +139,8 @@ static int masq_inet_event(struct notifier_block *this,
       unsigned long event,
       void *ptr)
 {
- const struct net_device *dev = ((struct in_ifaddr *)ptr)->ifa_dev->dev;
-
- if (event == NETDEV_DOWN) {
- /* IP address was deleted. Search entire table for
-     conntracks which were associated with that device,
-     and forget them. */
- NF_CT_ASSERT(dev->ifindex != 0);
-
- nf_ct_iterate_cleanup(device_cmp, (void *)(long)dev->ifindex);
- }
-
- return NOTIFY_DONE;
+ struct net_device *dev = ((struct in_ifaddr *)ptr)->ifa_dev->dev;
+ return masq_device_event(this, event, dev);
 }

 static struct notifier_block masq_dev_notifier = {
--
1.5.3.rc5

Subject: [PATCH 4/17 net-2.6.26] [NETNS]: Disable inetaddr notifiers in namespaces other than initial.

ip_fib_init is kept enabled. It is already namespace-aware.

Signed-off-by: Denis V. Lunev <den@openvz.org>
---
 drivers/net/bonding/bond_main.c | 3 +++
 drivers/net/via-velocity.c | 3 +++
 drivers/s390/net/qeth_main.c | 3 +++
 net/sctp/protocol.c | 3 +++
 4 files changed, 12 insertions(+), 0 deletions(-)

```
diff --git a/drivers/net/bonding/bond_main.c b/drivers/net/bonding/bond_main.c
index 0942d82..9666434 100644
--- a/drivers/net/bonding/bond_main.c
+++ b/drivers/net/bonding/bond_main.c
@@ -3511,6 +3511,9 @@ static int bond_inetaddr_event(struct notifier_block *this, unsigned
long event,
  struct bonding *bond, *bond_next;
  struct vlan_entry *vlan, *vlan_next;

+ if (ifa->ifa_dev->dev->nd_net != &init_net)
+  return NOTIFY_DONE;
+
  list_for_each_entry_safe(bond, bond_next, &bond_dev_list, bond_list) {
   if (bond->dev == event_dev) {
    switch (event) {
diff --git a/drivers/net/via-velocity.c b/drivers/net/via-velocity.c
index c50fdee..1525e8a 100644
--- a/drivers/net/via-velocity.c
+++ b/drivers/net/via-velocity.c
@@ -3464,6 +3464,9 @@ static int velocity_netdev_event(struct notifier_block *nb, unsigned
long notifi
  struct velocity_info *vptr;
  unsigned long flags;

+ if (dev->nd_net != &init_net)
+  return NOTIFY_DONE;
+
  spin_lock_irqsave(&velocity_dev_list_lock, flags);
  list_for_each_entry(vptr, &velocity_dev_list, list) {
   if (vptr->dev == dev) {
diff --git a/drivers/s390/net/qeth_main.c b/drivers/s390/net/qeth_main.c
index 62606ce..d063e9e 100644
--- a/drivers/s390/net/qeth_main.c
+++ b/drivers/s390/net/qeth_main.c
@@ -8622,6 +8622,9 @@ qeth_ip_event(struct notifier_block *this,
  struct qeth_ipaddr *addr;
```

```
  struct qeth_card *card;

+ if (dev->nd_net != &init_net)
+  return NOTIFY_DONE;
+
  QETH_DBF_TEXT(trace,3,"ipevent");
  card = qeth_get_card_from_dev(dev);
  if (!card)
diff --git a/net/sctp/protocol.c b/net/sctp/protocol.c
index 22a1657..4475f7e 100644
--- a/net/sctp/protocol.c
+++ b/net/sctp/protocol.c
@@ -629,6 +629,9 @@ static int sctp_inetaddr_event(struct notifier_block *this, unsigned long
ev,
  struct sctp_sockaddr_entry *addr = NULL;
  struct sctp_sockaddr_entry *temp;

+ if (ifa->ifa_dev->dev->nd_net != &init_net)
+  return NOTIFY_DONE;
+
  switch (ev) {
  case NETDEV_UP:
   addr = kmalloc(sizeof(struct sctp_sockaddr_entry), GFP_ATOMIC);
--
1.5.3.rc5
```

Subject: [PATCH 5/17 net-2.6.26] [NETNS]: Register neighbour table parameters in the correct namespace.

Posted by den on Tue, 19 Feb 2008 07:31:25 GMT

View Forum Message <> Reply to Message

neigh_sysctl_register should register sysctl entries inside correct namespace
to avoid naming conflict. Typical example is a loopback. Entries for it
present in all namespaces.

Required to make inetdev_event working.

Signed-off-by: Denis V. Lunev <den@openvz.org>

```
---
 net/core/neighbour.c |    3 ++-
 1 files changed, 2 insertions(+), 1 deletions(-)

diff --git a/net/core/neighbour.c b/net/core/neighbour.c
index 7bb6a9a..c895ad4 100644
--- a/net/core/neighbour.c
+++ b/net/core/neighbour.c
@@ -2732,7 +2732,8 @@ int neigh_sysctl_register(struct net_device *dev, struct neigh_parms
```

```
*p,
  neigh_path[NEIGH_CTL_PATH_PROTO].procname = p_name;
  neigh_path[NEIGH_CTL_PATH_PROTO].ctl_name = p_id;

- t->sysctl_header = register_sysctl_paths(neigh_path, t->neigh_vars);
+ t->sysctl_header =
+  register_net_sysctl_table(p->net, neigh_path, t->neigh_vars);
  if (!t->sysctl_header)
   goto free_procname;

--
1.5.3.rc5
```

Subject: [PATCH 6/17 net-2.6.26] [NETNS]: Default arp parameters lookup.

Posted by den on Tue, 19 Feb 2008 07:31:26 GMT

View Forum Message <> Reply to Message

Default ARP parameters should be findable regardless of the context.
Required to make inetdev_event working.

Signed-off-by: Denis V. Lunev <den@openvz.org>

```
---
 net/core/neighbour.c |    4 +---
 1 files changed, 1 insertions(+), 3 deletions(-)

diff --git a/net/core/neighbour.c b/net/core/neighbour.c
index c895ad4..45ed620 100644
--- a/net/core/neighbour.c
+++ b/net/core/neighbour.c
@@ -1275,9 +1275,7 @@ static inline struct neigh_parms *lookup_neigh_params(struct
neigh_table *tbl,
  struct neigh_parms *p;

  for (p = &tbl->parms; p; p = p->next) {
-  if (p->net != net)
-   continue;
-  if ((p->dev && p->dev->ifindex == ifindex) ||
+  if ((p->dev && p->dev->ifindex == ifindex && p->net == net) ||
       (!p->dev && !ifindex))
    return p;
  }
--
1.5.3.rc5
```

Subject: [PATCH 7/17 net-2.6.26] [NETNS]: Disable multicaststing configuration

inside non-initial namespace.
Posted by den on Tue, 19 Feb 2008 07:31:27 GMT
View Forum Message <> Reply to Message

Do not calls hooks from device notifiers and disallow configuration from
ioctl/netlink layer.

Signed-off-by: Denis V. Lunev <den@openvz.org>
---
 net/ipv4/igmp.c | 39 +++++++++++++++++++++++++++++++++++++++
 1 files changed, 39 insertions(+), 0 deletions(-)

diff --git a/net/ipv4/igmp.c b/net/ipv4/igmp.c
index 732cd07..d3f34a7 100644
--- a/net/ipv4/igmp.c
+++ b/net/ipv4/igmp.c
@@ -1198,6 +1198,9 @@ void ip_mc_inc_group(struct in_device *in_dev, __be32 addr)

  ASSERT_RTNL();

+ if (in_dev->dev->nd_net != &init_net)
+ return;
+
  for (im=in_dev->mc_list; im; im=im->next) {
   if (im->multiaddr == addr) {
    im->users++;
@@ -1277,6 +1280,9 @@ void ip_mc_dec_group(struct in_device *in_dev, __be32 addr)

  ASSERT_RTNL();

+ if (in_dev->dev->nd_net != &init_net)
+ return;
+
  for (ip=&in_dev->mc_list; (i=*ip)!=NULL; ip=&i->next) {
   if (i->multiaddr==addr) {
    if (--i->users == 0) {
@@ -1304,6 +1310,9 @@ void ip_mc_down(struct in_device *in_dev)

  ASSERT_RTNL();

+ if (in_dev->dev->nd_net != &init_net)
+ return;
+
  for (i=in_dev->mc_list; i; i=i->next)
   igmp_group_dropped(i);

@@ -1324,6 +1333,9 @@ void ip_mc_init_dev(struct in_device *in_dev)
 {
  ASSERT_RTNL();

```
+ if (in_dev->dev->nd_net != &init_net)
+  return;
+
  in_dev->mc_tomb = NULL;
 #ifdef CONFIG_IP_MULTICAST
  in_dev->mr_gq_running = 0;
@@ -1347,6 +1359,9 @@ void ip_mc_up(struct in_device *in_dev)

  ASSERT_RTNL();

+ if (in_dev->dev->nd_net != &init_net)
+  return;
+
  ip_mc_inc_group(in_dev, IGMP_ALL_HOSTS);

  for (i=in_dev->mc_list; i; i=i->next)
@@ -1363,6 +1378,9 @@ void ip_mc_destroy_dev(struct in_device *in_dev)

  ASSERT_RTNL();

+ if (in_dev->dev->nd_net != &init_net)
+  return;
+
  /* Deactivate timers */
  ip_mc_down(in_dev);

@@ -1744,6 +1762,9 @@ int ip_mc_join_group(struct sock *sk , struct ip_mreqn *imr)
  if (!ipv4_is_multicast(addr))
   return -EINVAL;

+ if (sk->sk_net != &init_net)
+  return -EPROTONOSUPPORT;
+
  rtnl_lock();

  in_dev = ip_mc_find_dev(imr);
@@ -1812,6 +1833,9 @@ int ip_mc_leave_group(struct sock *sk, struct ip_mreqn *imr)
  u32 ifindex;
  int ret = -EADDRNOTAVAIL;

+ if (sk->sk_net != &init_net)
+  return -EPROTONOSUPPORT;
+
  rtnl_lock();
  in_dev = ip_mc_find_dev(imr);
  ifindex = imr->imr_ifindex;
@@ -1857,6 +1881,9 @@ int ip_mc_source(int add, int omode, struct sock *sk, struct
```

```
  if (!ipv4_is_multicast(addr))
   return -EINVAL;

+ if (sk->sk_net != &init_net)
+  return -EPROTONOSUPPORT;
+
  rtnl_lock();

  imr.imr_multiaddr.s_addr = mreqs->imr_multiaddr;
@@ -1990,6 +2017,9 @@ int ip_mc_msfilter(struct sock *sk, struct ip_msfilter *msf, int ifindex)
      msf->imsf_fmode != MCAST_EXCLUDE)
   return -EINVAL;

+ if (sk->sk_net != &init_net)
+  return -EPROTONOSUPPORT;
+
  rtnl_lock();

  imr.imr_multiaddr.s_addr = msf->imsf_multiaddr;
@@ -2070,6 +2100,9 @@ int ip_mc_msfget(struct sock *sk, struct ip_msfilter *msf,
  if (!ipv4_is_multicast(addr))
   return -EINVAL;

+ if (sk->sk_net != &init_net)
+  return -EPROTONOSUPPORT;
+
  rtnl_lock();

  imr.imr_multiaddr.s_addr = msf->imsf_multiaddr;
@@ -2132,6 +2165,9 @@ int ip_mc_gsfget(struct sock *sk, struct group_filter *gsf,
  if (!ipv4_is_multicast(addr))
   return -EINVAL;

+ if (sk->sk_net != &init_net)
+  return -EPROTONOSUPPORT;
+
  rtnl_lock();

  err = -EADDRNOTAVAIL;
@@ -2216,6 +2252,9 @@ void ip_mc_drop_socket(struct sock *sk)
  if (inet->mc_list == NULL)
   return;

+ if (sk->sk_net != &init_net)
+  return;
+
  rtnl_lock();
  while ((iml = inet->mc_list) != NULL) {
```

```
   struct in_device *in_dev;
--
1.5.3.rc5
```

Subject: [PATCH 8/17 net-2.6.26] [NETNS]: Enable inetdev_event notifier.
Posted by den on Tue, 19 Feb 2008 07:31:28 GMT
View Forum Message <> Reply to Message

After all these preparations it is time to enable main IPv4 device
initialization routine inside namespace. It is safe do this now.

Signed-off-by: Denis V. Lunev <den@openvz.org>

```
---
 net/ipv4/devinet.c |    3 ---
 1 files changed, 0 insertions(+), 3 deletions(-)

diff --git a/net/ipv4/devinet.c b/net/ipv4/devinet.c
index f282b26..963e711 100644
--- a/net/ipv4/devinet.c
+++ b/net/ipv4/devinet.c
@@ -1044,9 +1044,6 @@ static int inetdev_event(struct notifier_block *this, unsigned long event,
  struct net_device *dev = ptr;
  struct in_device *in_dev = __in_dev_get_rtnl(dev);

- if (dev->nd_net != &init_net)
-  return NOTIFY_DONE;
-
  ASSERT_RTNL();

  if (!in_dev) {
--
1.5.3.rc5
```

Subject: [PATCH 9/17 net-2.6.26] [NETNS]: DST cleanup routines should be called inside namespace.
Posted by den on Tue, 19 Feb 2008 07:31:29 GMT
View Forum Message <> Reply to Message

Device inside the namespace can be started and downed. So, active routing
cache should be cleaned up on device stop.

Signed-off-by: Denis V. Lunev <den@openvz.org>

```
---
 net/core/dst.c |    3 ---
 1 files changed, 0 insertions(+), 3 deletions(-)
```

```
diff --git a/net/core/dst.c b/net/core/dst.c
index 7deef48..3a01a81 100644
--- a/net/core/dst.c
+++ b/net/core/dst.c
@@ -295,9 +295,6 @@ static int dst_dev_event(struct notifier_block *this, unsigned long event,
void
  struct net_device *dev = ptr;
  struct dst_entry *dst, *last = NULL;

- if (dev->nd_net != &init_net)
-  return NOTIFY_DONE;
-
  switch (event) {
  case NETDEV_UNREGISTER:
  case NETDEV_DOWN:
--
1.5.3.rc5
```

Subject: [PATCH 10/17 net-2.6.26] [NETNS]: Process ip_rt_redirect in the correct namespace.
Posted by den on Tue, 19 Feb 2008 07:31:30 GMT
View Forum Message <> Reply to Message

```
Signed-off-by: Denis V. Lunev <den@openvz.org>
---
 net/ipv4/route.c |    7 +++++--
 1 files changed, 5 insertions(+), 2 deletions(-)

diff --git a/net/ipv4/route.c b/net/ipv4/route.c
index 525787b..44708ab 100644
--- a/net/ipv4/route.c
+++ b/net/ipv4/route.c
@@ -1132,10 +1132,12 @@ void ip_rt_redirect(__be32 old_gw, __be32 daddr, __be32 new_gw,
  __be32  skeys[2] = { saddr, 0 };
  int  ikeys[2] = { dev->ifindex, 0 };
  struct netevent_redirect netevent;
+ struct net *net;

  if (!in_dev)
   return;

+ net = dev->nd_net;
  if (new_gw == old_gw || !IN_DEV_RX_REDIRECTS(in_dev)
      || ipv4_is_multicast(new_gw) || ipv4_is_lbcast(new_gw)
      || ipv4_is_zeronet(new_gw))
@@ -1147,7 +1149,7 @@ void ip_rt_redirect(__be32 old_gw, __be32 daddr, __be32 new_gw,
```

```
   if (IN_DEV_SEC_REDIRECTS(in_dev) && ip_fib_check_default(new_gw, dev))
    goto reject_redirect;
  } else {
-  if (inet_addr_type(&init_net, new_gw) != RTN_UNICAST)
+  if (inet_addr_type(net, new_gw) != RTN_UNICAST)
    goto reject_redirect;
  }

@@ -1165,7 +1167,8 @@ void ip_rt_redirect(__be32 old_gw, __be32 daddr, __be32 new_gw,
         rth->fl.fl4_src != skeys[i] ||
         rth->fl.oif != ikeys[k] ||
         rth->fl.iif != 0 ||
-        rth->rt_genid != atomic_read(&rt_genid)) {
+        rth->rt_genid != atomic_read(&rt_genid) ||
+        rth->u.dst.dev->nd_net != net) {
      rthp = &rth->u.dst.rt_next;
      continue;
     }
--
1.5.3.rc5
```

Subject: [PATCH 12/17 net-2.6.26] [NETNS]: Process /proc/net/rt_cache inside a namespace.

Posted by den on Tue, 19 Feb 2008 07:31:31 GMT

View Forum Message <> Reply to Message

Show routing cache for a particular namespace only.

Signed-off-by: Denis V. Lunev <den@openvz.org>

```
---
 net/ipv4/route.c |   10 +++++++---
 1 files changed, 7 insertions(+), 3 deletions(-)

diff --git a/net/ipv4/route.c b/net/ipv4/route.c
index 67df872..c11e6bf 100644
--- a/net/ipv4/route.c
+++ b/net/ipv4/route.c
@@ -273,6 +273,7 @@ static unsigned int rt_hash_code(u32 daddr, u32 saddr)

 #ifdef CONFIG_PROC_FS
 struct rt_cache_iter_state {
+ struct seq_net_private p;
  int bucket;
  int genid;
 };
@@ -285,7 +286,8 @@ static struct rtable *rt_cache_get_first(struct rt_cache_iter_state *st)
   rcu_read_lock_bh();
```

```
   r = rcu_dereference(rt_hash_table[st->bucket].chain);
   while (r) {
-   if (r->rt_genid == st->genid)
+   if (r->u.dst.dev->nd_net == st->p.net &&
+       r->rt_genid == st->genid)
     return r;
    r = rcu_dereference(r->u.dst.rt_next);
   }
@@ -312,6 +314,8 @@ static struct rtable *rt_cache_get_next(struct rt_cache_iter_state *st,
      struct rtable *r)
 {
  while ((r = __rt_cache_get_next(st, r)) != NULL) {
+  if (r->u.dst.dev->nd_net != st->p.net)
+   continue;
   if (r->rt_genid == st->genid)
    break;
  }
@@ -398,7 +402,7 @@ static const struct seq_operations rt_cache_seq_ops = {

 static int rt_cache_seq_open(struct inode *inode, struct file *file)
 {
- return seq_open_private(file, &rt_cache_seq_ops,
+ return seq_open_net(inode, file, &rt_cache_seq_ops,
    sizeof(struct rt_cache_iter_state));
 }

@@ -407,7 +411,7 @@ static const struct file_operations rt_cache_seq_fops = {
  .open  = rt_cache_seq_open,
  .read  = seq_read,
  .llseek  = seq_lseek,
- .release = seq_release_private,
+ .release = seq_release_net,
 };


--
1.5.3.rc5
```

Subject: [PATCH 11/17 net-2.6.26] [IPV4]: rt_cache_get_next should take rt_genid into account.

Posted by den on Tue, 19 Feb 2008 07:31:31 GMT

View Forum Message <> Reply to Message

In the other case /proc/net/rt_cache will look inconsistent in respect to genid.

Signed-off-by: Denis V. Lunev <den@openvz.org>

```
Acked-by: Alexey Kuznetsov <kuznet@ms2.inr.ac.ru>
---
 net/ipv4/route.c |   18 ++++++++++++-----
 1 files changed, 13 insertions(+), 5 deletions(-)

diff --git a/net/ipv4/route.c b/net/ipv4/route.c
index 44708ab..67df872 100644
--- a/net/ipv4/route.c
+++ b/net/ipv4/route.c
@@ -294,7 +294,8 @@ static struct rtable *rt_cache_get_first(struct rt_cache_iter_state *st)
  return r;
 }

-static struct rtable *rt_cache_get_next(struct rt_cache_iter_state *st, struct rtable *r)
+static struct rtable *__rt_cache_get_next(struct rt_cache_iter_state *st,
+       struct rtable *r)
 {
  r = r->u.dst.rt_next;
  while (!r) {
@@ -307,16 +308,23 @@ static struct rtable *rt_cache_get_next(struct rt_cache_iter_state *st,
struct r
  return rcu_dereference(r);
 }

+static struct rtable *rt_cache_get_next(struct rt_cache_iter_state *st,
+     struct rtable *r)
+{
+ while ((r = __rt_cache_get_next(st, r)) != NULL) {
+  if (r->rt_genid == st->genid)
+   break;
+ }
+ return r;
+}
+
 static struct rtable *rt_cache_get_idx(struct rt_cache_iter_state *st, loff_t pos)
 {
  struct rtable *r = rt_cache_get_first(st);

  if (r)
-  while (pos && (r = rt_cache_get_next(st, r))) {
-   if (r->rt_genid != st->genid)
-    continue;
+  while (pos && (r = rt_cache_get_next(st, r)))
    --pos;
-  }
  return pos ? NULL : r;
 }
```

--
1.5.3.rc5

Subject: [PATCH 13/17 net-2.6.26] [NETNS]: Register /proc/net/rt_cache for each namespace.
Posted by den on Tue, 19 Feb 2008 07:31:33 GMT
View Forum Message <> Reply to Message

Signed-off-by: Denis V. Lunev <den@openvz.org>
---
 net/ipv4/route.c | 24 ++++++++++++++++++++----
 1 files changed, 21 insertions(+), 3 deletions(-)

diff --git a/net/ipv4/route.c b/net/ipv4/route.c
index c11e6bf..5f67eba 100644
--- a/net/ipv4/route.c
+++ b/net/ipv4/route.c
@@ -545,7 +545,7 @@ static int ip_rt_acct_read(char *buffer, char **start, off_t offset,
 }
 #endif

-static __init int ip_rt_proc_init(struct net *net)
+static int __net_init ip_rt_do_proc_init(struct net *net)
 {
  struct proc_dir_entry *pde;

@@ -577,8 +577,26 @@ err2:
 err1:
  return -ENOMEM;
 }
+
+static void __net_exit ip_rt_do_proc_exit(struct net *net)
+{
+ remove_proc_entry("rt_cache", net->proc_net_stat);
+ remove_proc_entry("rt_cache", net->proc_net);
+ remove_proc_entry("rt_acct", net->proc_net);
+}
+
+static struct pernet_operations ip_rt_proc_ops __net_initdata = {
+ .init = ip_rt_do_proc_init,
+ .exit = ip_rt_do_proc_exit,
+};
+
+static int __init ip_rt_proc_init(void)
+{
+ return register_pernet_subsys(&ip_rt_proc_ops);
+}

```
+
 #else
-static inline int ip_rt_proc_init(struct net *net)
+static inline int ip_rt_proc_init(void)
 {
  return 0;
 }
@@ -3056,7 +3074,7 @@ int __init ip_rt_init(void)
    ip_rt_secret_interval;
   add_timer(&rt_secret_timer);

- if (ip_rt_proc_init(&init_net))
+ if (ip_rt_proc_init())
    printk(KERN_ERR "Unable to create route proc files\n");
 #ifdef CONFIG_XFRM
   xfrm_init();
--
1.5.3.rc5
```

Subject: [PATCH 14/17 net-2.6.26] [NETNS]: Process devinet ioctl in the correct namespace.

Posted by den on Tue, 19 Feb 2008 07:31:34 GMT

View Forum Message <> Reply to Message

Add namespace parameter to devinet_ioctl and locate device inside it for
state changes.

Signed-off-by: Denis V. Lunev <den@openvz.org>

```
---
 include/linux/inetdevice.h |    2 +-
 net/ipv4/af_inet.c         |    7 ++++---
 net/ipv4/devinet.c         |    6 +++---
 net/ipv4/ipconfig.c        |    2 +-
 4 files changed, 9 insertions(+), 8 deletions(-)

diff --git a/include/linux/inetdevice.h b/include/linux/inetdevice.h
index fc4e3db..da05ab4 100644
--- a/include/linux/inetdevice.h
+++ b/include/linux/inetdevice.h
@@ -129,7 +129,7 @@ extern int unregister_inetaddr_notifier(struct notifier_block *nb);

 extern struct net_device *ip_dev_find(struct net *net, __be32 addr);
 extern int  inet_addr_onlink(struct in_device *in_dev, __be32 a, __be32 b);
-extern int  devinet_ioctl(unsigned int cmd, void __user *);
+extern int  devinet_ioctl(struct net *net, unsigned int cmd, void __user *);
 extern void  devinet_init(void);
 extern struct in_device *inetdev_by_index(struct net *, int);
```

```
 extern __be32  inet_select_addr(const struct net_device *dev, __be32 dst, int scope);
diff --git a/net/ipv4/af_inet.c b/net/ipv4/af_inet.c
index 09ca529..c270080 100644
--- a/net/ipv4/af_inet.c
+++ b/net/ipv4/af_inet.c
@@ -784,6 +784,7 @@ int inet_ioctl(struct socket *sock, unsigned int cmd, unsigned long arg)
 {
  struct sock *sk = sock->sk;
  int err = 0;
+ struct net *net = sk->sk_net;

  switch (cmd) {
   case SIOCGSTAMP:
@@ -795,12 +796,12 @@ int inet_ioctl(struct socket *sock, unsigned int cmd, unsigned long arg)
   case SIOCADDRT:
   case SIOCDELRT:
   case SIOCRTMSG:
-   err = ip_rt_ioctl(sk->sk_net, cmd, (void __user *)arg);
+   err = ip_rt_ioctl(net, cmd, (void __user *)arg);
    break;
   case SIOCDARP:
   case SIOCGARP:
   case SIOCSARP:
-   err = arp_ioctl(sk->sk_net, cmd, (void __user *)arg);
+   err = arp_ioctl(net, cmd, (void __user *)arg);
    break;
   case SIOCGIFADDR:
   case SIOCSIFADDR:
@@ -813,7 +814,7 @@ int inet_ioctl(struct socket *sock, unsigned int cmd, unsigned long arg)
   case SIOCSIFPFLAGS:
   case SIOCGIFPFLAGS:
   case SIOCSIFFLAGS:
-   err = devinet_ioctl(cmd, (void __user *)arg);
+   err = devinet_ioctl(net, cmd, (void __user *)arg);
    break;
   default:
    if (sk->sk_prot->ioctl)
diff --git a/net/ipv4/devinet.c b/net/ipv4/devinet.c
index 963e711..f7e78b7 100644
--- a/net/ipv4/devinet.c
+++ b/net/ipv4/devinet.c
@@ -595,7 +595,7 @@ static __inline__ int inet_abc_len(__be32 addr)
 }


-int devinet_ioctl(unsigned int cmd, void __user *arg)
+int devinet_ioctl(struct net *net, unsigned int cmd, void __user *arg)
 {
```

```
  struct ifreq ifr;
  struct sockaddr_in sin_orig;
@@ -624,7 +624,7 @@ int devinet_ioctl(unsigned int cmd, void __user *arg)
    *colon = 0;

 #ifdef CONFIG_KMOD
- dev_load(&init_net, ifr.ifr_name);
+ dev_load(net, ifr.ifr_name);
 #endif

  switch (cmd) {
@@ -665,7 +665,7 @@ int devinet_ioctl(unsigned int cmd, void __user *arg)
  rtnl_lock();

  ret = -ENODEV;
- if ((dev = __dev_get_by_name(&init_net, ifr.ifr_name)) == NULL)
+ if ((dev = __dev_get_by_name(net, ifr.ifr_name)) == NULL)
    goto done;

  if (colon)
diff --git a/net/ipv4/ipconfig.c b/net/ipv4/ipconfig.c
index a52b585..009d78f 100644
--- a/net/ipv4/ipconfig.c
+++ b/net/ipv4/ipconfig.c
@@ -291,7 +291,7 @@ static int __init ic_dev_ioctl(unsigned int cmd, struct ifreq *arg)

  mm_segment_t oldfs = get_fs();
  set_fs(get_ds());
- res = devinet_ioctl(cmd, (struct ifreq __user *) arg);
+ res = devinet_ioctl(&init_net, cmd, (struct ifreq __user *) arg);
  set_fs(oldfs);
  return res;
 }
--
1.5.3.rc5
```

Subject: [PATCH 15/17 net-2.6.26] [NETNS]: Enable all routing manipulation via netlink inside namespace.

Posted by den on Tue, 19 Feb 2008 07:31:35 GMT

View Forum Message <> Reply to Message

```
Signed-off-by: Denis V. Lunev <den@openvz.org>
---
 net/ipv4/route.c |   16 ++++++++--------
 1 files changed, 8 insertions(+), 8 deletions(-)

diff --git a/net/ipv4/route.c b/net/ipv4/route.c
```

```
index 5f67eba..79e2e8a 100644
--- a/net/ipv4/route.c
+++ b/net/ipv4/route.c
@@ -2702,9 +2702,6 @@ 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;
@@ -2734,7 +2731,7 @@ static int inet_rtm_getroute(struct sk_buff *in_skb, struct nlmsghdr*
nlh, void
  if (iif) {
   struct net_device *dev;

-  dev = __dev_get_by_index(&init_net, iif);
+  dev = __dev_get_by_index(net, iif);
   if (dev == NULL) {
    err = -ENODEV;
    goto errout_free;
@@ -2760,7 +2757,7 @@ static int inet_rtm_getroute(struct sk_buff *in_skb, struct nlmsghdr*
nlh, void
    },
    .oif = tb[RTA_OIF] ? nla_get_u32(tb[RTA_OIF]) : 0,
   };
-  err = ip_route_output_key(&init_net, &rt, &fl);
+  err = ip_route_output_key(net, &rt, &fl);
  }

  if (err)
@@ -2771,11 +2768,11 @@ static int inet_rtm_getroute(struct sk_buff *in_skb, struct nlmsghdr*
nlh, void
   rt->rt_flags |= RTCF_NOTIFY;

  err = rt_fill_info(skb, NETLINK_CB(in_skb).pid, nlh->nlmsg_seq,
-    RTM_NEWROUTE, 0, 0);
+      RTM_NEWROUTE, 0, 0);
  if (err <= 0)
   goto errout_free;

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

```
@@ -2789,6 +2786,9 @@ int ip_rt_dump(struct sk_buff *skb,  struct netlink_callback *cb)
  struct rtable *rt;
  int h, s_h;
  int idx, s_idx;
+ struct net *net;
+
+ net = skb->sk->sk_net;

  s_h = cb->args[0];
  if (s_h < 0)
@@ -2798,7 +2798,7 @@ int ip_rt_dump(struct sk_buff *skb,  struct netlink_callback *cb)
   rcu_read_lock_bh();
   for (rt = rcu_dereference(rt_hash_table[h].chain), idx = 0; rt;
        rt = rcu_dereference(rt->u.dst.rt_next), idx++) {
-   if (idx < s_idx)
+   if (rt->u.dst.dev->nd_net != net || idx < s_idx)
     continue;
    if (rt->rt_genid != atomic_read(&rt_genid))
     continue;
--
1.5.3.rc5
```

Subject: [PATCH 16/17 net-2.6.26] [NETNS]: Enable IPv4 address manipulations inside namespace.
Posted by den on Tue, 19 Feb 2008 07:31:36 GMT
View Forum Message <> Reply to Message

```
Signed-off-by: Denis V. Lunev <den@openvz.org>
---
 net/ipv4/devinet.c |    9 ---------
 1 files changed, 0 insertions(+), 9 deletions(-)

diff --git a/net/ipv4/devinet.c b/net/ipv4/devinet.c
index f7e78b7..aa23d10 100644
--- a/net/ipv4/devinet.c
+++ b/net/ipv4/devinet.c
@@ -446,9 +446,6 @@ 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;
@@ -560,9 +557,6 @@ static int inet_rtm_newaddr(struct sk_buff *skb, struct nlmsghdr *nlh, void
*arg

  ASSERT_RTNL();

- if (net != &init_net)
-  return -EINVAL;
-
  ifa = rtm_to_ifaddr(net, nlh);
  if (IS_ERR(ifa))
   return PTR_ERR(ifa);
@@ -1169,9 +1163,6 @@ static int inet_dump_ifaddr(struct sk_buff *skb, struct netlink_callback
*cb)
  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(net, dev) {
--
1.5.3.rc5
```

Subject: [PATCH 17/17 net-2.6.26] [NETNS]: Process inet_select_addr inside a namespace.

Posted by den on Tue, 19 Feb 2008 07:31:37 GMT

View Forum Message <> Reply to Message

The context is available from a network device passed in.

Signed-off-by: Denis V. Lunev <den@openvz.org>

```
---
 net/ipv4/devinet.c |    3 ++-
 1 files changed, 2 insertions(+), 1 deletions(-)

diff --git a/net/ipv4/devinet.c b/net/ipv4/devinet.c
index aa23d10..033670d 100644
--- a/net/ipv4/devinet.c
+++ b/net/ipv4/devinet.c
@@ -871,6 +871,7 @@ __be32 inet_select_addr(const struct net_device *dev, __be32 dst, int
scope)
 {
  __be32 addr = 0;
  struct in_device *in_dev;
```

```
+ struct net *net = dev->nd_net;

  rcu_read_lock();
  in_dev = __in_dev_get_rcu(dev);
@@ -899,7 +900,7 @@ no_in_dev:
   */
  read_lock(&dev_base_lock);
  rcu_read_lock();
- for_each_netdev(&init_net, dev) {
+ for_each_netdev(net, dev) {
   if ((in_dev = __in_dev_get_rcu(dev)) == NULL)
    continue;

--
1.5.3.rc5
```

Subject: Re: [PATCH 6/17 net-2.6.26] [NETNS]: Default arp parameters lookup.
Posted by Daniel Lezcano on Tue, 19 Feb 2008 09:14:51 GMT
View Forum Message <> Reply to Message

Denis V. Lunev wrote:
> Default ARP parameters should be findable regardless of the context.
> Required to make inetdev_event working.
>
> Signed-off-by: Denis V. Lunev <den@openvz.org>
> ---
> net/core/neighbour.c | 4 +---
> 1 files changed, 1 insertions(+), 3 deletions(-)
>
> diff --git a/net/core/neighbour.c b/net/core/neighbour.c
> index c895ad4..45ed620 100644
> --- a/net/core/neighbour.c
> +++ b/net/core/neighbour.c
> @@ -1275,9 +1275,7 @@ static inline struct neigh_parms *lookup_neigh_params(struct
neigh_table *tbl,
>   struct neigh_parms *p;
>
>   for (p = &tbl->parms; p; p = p->next) {
> - if (p->net != net)
> -   continue;
> - if ((p->dev && p->dev->ifindex == ifindex) ||
> + if ((p->dev && p->dev->ifindex == ifindex && p->net == net) ||
>   (!p->dev && !ifindex))
>   return p;
>   }

If the values are:

p->dev == NULL
ifindex == 0
p->net != net

The parms should not be taken into account and the looping must continue. But with this modification it is not the case, if we specify parms ifindex == 0, the first parms with the dev field set to NULL will be taken belonging or not to the right net.

IMO the right test is:

if (p->net == net && ((p->dev && p->dev->ifindex == ifindex) || !p->dev && !ifindex)))

I definitively prefer the first notation :)

> - if (p->net != net)
> - continue;

Subject: Re: [PATCH 6/17 net-2.6.26] [NETNS]: Default arp parameters lookup.
Posted by den on Tue, 19 Feb 2008 09:39:40 GMT
View Forum Message <> Reply to Message

On Tue, 2008-02-19 at 10:14 +0100, Daniel Lezcano wrote:
> Denis V. Lunev wrote:
> > Default ARP parameters should be findable regardless of the context.
> > Required to make inetdev_event working.
> >
> > Signed-off-by: Denis V. Lunev <den@openvz.org>
> > ---
> > net/core/neighbour.c | 4 +---
> > 1 files changed, 1 insertions(+), 3 deletions(-)
> >
> > diff --git a/net/core/neighbour.c b/net/core/neighbour.c
> > index c895ad4..45ed620 100644
> > --- a/net/core/neighbour.c
> > +++ b/net/core/neighbour.c
> > @@ -1275,9 +1275,7 @@ static inline struct neigh_parms *lookup_neigh_params(struct neigh_table *tbl,
> > struct neigh_parms *p;
> >
> > for (p = &tbl->parms; p; p = p->next) {
> > - if (p->net != net)
> > - continue;
> > - if ((p->dev && p->dev->ifindex == ifindex) ||
> > + if ((p->dev && p->dev->ifindex == ifindex && p->net == net) ||
> > (!p->dev && !ifindex))

> >   return p;
> >   }
>
> If the values are:
> p->dev == NULL
> ifindex == 0
> p->net != net
>
> The parms should not be taken into account and the looping must
> continue. But with this modification it is not the case, if we specify
> parms ifindex == 0, the first parms with the dev field set to NULL will
> be taken belonging or not to the right net.

They should be taken. In the other case inetdev_event will fail for sure
in the middle. You could check.

These are ARP defaults and I do not see a problem for now to get them.

Subject: Re: [PATCH 6/17 net-2.6.26] [NETNS]: Default arp parameters lookup.
Posted by Daniel Lezcano on Tue, 19 Feb 2008 09:51:12 GMT
View Forum Message <> Reply to Message

Denis V. Lunev wrote:
> On Tue, 2008-02-19 at 10:14 +0100, Daniel Lezcano wrote:
>> Denis V. Lunev wrote:
>>> Default ARP parameters should be findable regardless of the context.
>>> Required to make inetdev_event working.
>>>
>>> Signed-off-by: Denis V. Lunev <den@openvz.org>
>>> ---
>>> net/core/neighbour.c | 4 +---
>>> 1 files changed, 1 insertions(+), 3 deletions(-)
>>>
>>> diff --git a/net/core/neighbour.c b/net/core/neighbour.c
>>> index c895ad4..45ed620 100644
>>> --- a/net/core/neighbour.c
>>> +++ b/net/core/neighbour.c
>>> @@ -1275,9 +1275,7 @@ static inline struct neigh_parms *lookup_neigh_params(struct
neigh_table *tbl,
>>>   struct neigh_parms *p;
>>>
>>>   for (p = &tbl->parms; p; p = p->next) {
>>> - if (p->net != net)
>>> -   continue;
>>> - if ((p->dev && p->dev->ifindex == ifindex) ||
>>> + if ((p->dev && p->dev->ifindex == ifindex && p->net == net) ||
>>>   (!p->dev && !ifindex))

>>>   return p;
>>>   }
>> If the values are:
>> p->dev == NULL
>> ifindex == 0
>> p->net != net
>>
>> The parms should not be taken into account and the looping must
>> continue. But with this modification it is not the case, if we specify
>> parms ifindex == 0, the first parms with the dev field set to NULL will
>> be taken belonging or not to the right net.
>
> They should be taken. In the other case inetdev_event will fail for sure
> in the middle. You could check.
>
> These are ARP defaults and I do not see a problem for now to get them.

Because there is a parms default per namespace. So several instances of
them per nd table. That was the initial approach with Eric's patchset.

Subject: Re: [PATCH 6/17 net-2.6.26] [NETNS]: Default arp parameters lookup.
Posted by den on Tue, 19 Feb 2008 10:05:49 GMT
View Forum Message <> Reply to Message

On Tue, 2008-02-19 at 10:51 +0100, Daniel Lezcano wrote:
> Denis V. Lunev wrote:
> > On Tue, 2008-02-19 at 10:14 +0100, Daniel Lezcano wrote:
> >> Denis V. Lunev wrote:
> >>> Default ARP parameters should be findable regardless of the context.
> >>> Required to make inetdev_event working.
> >>>
> >>> Signed-off-by: Denis V. Lunev <den@openvz.org>
> >>> ---
> >>> net/core/neighbour.c | 4 +---
> >>> 1 files changed, 1 insertions(+), 3 deletions(-)
> >>>
> >>> diff --git a/net/core/neighbour.c b/net/core/neighbour.c
> >>> index c895ad4..45ed620 100644
> >>> --- a/net/core/neighbour.c
> >>> +++ b/net/core/neighbour.c
> >>> @@ -1275,9 +1275,7 @@ static inline struct neigh_parms *lookup_neigh_params(struct
neigh_table *tbl,
> >>>   struct neigh_parms *p;
> >>>
> >>>   for (p = &tbl->parms; p; p = p->next) {
> >>> - if (p->net != net)
> >>> -   continue;

> >>> - if ((p->dev && p->dev->ifindex == ifindex) ||
> >>> + if ((p->dev && p->dev->ifindex == ifindex && p->net == net) ||
> >>>   (!p->dev && !ifindex))
> >>>   return p;
> >>>   }
> >> If the values are:
> >> p->dev == NULL
> >> ifindex == 0
> >> p->net != net
> >>
> >> The parms should not be taken into account and the looping must
> >> continue. But with this modification it is not the case, if we specify
> >> parms ifindex == 0, the first parms with the dev field set to NULL will
> >> be taken belonging or not to the right net.
> >
> > They should be taken. In the other case inetdev_event will fail for sure
> > in the middle. You could check.
> >
> > These are ARP defaults and I do not see a problem for now to get them.
>
> Because there is a parms default per namespace. So several instances of
> them per nd table. That was the initial approach with Eric's patchset.
>

These changes are not in mainstream and I do not want to touch ARP as
this is not a simple thing. In reality ARP will be needed only when
we'll have a real device inside a namespace.

Right now I prefer to have minimal set of working changes to finish IP
and upper layers.

Regards,
 Den

Subject: Re: [PATCH 6/17 net-2.6.26] [NETNS]: Default arp parameters lookup.
Posted by Daniel Lezcano on Tue, 19 Feb 2008 10:22:35 GMT
View Forum Message <> Reply to Message

Denis V. Lunev wrote:
> On Tue, 2008-02-19 at 10:51 +0100, Daniel Lezcano wrote:
>> Denis V. Lunev wrote:
>>> On Tue, 2008-02-19 at 10:14 +0100, Daniel Lezcano wrote:
>>>> Denis V. Lunev wrote:
>>>>> Default ARP parameters should be findable regardless of the context.
>>>>> Required to make inetdev_event working.
>>>>>
>>>>> Signed-off-by: Denis V. Lunev <den@openvz.org>

>>>>> ---
>>>>> net/core/neighbour.c | 4 +---
>>>>> 1 files changed, 1 insertions(+), 3 deletions(-)
>>>>>
>>>>> diff --git a/net/core/neighbour.c b/net/core/neighbour.c
>>>>> index c895ad4..45ed620 100644
>>>>> --- a/net/core/neighbour.c
>>>>> +++ b/net/core/neighbour.c
>>>>> @@ -1275,9 +1275,7 @@ static inline struct neigh_parms *lookup_neigh_params(struct
neigh_table *tbl,
>>>>>   struct neigh_parms *p;
>>>>>
>>>>>   for (p = &tbl->parms; p; p = p->next) {
>>>>> - if (p->net != net)
>>>>> -   continue;
>>>>> - if ((p->dev && p->dev->ifindex == ifindex) ||
>>>>> + if ((p->dev && p->dev->ifindex == ifindex && p->net == net) ||
>>>>>   (!p->dev && !ifindex))
>>>>>   return p;
>>>>>   }
>>>> If the values are:
>>>> p->dev == NULL
>>>> ifindex == 0
>>>> p->net != net
>>>>
>>>> The parms should not be taken into account and the looping must
>>>> continue. But with this modification it is not the case, if we specify
>>>> parms ifindex == 0, the first parms with the dev field set to NULL will
>>>> be taken belonging or not to the right net.
>>> They should be taken. In the other case inetdev_event will fail for sure
>>> in the middle. You could check.
>>>
>>> These are ARP defaults and I do not see a problem for now to get them.
>> Because there is a parms default per namespace. So several instances of
>> them per nd table. That was the initial approach with Eric's patchset.
>>
>
> These changes are not in mainstream and I do not want to touch ARP as
> this is not a simple thing. In reality ARP will be needed only when
> we'll have a real device inside a namespace.
>
> Right now I prefer to have minimal set of working changes to finish IP
> and upper layers.

core/neighbour.c is a common part between several protocols, especially
ipv4 and ipv6. If you modify this function just to fit your need in the
arp that will block me for ipv6 until you make parms default per
namespace. So please, find another way to do that, perhaps just add a

helper function.

I suggest you do parms default per namespace first, it is quite small
and easy :)

Just let me the time to send the copy-parms-default function.

Is it ok ?

-- Daniel

-- Daniel

Subject: Re: [PATCH 3/17 net-2.6.26] [NETFILTER]: Consolidate masq_inet_event and masq_device_event.
Posted by Patrick McHardy on Tue, 19 Feb 2008 14:10:11 GMT
View Forum Message <> Reply to Message

Denis V. Lunev wrote:
> They do exactly the same job.
>
> Signed-off-by: Denis V. Lunev <den@openvz.org>
> ---
> net/ipv4/netfilter/ipt_MASQUERADE.c | 14 ++----------
> 1 files changed, 2 insertions(+), 12 deletions(-)

Looks fine.

Subject: Re: [PATCH 6/17 net-2.6.26] [NETNS]: Default arp parameters lookup.
Posted by Daniel Lezcano on Tue, 19 Feb 2008 15:16:43 GMT
View Forum Message <> Reply to Message

Daniel Lezcano wrote:
> Denis V. Lunev wrote:
>> On Tue, 2008-02-19 at 10:51 +0100, Daniel Lezcano wrote:
>>> Denis V. Lunev wrote:
>>>> On Tue, 2008-02-19 at 10:14 +0100, Daniel Lezcano wrote:
>>>>> Denis V. Lunev wrote:
>>>>>> Default ARP parameters should be findable regardless of the context.
>>>>>> Required to make inetdev_event working.
>>>>>>
>>>>>> Signed-off-by: Denis V. Lunev <den@openvz.org>
>>>>>> ---

```
>>>>>>  net/core/neighbour.c |    4 +---
>>>>>>  1 files changed, 1 insertions(+), 3 deletions(-)
>>>>>>
>>>>>> diff --git a/net/core/neighbour.c b/net/core/neighbour.c
>>>>>> index c895ad4..45ed620 100644
>>>>>> --- a/net/core/neighbour.c
>>>>>> +++ b/net/core/neighbour.c
>>>>>> @@ -1275,9 +1275,7 @@ static inline struct neigh_parms
>>>>>> *lookup_neigh_params(struct neigh_table *tbl,
>>>>>>      struct neigh_parms *p;
>>>>>>
>>>>>>      for (p = &tbl->parms; p; p = p->next) {
>>>>>> -        if (p->net != net)
>>>>>> -            continue;
>>>>>> -        if ((p->dev && p->dev->ifindex == ifindex) ||
>>>>>> +        if ((p->dev && p->dev->ifindex == ifindex && p->net ==
>>>>>> net) ||
>>>>>>              (!p->dev && !ifindex))
>>>>>>              return p;
>>>>>>      }
>>>>> If the values are:
>>>>>     p->dev == NULL
>>>>>     ifindex == 0
>>>>>     p->net != net
>>>>>
>>>>> The parms should not be taken into account and the looping must
>>>>> continue. But with this modification it is not the case, if we
>>>>> specify parms ifindex == 0, the first parms with the dev field set
>>>>> to NULL will be taken belonging or not to the right net.
>>>> They should be taken. In the other case inetdev_event will fail for
>>>> sure
>>>> in the middle. You could check.
>>>>
>>>> These are ARP defaults and I do not see a problem for now to get them.
>>> Because there is a parms default per namespace. So several instances
>>> of them per nd table. That was the initial approach with Eric's
>>> patchset.
>>>
>>
>> These changes are not in mainstream and I do not want to touch ARP as
>> this is not a simple thing. In reality ARP will be needed only when
>> we'll have a real device inside a namespace.
>>
>> Right now I prefer to have minimal set of working changes to finish IP
>> and upper layers.
>
> core/neighbour.c is a common part between several protocols, especially
> ipv4 and ipv6. If you modify this function just to fit your need in the
```

> arp that will block me for ipv6 until you make parms default per
> namespace. So please, find another way to do that, perhaps just add a
> helper function.
>
> I suggest you do parms default per namespace first, it is quite small
> and easy :)
>
> Just let me the time to send the copy-parms-default function.

Ok, so after a long discussion with Denis about this patch, I will
change the ipv6 code to share the neigh->parms. It is not a problem.
Having the behavior of the neighbour subsystem per namespace is not a
must-have.

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

Subject: Re: [PATCH 0/17] Finish IPv4 infrastructure namespacing.
Posted by davem on Fri, 29 Feb 2008 04:53:09 GMT
View Forum Message <> Reply to Message

From: "Denis V. Lunev" <den@openvz.org>
Date: Tue, 19 Feb 2008 10:28:36 +0300

> This set finally allows to manipulate with network devices inside a
> namespace and allows to configure them [via netlink]. 'route' is not yet
> supported (but prepared to) as it requires a socket.
>
> Additionally, better routing cache support is added.
>
> Signed-off-by: Denis V. Lunev <den@openvz.org>

All applied to net-2.6.26, thanks Denis.
