Subject: [PATCH 0/8 2.6.25] a set of small code cleanups
Posted by den on Mon, 14 Jan 2008 14:59:23 GMT
View Forum Message <> Reply to Message

This set contains a set of small improvements found in IPv4 code during
preparations to support ARP in the network namespace. These fixes are
mostly independent except 2 last ones.

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

Subject: [PATCH 1/8 net-2.6.25] [ARP] Move inet_addr_type call after simple error checks in arp_contructor.
Posted by den on Mon, 14 Jan 2008 14:59:54 GMT
View Forum Message <> Reply to Message

The neighbour entry will be destroyed in the case of error, so it is pointless
to perform constly routing table lookup in this case.

Signed-off-by: Denis V. Lunev <den@openvz.org>
---
 net/ipv4/arp.c | 4 ++--
 1 files changed, 2 insertions(+), 2 deletions(-)

```
diff --git a/net/ipv4/arp.c b/net/ipv4/arp.c
index b715ec0..49c24ff 100644
--- a/net/ipv4/arp.c
+++ b/net/ipv4/arp.c
@@ -235,8 +235,6 @@ static int arp_constructor(struct neighbour *neigh)
  struct in_device *in_dev;
  struct neigh_parms *parms;

- neigh->type = inet_addr_type(&init_net, addr);
-
  rcu_read_lock();
  in_dev = __in_dev_get_rcu(dev);
  if (in_dev == NULL) {
@@ -244,6 +242,8 @@ static int arp_constructor(struct neighbour *neigh)
   return -EINVAL;
  }

+ neigh->type = inet_addr_type(&init_net, addr);
+
  parms = in_dev->arp_parms;
  __neigh_parms_put(neigh->parms);
  neigh->parms = neigh_parms_clone(parms);
--
1.5.3.rc5
```

Subject: [PATCH 2/8 net-2.6.25] [NETNS] Make arp code network namespace consistent.
Posted by den on Mon, 14 Jan 2008 14:59:55 GMT
View Forum Message <> Reply to Message

Some calls in the arp.c have network namespace as an argument. Getting
init_net inside these functions is simply inconsistent. Fix this.

Signed-off-by: Denis V. Lunev <den@openvz.org>
---
 net/ipv4/arp.c | 8 ++++----
 1 files changed, 4 insertions(+), 4 deletions(-)

```
diff --git a/net/ipv4/arp.c b/net/ipv4/arp.c
index 49c24ff..0db7d49 100644
--- a/net/ipv4/arp.c
+++ b/net/ipv4/arp.c
@@ -969,13 +969,13 @@ static int arp_req_set_public(struct net *net, struct arpreq *r,
  if (mask && mask != htonl(0xFFFFFFFF))
   return -EINVAL;
  if (!dev && (r->arp_flags & ATF_COM)) {
-  dev = dev_getbyhwaddr(&init_net, r->arp_ha.sa_family,
+  dev = dev_getbyhwaddr(net, r->arp_ha.sa_family,
      r->arp_ha.sa_data);
   if (!dev)
    return -ENODEV;
  }
  if (mask) {
-  if (pneigh_lookup(&arp_tbl, &init_net, &ip, dev, 1) == NULL)
+  if (pneigh_lookup(&arp_tbl, net, &ip, dev, 1) == NULL)
    return -ENOBUFS;
   return 0;
  }
@@ -1084,7 +1084,7 @@ static int arp_req_delete_public(struct net *net, struct arpreq *r,
  __be32 mask = ((struct sockaddr_in *)&r->arp_netmask)->sin_addr.s_addr;

  if (mask == htonl(0xFFFFFFFF))
-  return pneigh_delete(&arp_tbl, &init_net, &ip, dev);
+  return pneigh_delete(&arp_tbl, net, &ip, dev);

  if (mask)
   return -EINVAL;
@@ -1162,7 +1162,7 @@ int arp_ioctl(struct net *net, unsigned int cmd, void __user *arg)
  rtnl_lock();
  if (r.arp_dev[0]) {
   err = -ENODEV;
-  if ((dev = __dev_get_by_name(&init_net, r.arp_dev)) == NULL)
+  if ((dev = __dev_get_by_name(net, r.arp_dev)) == NULL)
    goto out;
```

```
   /* Mmmm... It is wrong... ARPHRD_NETROM==0 */
--
1.5.3.rc5
```

Subject: [PATCH 3/8 net-2.6.25] [IPV4] fib_rules_unregister is essentially void.
Posted by den on Mon, 14 Jan 2008 14:59:56 GMT
View Forum Message <> Reply to Message

fib_rules_unregister is called only after successful register and the return
code is never checked.

Signed-off-by: Denis V. Lunev <den@openvz.org>
---
 include/net/fib_rules.h | 2 +-
 net/core/fib_rules.c | 21 ++++----------------
 2 files changed, 5 insertions(+), 18 deletions(-)

```
diff --git a/include/net/fib_rules.h b/include/net/fib_rules.h
index 88f870f..34349f9 100644
--- a/include/net/fib_rules.h
+++ b/include/net/fib_rules.h
@@ -104,7 +104,7 @@ static inline u32 frh_get_table(struct fib_rule_hdr *frh, struct nlattr **nla)
 }

 extern int fib_rules_register(struct net *, struct fib_rules_ops *);
-extern int fib_rules_unregister(struct net *, struct fib_rules_ops *);
+extern void fib_rules_unregister(struct net *, struct fib_rules_ops *);
 extern void                    fib_rules_cleanup_ops(struct fib_rules_ops *);

 extern int   fib_rules_lookup(struct fib_rules_ops *,
diff --git a/net/core/fib_rules.c b/net/core/fib_rules.c
index 0eecf4c..42ccaf5 100644
--- a/net/core/fib_rules.c
+++ b/net/core/fib_rules.c
@@ -115,29 +115,16 @@ void fib_rules_cleanup_ops(struct fib_rules_ops *ops)
 }
 EXPORT_SYMBOL_GPL(fib_rules_cleanup_ops);

-int fib_rules_unregister(struct net *net, struct fib_rules_ops *ops)
+void fib_rules_unregister(struct net *net, struct fib_rules_ops *ops)
 {
- int err = 0;
- struct fib_rules_ops *o;

  spin_lock(&net->rules_mod_lock);
- list_for_each_entry(o, &net->rules_ops, list) {
```

```
-  if (o == ops) {
-   list_del_rcu(&o->list);
-   fib_rules_cleanup_ops(ops);
-   goto out;
-  }
- }
-
- err = -ENOENT;
-out:
+ list_del_rcu(&ops->list);
+ fib_rules_cleanup_ops(ops);
  spin_unlock(&net->rules_mod_lock);

  synchronize_rcu();
- if (!err)
-  release_net(net);
-
- return err;
+ release_net(net);
 }

 EXPORT_SYMBOL_GPL(fib_rules_unregister);
--
1.5.3.rc5
```

Subject: [PATCH 4/8 net-2.6.25] [ARP] Remove overkill checks from neigh_param_alloc.

Posted by den on Mon, 14 Jan 2008 14:59:57 GMT

View Forum Message <> Reply to Message

Valid network device is always passed into neigh_param_alloc, so remove
extra checking for dev == NULL. Additionally, cleanup bogus netns assignment.

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

```
---
 net/core/neighbour.c |   18 +++++++-----------
 1 files changed, 7 insertions(+), 11 deletions(-)

diff --git a/net/core/neighbour.c b/net/core/neighbour.c
index af49137..32f1a23 100644
--- a/net/core/neighbour.c
+++ b/net/core/neighbour.c
@@ -1301,10 +1301,7 @@ struct neigh_parms *neigh_parms_alloc(struct net_device *dev,
  struct neigh_parms *p, *ref;
  struct net *net;
```

```
- net = &init_net;
- if (dev)
-  net = dev->nd_net;
-
+ net = dev->nd_net;
  ref = lookup_neigh_params(tbl, net, 0);
  if (!ref)
   return NULL;
@@ -1316,15 +1313,14 @@ struct neigh_parms *neigh_parms_alloc(struct net_device *dev,
   INIT_RCU_HEAD(&p->rcu_head);
   p->reachable_time =
     neigh_rand_reach_time(p->base_reachable_time);
-  if (dev) {
-   if (dev->neigh_setup && dev->neigh_setup(dev, p)) {
-    kfree(p);
-    return NULL;
-   }

-   dev_hold(dev);
-   p->dev = dev;
+  if (dev->neigh_setup && dev->neigh_setup(dev, p)) {
+   kfree(p);
+   return NULL;
   }
+
+  dev_hold(dev);
+  p->dev = dev;
   p->net = hold_net(net);
   p->sysctl_table = NULL;
   write_lock_bh(&tbl->lock);
--
1.5.3.rc5
```

Subject: [PATCH 5/8 net-2.6.25] [ARP] Remove forward declaration of neigh_changeaddr.

Posted by den on Mon, 14 Jan 2008 14:59:58 GMT

View Forum Message <> Reply to Message

No need for this. It is declared in the neighbour.h

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

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

diff --git a/net/core/neighbour.c b/net/core/neighbour.c
index 2eab6a5..9b0b773 100644
```

```
--- a/net/core/neighbour.c
+++ b/net/core/neighbour.c
@@ -59,7 +59,6 @@ static void neigh_timer_handler(unsigned long arg);
 static void __neigh_notify(struct neighbour *n, int type, int flags);
 static void neigh_update_notify(struct neighbour *neigh);
 static int pneigh_ifdown(struct neigh_table *tbl, struct net_device *dev);
-void neigh_changeaddr(struct neigh_table *tbl, struct net_device *dev);

 static struct neigh_table *neigh_tables;
 #ifdef CONFIG_PROC_FS
--
1.5.3.rc5
```

Subject: [PATCH 6/8 net-2.6.25] [ARP] neigh_parms_put(destroy) are essentially local to core/neighbour.c.

Posted by den on Mon, 14 Jan 2008 14:59:59 GMT

View Forum Message <> Reply to Message

Make them static.

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

```
---
 include/net/neighbour.h |    7 -------
 net/core/neighbour.c    |   11 ++++++++++-
 2 files changed, 10 insertions(+), 8 deletions(-)

diff --git a/include/net/neighbour.h b/include/net/neighbour.h
index a0d42a5..ebbfb50 100644
--- a/include/net/neighbour.h
+++ b/include/net/neighbour.h
@@ -213,7 +213,6 @@ extern struct neighbour  *neigh_event_ns(struct neigh_table *tbl,

 extern struct neigh_parms *neigh_parms_alloc(struct net_device *dev, struct neigh_table *tbl);
 extern void    neigh_parms_release(struct neigh_table *tbl, struct neigh_parms *parms);
-extern void    neigh_parms_destroy(struct neigh_parms *parms);
 extern unsigned long  neigh_rand_reach_time(unsigned long base);

 extern void    pneigh_enqueue(struct neigh_table *tbl, struct neigh_parms *p,
@@ -254,12 +253,6 @@ static inline void __neigh_parms_put(struct neigh_parms *parms)
   atomic_dec(&parms->refcnt);
 }

-static inline void neigh_parms_put(struct neigh_parms *parms)
-{
- if (atomic_dec_and_test(&parms->refcnt))
-  neigh_parms_destroy(parms);
-}
```

```
-
 static inline struct neigh_parms *neigh_parms_clone(struct neigh_parms *parms)
 {
  atomic_inc(&parms->refcnt);
diff --git a/net/core/neighbour.c b/net/core/neighbour.c
index 9b0b773..41394db 100644
--- a/net/core/neighbour.c
+++ b/net/core/neighbour.c
@@ -55,6 +55,8 @@

 #define PNEIGH_HASHMASK  0xF

+static inline void neigh_parms_put(struct neigh_parms *parms);
+
 static void neigh_timer_handler(unsigned long arg);
 static void __neigh_notify(struct neighbour *n, int type, int flags);
 static void neigh_update_notify(struct neighbour *neigh);
@@ -1348,7 +1350,7 @@ void neigh_parms_release(struct neigh_table *tbl, struct neigh_parms
*parms)
  NEIGH_PRINTK1("neigh_parms_release: not found\n");
 }

-void neigh_parms_destroy(struct neigh_parms *parms)
+static void neigh_parms_destroy(struct neigh_parms *parms)
 {
  release_net(parms->net);
  if (parms->dev)
@@ -1356,6 +1358,13 @@ void neigh_parms_destroy(struct neigh_parms *parms)
  kfree(parms);
 }

+static inline void neigh_parms_put(struct neigh_parms *parms)
+{
+ if (atomic_dec_and_test(&parms->refcnt))
+  neigh_parms_destroy(parms);
+}
+
+
 static struct lock_class_key neigh_table_proxy_queue_class;

 void neigh_table_init_no_netlink(struct neigh_table *tbl)
--
1.5.3.rc5
```

Subject: [PATCH 7/8 net-2.6.25] [IPV4] Remove extra argument from arp_ignore.
Posted by den on Mon, 14 Jan 2008 15:00:00 GMT

arp_ignore has two arguments: dev & in_dev. dev is used for inet_confirm_addr
calling only.

inet_confirm_addr, in turn, either gets in_dev from the device passed or
iterates over all network devices if the device passed is NULL. It seems
logical to directly pass in_dev into inet_confirm_addr.

Signed-off-by: Denis V. Lunev <den@openvz.org>
---
 include/linux/inetdevice.h | 2 +-
 net/ipv4/arp.c | 11 +++++------
 net/ipv4/devinet.c | 17 ++++++++-----------
 3 files changed, 12 insertions(+), 18 deletions(-)

```
diff --git a/include/linux/inetdevice.h b/include/linux/inetdevice.h
index b3c5081..45f3731 100644
--- a/include/linux/inetdevice.h
+++ b/include/linux/inetdevice.h
@@ -135,7 +135,7 @@ extern int  devinet_ioctl(unsigned int cmd, void __user *);
 extern void  devinet_init(void);
 extern struct in_device *inetdev_by_index(int);
 extern __be32  inet_select_addr(const struct net_device *dev, __be32 dst, int scope);
-extern __be32  inet_confirm_addr(const struct net_device *dev, __be32 dst, __be32 local, int
scope);
+extern __be32  inet_confirm_addr(struct in_device *in_dev, __be32 dst, __be32 local, int scope);
 extern struct in_ifaddr *inet_ifa_byprefix(struct in_device *in_dev, __be32 prefix, __be32 mask);

 static __inline__ int inet_ifa_match(__be32 addr, struct in_ifaddr *ifa)
diff --git a/net/ipv4/arp.c b/net/ipv4/arp.c
index e944d98..f38e1a9 100644
--- a/net/ipv4/arp.c
+++ b/net/ipv4/arp.c
@@ -382,8 +382,7 @@ static void arp_solicit(struct neighbour *neigh, struct sk_buff *skb)
   read_unlock_bh(&neigh->lock);
 }

-static int arp_ignore(struct in_device *in_dev, struct net_device *dev,
-       __be32 sip, __be32 tip)
+static int arp_ignore(struct in_device *in_dev, __be32 sip, __be32 tip)
 {
  int scope;

@@ -403,7 +402,7 @@ static int arp_ignore(struct in_device *in_dev, struct net_device *dev,
  case 3: /* Do not reply for scope host addresses */
   sip = 0;
   scope = RT_SCOPE_LINK;
-  dev = NULL;
```

```
+  in_dev = NULL;
   break;
  case 4: /* Reserved */
  case 5:
@@ -415,7 +414,7 @@ static int arp_ignore(struct in_device *in_dev, struct net_device *dev,
  default:
   return 0;
  }
- return !inet_confirm_addr(dev, sip, tip, scope);
+ return !inet_confirm_addr(in_dev, sip, tip, scope);
 }

 static int arp_filter(__be32 sip, __be32 tip, struct net_device *dev)
@@ -807,7 +806,7 @@ static int arp_process(struct sk_buff *skb)
  if (sip == 0) {
   if (arp->ar_op == htons(ARPOP_REQUEST) &&
       inet_addr_type(&init_net, tip) == RTN_LOCAL &&
-      !arp_ignore(in_dev,dev,sip,tip))
+      !arp_ignore(in_dev, sip, tip))
    arp_send(ARPOP_REPLY, ETH_P_ARP, sip, dev, tip, sha,
      dev->dev_addr, sha);
   goto out;
@@ -825,7 +824,7 @@ static int arp_process(struct sk_buff *skb)
     int dont_send = 0;

     if (!dont_send)
-     dont_send |= arp_ignore(in_dev,dev,sip,tip);
+     dont_send |= arp_ignore(in_dev,sip,tip);
     if (!dont_send && IN_DEV_ARPFILTER(in_dev))
      dont_send |= arp_filter(sip,tip,dev);
     if (!dont_send)
diff --git a/net/ipv4/devinet.c b/net/ipv4/devinet.c
index 03db15b..dc1665a 100644
--- a/net/ipv4/devinet.c
+++ b/net/ipv4/devinet.c
@@ -968,24 +968,19 @@ static __be32 confirm_addr_indev(struct in_device *in_dev, __be32
dst,

 /*
  * Confirm that local IP address exists using wildcards:
- * - dev: only on this interface, 0=any interface
+ * - in_dev: only on this interface, 0=any interface
  * - dst: only in the same subnet as dst, 0=any dst
  * - local: address, 0=autoselect the local address
  * - scope: maximum allowed scope value for the local address
  */
-__be32 inet_confirm_addr(const struct net_device *dev, __be32 dst, __be32 local, int scope)
+__be32 inet_confirm_addr(struct in_device *in_dev,
```

```
+    __be32 dst, __be32 local, int scope)
 {
   __be32 addr = 0;
- struct in_device *in_dev;
-
- if (dev) {
-  rcu_read_lock();
-  if ((in_dev = __in_dev_get_rcu(dev)))
-   addr = confirm_addr_indev(in_dev, dst, local, scope);
-  rcu_read_unlock();
+ struct net_device *dev;

-  return addr;
- }
+ if (in_dev != NULL)
+  return confirm_addr_indev(in_dev, dst, local, scope);

   read_lock(&dev_base_lock);
   rcu_read_lock();
--
1.5.3.rc5
```

Subject: [PATCH 8/8 net-2.6.25] [NETNS] Process inet_confirm_addr in the correct namespace.
Posted by den on Mon, 14 Jan 2008 15:00:01 GMT
View Forum Message <> Reply to Message

inet_confirm_addr can be called with NULL in_dev from arp_ignore iff
scope is RT_SCOPE_LINK.

Lets always pass the device and check for RT_SCOPE_LINK scope inside
inet_confirm_addr. This let us take network namespace from in_device a
need for an additional argument.

Signed-off-by: Denis V. Lunev <den@openvz.org>
---
 net/ipv4/arp.c | 1 -
 net/ipv4/devinet.c | 6 ++++--
 2 files changed, 4 insertions(+), 3 deletions(-)

```
diff --git a/net/ipv4/arp.c b/net/ipv4/arp.c
index f38e1a9..b715ec0 100644
--- a/net/ipv4/arp.c
+++ b/net/ipv4/arp.c
@@ -402,7 +400,6 @@ static int arp_ignore(struct in_device *in_dev, __be32 sip, __be32 tip)
   case 3: /* Do not reply for scope host addresses */
    sip = 0;
```

```
   scope = RT_SCOPE_LINK;
-  in_dev = NULL;
   break;
  case 4: /* Reserved */
  case 5:
diff --git a/net/ipv4/devinet.c b/net/ipv4/devinet.c
index dc1665a..4569c69 100644
--- a/net/ipv4/devinet.c
+++ b/net/ipv4/devinet.c
@@ -978,13 +978,15 @@ __be32 inet_confirm_addr(struct in_device *in_dev,
 {
  __be32 addr = 0;
  struct net_device *dev;
+ struct net *net;

- if (in_dev != NULL)
+ if (scope != RT_SCOPE_LINK)
   return confirm_addr_indev(in_dev, dst, local, scope);

+ net = in_dev->dev->nd_net;
  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))) {
    addr = confirm_addr_indev(in_dev, dst, local, scope);
    if (addr)
--
1.5.3.rc5
```

Subject: Re: [PATCH 6/8 net-2.6.25] [ARP] neigh_parms_put(destroy) are essentially local to core/neighbour.c.
Posted by davem on Tue, 15 Jan 2008 07:04:38 GMT
View Forum Message <> Reply to Message

From: "Denis V. Lunev" <den@openvz.org>
Date: Mon, 14 Jan 2008 17:59:59 +0300

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

It's completely awkward to put the inline after all
the call sites. A non-global-optimizing compiler
won't be able to even inline it.

I've reworked this patch as follows when applying

it.

```
diff --git a/include/net/neighbour.h b/include/net/neighbour.h
index a0d42a5..ebbfb50 100644
--- a/include/net/neighbour.h
+++ b/include/net/neighbour.h
@@ -213,7 +213,6 @@ extern struct neighbour  *neigh_event_ns(struct neigh_table *tbl,

 extern struct neigh_parms *neigh_parms_alloc(struct net_device *dev, struct neigh_table *tbl);
 extern void    neigh_parms_release(struct neigh_table *tbl, struct neigh_parms *parms);
-extern void    neigh_parms_destroy(struct neigh_parms *parms);
 extern unsigned long  neigh_rand_reach_time(unsigned long base);

 extern void    pneigh_enqueue(struct neigh_table *tbl, struct neigh_parms *p,
@@ -254,12 +253,6 @@ static inline void __neigh_parms_put(struct neigh_parms *parms)
  atomic_dec(&parms->refcnt);
 }

-static inline void neigh_parms_put(struct neigh_parms *parms)
-{
- if (atomic_dec_and_test(&parms->refcnt))
-  neigh_parms_destroy(parms);
-}
-
 static inline struct neigh_parms *neigh_parms_clone(struct neigh_parms *parms)
 {
  atomic_inc(&parms->refcnt);
diff --git a/net/core/neighbour.c b/net/core/neighbour.c
index 9b0b773..7cc772a 100644
--- a/net/core/neighbour.c
+++ b/net/core/neighbour.c
@@ -577,6 +577,13 @@ static int pneigh_ifdown(struct neigh_table *tbl, struct net_device *dev)
  return -ENOENT;
 }

+static void neigh_parms_destroy(struct neigh_parms *parms);
+
+static inline void neigh_parms_put(struct neigh_parms *parms)
+{
+ if (atomic_dec_and_test(&parms->refcnt))
+  neigh_parms_destroy(parms);
+}

 /*
  * neighbour must already be out of the table;
@@ -1348,7 +1355,7 @@ void neigh_parms_release(struct neigh_table *tbl, struct neigh_parms
*parms)
  NEIGH_PRINTK1("neigh_parms_release: not found\n");
```

```
 }

-void neigh_parms_destroy(struct neigh_parms *parms)
+static void neigh_parms_destroy(struct neigh_parms *parms)
 {
  release_net(parms->net);
  if (parms->dev)
```

Subject: Re: [PATCH 0/8 2.6.25] a set of small code cleanups
Posted by davem on Tue, 15 Jan 2008 07:06:39 GMT
View Forum Message <> Reply to Message

From: "Denis V. Lunev" <den@openvz.org>
Date: Mon, 14 Jan 2008 17:59:23 +0300

> This set contains a set of small improvements found in IPv4 code during
> preparations to support ARP in the network namespace. These fixes are
> mostly independent except 2 last ones.
>
> Signed-off-by: Denis V. Lunev <den@openvz.org>

All applied, thanks Denis.
