Subject: [PATCH net-2.6.25 0/6][NETNS]: Make ipv6_devconf (all and default) live in net namespaces
Posted by Pavel Emelianov on Thu, 10 Jan 2008 13:55:12 GMT
View Forum Message <> Reply to Message

The ipv6_devconf_(all) and ipv6_devconf_dflt are currently
global, but should be per-namespace.

This set moves them on the struct net. Or, more precisely,
on the struct netns_ipv6, which is already added.

Unfortunately, many code in the ipv6 cannot yet provide a
correct struct net to get the ipv6_devconf from (e.g. routing
code), so this part of job is to be done after the appropriate
parts are virtualized.

However, after this set user can play with the ipv6_devconf
inside a namespace not affecting the others.

Signed-off-by: Pavel Emelyanov <xemul@openvz.org>

Subject: [PATCH net-2.6.25 1/6][NETNS]: Clean out the ipv6-related sysctls creation/destruction
Posted by Pavel Emelianov on Thu, 10 Jan 2008 13:58:53 GMT
View Forum Message <> Reply to Message

The addrconf sysctls and neigh sysctls are registered and
unregistered always in pairs, so they can be joined into
one (well, two) functions, that accept the struct inet6_dev
and do all the job.

This also get rids of unneeded ifdefs inside the code.

Signed-off-by: Pavel Emelyanov <xemul@openvz.org>

---
 net/ipv6/addrconf.c | 63 ++++++++++++++++++++++++++----------------
 1 files changed, 34 insertions(+), 29 deletions(-)

diff --git a/net/ipv6/addrconf.c b/net/ipv6/addrconf.c
index 6a48bb8..27b35dd 100644
--- a/net/ipv6/addrconf.c
+++ b/net/ipv6/addrconf.c
@@ -102,7 +102,15 @@

 #ifdef CONFIG_SYSCTL
 static void addrconf_sysctl_register(struct inet6_dev *idev);

```
-static void addrconf_sysctl_unregister(struct ipv6_devconf *p);
+static void addrconf_sysctl_unregister(struct inet6_dev *idev);
+#else
+static inline void addrconf_sysctl_register(struct inet6_dev *idev)
+{
+}
+
+static inline void addrconf_sysctl_unregister(struct inet6_dev *idev)
+{
+}
 #endif

 #ifdef CONFIG_IPV6_PRIVACY
@@ -392,13 +400,7 @@ static struct inet6_dev * ipv6_add_dev(struct net_device *dev)

  ipv6_mc_init_dev(ndev);
  ndev->tstamp = jiffies;
-#ifdef CONFIG_SYSCTL
- neigh_sysctl_register(dev, ndev->nd_parms, NET_IPV6,
-         NET_IPV6_NEIGH, "ipv6",
-         &ndisc_ifinfo_sysctl_change,
-         NULL);
  addrconf_sysctl_register(ndev);
-#endif
  /* protected by rtnl_lock */
  rcu_assign_pointer(dev->ip6_ptr, ndev);

@@ -2391,15 +2393,8 @@ static int addrconf_notify(struct notifier_block *this, unsigned long
event,
  case NETDEV_CHANGENAME:
   if (idev) {
    snmp6_unregister_dev(idev);
-#ifdef CONFIG_SYSCTL
-   addrconf_sysctl_unregister(&idev->cnf);
-   neigh_sysctl_unregister(idev->nd_parms);
-   neigh_sysctl_register(dev, idev->nd_parms,
-           NET_IPV6, NET_IPV6_NEIGH, "ipv6",
-           &ndisc_ifinfo_sysctl_change,
-           NULL);
+   addrconf_sysctl_unregister(idev);
    addrconf_sysctl_register(idev);
-#endif
    err = snmp6_register_dev(idev);
    if (err)
     return notifier_from_errno(err);
@@ -2523,10 +2518,7 @@ static int addrconf_ifdown(struct net_device *dev, int how)
  /* Shot the device (if unregistered) */
```

```
  if (how == 1) {
-#ifdef CONFIG_SYSCTL
-  addrconf_sysctl_unregister(&idev->cnf);
-  neigh_sysctl_unregister(idev->nd_parms);
-#endif
+  addrconf_sysctl_unregister(idev);
   neigh_parms_release(&nd_tbl, idev->nd_parms);
   neigh_ifdown(&nd_tbl, dev);
   in6_dev_put(idev);
@@ -4106,21 +4098,34 @@ out:
  return;
 }

+static void __addrconf_sysctl_unregister(struct ipv6_devconf *p)
+{
+ struct addrconf_sysctl_table *t;
+
+ if (p->sysctl == NULL)
+  return;
+
+ t = p->sysctl;
+ p->sysctl = NULL;
+ unregister_sysctl_table(t->sysctl_header);
+ kfree(t->dev_name);
+ kfree(t);
+}
+
 static void addrconf_sysctl_register(struct inet6_dev *idev)
 {
+ neigh_sysctl_register(idev->dev, idev->nd_parms, NET_IPV6,
+         NET_IPV6_NEIGH, "ipv6",
+         &ndisc_ifinfo_sysctl_change,
+         NULL);
   __addrconf_sysctl_register(idev->dev->name, idev->dev->ifindex,
    idev, &idev->cnf);
 }

-static void addrconf_sysctl_unregister(struct ipv6_devconf *p)
+static void addrconf_sysctl_unregister(struct inet6_dev *idev)
 {
- if (p->sysctl) {
-  struct addrconf_sysctl_table *t = p->sysctl;
-  p->sysctl = NULL;
-  unregister_sysctl_table(t->sysctl_header);
-  kfree(t->dev_name);
-  kfree(t);
- }
+ __addrconf_sysctl_unregister(&idev->cnf);
```

```
+ neigh_sysctl_unregister(idev->nd_parms);
 }


@@ -4232,8 +4237,8 @@ void addrconf_cleanup(void)
  unregister_netdevice_notifier(&ipv6_dev_notf);

 #ifdef CONFIG_SYSCTL
- addrconf_sysctl_unregister(&ipv6_devconf_dflt);
- addrconf_sysctl_unregister(&ipv6_devconf);
+ __addrconf_sysctl_unregister(&ipv6_devconf_dflt);
+ __addrconf_sysctl_unregister(&ipv6_devconf);
 #endif

  rtnl_lock();
--
1.5.3.4
```

Subject: [PATCH net-2.6.25 2/6][NETNS]: Make the __addrconf_sysctl_register return an error

Posted by Pavel Emelianov on Thu, 10 Jan 2008 14:01:13 GMT

View Forum Message <> Reply to Message

This error code will be needed to abort the namespace
creation if needed.

Probably, this is to be checked when a new device is
created (currently it is ignored).

Signed-off-by: Pavel Emelyanov <xemul@openvz.org>

```
---
 net/ipv6/addrconf.c |    6 +++---
 1 files changed, 3 insertions(+), 3 deletions(-)

diff --git a/net/ipv6/addrconf.c b/net/ipv6/addrconf.c
index 27b35dd..18d4334 100644
--- a/net/ipv6/addrconf.c
+++ b/net/ipv6/addrconf.c
@@ -4044,7 +4044,7 @@ static struct addrconf_sysctl_table
  },
 };

-static void __addrconf_sysctl_register(char *dev_name, int ctl_name,
+static int __addrconf_sysctl_register(char *dev_name, int ctl_name,
   struct inet6_dev *idev, struct ipv6_devconf *p)
 {
```

```
  int i;
@@ -4088,14 +4088,14 @@ static void __addrconf_sysctl_register(char *dev_name, int
ctl_name,
    goto free_procname;

   p->sysctl = t;
- return;
+ return 0;

  free_procname:
   kfree(t->dev_name);
  free:
   kfree(t);
  out:
- return;
+ return -ENOBUFS;
  }

  static void __addrconf_sysctl_unregister(struct ipv6_devconf *p)
--
1.5.3.4
```

Subject: [PATCH net-2.6.25 3/6][NETNS]: Make the ctl-tables per-namespace
Posted by Pavel Emelianov on Thu, 10 Jan 2008 14:03:10 GMT
View Forum Message <> Reply to Message

This includes passing the net to __addrconf_sysctl_register
and saving this on the ctl_table->extra2 to be used in
handlers (those, needing it).

Signed-off-by: Pavel Emelyanov <xemul@openvz.org>

```
---
 net/ipv6/addrconf.c |   24 ++++++++++++++----------
 1 files changed, 14 insertions(+), 10 deletions(-)

diff --git a/net/ipv6/addrconf.c b/net/ipv6/addrconf.c
index 18d4334..bde50c6 100644
--- a/net/ipv6/addrconf.c
+++ b/net/ipv6/addrconf.c
@@ -456,13 +456,13 @@ static void dev_forward_change(struct inet6_dev *idev)
 }


-static void addrconf_forward_change(void)
+static void addrconf_forward_change(struct net *net)
 {
```

```
  struct net_device *dev;
  struct inet6_dev *idev;

  read_lock(&dev_base_lock);
- for_each_netdev(&init_net, dev) {
+ for_each_netdev(net, dev) {
   rcu_read_lock();
   idev = __in6_dev_get(dev);
   if (idev) {
@@ -478,12 +478,15 @@ static void addrconf_forward_change(void)

 static void addrconf_fixup_forwarding(struct ctl_table *table, int *p, int old)
 {
+ struct net *net;
+
+ net = (struct net *)table->extra2;
  if (p == &ipv6_devconf_dflt.forwarding)
   return;

  if (p == &ipv6_devconf.forwarding) {
   ipv6_devconf_dflt.forwarding = ipv6_devconf.forwarding;
-  addrconf_forward_change();
+  addrconf_forward_change(net);
  } else if ((!*p) ^ (!old))
   dev_forward_change((struct inet6_dev *)table->extra1);

@@ -4044,8 +4047,8 @@ static struct addrconf_sysctl_table
  },
 };

-static int __addrconf_sysctl_register(char *dev_name, int ctl_name,
-  struct inet6_dev *idev, struct ipv6_devconf *p)
+static int __addrconf_sysctl_register(struct net *net, char *dev_name,
+  int ctl_name, struct inet6_dev *idev, struct ipv6_devconf *p)
 {
  int i;
  struct addrconf_sysctl_table *t;
@@ -4068,6 +4071,7 @@ static int __addrconf_sysctl_register(char *dev_name, int ctl_name,
  for (i=0; t->addrconf_vars[i].data; i++) {
   t->addrconf_vars[i].data += (char*)p - (char*)&ipv6_devconf;
   t->addrconf_vars[i].extra1 = idev; /* embedded; no ref */
+  t->addrconf_vars[i].extra2 = net;
  }

  /*
@@ -4082,7 +4086,7 @@ static int __addrconf_sysctl_register(char *dev_name, int ctl_name,
  addrconf_ctl_path[ADDRCONF_CTL_PATH_DEV].procname = t->dev_name;
  addrconf_ctl_path[ADDRCONF_CTL_PATH_DEV].ctl_name = ctl_name;
```

```
- t->sysctl_header = register_sysctl_paths(addrconf_ctl_path,
+ t->sysctl_header = register_net_sysctl_table(net, addrconf_ctl_path,
    t->addrconf_vars);
  if (t->sysctl_header == NULL)
   goto free_procname;
@@ -4118,8 +4122,8 @@ static void addrconf_sysctl_register(struct inet6_dev *idev)
         NET_IPV6_NEIGH, "ipv6",
         &ndisc_ifinfo_sysctl_change,
         NULL);
- __addrconf_sysctl_register(idev->dev->name, idev->dev->ifindex,
-   idev, &idev->cnf);
+ __addrconf_sysctl_register(idev->dev->nd_net, idev->dev->name,
+   idev->dev->ifindex, idev, &idev->cnf);
 }

 static void addrconf_sysctl_unregister(struct inet6_dev *idev)
@@ -4215,9 +4219,9 @@ int __init addrconf_init(void)
  ipv6_addr_label_rtnl_register();

 #ifdef CONFIG_SYSCTL
- __addrconf_sysctl_register("all", NET_PROTO_CONF_ALL,
+ __addrconf_sysctl_register(&init_net, "all", NET_PROTO_CONF_ALL,
    NULL, &ipv6_devconf);
- __addrconf_sysctl_register("default", NET_PROTO_CONF_DEFAULT,
+ __addrconf_sysctl_register(&init_net, "default", NET_PROTO_CONF_DEFAULT,
    NULL, &ipv6_devconf_dflt);
 #endif

--
1.5.3.4
```

Subject: [PATCH net-2.6.25 4/6][NETNS]: Create ipv6 devconf-s for namespaces
Posted by Pavel Emelianov on Thu, 10 Jan 2008 14:06:49 GMT
View Forum Message <> Reply to Message

This is the core. Declare and register the pernet subsys for
addrconf. The init callback the will create the devconf-s.

The init_net will reuse the existing statically declared confs,
so that accessing them from inside the ipv6 code will still
work.

The register_pernet_subsys() is moved above the ipv6_add_dev()
call for loopback, because this function will need the
net->devconf_dflt pointer to be already set.

Signed-off-by: Pavel Emelyanov <xemul@openvz.org>

---
 include/net/netns/ipv6.h | 2 +
 net/ipv6/addrconf.c | 82 ++++++++++++++++++++++++++++++++++++++-------
 2 files changed, 72 insertions(+), 12 deletions(-)

diff --git a/include/net/netns/ipv6.h b/include/net/netns/ipv6.h
index 10733a6..06b4dc0 100644
--- a/include/net/netns/ipv6.h
+++ b/include/net/netns/ipv6.h
@@ -28,5 +28,7 @@ struct netns_sysctl_ipv6 {

 struct netns_ipv6 {
  struct netns_sysctl_ipv6 sysctl;
+ struct ipv6_devconf *devconf_all;
+ struct ipv6_devconf *devconf_dflt;
 };
 #endif
diff --git a/net/ipv6/addrconf.c b/net/ipv6/addrconf.c
index bde50c6..3ad081e 100644
--- a/net/ipv6/addrconf.c
+++ b/net/ipv6/addrconf.c
@@ -4135,6 +4135,70 @@ static void addrconf_sysctl_unregister(struct inet6_dev *idev)

 #endif

+static int addrconf_init_net(struct net *net)
+{
+ int err;
+ struct ipv6_devconf *all, *dflt;
+
+ err = -ENOMEM;
+ all = &ipv6_devconf;
+ dflt = &ipv6_devconf_dflt;
+
+ if (net != &init_net) {
+ all = kmemdup(all, sizeof(ipv6_devconf), GFP_KERNEL);
+ if (all == NULL)
+   goto err_alloc_all;
+
+ dflt = kmemdup(dflt, sizeof(ipv6_devconf_dflt), GFP_KERNEL);
+ if (dflt == NULL)
+   goto err_alloc_dflt;
+ }
+
+ net->ipv6.devconf_all = all;
+ net->ipv6.devconf_dflt = dflt;

```
+
+#ifdef CONFIG_SYSCTL
+ err = __addrconf_sysctl_register(net, "all", NET_PROTO_CONF_ALL,
+   NULL, all);
+ if (err < 0)
+  goto err_reg_all;
+
+ err = __addrconf_sysctl_register(net, "default", NET_PROTO_CONF_DEFAULT,
+   NULL, dflt);
+ if (err < 0)
+  goto err_reg_dflt;
+#endif
+ return 0;
+
+#ifdef CONFIG_SYSCTL
+err_reg_dflt:
+ __addrconf_sysctl_unregister(all);
+err_reg_all:
+ kfree(dflt);
+#endif
+err_alloc_dflt:
+ kfree(all);
+err_alloc_all:
+ return err;
+}
+
+static void addrconf_exit_net(struct net *net)
+{
+#ifdef CONFIG_SYSCTL
+ __addrconf_sysctl_unregister(net->ipv6.devconf_dflt);
+ __addrconf_sysctl_unregister(net->ipv6.devconf_all);
+#endif
+ if (net != &init_net) {
+  kfree(net->ipv6.devconf_dflt);
+  kfree(net->ipv6.devconf_all);
+ }
+}
+
+static struct pernet_operations addrconf_ops = {
+ .init = addrconf_init_net,
+ .exit = addrconf_exit_net,
+};
+
 /*
  *      Device notifier
  */
@@ -4167,6 +4231,8 @@ int __init addrconf_init(void)
   return err;
```

```
  }

+ register_pernet_subsys(&addrconf_ops);
+
  /* The addrconf netdev notifier requires that loopback_dev
   * has it's ipv6 private information allocated and setup
   * before it can bring up and give link-local addresses
@@ -4190,7 +4256,7 @@ int __init addrconf_init(void)
   err = -ENOMEM;
  rtnl_unlock();
  if (err)
-  return err;
+  goto errlo;

  ip6_null_entry.u.dst.dev = init_net.loopback_dev;
  ip6_null_entry.rt6i_idev = in6_dev_get(init_net.loopback_dev);
@@ -4218,16 +4284,11 @@ int __init addrconf_init(void)

  ipv6_addr_label_rtnl_register();

-#ifdef CONFIG_SYSCTL
- __addrconf_sysctl_register(&init_net, "all", NET_PROTO_CONF_ALL,
-   NULL, &ipv6_devconf);
- __addrconf_sysctl_register(&init_net, "default", NET_PROTO_CONF_DEFAULT,
-   NULL, &ipv6_devconf_dflt);
-#endif
-
  return 0;
 errout:
  unregister_netdevice_notifier(&ipv6_dev_notf);
+errlo:
+ unregister_pernet_subsys(&addrconf_ops);

  return err;
 }
@@ -4240,10 +4301,7 @@ void addrconf_cleanup(void)

  unregister_netdevice_notifier(&ipv6_dev_notf);

-#ifdef CONFIG_SYSCTL
- __addrconf_sysctl_unregister(&ipv6_devconf_dflt);
- __addrconf_sysctl_unregister(&ipv6_devconf);
-#endif
+ unregister_pernet_subsys(&addrconf_ops);

  rtnl_lock();

--
```

1.5.3.4

Subject: [PATCH net-2.6.25 5/6][NETNS]: Use the per-net ipv6_devconf_dflt
Posted by Pavel Emelianov on Thu, 10 Jan 2008 14:08:24 GMT
View Forum Message <> Reply to Message

All its users are in net/ipv6/addrconf.c's sysctl handlers.
Since they already have the struct net to get from, the
per-net ipv6_devconf_dflt can already be used.

Signed-off-by: Pavel Emelyanov <xemul@openvz.org>

---
 net/ipv6/addrconf.c | 6 +++---
 1 files changed, 3 insertions(+), 3 deletions(-)

diff --git a/net/ipv6/addrconf.c b/net/ipv6/addrconf.c
index 3ad081e..9b96de3 100644
--- a/net/ipv6/addrconf.c
+++ b/net/ipv6/addrconf.c
@@ -334,7 +334,7 @@ static struct inet6_dev * ipv6_add_dev(struct net_device *dev)

  rwlock_init(&ndev->lock);
  ndev->dev = dev;
- memcpy(&ndev->cnf, &ipv6_devconf_dflt, sizeof(ndev->cnf));
+ memcpy(&ndev->cnf, dev->nd_net->ipv6.devconf_dflt, sizeof(ndev->cnf));
  ndev->cnf.mtu6 = dev->mtu;
  ndev->cnf.sysctl = NULL;
  ndev->nd_parms = neigh_parms_alloc(dev, &nd_tbl);
@@ -481,11 +481,11 @@ static void addrconf_fixup_forwarding(struct ctl_table *table, int *p, int
old)
  struct net *net;

  net = (struct net *)table->extra2;
- if (p == &ipv6_devconf_dflt.forwarding)
+ if (p == &net->ipv6.devconf_dflt->forwarding)
   return;

  if (p == &ipv6_devconf.forwarding) {
- ipv6_devconf_dflt.forwarding = ipv6_devconf.forwarding;
+ net->ipv6.devconf_dflt->forwarding = ipv6_devconf.forwarding;
   addrconf_forward_change(net);
  } else if ((!*p) ^ (!old))
   dev_forward_change((struct inet6_dev *)table->extra1);
--
1.5.3.4

Subject: [PATCH net-2.6.25 6/6][NETNS]: Use the per-net ipv6_devconf(_all) in sysctl handlers
Posted by Pavel Emelianov on Thu, 10 Jan 2008 14:10:44 GMT
View Forum Message <> Reply to Message

Actually the net->ipv6.devconf_all can be used in a few places,
but to keep the /proc/sys/net/ipv6/conf/ sysctls work consistently
in the namespace we should use the per-net devconf_all in the
sysctl "forwarding" handler.

Signed-off-by: Pavel Emelyanov <xemul@openvz.org>

---
 net/ipv6/addrconf.c | 13 +++++++------
 1 files changed, 7 insertions(+), 6 deletions(-)

diff --git a/net/ipv6/addrconf.c b/net/ipv6/addrconf.c
index 9b96de3..cd90f9a 100644
--- a/net/ipv6/addrconf.c
+++ b/net/ipv6/addrconf.c
@@ -456,7 +456,7 @@ static void dev_forward_change(struct inet6_dev *idev)
 }

-static void addrconf_forward_change(struct net *net)
+static void addrconf_forward_change(struct net *net, __s32 newf)
 {
  struct net_device *dev;
  struct inet6_dev *idev;
@@ -466,8 +466,8 @@ static void addrconf_forward_change(struct net *net)
   rcu_read_lock();
   idev = __in6_dev_get(dev);
   if (idev) {
-   int changed = (!idev->cnf.forwarding) ^ (!ipv6_devconf.forwarding);
-   idev->cnf.forwarding = ipv6_devconf.forwarding;
+   int changed = (!idev->cnf.forwarding) ^ (!newf);
+   idev->cnf.forwarding = newf;
    if (changed)
     dev_forward_change(idev);
   }
@@ -484,9 +484,10 @@ static void addrconf_fixup_forwarding(struct ctl_table *table, int *p, int
old)
  if (p == &net->ipv6.devconf_dflt->forwarding)
   return;

- if (p == &ipv6_devconf.forwarding) {
- net->ipv6.devconf_dflt->forwarding = ipv6_devconf.forwarding;
- addrconf_forward_change(net);
+ if (p == &net->ipv6.devconf_all->forwarding) {

```
+  __s32 newf = net->ipv6.devconf_all->forwarding;
+  net->ipv6.devconf_dflt->forwarding = newf;
+  addrconf_forward_change(net, newf);
  } else if ((!*p) ^ (!old))
   dev_forward_change((struct inet6_dev *)table->extra1);

--
1.5.3.4
```

Subject: Re: [PATCH net-2.6.25 0/6][NETNS]: Make ipv6_devconf (all and default) live in net namespaces
Posted by davem on Fri, 11 Jan 2008 01:54:12 GMT
View Forum Message <> Reply to Message

From: Pavel Emelyanov <xemul@openvz.org>
Date: Thu, 10 Jan 2008 16:55:12 +0300

> The ipv6_devconf_(all) and ipv6_devconf_dflt are currently
> global, but should be per-namespace.
>
> This set moves them on the struct net. Or, more precisely,
> on the struct netns_ipv6, which is already added.
>
> Unfortunately, many code in the ipv6 cannot yet provide a
> correct struct net to get the ipv6_devconf from (e.g. routing
> code), so this part of job is to be done after the appropriate
> parts are virtualized.
>
> However, after this set user can play with the ipv6_devconf
> inside a namespace not affecting the others.
>
> Signed-off-by: Pavel Emelyanov <xemul@openvz.org>

All 6 patches applied, thanks Pavel.
