Subject: [patch 37/38][IPV6] ndisc - dynamically allocate default neigh_parms for ndisc
Posted by Daniel Lezcano on Mon, 03 Dec 2007 16:17:13 GMT
View Forum Message <> Reply to Message

This patch dynamically allocates default neigh_parms for IPv6 network
discovery and store them in struct net. That provides the basic changes
needed to support ndisc for multiple network namespaces.

Signed-off-by: Benjamin Thery <benjamin.thery@bull.net>
Signed-off-by: Daniel Lezcano <dlezcano@fr.ibm.com>
---
 include/net/net_namespace.h | 3 +++
 net/ipv6/ndisc.c | 38 ++++++++++++++++++++++++++++++++------
 2 files changed, 34 insertions(+), 7 deletions(-)

Index: linux-2.6-netns/include/net/net_namespace.h
===
--- linux-2.6-netns.orig/include/net/net_namespace.h
+++ linux-2.6-netns/include/net/net_namespace.h
@@ -83,6 +83,9 @@ struct net {
  unsigned int fib_info_hash_size;
  unsigned int fib_info_cnt;
  struct hlist_head *fib_info_devhash;
+
+ /* IPv6 ndisc.c */
+ struct neigh_parms *ndisc_neigh_parms_default;
 };

 #ifdef CONFIG_NET
Index: linux-2.6-netns/net/ipv6/ndisc.c
===
--- linux-2.6-netns.orig/net/ipv6/ndisc.c
+++ linux-2.6-netns/net/ipv6/ndisc.c
@@ -1764,23 +1764,47 @@ int __init ndisc_init(struct net_proto_f

  neigh_table_init(&nd_tbl);

+ init_net.ndisc_neigh_parms_default =
+ neigh_parms_alloc_default(&nd_tbl, &init_net);
+ if (!init_net.ndisc_neigh_parms_default) {
+ err = -ENOMEM;
+ goto out_neigh_parms;
+ }
+
 #ifdef CONFIG_SYSCTL
- neigh_sysctl_register(NULL, &nd_tbl.parms, NET_IPV6, NET_IPV6_NEIGH,
-         "ipv6",

```
-         &ndisc_ifinfo_sysctl_change,
-         &ndisc_ifinfo_sysctl_strategy);
+ if ((err = neigh_sysctl_register(NULL,
+       init_net.ndisc_neigh_parms_default,
+       NET_IPV6, NET_IPV6_NEIGH,
+       "ipv6",
+       &ndisc_ifinfo_sysctl_change,
+       &ndisc_ifinfo_sysctl_strategy)))
+       goto out_sysctl;
 #endif
-
  register_netdevice_notifier(&ndisc_netdev_notifier);
- return 0;
+out:
+ return err;
+#ifdef CONFIG_SYSCTL
+out_sysctl:
+ neigh_parms_release(&nd_tbl, init_net.ndisc_neigh_parms_default);
+#endif
+out_neigh_parms:
+ sock_release(ndisc_socket);
+ goto out;
 }

 void ndisc_cleanup(void)
 {
+ struct neigh_parms *parms = init_net.ndisc_neigh_parms_default;
+
  unregister_netdevice_notifier(&ndisc_netdev_notifier);
+
+ if (parms) {
 #ifdef CONFIG_SYSCTL
- neigh_sysctl_unregister(&nd_tbl.parms);
+  neigh_sysctl_unregister(parms);
 #endif
+  neigh_parms_release(&nd_tbl, parms);
+  init_net.ndisc_neigh_parms_default = NULL;
+ }
  neigh_table_clear(&nd_tbl);
  sock_release(ndisc_socket);
  ndisc_socket = NULL; /* For safety. */

--
```