Subject: [PATCH 11/13 net-2.6.26] [IPV6]: Simplify IPv6 control sockets creation.
Posted by den on Mon, 31 Mar 2008 13:16:51 GMT
View Forum Message <> Reply to Message

Do this by replacing sock_create_kern with inet_ctl_sock_create.

Signed-off-by: Denis V. Lunev <den@openvz.org>
---
 net/ipv6/icmp.c | 16 ++++++----------
 net/ipv6/mcast.c | 19 ++++++----------
 net/ipv6/ndisc.c | 12 +++++-------
 3 files changed, 17 insertions(+), 30 deletions(-)

diff --git a/net/ipv6/icmp.c b/net/ipv6/icmp.c
index 63309d1..24a657e 100644
--- a/net/ipv6/icmp.c
+++ b/net/ipv6/icmp.c
@@ -64,6 +64,7 @@
 #include <net/addrconf.h>
 #include <net/icmp.h>
 #include <net/xfrm.h>
+#include <net/inet_common.h>

 #include <asm/uaccess.h>
 #include <asm/system.h>
@@ -808,9 +809,8 @@ static int __net_init icmpv6_sk_init(struct net *net)
   return -ENOMEM;

  for_each_possible_cpu(i) {
- struct socket *sock;
- err = sock_create_kern(PF_INET6, SOCK_RAW, IPPROTO_ICMPV6,
-           &sock);
+ err = inet_ctl_sock_create(&sk, PF_INET6,
+   SOCK_RAW, IPPROTO_ICMPV6, net);
   if (err < 0) {
    printk(KERN_ERR
           "Failed to initialize the ICMP6 control socket "
@@ -819,10 +819,8 @@ static int __net_init icmpv6_sk_init(struct net *net)
    goto fail;
   }

- net->ipv6.icmp_sk[i] = sk = sock->sk;
- sk_change_net(sk, net);
+ net->ipv6.icmp_sk[i] = sk;

- sk->sk_allocation = GFP_ATOMIC;
   /*
    * Split off their lock-class, because sk->sk_dst_lock

```
    * gets used from softirqs, which is safe for
@@ -837,14 +835,12 @@ static int __net_init icmpv6_sk_init(struct net *net)
    */
   sk->sk_sndbuf =
    (2 * ((64 * 1024) + sizeof(struct sk_buff)));
-
-  sk->sk_prot->unhash(sk);
  }
  return 0;

  fail:
  for (j = 0; j < i; j++)
-  sk_release_kernel(net->ipv6.icmp_sk[j]);
+  inet_ctl_sock_destroy(net->ipv6.icmp_sk[j]);
  kfree(net->ipv6.icmp_sk);
  return err;
 }
@@ -854,7 +850,7 @@ static void __net_exit icmpv6_sk_exit(struct net *net)
  int i;

  for_each_possible_cpu(i) {
-  sk_release_kernel(net->ipv6.icmp_sk[i]);
+  inet_ctl_sock_destroy(net->ipv6.icmp_sk[i]);
  }
  kfree(net->ipv6.icmp_sk);
 }
diff --git a/net/ipv6/mcast.c b/net/ipv6/mcast.c
index d810cff..2e6a53f 100644
--- a/net/ipv6/mcast.c
+++ b/net/ipv6/mcast.c
@@ -59,6 +59,7 @@
 #include <net/ndisc.h>
 #include <net/addrconf.h>
 #include <net/ip6_route.h>
+#include <net/inet_common.h>

 #include <net/ip6_checksum.h>

@@ -2672,12 +2673,10 @@ static void igmp6_proc_exit(struct net *net)

 static int igmp6_net_init(struct net *net)
 {
- struct ipv6_pinfo *np;
- struct socket *sock;
- struct sock *sk;
  int err;

- err = sock_create_kern(PF_INET6, SOCK_RAW, IPPROTO_ICMPV6, &sock);
```

```
+ err = inet_ctl_sock_create(&net->ipv6.igmp_sk, PF_INET6,
+       SOCK_RAW, IPPROTO_ICMPV6, net);
  if (err < 0) {
   printk(KERN_ERR
          "Failed to initialize the IGMP6 control socket (err %d).\n",
@@ -2685,13 +2684,7 @@ static int igmp6_net_init(struct net *net)
   goto out;
  }

- net->ipv6.igmp_sk = sk = sock->sk;
- sk_change_net(sk, net);
- sk->sk_allocation = GFP_ATOMIC;
- sk->sk_prot->unhash(sk);
-
- np = inet6_sk(sk);
- np->hop_limit = 1;
+ inet6_sk(net->ipv6.igmp_sk)->hop_limit = 1;

  err = igmp6_proc_init(net);
  if (err)
@@ -2700,13 +2693,13 @@ out:
  return err;

 out_sock_create:
- sk_release_kernel(net->ipv6.igmp_sk);
+ inet_ctl_sock_destroy(net->ipv6.igmp_sk);
  goto out;
 }

 static void igmp6_net_exit(struct net *net)
 {
- sk_release_kernel(net->ipv6.igmp_sk);
+ inet_ctl_sock_destroy(net->ipv6.igmp_sk);
  igmp6_proc_exit(net);
 }

diff --git a/net/ipv6/ndisc.c b/net/ipv6/ndisc.c
index 510aa74..06d80c6 100644
--- a/net/ipv6/ndisc.c
+++ b/net/ipv6/ndisc.c
@@ -84,6 +84,7 @@

 #include <net/flow.h>
 #include <net/ip6_checksum.h>
+#include <net/inet_common.h>
 #include <linux/proc_fs.h>

 #include <linux/netfilter.h>
```

```
@@ -1731,12 +1732,12 @@ static int ndisc_ifinfo_sysctl_strategy(ctl_table *ctl, int __user
*name,

 static int ndisc_net_init(struct net *net)
 {
- struct socket *sock;
  struct ipv6_pinfo *np;
  struct sock *sk;
  int err;

- err = sock_create_kern(PF_INET6, SOCK_RAW, IPPROTO_ICMPV6, &sock);
+ err = inet_ctl_sock_create(&sk, PF_INET6,
+       SOCK_RAW, IPPROTO_ICMPV6, net);
  if (err < 0) {
   ND_PRINTK0(KERN_ERR
       "ICMPv6 NDISC: Failed to initialize the control socket (err %d).\n",
@@ -1744,22 +1745,19 @@ static int ndisc_net_init(struct net *net)
   return err;
  }

- net->ipv6.ndisc_sk = sk = sock->sk;
- sk_change_net(sk, net);
+ net->ipv6.ndisc_sk = sk;

  np = inet6_sk(sk);
- sk->sk_allocation = GFP_ATOMIC;
  np->hop_limit = 255;
  /* Do not loopback ndisc messages */
  np->mc_loop = 0;
- sk->sk_prot->unhash(sk);

  return 0;
 }

 static void ndisc_net_exit(struct net *net)
 {
- sk_release_kernel(net->ipv6.ndisc_sk);
+ inet_ctl_sock_destroy(net->ipv6.ndisc_sk);
 }

 static struct pernet_operations ndisc_net_ops = {
--
1.5.3.rc5
```