Subject: [PATCH 7/13 net-2.6.26] [INET]: Let inet_ctl_sock_create return sock rather than socket.
Posted by den on Mon, 31 Mar 2008 13:16:47 GMT
View Forum Message <> Reply to Message

All upper protocol layers are already use sock internally.

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

```
 include/net/inet_common.h |    2 +-
 net/dccp/ipv4.c           |    4 +---
 net/dccp/ipv6.c           |    4 +---
 net/ipv4/af_inet.c        |   12 +++++++-----
 net/ipv4/tcp_ipv4.c       |    4 +---
 net/ipv6/tcp_ipv6.c       |    5 ++---
 net/sctp/protocol.c       |    4 +---
 7 files changed, 14 insertions(+), 21 deletions(-)

diff --git a/include/net/inet_common.h b/include/net/inet_common.h
index d6238bd..4bfcf3f 100644
--- a/include/net/inet_common.h
+++ b/include/net/inet_common.h
@@ -39,7 +39,7 @@ extern int   inet_getname(struct socket *sock,
 extern int   inet_ioctl(struct socket *sock,
        unsigned int cmd, unsigned long arg);

-extern int   inet_ctl_sock_create(struct socket **sock,
+extern int   inet_ctl_sock_create(struct sock **sk,
           unsigned short family,
           unsigned short type,
           unsigned char protocol);
diff --git a/net/dccp/ipv4.c b/net/dccp/ipv4.c
index feb3fa5..5669c89 100644
--- a/net/dccp/ipv4.c
+++ b/net/dccp/ipv4.c
@@ -991,7 +991,6 @@ static struct inet_protosw dccp_v4_protosw = {

 static int __init dccp_v4_init(void)
 {
- struct socket *socket;
  int err = proto_register(&dccp_v4_prot, 1);

  if (err != 0)
@@ -1003,11 +1002,10 @@ static int __init dccp_v4_init(void)

  inet_register_protosw(&dccp_v4_protosw);

- err = inet_ctl_sock_create(&socket, PF_INET,
```

```
+ err = inet_ctl_sock_create(&dccp_v4_ctl_sk, PF_INET,
        SOCK_DCCP, IPPROTO_DCCP);
  if (err)
   goto out_unregister_protosw;
- dccp_v4_ctl_sk = socket->sk;
 out:
  return err;
 out_unregister_protosw:
diff --git a/net/dccp/ipv6.c b/net/dccp/ipv6.c
index 5690fbd..cf598bf 100644
--- a/net/dccp/ipv6.c
+++ b/net/dccp/ipv6.c
@@ -1173,7 +1173,6 @@ static struct inet_protosw dccp_v6_protosw = {

 static int __init dccp_v6_init(void)
 {
- struct socket *socket;
  int err = proto_register(&dccp_v6_prot, 1);

  if (err != 0)
@@ -1185,11 +1184,10 @@ static int __init dccp_v6_init(void)

  inet6_register_protosw(&dccp_v6_protosw);

- err = inet_ctl_sock_create(&socket, PF_INET6,
+ err = inet_ctl_sock_create(&dccp_v6_ctl_sk, PF_INET6,
        SOCK_DCCP, IPPROTO_DCCP);
  if (err != 0)
   goto out_unregister_protosw;
- dccp_v6_ctl_sk = socket->sk;
 out:
  return err;
 out_unregister_protosw:
diff --git a/net/ipv4/af_inet.c b/net/ipv4/af_inet.c
index 7ab0bd6..cad664b 100644
--- a/net/ipv4/af_inet.c
+++ b/net/ipv4/af_inet.c
@@ -1250,19 +1250,21 @@ out:
  return segs;
 }

-int inet_ctl_sock_create(struct socket **sock, unsigned short family,
+int inet_ctl_sock_create(struct sock **sk, unsigned short family,
     unsigned short type, unsigned char protocol)
 {
- int rc = sock_create_kern(family, type, protocol, sock);
+ struct socket *sock;
+ int rc = sock_create_kern(family, type, protocol, &sock);
```

```
  if (rc == 0) {
-  (*sock)->sk->sk_allocation = GFP_ATOMIC;
-  inet_sk((*sock)->sk)->uc_ttl = -1;
+  *sk = sock->sk;
+  (*sk)->sk_allocation = GFP_ATOMIC;
+  inet_sk(*sk)->uc_ttl = -1;
   /*
    * Unhash it so that IP input processing does not even see it,
    * we do not wish this socket to see incoming packets.
    */
-  (*sock)->sk->sk_prot->unhash((*sock)->sk);
+  (*sk)->sk_prot->unhash(*sk);
  }
  return rc;
 }
diff --git a/net/ipv4/tcp_ipv4.c b/net/ipv4/tcp_ipv4.c
index edf5a37..cfe5df7 100644
--- a/net/ipv4/tcp_ipv4.c
+++ b/net/ipv4/tcp_ipv4.c
@@ -2490,11 +2490,9 @@ struct proto tcp_prot = {

 void __init tcp_v4_init(void)
 {
- struct socket *__tcp_socket;
- if (inet_ctl_sock_create(&__tcp_socket, PF_INET, SOCK_RAW,
+ if (inet_ctl_sock_create(&tcp_sock, PF_INET, SOCK_RAW,
      IPPROTO_TCP) < 0)
   panic("Failed to create the TCP control socket.\n");
- tcp_sock = __tcp_socket->sk;
 }

 EXPORT_SYMBOL(ipv4_specific);
diff --git a/net/ipv6/tcp_ipv6.c b/net/ipv6/tcp_ipv6.c
index d98222f..2882cc5 100644
--- a/net/ipv6/tcp_ipv6.c
+++ b/net/ipv6/tcp_ipv6.c
@@ -2200,14 +2200,13 @@ static struct inet_protosw tcpv6_protosw = {
 static int tcpv6_net_init(struct net *net)
 {
  int err;
- struct socket *sock;
  struct sock *sk;

- err = inet_ctl_sock_create(&sock, PF_INET6, SOCK_RAW, IPPROTO_TCP);
+ err = inet_ctl_sock_create(&sk, PF_INET6, SOCK_RAW, IPPROTO_TCP);
  if (err)
   return err;
```

```
- net->ipv6.tcp_sk = sk = sock->sk;
+ net->ipv6.tcp_sk = sk;
  sk_change_net(sk, net);
  return err;
 }
diff --git a/net/sctp/protocol.c b/net/sctp/protocol.c
index ac0833c..3c08d33 100644
--- a/net/sctp/protocol.c
+++ b/net/sctp/protocol.c
@@ -674,21 +674,19 @@ static int sctp_ctl_sock_init(void)
 {
  int err;
  sa_family_t family;
- struct socket *socket;

  if (sctp_get_pf_specific(PF_INET6))
   family = PF_INET6;
  else
   family = PF_INET;

- err = inet_ctl_sock_create(&socket, family,
+ err = inet_ctl_sock_create(&sctp_ctl_sock, family,
        SOCK_SEQPACKET, IPPROTO_SCTP);
  if (err < 0) {
   printk(KERN_ERR
          "SCTP: Failed to create the SCTP control socket.\n");
   return err;
  }
- sctp_ctl_sock = socket->sk;
  return 0;
 }

--
1.5.3.rc5
```

_____
Containers mailing list
Containers@lists.linux-foundation.org
https://lists.linux-foundation.org/mailman/listinfo/containers