
Subject: [PATCH 8/13 net-2.6.26] [NETNS]: Inet control socket should not hold a namespace.

Posted by [den](#) on Mon, 31 Mar 2008 13:16:48 GMT

[View Forum Message](#) <> [Reply to Message](#)

This is a generic requirement, so make inet_ctl_sock_create namespace aware and create a inet_ctl_sock_destroy wrapper around sk_release_kernel.

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

include/net/inet_common.h | 8 +++++++-

net/dccp/ipv4.c | 4 +-+-

net/dccp/ipv6.c | 4 +-+-

net/ipv4/af_inet.c | 5 +----

net/ipv4/tcp_ipv4.c | 2 +-

net/ipv6/tcp_ipv6.c | 14 +-----

net/sctp/protocol.c | 6 +----

7 files changed, 22 insertions(+), 21 deletions(-)

diff --git a/include/net/inet_common.h b/include/net/inet_common.h

index 4bfcf3f..18c7732 100644

--- a/include/net/inet_common.h

+++ b/include/net/inet_common.h

@@ -42,7 +42,13 @@ extern int inet_ioctl(struct socket *sock,
extern int inet_ctl_sock_create(struct sock **sk,

 unsigned short family,

 unsigned short type,

- unsigned char protocol);

+ unsigned char protocol,

+ struct net *net);

+

+static inline void inet_ctl_sock_destroy(struct sock *sk)

+{

+ sk_release_kernel(sk);

}

#endif

diff --git a/net/dccp/ipv4.c b/net/dccp/ipv4.c

index 5669c89..b12803b 100644

--- a/net/dccp/ipv4.c

+++ b/net/dccp/ipv4.c

@@ -1003,7 +1003,7 @@ static int __init dccp_v4_init(void)

 inet_register_protosw(&dccp_v4_protosw);

 err = inet_ctl_sock_create(&dccp_v4_ctl_sk, PF_INET,

- SOCK_DCCP, IPPROTO_DCCP);

+ SOCK_DCCP, IPPROTO_DCCP, &init_net);

```

if (err)
    goto out_unregister_protosw;
out:
@@ -1018,7 +1018,7 @@ out_proto_unregister:

static void __exit dccp_v4_exit(void)
{
- sock_release(dccp_v4_ctl_sk->sk_socket);
+ inet_ctl_sock_destroy(dccp_v4_ctl_sk);
inet_unregister_protosw(&dccp_v4_protosw);
inet_del_protocol(&dccp_v4_protocol, IPPROTO_DCCP);
proto_unregister(&dccp_v4_prot);
diff --git a/net/dccp/ipv6.c b/net/dccp/ipv6.c
index cf598bf..94d749e 100644
--- a/net/dccp/ipv6.c
+++ b/net/dccp/ipv6.c
@@ -1185,7 +1185,7 @@ static int __init dccp_v6_init(void)
inet6_register_protosw(&dccp_v6_protosw);

err = inet_ctl_sock_create(&dccp_v6_ctl_sk, PF_INET6,
-    SOCK_DCCP, IPPROTO_DCCP);
+    SOCK_DCCP, IPPROTO_DCCP, &init_net);
if (err != 0)
    goto out_unregister_protosw;
out:
@@ -1200,7 +1200,7 @@ out_unregister_proto:

static void __exit dccp_v6_exit(void)
{
- sock_release(dccp_v6_ctl_sk->sk_socket);
+ inet_ctl_sock_destroy(dccp_v6_ctl_sk);
inet6_del_protocol(&dccp_v6_protocol, IPPROTO_DCCP);
inet6_unregister_protosw(&dccp_v6_protosw);
proto_unregister(&dccp_v6_prot);
diff --git a/net/ipv4/af_inet.c b/net/ipv4/af_inet.c
index cad664b..cf766ad 100644
--- a/net/ipv4/af_inet.c
+++ b/net/ipv4/af_inet.c
@@ -1251,7 +1251,8 @@ out:
}

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

```

```

@@ -1265,6 +1266,8 @@ int inet_ctl_sock_create(struct sock **sk, unsigned short family,
 * we do not wish this socket to see incoming packets.
 */
(*sk)->sk_prot->unhash(*sk);
+
+ sk_change_net(*sk, net);
}
return rc;
}

diff --git a/net/ipv4/tcp_ipv4.c b/net/ipv4/tcp_ipv4.c
index cfe5df7..dc8c3dc 100644
--- a/net/ipv4/tcp_ipv4.c
+++ b/net/ipv4/tcp_ipv4.c
@@ -2491,7 +2491,7 @@ struct proto tcp_prot = {
void __init tcp_v4_init(void)
{
if (inet_ctl_sock_create(&tcp_sock, PF_INET, SOCK_RAW,
- IPPROTO_TCP) < 0)
+ IPPROTO_TCP, &init_net) < 0)
panic("Failed to create the TCP control socket.\n");
}

diff --git a/net/ipv6/tcp_ipv6.c b/net/ipv6/tcp_ipv6.c
index 2882cc5..378cc40 100644
--- a/net/ipv6/tcp_ipv6.c
+++ b/net/ipv6/tcp_ipv6.c
@@ -2199,21 +2199,13 @@ static struct inet_protosw tcpv6_protosw = {

static int tcpv6_net_init(struct net *net)
{
- int err;
- struct sock *sk;
-
- err = inet_ctl_sock_create(&sk, PF_INET6, SOCK_RAW, IPPROTO_TCP);
- if (err)
- return err;
-
- net->ipv6.tcp_sk = sk;
- sk_change_net(sk, net);
- return err;
+ return inet_ctl_sock_create(&net->ipv6.tcp_sk, PF_INET6,
+ SOCK_RAW, IPPROTO_TCP, net);
}

static void tcpv6_net_exit(struct net *net)
{
- sk_release_kernel(net->ipv6.tcp_sk);
+ inet_ctl_sock_destroy(net->ipv6.tcp_sk);

```

```

}

static struct pernet_operations tcpv6_net_ops = {
diff --git a/net/sctp/protocol.c b/net/sctp/protocol.c
index 3c08d33..067c8a1 100644
--- a/net/sctp/protocol.c
+++ b/net/sctp/protocol.c
@@ -681,7 +681,7 @@ static int sctp_ctl_sock_init(void)
    family = PF_INET;

    err = inet_ctl_sock_create(&sctp_ctl_sock, family,
-     SOCK_SEQPACKET, IPPROTO_SCTP);
+     SOCK_SEQPACKET, IPPROTO_SCTP, &init_net);
    if (err < 0) {
        printk(KERN_ERR
              "SCTP: Failed to create the SCTP control socket.\n");
@@ -1284,7 +1284,7 @@ err_v6_add_protocol:
    sctp_v6_del_protocol();
err_add_protocol:
    sctp_v4_del_protocol();
- sock_release(sctp_ctl_sock->sk_socket);
+ inet_ctl_sock_destroy(sctp_ctl_sock);
err_ctl_sock_init:
    sctp_v6_protosw_exit();
err_v6_protosw_init:
@@ -1328,7 +1328,7 @@ SCTP_STATIC __exit void sctp_exit(void)
    sctp_v4_del_protocol();

/* Free the control endpoint. */
- sock_release(sctp_ctl_sock->sk_socket);
+ inet_ctl_sock_destroy(sctp_ctl_sock);

/* Free protosw registrations */
    sctp_v6_protosw_exit();
--
```

1.5.3.rc5

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