
Subject: [PATCH 4/13 net-2.6.26] [INET]: Rename inet_csk_ctl_sock_create to inet_ctl_sock_create.

Posted by [den](#) on Mon, 31 Mar 2008 13:16:44 GMT

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

This call is nothing common with INET connection sockets code. It simply creates an unhashes kernel sockets for protocol messages.

Move the new call into af_inet.c after the rename.

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

```
include/net/inet_common.h      | 5 +++++
include/net/inet_connection_sock.h | 5 -----
net/dccp/ipv4.c                | 4 ++--
net/dccp/ipv6.c                | 4 ++--
net/ipv4/af_inet.c             | 19 +++++++++++++++++++++
net/ipv4/inet_connection_sock.c | 19 -----
net/ipv4/tcp_ipv4.c            | 4 ++--
net/ipv6/tcp_ipv6.c            | 3 ++-
8 files changed, 32 insertions(+), 31 deletions(-)
```

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

index 38d5a1e..d6238bd 100644

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

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

```
@@ -39,6 +39,11 @@ 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,
+    unsigned short family,
+    unsigned short type,
+    unsigned char protocol);
+
#endif
```

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

index f00f057..2ff545a 100644

--- a/include/net/inet_connection_sock.h

+++ b/include/net/inet_connection_sock.h

```
@@ -327,11 +327,6 @@ extern void inet_csk_listen_stop(struct sock *sk);
```

```
extern void inet_csk_addr2sockaddr(struct sock *sk, struct sockaddr *uaddr);
```

```
-extern int inet_csk_ctl_sock_create(struct socket **sock,
-    unsigned short family,
```

```

- unsigned short type,
- unsigned char protocol);
-
extern int inet_csk_compat_getsockopt(struct sock *sk, int level, int optname,
    char __user *optval, int __user *optlen);
extern int inet_csk_compat_setsockopt(struct sock *sk, int level, int optname,
diff --git a/net/dccp/ipv4.c b/net/dccp/ipv4.c
index 6d8f684..feb3fa5 100644
--- a/net/dccp/ipv4.c
+++ b/net/dccp/ipv4.c
@@ -1003,8 +1003,8 @@ static int __init dccp_v4_init(void)

    inet_register_protosw(&dccp_v4_protosw);

- err = inet_csk_ctl_sock_create(&socket, PF_INET,
-     SOCK_DCCP, IPPROTO_DCCP);
+ err = inet_ctl_sock_create(&socket, PF_INET,
+     SOCK_DCCP, IPPROTO_DCCP);
    if (err)
        goto out_unregister_protosw;
    dccp_v4_ctl_sk = socket->sk;
diff --git a/net/dccp/ipv6.c b/net/dccp/ipv6.c
index c5d9d1b..5690fbd 100644
--- a/net/dccp/ipv6.c
+++ b/net/dccp/ipv6.c
@@ -1185,8 +1185,8 @@ static int __init dccp_v6_init(void)

    inet6_register_protosw(&dccp_v6_protosw);

- err = inet_csk_ctl_sock_create(&socket, PF_INET6,
-     SOCK_DCCP, IPPROTO_DCCP);
+ err = inet_ctl_sock_create(&socket, PF_INET6,
+     SOCK_DCCP, IPPROTO_DCCP);
    if (err != 0)
        goto out_unregister_protosw;
    dccp_v6_ctl_sk = socket->sk;
diff --git a/net/ipv4/af_inet.c b/net/ipv4/af_inet.c
index 5882a13..7ab0bd6 100644
--- a/net/ipv4/af_inet.c
+++ b/net/ipv4/af_inet.c
@@ -1250,6 +1250,25 @@ out:
    return segs;
}

+int inet_ctl_sock_create(struct socket **sock, unsigned short family,
+    unsigned short type, unsigned char protocol)
+{
+    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;
+ /*
+  * 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);
+ }
+ return rc;
+}
+
+EXPORT_SYMBOL_GPL(inet_ctl_sock_create);
+
unsigned long snmp_fold_field(void *mib[], int offt)
{
    unsigned long res = 0;
diff --git a/net/ipv4/inet_connection_sock.c b/net/ipv4/inet_connection_sock.c
index a7fcaf2..ee55678 100644
--- a/net/ipv4/inet_connection_sock.c
+++ b/net/ipv4/inet_connection_sock.c
@@ -651,25 +651,6 @@ void inet_csk_addr2sockaddr(struct sock *sk, struct sockaddr *uaddr)

EXPORT_SYMBOL_GPL(inet_csk_addr2sockaddr);

-int inet_csk_ctl_sock_create(struct socket **sock, unsigned short family,
-    unsigned short type, unsigned char protocol)
-{
- 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;
- /*
-  * 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);
- }
- return rc;
-}
-
-EXPORT_SYMBOL_GPL(inet_csk_ctl_sock_create);
-
#ifdef CONFIG_COMPAT
int inet_csk_compat_getsockopt(struct sock *sk, int level, int optname,
    char __user *optval, int __user *optlen)

```

```

diff --git a/net/ipv4/tcp_ipv4.c b/net/ipv4/tcp_ipv4.c
index 1d77f37..edf5a37 100644
--- a/net/ipv4/tcp_ipv4.c
+++ b/net/ipv4/tcp_ipv4.c
@@ -2491,8 +2491,8 @@ struct proto tcp_prot = {
void __init tcp_v4_init(void)
{
    struct socket *__tcp_socket;
- if (inet_csk_ctl_sock_create(&__tcp_socket, PF_INET, SOCK_RAW,
-     IPPROTO_TCP) < 0)
+ if (inet_ctl_sock_create(&__tcp_socket, PF_INET, SOCK_RAW,
+     IPPROTO_TCP) < 0)
    panic("Failed to create the TCP control socket.\n");
    tcp_sock = __tcp_socket->sk;
}
diff --git a/net/ipv6/tcp_ipv6.c b/net/ipv6/tcp_ipv6.c
index 6d851c3..d98222f 100644
--- a/net/ipv6/tcp_ipv6.c
+++ b/net/ipv6/tcp_ipv6.c
@@ -60,6 +60,7 @@
#include <net/dsfield.h>
#include <net/timewait_sock.h>
#include <net/netdma.h>
+#include <net/inet_common.h>

#include <asm/uaccess.h>

@@ -2202,7 +2203,7 @@ static int tcpv6_net_init(struct net *net)
    struct socket *sock;
    struct sock *sk;

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

--
1.5.3.rc5

```

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