
Subject: [PATCH 2/13 net-2.6.26] [DCCP]: Replace socket with sock for reset sending.

Posted by [den](#) on Mon, 31 Mar 2008 13:16:42 GMT

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

Replace dccp_v(4|6)_ctl_socket with sock to unify a code with TCP/ICMP.

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

```
net/dccp/dccp.h | 2 +-
net/dccp/ipv4.c | 16 ++++++++-----
net/dccp/ipv6.c | 10 ++++++----
net/dccp/output.c | 6 +++--
4 files changed, 19 insertions(+), 15 deletions(-)
```

```
diff --git a/net/dccp/dccp.h b/net/dccp/dccp.h
```

```
index e1b7c9c..fe7726b 100644
```

```
--- a/net/dccp/dccp.h
```

```
+++ b/net/dccp/dccp.h
```

```
@@ -296,7 +296,7 @@ extern unsigned int dccp_poll(struct file *file, struct socket *sock,
extern int dccp_v4_connect(struct sock *sk, struct sockaddr *uaddr,
int addr_len);
```

```
-extern struct sk_buff *dccp_ctl_make_reset(struct socket *ctl,
```

```
+extern struct sk_buff *dccp_ctl_make_reset(struct sock *sk,
struct sk_buff *skb);
```

```
extern int dccp_send_reset(struct sock *sk, enum dccp_reset_codes code);
```

```
extern void dccp_send_close(struct sock *sk, const int active);
```

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

```
index 7d62f7e..f97049b 100644
```

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

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

```
@@ -36,7 +36,7 @@
```

```
* the Out-of-the-blue (OOTB) packets. A control sock will be created
```

```
* for this socket at the initialization time.
```

```
*/
```

```
-static struct socket *dccp_v4_ctl_socket;
```

```
+static struct sock *dccp_v4_ctl_sk;
```

```
int dccp_v4_connect(struct sock *sk, struct sockaddr *uaddr, int addr_len)
```

```
{
@@ -514,11 +514,11 @@ static void dccp_v4_ctl_send_reset(struct sock *sk, struct sk_buff
*rxskb)
if (rxskb->rtable->rt_type != RTN_LOCAL)
return;
```

```
- dst = dccp_v4_route_skb(dccp_v4_ctl_socket->sk, rxskb);
```

```
+ dst = dccp_v4_route_skb(dccp_v4_ctl_sk, rxskb);
```

```

if (dst == NULL)
    return;

- skb = dccp_ctl_make_reset(dccp_v4_ctl_socket, rxskb);
+ skb = dccp_ctl_make_reset(dccp_v4_ctl_sk, rxskb);
    if (skb == NULL)
        goto out;

@@ -527,10 +527,10 @@ static void dccp_v4_ctl_send_reset(struct sock *sk, struct sk_buff
*rxskb)
    rxiph->daddr;
    skb->dst = dst_clone(dst);

- bh_lock_sock(dccp_v4_ctl_socket->sk);
- err = ip_build_and_send_pkt(skb, dccp_v4_ctl_socket->sk,
+ bh_lock_sock(dccp_v4_ctl_sk);
+ err = ip_build_and_send_pkt(skb, dccp_v4_ctl_sk,
    rxiph->daddr, rxiph->saddr, NULL);
- bh_unlock_sock(dccp_v4_ctl_socket->sk);
+ bh_unlock_sock(dccp_v4_ctl_sk);

    if (net_xmit_eval(err) == 0) {
        DCCP_INC_STATS_BH(DCCP_MIB_OUTSEGS);
@@ -991,6 +991,7 @@ 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)
@@ -1002,10 +1003,11 @@ static int __init dccp_v4_init(void)

    inet_register_protosw(&dccp_v4_protosw);

- err = inet_csk_ctl_sock_create(&dccp_v4_ctl_socket, PF_INET,
+ err = inet_csk_ctl_sock_create(&socket, 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 ea3f326..44e8b33 100644
--- a/net/dccp/ipv6.c
+++ b/net/dccp/ipv6.c

```

```
@@ -34,7 +34,7 @@
#include "feat.h"
```

```
/* Socket used for sending RSTs and ACKs */
```

```
-static struct socket *dccp_v6_ctl_socket;
```

```
+static struct sock *dccp_v6_ctl_sk;
```

```
static struct inet_connection_sock_af_ops dccp_ipv6_mapped;
```

```
static struct inet_connection_sock_af_ops dccp_ipv6_af_ops;
```

```
@@ -303,7 +303,7 @@ static void dccp_v6_ctl_send_reset(struct sock *sk, struct sk_buff *rxskb)
    if (!ipv6_unicast_destination(rxskb))
        return;
```

```
- skb = dccp_ctl_make_reset(dccp_v6_ctl_socket, rxskb);
```

```
+ skb = dccp_ctl_make_reset(dccp_v6_ctl_sk, rxskb);
```

```
    if (skb == NULL)
```

```
        return;
```

```
@@ -324,7 +324,7 @@ static void dccp_v6_ctl_send_reset(struct sock *sk, struct sk_buff *rxskb)
    /* sk = NULL, but it is safe for now. RST socket required. */
```

```
    if (!ip6_dst_lookup(NULL, &skb->dst, &fl)) {
```

```
        if (xfrm_lookup(&skb->dst, &fl, NULL, 0) >= 0) {
```

```
-    ip6_xmit(dccp_v6_ctl_socket->sk, skb, &fl, NULL, 0);
```

```
+    ip6_xmit(dccp_v6_ctl_sk, skb, &fl, NULL, 0);
```

```
        DCCP_INC_STATS_BH(DCCP_MIB_OUTSEGS);
```

```
        DCCP_INC_STATS_BH(DCCP_MIB_OUTRSTS);
```

```
        return;
```

```
@@ -1173,6 +1173,7 @@ 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)
```

```
@@ -1184,10 +1185,11 @@ static int __init dccp_v6_init(void)
```

```
    inet6_register_protosw(&dccp_v6_protosw);
```

```
- err = inet_csk_ctl_sock_create(&dccp_v6_ctl_socket, PF_INET6,
```

```
+ err = inet_csk_ctl_sock_create(&socket, 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/dccp/output.c b/net/dccp/output.c
index 3b763db..f32a84e 100644
--- a/net/dccp/output.c
+++ b/net/dccp/output.c
@@ -348,7 +348,7 @@ struct sk_buff *dccp_make_response(struct sock *sk, struct dst_entry
 *dst,
EXPORT_SYMBOL_GPL(dccp_make_response);

/* answer offending packet in @rcv_skb with Reset from control socket @ctl */
-struct sk_buff *dccp_ctl_make_reset(struct socket *ctl, struct sk_buff *rcv_skb)
+struct sk_buff *dccp_ctl_make_reset(struct sock *sk, struct sk_buff *rcv_skb)
{
    struct dccp_hdr *rxdh = dccp_hdr(rcv_skb), *dh;
    struct dccp_skb_cb *dcb = DCCP_SKB_CB(rcv_skb);
@@ -358,11 +358,11 @@ struct sk_buff *dccp_ctl_make_reset(struct socket *ctl, struct sk_buff
*rcv_skb)
    struct dccp_hdr_reset *dhr;
    struct sk_buff *skb;

- skb = alloc_skb(ctl->sk->sk_prot->max_header, GFP_ATOMIC);
+ skb = alloc_skb(sk->sk_prot->max_header, GFP_ATOMIC);
    if (skb == NULL)
        return NULL;

- skb_reserve(skb, ctl->sk->sk_prot->max_header);
+ skb_reserve(skb, sk->sk_prot->max_header);

    /* Swap the send and the receive. */
    dh = dccp_zeroed_hdr(skb, dccp_hdr_reset_len);
--
1.5.3.rc5

```

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