Subject: Re: [NETLINK] Don't attach callback to a going-away netlink socket
Posted by Herbert Xu on Thu, 19 Apr 2007 02:13:51 GMT
View Forum Message <> Reply to Message

David Miller <davem@davemloft.net> wrote:
>
> As discussed in this thread there might be other ways to a
> approach this, but this fix is good for now.
>
> Patch applied, thank you.

Actually I was going to suggest something like this:

[NETLINK]: Kill CB only when socket is unused

Since we can still receive packets until all references to the
socket are gone, we don't need to kill the CB until that happens.
This also aligns ourselves with the receive queue purging which
happens at that point.

Original patch by Pavel Emelianov who noticed this race condition.

Signed-off-by: Herbert Xu <herbert@gondor.apana.org.au>

Cheers,
--
Visit Openswan at http://www.openswan.org/
Email: Herbert Xu ~{PmV>HI~} <herbert@gondor.apana.org.au>
Home Page: http://gondor.apana.org.au/~herbert/
PGP Key: http://gondor.apana.org.au/~herbert/pubkey.txt
--
diff --git a/net/netlink/af_netlink.c b/net/netlink/af_netlink.c
index 0be19b7..914884c 100644
--- a/net/netlink/af_netlink.c
+++ b/net/netlink/af_netlink.c
@@ -139,6 +139,15 @@ static struct hlist_head *nl_pid_hashfn(struct nl_pid_hash *hash, u32 pid)

 static void netlink_sock_destruct(struct sock *sk)
 {
+ struct netlink_sock *nlk = nlk_sk(sk);
+
+ WARN_ON(mutex_is_locked(nlk_sk(sk)->cb_mutex));
+ if (nlk->cb) {
+  if (nlk->cb->done)
+   nlk->cb->done(nlk->cb);
+  netlink_destroy_callback(nlk->cb);
+ }

```
+
  skb_queue_purge(&sk->sk_receive_queue);

  if (!sock_flag(sk, SOCK_DEAD)) {
@@ -147,7 +156,6 @@ static void netlink_sock_destruct(struct sock *sk)
  }
  BUG_TRAP(!atomic_read(&sk->sk_rmem_alloc));
  BUG_TRAP(!atomic_read(&sk->sk_wmem_alloc));
- BUG_TRAP(!nlk_sk(sk)->cb);
  BUG_TRAP(!nlk_sk(sk)->groups);
 }

@@ -450,17 +458,7 @@ static int netlink_release(struct socket *sock)
  netlink_remove(sk);
  nlk = nlk_sk(sk);

- mutex_lock(nlk->cb_mutex);
- if (nlk->cb) {
-  if (nlk->cb->done)
-   nlk->cb->done(nlk->cb);
-  netlink_destroy_callback(nlk->cb);
-  nlk->cb = NULL;
- }
- mutex_unlock(nlk->cb_mutex);
-
- /* OK. Socket is unlinked, and, therefore,
-    no new packets will arrive */
+ /* OK. Socket is unlinked. */

  sock_orphan(sk);
  sock->sk = NULL;
```