Subject: Re: [NETLINK] Don't attach callback to a going-away netlink socket
Posted by Herbert Xu on Wed, 02 May 2007 04:12:22 GMT
View Forum Message <> Reply to Message

On Sat, Apr 28, 2007 at 11:18:49PM -0700, David Miller wrote:
>
> Herbert, could you refresh this refinement to the current
> tree?

Dave, thanks for reminding me. Here it is.

[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.

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 42d2fb9..7fc6b4d 100644
--- a/net/netlink/af_netlink.c
+++ b/net/netlink/af_netlink.c
@@ -140,6 +140,15 @@ 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);
+
+ BUG_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)) {

```
@@ -148,7 +157,6 @@ 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);
 }

@@ -456,17 +464,10 @@ static int netlink_release(struct socket *sock)
  sock_orphan(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, any packets that arrive now
+  * will be purged.
+  */

  sock->sk = NULL;
  wake_up_interruptible_all(&nlk->wait);
@@ -1426,9 +1427,9 @@ int netlink_dump_start(struct sock *ssk, struct sk_buff *skb,
   return -ECONNREFUSED;
  }
  nlk = nlk_sk(sk);
- /* A dump or destruction is in progress... */
+ /* A dump is in progress... */
  mutex_lock(nlk->cb_mutex);
- if (nlk->cb || sock_flag(sk, SOCK_DEAD)) {
+ if (nlk->cb) {
   mutex_unlock(nlk->cb_mutex);
   netlink_destroy_callback(cb);
   sock_put(sk);
```