
Subject: [PATCH 5/9] network namespaces: async socket operations
Posted by [Andrey Savochkin](#) on Tue, 15 Aug 2006 14:48:45 GMT
[View Forum Message](#) <> [Reply to Message](#)

Non-trivial part of socket namespaces: asynchronous events
should be run in proper context.

Signed-off-by: Andrey Savochkin <saw@swsoft.com>

```
af_inet.c          | 10 ++++++++
inet_timewait_sock.c | 8 +++++
tcp_timer.c        | 9 ++++++
3 files changed, 27 insertions(+)
```

--- ./net/ipv4/af_inet.c.venussock-asyn Mon Aug 14 17:04:07 2006

+++ ./net/ipv4/af_inet.c Tue Aug 15 13:45:44 2006

@@ -366,10 +366,17 @@ out_rcu_unlock:

int inet_release(struct socket *sock)

```
{
    struct sock *sk = sock->sk;
+ struct net_namespace *ns, *orig_net_ns;
```

```
    if (sk) {
        long timeout;
```

```
+ /* Need to change context here since protocol ->close
```

```
+ * operation may send packets.
```

```
+ */
```

```
+ ns = get_net_ns(sk->sk_net_ns);
```

```
+ push_net_ns(ns, orig_net_ns);
```

```
+
```

```
    /* Applications forget to leave groups before exiting */
```

```
    ip_mc_drop_socket(sk);
```

@@ -386,6 +393,9 @@ int inet_release(struct socket *sock)

```
    timeout = sk->sk_lingertime;
```

```
    sock->sk = NULL;
```

```
    sk->sk_prot->close(sk, timeout);
```

```
+
```

```
+ pop_net_ns(orig_net_ns);
```

```
+ put_net_ns(ns);
```

```
}
```

```
    return 0;
```

```
}
```

--- ./net/ipv4/inet_timewait_sock.c.venussock-asyn Tue Aug 15 13:45:44 2006

+++ ./net/ipv4/inet_timewait_sock.c Tue Aug 15 13:45:44 2006

@@ -129,6 +129,7 @@ static int inet_twdr_do_twkill_work(stru

```
{
```

```

    struct inet_timewait_sock *tw;
    struct hlist_node *node;
+ struct net_namespace *orig_net_ns;
    unsigned int killed;
    int ret;

@@ -140,8 +141,10 @@ static int inet_twdr_do_twkill_work(stru
    */
    killed = 0;
    ret = 0;
+ push_net_ns(current_net_ns, orig_net_ns);
    rescan:
    inet_twsd_for_each_inmate(tw, node, &twdr->cells[slot]) {
+ switch_net_ns(tw->tw_net_ns);
    __inet_twsd_del_dead_node(tw);
    spin_unlock(&twdr->death_lock);
    __inet_twsd_kill(tw, twdr->hashinfo);
@@ -164,6 +167,7 @@ rescan:

    twdr->tw_count -= killed;
    NET_ADD_STATS_BH(LINUX_MIB_TIMEWAITED, killed);
+ pop_net_ns(orig_net_ns);

    return ret;
}
@@ -338,10 +342,12 @@ void inet_twdr_twcal_tick(unsigned long
    int n, slot;
    unsigned long j;
    unsigned long now = jiffies;
+ struct net_namespace *orig_net_ns;
    int killed = 0;
    int adv = 0;

    twdr = (struct inet_timewait_death_row *)data;
+ push_net_ns(current_net_ns, orig_net_ns);

    spin_lock(&twdr->death_lock);
    if (twdr->twcal_hand < 0)
@@ -357,6 +363,7 @@ void inet_twdr_twcal_tick(unsigned long

    inet_twsd_for_each_inmate_safe(tw, node, safe,
        &twdr->twcal_row[slot]) {
+ switch_net_ns(tw->tw_net_ns);
    __inet_twsd_del_dead_node(tw);
    __inet_twsd_kill(tw, twdr->hashinfo);
    inet_twsd_put(tw);
@@ -384,6 +391,7 @@ out:
    del_timer(&twdr->tw_timer);

```

```

NET_ADD_STATS_BH(LINUX_MIB_TIMEWAITKILLED, killed);
spin_unlock(&twdr->death_lock);
+ pop_net_ns(orig_net_ns);
}

EXPORT_SYMBOL_GPL(inet_twdr_twcal_tick);
--- ./net/ipv4/tcp_timer.c.vensock-asyn Mon Aug 14 16:43:51 2006
+++ ./net/ipv4/tcp_timer.c Tue Aug 15 13:45:44 2006
@@ -171,7 +171,9 @@ static void tcp_delack_timer(unsigned lo
    struct sock *sk = (struct sock*)data;
    struct tcp_sock *tp = tcp_sk(sk);
    struct inet_connection_sock *icsk = inet_csk(sk);
+ struct net_namespace *orig_net_ns;

+ push_net_ns(sk->sk_net_ns, orig_net_ns);
    bh_lock_sock(sk);
    if (sock_owned_by_user(sk)) {
        /* Try again later. */
@@ -225,6 +227,7 @@ out:
out_unlock:
    bh_unlock_sock(sk);
    sock_put(sk);
+ pop_net_ns(orig_net_ns);
}

static void tcp_probe_timer(struct sock *sk)
@@ -384,8 +387,10 @@ static void tcp_write_timer(unsigned lon
{
    struct sock *sk = (struct sock*)data;
    struct inet_connection_sock *icsk = inet_csk(sk);
+ struct net_namespace *orig_net_ns;
    int event;

+ push_net_ns(sk->sk_net_ns, orig_net_ns);
    bh_lock_sock(sk);
    if (sock_owned_by_user(sk)) {
        /* Try again later */
@@ -419,6 +424,7 @@ out:
out_unlock:
    bh_unlock_sock(sk);
    sock_put(sk);
+ pop_net_ns(orig_net_ns);
}

/*
@@ -447,9 +453,11 @@ static void tcp_keepalive_timer (unsigne
{
    struct sock *sk = (struct sock *) data;

```

```
    struct inet_connection_sock *icsk = inet_csk(sk);
+ struct net_namespace *orig_net_ns;
    struct tcp_sock *tp = tcp_sk(sk);
    __u32 elapsed;

+ push_net_ns(sk->sk_net_ns, orig_net_ns);
    /* Only process if socket is not in use. */
    bh_lock_sock(sk);
    if (sock_owned_by_user(sk)) {
@@ -521,4 +529,5 @@ death:
    out:
        bh_unlock_sock(sk);
        sock_put(sk);
+ put_net_ns(orig_net_ns);
    }
```
