Subject: [patch 2/3][netns] make timewait unhash lock free
Posted by Daniel Lezcano on Mon, 24 Sep 2007 13:29:37 GMT
View Forum Message <> Reply to Message

From: Daniel Lezcano <dlezcano@fr.ibm.com>

The network namespace cleanup will remove all timewait sockets
related to it because there are pointless.

The problem is we need to browse the established hash table and
for that we need to take the lock. For each timesocket we call
inet_deschedule and this one take the established hash table lock
too.

The following patchset split the removing of the established hash
into two parts, one removing the node from the hash and another
taking the lock and calling the first one.

The network namespace cleanup can be done calling the lock free
function.

Signed-off-by: Daniel Lezcano <dlezcano@fr.ibm.com>

```
---
 include/net/inet_timewait_sock.h |   13 ++++++++++++
 net/ipv4/inet_timewait_sock.c    |   40 +++++++++++++++++++++++++++------------
 2 files changed, 41 insertions(+), 12 deletions(-)

Index: linux-2.6-netns/net/ipv4/inet_timewait_sock.c
===================================================================
--- linux-2.6-netns.orig/net/ipv4/inet_timewait_sock.c
+++ linux-2.6-netns/net/ipv4/inet_timewait_sock.c
@@ -13,25 +13,28 @@
 #include <net/inet_timewait_sock.h>
 #include <net/ip.h>

-/* Must be called with locally disabled BHs. */
-static void __inet_twsk_kill(struct inet_timewait_sock *tw,
-			     struct inet_hashinfo *hashinfo)
+static inline int inet_twsk_unehash(struct inet_timewait_sock *tw,
+				     struct inet_hashinfo *hashinfo)
 {
-	struct inet_bind_hashbucket *bhead;
-	struct inet_bind_bucket *tb;
-	/* Unlink from established hashes. */
-	struct inet_ehash_bucket *ehead = inet_ehash_bucket(hashinfo, tw->tw_hash);
+	struct inet_ehash_bucket *ehead =
+		inet_ehash_bucket(hashinfo, tw->tw_hash);
```

```
  write_lock(&ehead->lock);
- if (hlist_unhashed(&tw->tw_node)) {
+ if (__inet_twsk_unehash(tw)) {
   write_unlock(&ehead->lock);
-  return;
+  return 1;
  }
- __hlist_del(&tw->tw_node);
- sk_node_init(&tw->tw_node);
  write_unlock(&ehead->lock);

- /* Disassociate with bind bucket. */
+ return 0;
+}
+
+void inet_twsk_unbhash(struct inet_timewait_sock *tw,
+         struct inet_hashinfo *hashinfo)
+{
+ struct inet_bind_hashbucket *bhead;
+ struct inet_bind_bucket *tb;
+
  bhead = &hashinfo->bhash[inet_bhashfn(tw->tw_net, tw->tw_num, hashinfo->bhash_size)];
  spin_lock(&bhead->lock);
  tb = tw->tw_tb;
@@ -39,6 +42,19 @@ static void __inet_twsk_kill(struct inet
  tw->tw_tb = NULL;
  inet_bind_bucket_destroy(hashinfo->bind_bucket_cachep, tb);
  spin_unlock(&bhead->lock);
+}
+
+/* Must be called with locally disabled BHs. */
+static void __inet_twsk_kill(struct inet_timewait_sock *tw,
+        struct inet_hashinfo *hashinfo)
+{
+ /* Unlink from established hashes. */
+ if (inet_twsk_unehash(tw, hashinfo))
+  return;
+
+ /* Disassociate with bind bucket. */
+ inet_twsk_unbhash(tw, hashinfo);
+
 #ifdef SOCK_REFCNT_DEBUG
  if (atomic_read(&tw->tw_refcnt) != 1) {
   printk(KERN_DEBUG "%s timewait_sock %p refcnt=%d\n",
Index: linux-2.6-netns/include/net/inet_timewait_sock.h
===================================================================
--- linux-2.6-netns.orig/include/net/inet_timewait_sock.h
+++ linux-2.6-netns/include/net/inet_timewait_sock.h
```

```
@@ -173,6 +173,19 @@ static inline int inet_twsk_del_dead_nod
  return 0;
 }

+static inline int __inet_twsk_unehash(struct inet_timewait_sock *tw)
+{
+ if (hlist_unhashed(&tw->tw_node))
+  return 1;
+ __hlist_del(&tw->tw_node);
+ sk_node_init(&tw->tw_node);
+
+ return 0;
+}
+
+extern void inet_twsk_unbhash(struct inet_timewait_sock *tw,
+         struct inet_hashinfo *hashinfo);
+
 #define inet_twsk_for_each(tw, node, head) \
  hlist_for_each_entry(tw, node, head, tw_node)


--
```

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