Subject: [PATCH] netns: Simplify the network namespace list locking rules.
Posted by ebiederm on Thu, 27 Sep 2007 03:54:47 GMT
View Forum Message <> Reply to Message

Denis V. Lunev <den@sw.ru> noticed that the locking rules
for the network namespace list are over complicated and broken.

In particular the current register_netdev_notifier currently
does not take any lock making the for_each_net iteration racy
with network namespace creation and destruction. Oops.

The fact that we need to use for_each_net in rtnl_unlock() when
the rtnetlink support becomes per network namespace makes designing
the proper locking tricky. In addition we need to be able to call
rtnl_lock() and rtnl_unlock() when we have the net_mutex held.

After thinking about it and looking at the alternatives carefully
it looks like the simplest and most maintainable solution is
to remove net_list_mutex altogether, and to use the rtnl_mutex instead.

Signed-off-by: Eric W. Biederman <ebiederm@xmission.com>
---
 include/net/net_namespace.h | 3 ---
 net/core/net_namespace.c | 23 ++++++----------
 2 files changed, 6 insertions(+), 20 deletions(-)

diff --git a/include/net/net_namespace.h b/include/net/net_namespace.h
index 13b0e3b..934c840 100644
--- a/include/net/net_namespace.h
+++ b/include/net/net_namespace.h
@@ -96,9 +96,6 @@ static inline void release_net(struct net *net)
 #endif
 }

-extern void net_lock(void);
-extern void net_unlock(void);
-
 #define for_each_net(VAR) \
  list_for_each_entry(VAR, &net_namespace_list, list)

diff --git a/net/core/net_namespace.c b/net/core/net_namespace.c
index e478e35..0e0ca6d 100644
--- a/net/core/net_namespace.c
+++ b/net/core/net_namespace.c
@@ -15,7 +15,6 @@ static LIST_HEAD(pernet_list);
 static struct list_head *first_device = &pernet_list;
 static DEFINE_MUTEX(net_mutex);

```
-static DEFINE_MUTEX(net_list_mutex);
 LIST_HEAD(net_namespace_list);

 static struct kmem_cache *net_cachep;
@@ -23,16 +22,6 @@ static struct kmem_cache *net_cachep;
 struct net init_net;
 EXPORT_SYMBOL_GPL(init_net);

-void net_lock(void)
-{
- mutex_lock(&net_list_mutex);
-}
-
-void net_unlock(void)
-{
- mutex_unlock(&net_list_mutex);
-}
-
 static struct net *net_alloc(void)
 {
  return kmem_cache_alloc(net_cachep, GFP_KERNEL);
@@ -62,9 +51,9 @@ static void cleanup_net(struct work_struct *work)
  mutex_lock(&net_mutex);

  /* Don't let anyone else find us. */
- net_lock();
+ rtnl_lock();
  list_del(&net->list);
- net_unlock();
+ rtnl_unlock();

  /* Run all of the network namespace exit methods */
  list_for_each_entry_reverse(ops, &pernet_list, list) {
@@ -151,9 +140,9 @@ struct net *copy_net_ns(unsigned long flags, struct net *old_net)
  if (err)
   goto out_unlock;

- net_lock();
+ rtnl_lock();
  list_add_tail(&new_net->list, &net_namespace_list);
- net_unlock();
+ rtnl_unlock();


 out_unlock:
@@ -178,9 +167,9 @@ static int __init net_ns_init(void)
  mutex_lock(&net_mutex);
  err = setup_net(&init_net);
```

```
- net_lock();
+ rtnl_lock();
  list_add_tail(&init_net.list, &net_namespace_list);
- net_unlock();
+ rtnl_unlock();

  mutex_unlock(&net_mutex);
  if (err)
--
1.5.3.rc6.17.g1911
```