Subject: [PATCH 1/8 net-2.6.26] [NETNS]: Make netns refconting debug like a socket one.
Posted by den on Tue, 15 Apr 2008 12:37:48 GMT
View Forum Message <> Reply to Message

Make release_net/hold_net noop for performance-hungry people. This is a debug
staff and should be used in the debug mode only.

Add check for net != NULL in hold/release calls. This will be required
later on.

Signed-off-by: Denis V. Lunev <den@openvz.org>

```
---
 include/net/net_namespace.h |   44 +++++++++++++++++++++++++++++---------------
 net/core/net_namespace.c    |    4 +++
 2 files changed, 30 insertions(+), 16 deletions(-)

diff --git a/include/net/net_namespace.h b/include/net/net_namespace.h
index e2aee26..269a681 100644
--- a/include/net/net_namespace.h
+++ b/include/net/net_namespace.h
@@ -21,9 +21,11 @@ struct net_device;
  atomic_t  count;  /* To decided when the network
        *  namespace should be freed.
        */
+#ifdef NETNS_REFCNT_DEBUG
  atomic_t  use_count; /* To track references we
        * destroy on demand
        */
+#endif
  struct list_head list;  /* list of network namespaces */
  struct work_struct work;  /* work struct for freeing */

@@ -115,17 +119,6 @@ static inline void put_net(struct net *net)
   __put_net(net);
 }

-static inline struct net *hold_net(struct net *net)
-{
- atomic_inc(&net->use_count);
- return net;
-}
-
-static inline void release_net(struct net *net)
-{
- atomic_dec(&net->use_count);
-}
-
```

```
 static inline
 int net_eq(const struct net *net1, const struct net *net2)
 {
@@ -141,27 +134,46 @@ static inline void put_net(struct net *net)
 {
 }

+static inline struct net *maybe_get_net(struct net *net)
+{
+ return net;
+}
+
+static inline
+int net_eq(const struct net *net1, const struct net *net2)
+{
+ return 1;
+}
+#endif
+
+
+#ifdef NETNS_REFCNT_DEBUG
 static inline struct net *hold_net(struct net *net)
 {
+ if (net == NULL)
+  return NULL;
+ atomic_inc(&net->use_count);
  return net;
 }

 static inline void release_net(struct net *net)
 {
+ if (net == NULL)
+  return;
+ atomic_dec(&net->use_count);
 }
-
-static inline struct net *maybe_get_net(struct net *net)
+#else
+static inline struct net *hold_net(struct net *net)
 {
  return net;
 }

-static inline
-int net_eq(const struct net *net1, const struct net *net2)
+static inline void release_net(struct net *net)
 {
- return 1;
```

```
 }
 #endif

+
 #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 7b66083..f310ef3 100644
--- a/net/core/net_namespace.c
+++ b/net/core/net_namespace.c
@@ -30,7 +30,9 @@ static __net_init int setup_net(struct net *net)
  int error;

  atomic_set(&net->count, 1);
+#ifdef NETNS_REFCNT_DEBUG
  atomic_set(&net->use_count, 0);
+#endif

  error = 0;
  list_for_each_entry(ops, &pernet_list, list) {
@@ -70,11 +72,13 @@ static void net_free(struct net *net)
  if (!net)
   return;

+#ifdef NETNS_REFCNT_DEBUG
  if (unlikely(atomic_read(&net->use_count) != 0)) {
   printk(KERN_EMERG "network namespace not free! Usage: %d\n",
    atomic_read(&net->use_count));
   return;
  }
+#endif

  kmem_cache_free(net_cachep, net);
 }
--
1.5.3.rc5


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