Subject: Re: [PATCH] Teach cifs about network namespaces (take 3)
Posted by Rob Landley on Thu, 13 Jan 2011 18:55:04 GMT
View Forum Message <> Reply to Message

From: Rob Landley <rlandley@parallels.com>

Teach cifs about network namespaces, so mounting uses adresses/routing
visible from the container rather than from init context.

Signed-off-by: Rob Landley <rlandley@parallels.com>

Now using net_eq(), with the initialization moved up so the error path doesn't
dereference a null on the put.

 fs/cifs/cifsglob.h | 33 ++++++++++++++++++++++++++++++++++
 fs/cifs/connect.c | 12 ++++++++++--
 2 files changed, 43 insertions(+), 2 deletions(-)

```
diff --git a/fs/cifs/cifsglob.h b/fs/cifs/cifsglob.h
index 606ca8b..54cd4ab 100644
--- a/fs/cifs/cifsglob.h
+++ b/fs/cifs/cifsglob.h
@@ -165,6 +165,9 @@ struct TCP_Server_Info {
  struct socket *ssocket;
  struct sockaddr_storage dstaddr;
  struct sockaddr_storage srcaddr; /* locally bind to this IP */
+#ifdef CONFIG_NET_NS
+ struct net *net;
+#endif
  wait_queue_head_t response_q;
  wait_queue_head_t request_q; /* if more than maxmpx to srvr must block*/
  struct list_head pending_mid_q;
@@ -224,6 +227,36 @@ struct TCP_Server_Info {
 };

 /*
+ * Macros to allow the TCP_Server_Info->net field and related code to drop out
+ * when CONFIG_NET_NS isn't set.
+ */
+
+#ifdef CONFIG_NET_NS
+
+static inline struct net *cifs_net_ns(struct TCP_Server_Info *srv)
+{
+ return srv->net;
+}
+
```

```
+static inline void cifs_set_net_ns(struct TCP_Server_Info *srv, struct net *net)
+{
+ srv->net = net;
+}
+
+#else
+
+static inline struct net *cifs_net_ns(struct TCP_Server_Info *srv)
+{
+ return &init_net;
+}
+
+static inline void cifs_set_net_ns(struct TCP_Server_Info *srv, struct net *net)
+{
+}
+
+#endif
+
+/*
  * Session structure.  One of these for each uid session with a particular host
  */
 struct cifsSesInfo {
diff --git a/fs/cifs/connect.c b/fs/cifs/connect.c
index a65d311..53679f6 100644
--- a/fs/cifs/connect.c
+++ b/fs/cifs/connect.c
@@ -1577,6 +1577,9 @@ cifs_find_tcp_session(struct sockaddr *addr, struct smb_vol *vol)

  spin_lock(&cifs_tcp_ses_lock);
  list_for_each_entry(server, &cifs_tcp_ses_list, tcp_ses_list) {
+  if (!net_eq(cifs_net_ns(server), current->nsproxy->net_ns))
+   continue;
+
   if (!match_address(server, addr,
        (struct sockaddr *)&vol->srcaddr))
    continue;
@@ -1607,6 +1610,8 @@ cifs_put_tcp_session(struct TCP_Server_Info *server)
   return;
  }

+ put_net(cifs_net_ns(server));
+
  list_del_init(&server->tcp_ses_list);
  spin_unlock(&cifs_tcp_ses_lock);

@@ -1679,6 +1684,7 @@ cifs_get_tcp_session(struct smb_vol *volume_info)
   goto out_err;
  }
```

```
+ cifs_set_net_ns(tcp_ses, get_net(current->nsproxy->net_ns));
  tcp_ses->hostname = extract_hostname(volume_info->UNC);
  if (IS_ERR(tcp_ses->hostname)) {
   rc = PTR_ERR(tcp_ses->hostname);
@@ -1754,6 +1760,8 @@ cifs_get_tcp_session(struct smb_vol *volume_info)
 out_err_crypto_release:
  cifs_crypto_shash_release(tcp_ses);

+ put_net(cifs_net_ns(tcp_ses));
+
 out_err:
  if (tcp_ses) {
   if (!IS_ERR(tcp_ses->hostname))
@@ -2265,8 +2273,8 @@ generic_ip_connect(struct TCP_Server_Info *server)
  }

  if (socket == NULL) {
-  rc = sock_create_kern(sfamily, SOCK_STREAM,
-          IPPROTO_TCP, &socket);
+  rc = __sock_create(cifs_net_ns(server), sfamily, SOCK_STREAM,
+        IPPROTO_TCP, &socket, 1);
   if (rc < 0) {
    cERROR(1, "Error %d creating socket", rc);
    server->ssocket = NULL;
```

_____

Containers mailing list
Containers@lists.linux-foundation.org
https://lists.linux-foundation.org/mailman/listinfo/containe rs