Subject: [PATCH 16/16] net: netlink support for moving devices between network namespaces.
Posted by ebiederm on Sat, 08 Sep 2007 21:43:44 GMT
View Forum Message <> Reply to Message

The simplest thing to implement is moving network devices between
namespaces. However with the same attribute IFLA_NET_NS_PID we can
easily implement creating devices in the destination network
namespace as well. However that is a little bit trickier so this
patch sticks to what is simple and easy.

A pid is used to identify a process that happens to be a member
of the network namespace we want to move the network device to.

Signed-off-by: Eric W. Biederman <ebiederm@xmission.com>
---
include/linux/if_link.h | 1 +
net/core/rtnetlink.c | 35 ++++++++++++++++++++++++++++++++++
2 files changed, 36 insertions(+), 0 deletions(-)

```
diff --git a/include/linux/if_link.h b/include/linux/if_link.h
index 422084d..84c3492 100644
--- a/include/linux/if_link.h
+++ b/include/linux/if_link.h
@@ -78,6 +78,7 @@ enum
  IFLA_LINKMODE,
  IFLA_LINKINFO,
 #define IFLA_LINKINFO IFLA_LINKINFO
+ IFLA_NET_NS_PID,
  __IFLA_MAX
 };

diff --git a/net/core/rtnetlink.c b/net/core/rtnetlink.c
index 44f91bb..1b9c32d 100644
--- a/net/core/rtnetlink.c
+++ b/net/core/rtnetlink.c
@@ -35,6 +35,7 @@
 #include <linux/security.h>
 #include <linux/mutex.h>
 #include <linux/if_addr.h>
+#include <linux/nsproxy.h>

 #include <asm/uaccess.h>
 #include <asm/system.h>
@@ -727,6 +728,7 @@ const struct nla_policy ifla_policy[IFLA_MAX+1] = {
  [IFLA_WEIGHT]  = { .type = NLA_U32 },
  [IFLA_OPERSTATE] = { .type = NLA_U8 },
  [IFLA_LINKMODE]  = { .type = NLA_U8 },
```

```
+ [IFLA_NET_NS_PID] = { .type = NLA_U32 },
 };

 static const struct nla_policy ifla_info_policy[IFLA_INFO_MAX+1] = {
@@ -734,12 +736,45 @@ static const struct nla_policy ifla_info_policy[IFLA_INFO_MAX+1] = {
  [IFLA_INFO_DATA] = { .type = NLA_NESTED },
 };

+static struct net *get_net_ns_by_pid(pid_t pid)
+{
+ struct task_struct *tsk;
+ struct net *net;
+
+ /* Lookup the network namespace */
+ net = ERR_PTR(-ESRCH);
+ rcu_read_lock();
+ tsk = find_task_by_pid(pid);
+ if (tsk) {
+  task_lock(tsk);
+  if (tsk->nsproxy)
+   net = get_net(tsk->nsproxy->net_ns);
+  task_unlock(tsk);
+ }
+ rcu_read_unlock();
+ return net;
+}
+
 static int do_setlink(struct net_device *dev, struct ifinfomsg *ifm,
         struct nlattr **tb, char *ifname, int modified)
 {
  int send_addr_notify = 0;
  int err;

+ if (tb[IFLA_NET_NS_PID]) {
+  struct net *net;
+  net = get_net_ns_by_pid(nla_get_u32(tb[IFLA_NET_NS_PID]));
+  if (IS_ERR(net)) {
+   err = PTR_ERR(net);
+   goto errout;
+  }
+  err = dev_change_net_namespace(dev, net, ifname);
+  put_net(net);
+  if (err)
+   goto errout;
+  modified = 1;
+ }
+
  if (tb[IFLA_MAP]) {
```

```
   struct rtnl_link_ifmap *u_map;
   struct ifmap k_map;
--
1.5.3.rc6.17.g1911
```