Subject: [PATCH RFC 10/31] net: Make socket creation namespace safe.
Posted by ebiederm on Thu, 25 Jan 2007 19:00:12 GMT
View Forum Message <> Reply to Message

From: Eric W. Biederman <ebiederm@xmission.com> - unquoted

This patch passes in the namespace a new socket should be created in
and has the socket code do the appropriate reference counting. By
virtue of this all socket create methods are touched. In addition
the socket create methods are modified so that they will fail if
you attempt to create a socket in a non-default network namespace.

Failing if we attempt to create a socket outside of the default
socket namespace ensures that as we incrementally make the network stack
network namespace aware we will not export functionality that someone
has not audited and made certain is network namespace safe.
Allowing us to partially enable network namespaces before all of the
exotic protocols are supported.

Any protocol layers I have missed will fail to compile because I now
pass an extra parameter into the socket creation code.

Signed-off-by: Eric W. Biederman <ebiederm@xmission.com>
---
drivers/net/pppoe.c | 4 ++--
drivers/net/pppox.c | 7 +++++--
include/linux/if_pppox.h | 2 +-
include/linux/net.h | 3 ++-
include/net/llc_conn.h | 2 +-
include/net/sock.h | 4 +++-
net/appletalk/ddp.c | 7 +++++--
net/atm/common.c | 4 ++--
net/atm/common.h | 2 +-
net/atm/pvc.c | 7 +++++--
net/atm/svc.c | 11 +++++++----
net/ax25/af_ax25.c | 9 ++++++---
net/bluetooth/af_bluetooth.c | 7 +++++--
net/bluetooth/bnep/sock.c | 4 ++--
net/bluetooth/cmtp/sock.c | 4 ++--
net/bluetooth/hci_sock.c | 4 ++--
net/bluetooth/hidp/sock.c | 4 ++--
net/bluetooth/l2cap.c | 10 +++++-----
net/bluetooth/rfcomm/sock.c | 10 +++++-----
net/bluetooth/sco.c | 10 +++++-----
net/core/sock.c | 6 ++++--
net/decnet/af_decnet.c | 13 +++++++++---
net/econet/af_econet.c | 7 +++++--
net/ipv4/af_inet.c | 7 +++++--

```
 net/ipv6/af_inet6.c          |    7 +++++--
 net/ipx/af_ipx.c             |    7 +++++--
 net/irda/af_irda.c           |   11 +++++++----
 net/key/af_key.c             |    7 +++++--
 net/llc/af_llc.c             |    7 +++++--
 net/llc/llc_conn.c           |    6 +++---
 net/netlink/af_netlink.c     |   13 ++++++++-----
 net/netrom/af_netrom.c       |    9 ++++++---
 net/packet/af_packet.c       |    7 +++++--
 net/rose/af_rose.c           |    9 ++++++---
 net/sctp/ipv6.c              |    2 +-
 net/sctp/protocol.c          |    2 +-
 net/socket.c                 |    8 ++++----
 net/tipc/socket.c            |    9 ++++++---
 net/unix/af_unix.c           |   13 ++++++++-----
 net/wanrouter/af_wanpipe.c   |   15 +++++++++------
 net/x25/af_x25.c             |   13 ++++++++-----
 41 files changed, 182 insertions(+), 111 deletions(-)

diff --git a/drivers/net/pppoe.c b/drivers/net/pppoe.c
index d34fe16..d09334d 100644
--- a/drivers/net/pppoe.c
+++ b/drivers/net/pppoe.c
@@ -475,12 +475,12 @@ static struct proto pppoe_sk_proto = {
  * Initialize a new struct sock.
  *
  **********************************************************************/
-static int pppoe_create(struct socket *sock)
+static int pppoe_create(net_t net, struct socket *sock)
 {
  int error = -ENOMEM;
  struct sock *sk;

- sk = sk_alloc(PF_PPPOX, GFP_KERNEL, &pppoe_sk_proto, 1);
+ sk = sk_alloc(net, PF_PPPOX, GFP_KERNEL, &pppoe_sk_proto, 1);
  if (!sk)
   goto out;

diff --git a/drivers/net/pppox.c b/drivers/net/pppox.c
index 9315046..0d5c7bc 100644
--- a/drivers/net/pppox.c
+++ b/drivers/net/pppox.c
@@ -106,10 +106,13 @@ int pppox_ioctl(struct socket *sock, unsigned int cmd, unsigned long
arg)

 EXPORT_SYMBOL(pppox_ioctl);

-static int pppox_create(struct socket *sock, int protocol)
```

```
+static int pppox_create(net_t net, struct socket *sock, int protocol)
 {
  int rc = -EPROTOTYPE;

+ if (!net_eq(net, init_net()))
+  return -EAFNOSUPPORT;
+
  if (protocol < 0 || protocol > PX_MAX_PROTO)
   goto out;

@@ -118,7 +121,7 @@ static int pppox_create(struct socket *sock, int protocol)
      !try_module_get(pppox_protos[protocol]->owner))
   goto out;

- rc = pppox_protos[protocol]->create(sock);
+ rc = pppox_protos[protocol]->create(net, sock);

  module_put(pppox_protos[protocol]->owner);
 out:
diff --git a/include/linux/if_pppox.h b/include/linux/if_pppox.h
index 4fab3d0..f6ffd83 100644
--- a/include/linux/if_pppox.h
+++ b/include/linux/if_pppox.h
@@ -148,7 +148,7 @@ static inline struct sock *sk_pppox(struct pppox_sock *po)
 struct module;

 struct pppox_proto {
- int  (*create)(struct socket *sock);
+ int  (*create)(net_t net, struct socket *sock);
  int  (*ioctl)(struct socket *sock, unsigned int cmd,
       unsigned long arg);
  struct module *owner;
diff --git a/include/linux/net.h b/include/linux/net.h
index f28d8a2..4136768 100644
--- a/include/linux/net.h
+++ b/include/linux/net.h
@@ -19,6 +19,7 @@
 #define _LINUX_NET_H

 #include <linux/wait.h>
+#include <linux/net_namespace_type.h>
 #include <asm/socket.h>

 struct poll_table_struct;
@@ -169,7 +170,7 @@ struct proto_ops {

 struct net_proto_family {
  int  family;
```

```
- int  (*create)(struct socket *sock, int protocol);
+ int  (*create)(net_t net, struct socket *sock, int protocol);
  struct module *owner;
 };

diff --git a/include/net/llc_conn.h b/include/net/llc_conn.h
index 00730d2..e4f7104 100644
--- a/include/net/llc_conn.h
+++ b/include/net/llc_conn.h
@@ -93,7 +93,7 @@ static __inline__ char llc_backlog_type(struct sk_buff *skb)
  return skb->cb[sizeof(skb->cb) - 1];
 }

-extern struct sock *llc_sk_alloc(int family, gfp_t priority,
+extern struct sock *llc_sk_alloc(net_t net, int family, gfp_t priority,
       struct proto *prot);
 extern void llc_sk_free(struct sock *sk);

diff --git a/include/net/sock.h b/include/net/sock.h
index 01a2781..ebcaa7f 100644
--- a/include/net/sock.h
+++ b/include/net/sock.h
@@ -55,6 +55,7 @@
 #include <asm/atomic.h>
 #include <net/dst.h>
 #include <net/checksum.h>
+#include <net/net_namespace.h>

 /*
  * This structure really needs to be cleaned up.
@@ -784,7 +785,7 @@ extern void FASTCALL(release_sock(struct sock *sk));
     SINGLE_DEPTH_NESTING)
 #define bh_unlock_sock(__sk) spin_unlock(&((__sk)->sk_lock.slock))

-extern struct sock  *sk_alloc(int family,
+extern struct sock  *sk_alloc(net_t net, int family,
         gfp_t priority,
         struct proto *prot, int zero_it);
 extern void   sk_free(struct sock *sk);
@@ -1013,6 +1014,7 @@ static inline void sock_copy(struct sock *nsk, const struct sock *osk)
 #endif

  memcpy(nsk, osk, osk->sk_prot->obj_size);
+ get_net(nsk->sk_net);
 #ifdef CONFIG_SECURITY_NETWORK
  nsk->sk_security = sptr;
  security_sk_clone(osk, nsk);
diff --git a/net/appletalk/ddp.c b/net/appletalk/ddp.c
```

```
index 5b8a8ce..e08367b 100644
--- a/net/appletalk/ddp.c
+++ b/net/appletalk/ddp.c
@@ -1026,11 +1026,14 @@ static struct proto ddp_proto = {
  * Create a socket. Initialise the socket, blank the addresses
  * set the state.
  */
-static int atalk_create(struct socket *sock, int protocol)
+static int atalk_create(net_t net, struct socket *sock, int protocol)
 {
  struct sock *sk;
  int rc = -ESOCKTNOSUPPORT;

+ if (!net_eq(net, init_net()))
+  return -EAFNOSUPPORT;
+
  /*
   * We permit SOCK_DGRAM and RAW is an extension. It is trivial to do
   * and gives you the full ELAP frame. Should be handy for CAP 8)
@@ -1038,7 +1041,7 @@ static int atalk_create(struct socket *sock, int protocol)
  if (sock->type != SOCK_RAW && sock->type != SOCK_DGRAM)
   goto out;
  rc = -ENOMEM;
- sk = sk_alloc(PF_APPLETALK, GFP_KERNEL, &ddp_proto, 1);
+ sk = sk_alloc(net, PF_APPLETALK, GFP_KERNEL, &ddp_proto, 1);
  if (!sk)
   goto out;
  rc = 0;
diff --git a/net/atm/common.c b/net/atm/common.c
index fbabff4..c4329f0 100644
--- a/net/atm/common.c
+++ b/net/atm/common.c
@@ -132,7 +132,7 @@ static struct proto vcc_proto = {
  .obj_size = sizeof(struct atm_vcc),
 };

-int vcc_create(struct socket *sock, int protocol, int family)
+int vcc_create(net_t net, struct socket *sock, int protocol, int family)
 {
  struct sock *sk;
  struct atm_vcc *vcc;
@@ -140,7 +140,7 @@ int vcc_create(struct socket *sock, int protocol, int family)
  sock->sk = NULL;
  if (sock->type == SOCK_STREAM)
   return -EINVAL;
- sk = sk_alloc(family, GFP_KERNEL, &vcc_proto, 1);
+ sk = sk_alloc(net, family, GFP_KERNEL, &vcc_proto, 1);
  if (!sk)
```

```
   return -ENOMEM;
  sock_init_data(sock, sk);
diff --git a/net/atm/common.h b/net/atm/common.h
index a422da7..c7101c7 100644
--- a/net/atm/common.h
+++ b/net/atm/common.h
@@ -10,7 +10,7 @@
 #include <linux/poll.h> /* for poll_table */


-int vcc_create(struct socket *sock, int protocol, int family);
+int vcc_create(net_t net, struct socket *sock, int protocol, int family);
 int vcc_release(struct socket *sock);
 int vcc_connect(struct socket *sock, int itf, short vpi, int vci);
 int vcc_recvmsg(struct kiocb *iocb, struct socket *sock, struct msghdr *msg,
diff --git a/net/atm/pvc.c b/net/atm/pvc.c
index b2148b4..13bf58e 100644
--- a/net/atm/pvc.c
+++ b/net/atm/pvc.c
@@ -124,10 +124,13 @@ static const struct proto_ops pvc_proto_ops = {
 };


-static int pvc_create(struct socket *sock,int protocol)
+static int pvc_create(net_t net, struct socket *sock,int protocol)
 {
+ if (!net_eq(net, init_net()))
+  return -EAFNOSUPPORT;
+
  sock->ops = &pvc_proto_ops;
- return vcc_create(sock, protocol, PF_ATMPVC);
+ return vcc_create(net, sock, protocol, PF_ATMPVC);
 }


diff --git a/net/atm/svc.c b/net/atm/svc.c
index 3a180cf..e78d9f7 100644
--- a/net/atm/svc.c
+++ b/net/atm/svc.c
@@ -33,7 +33,7 @@
 #endif


-static int svc_create(struct socket *sock,int protocol);
+static int svc_create(net_t net, struct socket *sock,int protocol);


 /*
```

```
@@ -335,7 +335,7 @@ static int svc_accept(struct socket *sock,struct socket *newsock,int flags)

  lock_sock(sk);

- error = svc_create(newsock,0);
+ error = svc_create(sk->sk_net, newsock,0);
  if (error)
   goto out;

@@ -636,12 +636,15 @@ static const struct proto_ops svc_proto_ops = {
 };


-static int svc_create(struct socket *sock,int protocol)
+static int svc_create(net_t net, struct socket *sock,int protocol)
 {
  int error;

+ if (!net_eq(net, init_net()))
+  return -EAFNOSUPPORT;
+
  sock->ops = &svc_proto_ops;
- error = vcc_create(sock, protocol, AF_ATMSVC);
+ error = vcc_create(net, sock, protocol, AF_ATMSVC);
  if (error) return error;
  ATM_SD(sock)->local.sas_family = AF_ATMSVC;
  ATM_SD(sock)->remote.sas_family = AF_ATMSVC;
diff --git a/net/ax25/af_ax25.c b/net/ax25/af_ax25.c
index e60af4e..cdbf3f6 100644
--- a/net/ax25/af_ax25.c
+++ b/net/ax25/af_ax25.c
@@ -781,11 +781,14 @@ static struct proto ax25_proto = {
  .obj_size = sizeof(struct sock),
 };

-static int ax25_create(struct socket *sock, int protocol)
+static int ax25_create(net_t net, struct socket *sock, int protocol)
 {
  struct sock *sk;
  ax25_cb *ax25;

+ if (!net_eq(net, init_net()))
+  return -EAFNOSUPPORT;
+
  switch (sock->type) {
  case SOCK_DGRAM:
   if (protocol == 0 || protocol == PF_AX25)
@@ -831,7 +834,7 @@ static int ax25_create(struct socket *sock, int protocol)
```

```
   return -ESOCKTNOSUPPORT;
  }

- if ((sk = sk_alloc(PF_AX25, GFP_ATOMIC, &ax25_proto, 1)) == NULL)
+ if ((sk = sk_alloc(net, PF_AX25, GFP_ATOMIC, &ax25_proto, 1)) == NULL)
   return -ENOMEM;

  ax25 = sk->sk_protinfo = ax25_create_cb();
@@ -856,7 +859,7 @@ struct sock *ax25_make_new(struct sock *osk, struct ax25_dev
*ax25_dev)
  struct sock *sk;
  ax25_cb *ax25, *oax25;

- if ((sk = sk_alloc(PF_AX25, GFP_ATOMIC, osk->sk_prot, 1)) == NULL)
+ if ((sk = sk_alloc(osk->sk_net, PF_AX25, GFP_ATOMIC, osk->sk_prot, 1)) == NULL)
   return NULL;

  if ((ax25 = ax25_create_cb()) == NULL) {
diff --git a/net/bluetooth/af_bluetooth.c b/net/bluetooth/af_bluetooth.c
index 67df99e..7110360 100644
--- a/net/bluetooth/af_bluetooth.c
+++ b/net/bluetooth/af_bluetooth.c
@@ -95,10 +95,13 @@ int bt_sock_unregister(int proto)
 }
 EXPORT_SYMBOL(bt_sock_unregister);

-static int bt_sock_create(struct socket *sock, int proto)
+static int bt_sock_create(net_t net, struct socket *sock, int proto)
 {
  int err;

+ if (!net_eq(net, init_net()))
+  return -EAFNOSUPPORT;
+
  if (proto < 0 || proto >= BT_MAX_PROTO)
   return -EINVAL;

@@ -113,7 +116,7 @@ static int bt_sock_create(struct socket *sock, int proto)
  read_lock(&bt_proto_lock);

  if (bt_proto[proto] && try_module_get(bt_proto[proto]->owner)) {
-  err = bt_proto[proto]->create(sock, proto);
+  err = bt_proto[proto]->create(net, sock, proto);
   module_put(bt_proto[proto]->owner);
  }

diff --git a/net/bluetooth/bnep/sock.c b/net/bluetooth/bnep/sock.c
index 5563db1..dc9b1ef 100644
```

```
--- a/net/bluetooth/bnep/sock.c
+++ b/net/bluetooth/bnep/sock.c
@@ -205,7 +205,7 @@ static struct proto bnep_proto = {
  .obj_size = sizeof(struct bt_sock)
 };

-static int bnep_sock_create(struct socket *sock, int protocol)
+static int bnep_sock_create(net_t net, struct socket *sock, int protocol)
 {
  struct sock *sk;

@@ -214,7 +214,7 @@ static int bnep_sock_create(struct socket *sock, int protocol)
  if (sock->type != SOCK_RAW)
   return -ESOCKTNOSUPPORT;

- sk = sk_alloc(PF_BLUETOOTH, GFP_ATOMIC, &bnep_proto, 1);
+ sk = sk_alloc(net, PF_BLUETOOTH, GFP_ATOMIC, &bnep_proto, 1);
  if (!sk)
   return -ENOMEM;

diff --git a/net/bluetooth/cmtp/sock.c b/net/bluetooth/cmtp/sock.c
index 53295d3..107dbfe 100644
--- a/net/bluetooth/cmtp/sock.c
+++ b/net/bluetooth/cmtp/sock.c
@@ -196,7 +196,7 @@ static struct proto cmtp_proto = {
  .obj_size = sizeof(struct bt_sock)
 };

-static int cmtp_sock_create(struct socket *sock, int protocol)
+static int cmtp_sock_create(net_t net, struct socket *sock, int protocol)
 {
  struct sock *sk;

@@ -205,7 +205,7 @@ static int cmtp_sock_create(struct socket *sock, int protocol)
  if (sock->type != SOCK_RAW)
   return -ESOCKTNOSUPPORT;

- sk = sk_alloc(PF_BLUETOOTH, GFP_ATOMIC, &cmtp_proto, 1);
+ sk = sk_alloc(net, PF_BLUETOOTH, GFP_ATOMIC, &cmtp_proto, 1);
  if (!sk)
   return -ENOMEM;

diff --git a/net/bluetooth/hci_sock.c b/net/bluetooth/hci_sock.c
index dbf98c4..3a15a31 100644
--- a/net/bluetooth/hci_sock.c
+++ b/net/bluetooth/hci_sock.c
@@ -610,7 +610,7 @@ static struct proto hci_sk_proto = {
  .obj_size = sizeof(struct hci_pinfo)
```

```
 };

-static int hci_sock_create(struct socket *sock, int protocol)
+static int hci_sock_create(net_t net, struct socket *sock, int protocol)
 {
  struct sock *sk;

@@ -621,7 +621,7 @@ static int hci_sock_create(struct socket *sock, int protocol)

  sock->ops = &hci_sock_ops;

- sk = sk_alloc(PF_BLUETOOTH, GFP_ATOMIC, &hci_sk_proto, 1);
+ sk = sk_alloc(net, PF_BLUETOOTH, GFP_ATOMIC, &hci_sk_proto, 1);
  if (!sk)
   return -ENOMEM;

diff --git a/net/bluetooth/hidp/sock.c b/net/bluetooth/hidp/sock.c
index 407fba4..647f85e 100644
--- a/net/bluetooth/hidp/sock.c
+++ b/net/bluetooth/hidp/sock.c
@@ -247,7 +247,7 @@ static struct proto hidp_proto = {
  .obj_size = sizeof(struct bt_sock)
 };

-static int hidp_sock_create(struct socket *sock, int protocol)
+static int hidp_sock_create(net_t net, struct socket *sock, int protocol)
 {
  struct sock *sk;

@@ -256,7 +256,7 @@ static int hidp_sock_create(struct socket *sock, int protocol)
  if (sock->type != SOCK_RAW)
   return -ESOCKTNOSUPPORT;

- sk = sk_alloc(PF_BLUETOOTH, GFP_ATOMIC, &hidp_proto, 1);
+ sk = sk_alloc(net, PF_BLUETOOTH, GFP_ATOMIC, &hidp_proto, 1);
  if (!sk)
   return -ENOMEM;

diff --git a/net/bluetooth/l2cap.c b/net/bluetooth/l2cap.c
index 29a8fa4..13e9b5b 100644
--- a/net/bluetooth/l2cap.c
+++ b/net/bluetooth/l2cap.c
@@ -517,11 +517,11 @@ static struct proto l2cap_proto = {
  .obj_size = sizeof(struct l2cap_pinfo)
 };

-static struct sock *l2cap_sock_alloc(struct socket *sock, int proto, gfp_t prio)
+static struct sock *l2cap_sock_alloc(net_t net, struct socket *sock, int proto, gfp_t prio)
```

```
 {
  struct sock *sk;

- sk = sk_alloc(PF_BLUETOOTH, prio, &l2cap_proto, 1);
+ sk = sk_alloc(net, PF_BLUETOOTH, prio, &l2cap_proto, 1);
  if (!sk)
   return NULL;

@@ -542,7 +542,7 @@ static struct sock *l2cap_sock_alloc(struct socket *sock, int proto, gfp_t
prio)
  return sk;
 }

-static int l2cap_sock_create(struct socket *sock, int protocol)
+static int l2cap_sock_create(net_t net, struct socket *sock, int protocol)
 {
  struct sock *sk;

@@ -559,7 +559,7 @@ static int l2cap_sock_create(struct socket *sock, int protocol)

  sock->ops = &l2cap_sock_ops;

- sk = l2cap_sock_alloc(sock, protocol, GFP_ATOMIC);
+ sk = l2cap_sock_alloc(net, sock, protocol, GFP_ATOMIC);
  if (!sk)
   return -ENOMEM;

@@ -1412,7 +1412,7 @@ static inline int l2cap_connect_req(struct l2cap_conn *conn, struct
l2cap_cmd_hd
   goto response;
  }

- sk = l2cap_sock_alloc(NULL, BTPROTO_L2CAP, GFP_ATOMIC);
+ sk = l2cap_sock_alloc(parent->sk_net, NULL, BTPROTO_L2CAP, GFP_ATOMIC);
  if (!sk)
   goto response;

diff --git a/net/bluetooth/rfcomm/sock.c b/net/bluetooth/rfcomm/sock.c
index cb7e855..12ff829 100644
--- a/net/bluetooth/rfcomm/sock.c
+++ b/net/bluetooth/rfcomm/sock.c
@@ -282,12 +282,12 @@ static struct proto rfcomm_proto = {
  .obj_size = sizeof(struct rfcomm_pinfo)
 };

-static struct sock *rfcomm_sock_alloc(struct socket *sock, int proto, gfp_t prio)
+static struct sock *rfcomm_sock_alloc(net_t net, struct socket *sock, int proto, gfp_t prio)
 {
```

```
  struct rfcomm_dlc *d;
  struct sock *sk;

- sk = sk_alloc(PF_BLUETOOTH, prio, &rfcomm_proto, 1);
+ sk = sk_alloc(net, PF_BLUETOOTH, prio, &rfcomm_proto, 1);
  if (!sk)
   return NULL;

@@ -323,7 +323,7 @@ static struct sock *rfcomm_sock_alloc(struct socket *sock, int proto, gfp_t
prio
  return sk;
 }

-static int rfcomm_sock_create(struct socket *sock, int protocol)
+static int rfcomm_sock_create(net_t net, struct socket *sock, int protocol)
 {
  struct sock *sk;

@@ -336,7 +336,7 @@ static int rfcomm_sock_create(struct socket *sock, int protocol)

  sock->ops = &rfcomm_sock_ops;

- sk = rfcomm_sock_alloc(sock, protocol, GFP_ATOMIC);
+ sk = rfcomm_sock_alloc(net, sock, protocol, GFP_ATOMIC);
  if (!sk)
   return -ENOMEM;

@@ -868,7 +868,7 @@ int rfcomm_connect_ind(struct rfcomm_session *s, u8 channel, struct
rfcomm_dlc *
   goto done;
  }

- sk = rfcomm_sock_alloc(NULL, BTPROTO_RFCOMM, GFP_ATOMIC);
+ sk = rfcomm_sock_alloc(parent->sk_net, NULL, BTPROTO_RFCOMM, GFP_ATOMIC);
  if (!sk)
   goto done;

diff --git a/net/bluetooth/sco.c b/net/bluetooth/sco.c
index 5d13d4f..6d424ea 100644
--- a/net/bluetooth/sco.c
+++ b/net/bluetooth/sco.c
@@ -414,11 +414,11 @@ static struct proto sco_proto = {
  .obj_size = sizeof(struct sco_pinfo)
 };

-static struct sock *sco_sock_alloc(struct socket *sock, int proto, gfp_t prio)
+static struct sock *sco_sock_alloc(net_t net, struct socket *sock, int proto, gfp_t prio)
 {
```

```
  struct sock *sk;

- sk = sk_alloc(PF_BLUETOOTH, prio, &sco_proto, 1);
+ sk = sk_alloc(net, PF_BLUETOOTH, prio, &sco_proto, 1);
  if (!sk)
   return NULL;

@@ -439,7 +439,7 @@ static struct sock *sco_sock_alloc(struct socket *sock, int proto, gfp_t
prio)
  return sk;
 }

-static int sco_sock_create(struct socket *sock, int protocol)
+static int sco_sock_create(net_t net, struct socket *sock, int protocol)
 {
  struct sock *sk;

@@ -452,7 +452,7 @@ static int sco_sock_create(struct socket *sock, int protocol)

  sock->ops = &sco_sock_ops;

- sk = sco_sock_alloc(sock, protocol, GFP_ATOMIC);
+ sk = sco_sock_alloc(net, sock, protocol, GFP_ATOMIC);
  if (!sk)
   return -ENOMEM;

@@ -807,7 +807,7 @@ static void sco_conn_ready(struct sco_conn *conn)

   bh_lock_sock(parent);

-  sk = sco_sock_alloc(NULL, BTPROTO_SCO, GFP_ATOMIC);
+  sk = sco_sock_alloc(parent->sk_net, NULL, BTPROTO_SCO, GFP_ATOMIC);
   if (!sk) {
    bh_unlock_sock(parent);
    goto done;
diff --git a/net/core/sock.c b/net/core/sock.c
index 5555364..e42f7df 100644
--- a/net/core/sock.c
+++ b/net/core/sock.c
@@ -825,7 +825,7 @@ static void inline sock_lock_init(struct sock *sk)
  * @prot: struct proto associated with this new sock instance
  * @zero_it: if we should zero the newly allocated sock
  */
-struct sock *sk_alloc(int family, gfp_t priority,
+struct sock *sk_alloc(net_t net, int family, gfp_t priority,
        struct proto *prot, int zero_it)
 {
  struct sock *sk = NULL;
```

```
@@ -846,6 +846,7 @@ struct sock *sk_alloc(int family, gfp_t priority,
     */
    sk->sk_prot = sk->sk_prot_creator = prot;
    sock_lock_init(sk);
+   sk->sk_net = get_net(net);
   }

   if (security_sk_alloc(sk, family, priority))
@@ -885,6 +886,7 @@ void sk_free(struct sock *sk)
         __FUNCTION__, atomic_read(&sk->sk_omem_alloc));

  security_sk_free(sk);
+ put_net(sk->sk_net);
  if (sk->sk_prot_creator->slab != NULL)
   kmem_cache_free(sk->sk_prot_creator->slab, sk);
  else
@@ -894,7 +896,7 @@ void sk_free(struct sock *sk)

 struct sock *sk_clone(const struct sock *sk, const gfp_t priority)
 {
- struct sock *newsk = sk_alloc(sk->sk_family, priority, sk->sk_prot, 0);
+ struct sock *newsk = sk_alloc(sk->sk_net, sk->sk_family, priority, sk->sk_prot, 0);

  if (newsk != NULL) {
   struct sk_filter *filter;
diff --git a/net/decnet/af_decnet.c b/net/decnet/af_decnet.c
index 77cd802..f1553fa 100644
--- a/net/decnet/af_decnet.c
+++ b/net/decnet/af_decnet.c
@@ -471,10 +471,10 @@ static struct proto dn_proto = {
  .obj_size  = sizeof(struct dn_sock),
 };

-static struct sock *dn_alloc_sock(struct socket *sock, gfp_t gfp)
+static struct sock *dn_alloc_sock(net_t net, struct socket *sock, gfp_t gfp)
 {
  struct dn_scp *scp;
- struct sock *sk = sk_alloc(PF_DECnet, gfp, &dn_proto, 1);
+ struct sock *sk = sk_alloc(net, PF_DECnet, gfp, &dn_proto, 1);

  if  (!sk)
   goto out;
@@ -675,10 +675,13 @@ char *dn_addr2asc(__u16 addr, char *buf)



-static int dn_create(struct socket *sock, int protocol)
+static int dn_create(net_t net, struct socket *sock, int protocol)
```

```
 {
  struct sock *sk;

+ if (!net_eq(net, init_net()))
+  return -EAFNOSUPPORT;
+
  switch(sock->type) {
   case SOCK_SEQPACKET:
    if (protocol != DNPROTO_NSP)
@@ -691,7 +694,7 @@ static int dn_create(struct socket *sock, int protocol)
  }


- if ((sk = dn_alloc_sock(sock, GFP_KERNEL)) == NULL)
+ if ((sk = dn_alloc_sock(net, sock, GFP_KERNEL)) == NULL)
   return -ENOBUFS;

  sk->sk_protocol = protocol;
@@ -1088,7 +1091,7 @@ static int dn_accept(struct socket *sock, struct socket *newsock, int
flags)

  cb = DN_SKB_CB(skb);
  sk->sk_ack_backlog--;
- newsk = dn_alloc_sock(newsock, sk->sk_allocation);
+ newsk = dn_alloc_sock(sk->sk_net, newsock, sk->sk_allocation);
  if (newsk == NULL) {
   release_sock(sk);
   kfree_skb(skb);
diff --git a/net/econet/af_econet.c b/net/econet/af_econet.c
index 4d66aac..a0b3fc5 100644
--- a/net/econet/af_econet.c
+++ b/net/econet/af_econet.c
@@ -609,12 +609,15 @@ static struct proto econet_proto = {
  * Create an Econet socket
  */

-static int econet_create(struct socket *sock, int protocol)
+static int econet_create(net_t net, struct socket *sock, int protocol)
 {
  struct sock *sk;
  struct econet_sock *eo;
  int err;

+ if (!net_eq(net, init_net()))
+  return -EAFNOSUPPORT;
+
  /* Econet only provides datagram services. */
  if (sock->type != SOCK_DGRAM)
```

```
   return -ESOCKTNOSUPPORT;
@@ -622,7 +625,7 @@ static int econet_create(struct socket *sock, int protocol)
  sock->state = SS_UNCONNECTED;

  err = -ENOBUFS;
- sk = sk_alloc(PF_ECONET, GFP_KERNEL, &econet_proto, 1);
+ sk = sk_alloc(net, PF_ECONET, GFP_KERNEL, &econet_proto, 1);
  if (sk == NULL)
   goto out;

diff --git a/net/ipv4/af_inet.c b/net/ipv4/af_inet.c
index 8640096..cb07cb6 100644
--- a/net/ipv4/af_inet.c
+++ b/net/ipv4/af_inet.c
@@ -221,7 +221,7 @@ out:
  * Create an inet socket.
  */

-static int inet_create(struct socket *sock, int protocol)
+static int inet_create(net_t net, struct socket *sock, int protocol)
 {
  struct sock *sk;
  struct list_head *p;
@@ -233,6 +233,9 @@ static int inet_create(struct socket *sock, int protocol)
  int try_loading_module = 0;
  int err;

+ if (!net_eq(net, init_net()))
+  return -EAFNOSUPPORT;
+
  sock->state = SS_UNCONNECTED;

  /* Look for the requested type/protocol pair. */
@@ -295,7 +298,7 @@ lookup_protocol:
  BUG_TRAP(answer_prot->slab != NULL);

  err = -ENOBUFS;
- sk = sk_alloc(PF_INET, GFP_KERNEL, answer_prot, 1);
+ sk = sk_alloc(net, PF_INET, GFP_KERNEL, answer_prot, 1);
  if (sk == NULL)
   goto out;

diff --git a/net/ipv6/af_inet6.c b/net/ipv6/af_inet6.c
index 0e0e426..00bd55a 100644
--- a/net/ipv6/af_inet6.c
+++ b/net/ipv6/af_inet6.c
@@ -86,7 +86,7 @@ static __inline__ struct ipv6_pinfo *inet6_sk_generic(struct sock *sk)
  return (struct ipv6_pinfo *)(((u8 *)sk) + offset);
```

```
 }

-static int inet6_create(struct socket *sock, int protocol)
+static int inet6_create(net_t net, struct socket *sock, int protocol)
 {
  struct inet_sock *inet;
  struct ipv6_pinfo *np;
@@ -99,6 +99,9 @@ static int inet6_create(struct socket *sock, int protocol)
  int try_loading_module = 0;
  int err;

+ if (!net_eq(net, init_net()))
+  return -EAFNOSUPPORT;
+
  /* Look for the requested type/protocol pair. */
  answer = NULL;
 lookup_protocol:
@@ -159,7 +162,7 @@ lookup_protocol:
  BUG_TRAP(answer_prot->slab != NULL);

  err = -ENOBUFS;
- sk = sk_alloc(PF_INET6, GFP_KERNEL, answer_prot, 1);
+ sk = sk_alloc(net, PF_INET6, GFP_KERNEL, answer_prot, 1);
  if (sk == NULL)
   goto out;

diff --git a/net/ipx/af_ipx.c b/net/ipx/af_ipx.c
index 76c6615..2ec4a3c 100644
--- a/net/ipx/af_ipx.c
+++ b/net/ipx/af_ipx.c
@@ -1358,11 +1358,14 @@ static struct proto ipx_proto = {
  .obj_size = sizeof(struct ipx_sock),
 };

-static int ipx_create(struct socket *sock, int protocol)
+static int ipx_create(net_t net, struct socket *sock, int protocol)
 {
  int rc = -ESOCKTNOSUPPORT;
  struct sock *sk;

+ if (!net_eq(net, init_net()))
+  return -EAFNOSUPPORT;
+
  /*
   * SPX support is not anymore in the kernel sources. If you want to
   * ressurrect it, completing it and making it understand shared skbs,
@@ -1373,7 +1376,7 @@ static int ipx_create(struct socket *sock, int protocol)
   goto out;
```

```
        rc = -ENOMEM;
- sk = sk_alloc(PF_IPX, GFP_KERNEL, &ipx_proto, 1);
+ sk = sk_alloc(net, PF_IPX, GFP_KERNEL, &ipx_proto, 1);
  if (!sk)
   goto out;
 #ifdef IPX_REFCNT_DEBUG
diff --git a/net/irda/af_irda.c b/net/irda/af_irda.c
index 7e1aea8..e3344c3 100644
--- a/net/irda/af_irda.c
+++ b/net/irda/af_irda.c
@@ -60,7 +60,7 @@

 #include <net/irda/af_irda.h>

-static int irda_create(struct socket *sock, int protocol);
+static int irda_create(net_t net, struct socket *sock, int protocol);

 static const struct proto_ops irda_stream_ops;
 static const struct proto_ops irda_seqpacket_ops;
@@ -844,7 +844,7 @@ static int irda_accept(struct socket *sock, struct socket *newsock, int
flags)

  IRDA_ASSERT(self != NULL, return -1;);

- err = irda_create(newsock, sk->sk_protocol);
+ err = irda_create(sk->sk_net, newsock, sk->sk_protocol);
  if (err)
   return err;

@@ -1085,13 +1085,16 @@ static struct proto irda_proto = {
  *    Create IrDA socket
  *
  */
-static int irda_create(struct socket *sock, int protocol)
+static int irda_create(net_t net, struct socket *sock, int protocol)
 {
  struct sock *sk;
  struct irda_sock *self;

  IRDA_DEBUG(2, "%s()\n", __FUNCTION__);

+ if (!net_eq(net, init_net()))
+  return -EAFNOSUPPORT;
+
  /* Check for valid socket type */
  switch (sock->type) {
  case SOCK_STREAM:     /* For TTP connections with SAR disabled */
```

```
@@ -1103,7 +1106,7 @@ static int irda_create(struct socket *sock, int protocol)
  }

  /* Allocate networking socket */
- sk = sk_alloc(PF_IRDA, GFP_ATOMIC, &irda_proto, 1);
+ sk = sk_alloc(net, PF_IRDA, GFP_ATOMIC, &irda_proto, 1);
  if (sk == NULL)
   return -ENOMEM;

diff --git a/net/key/af_key.c b/net/key/af_key.c
index c79f9c4..244ab5b 100644
--- a/net/key/af_key.c
+++ b/net/key/af_key.c
@@ -137,11 +137,14 @@ static struct proto key_proto = {
  .obj_size = sizeof(struct pfkey_sock),
 };

-static int pfkey_create(struct socket *sock, int protocol)
+static int pfkey_create(net_t net, struct socket *sock, int protocol)
 {
  struct sock *sk;
  int err;

+ if (!net_eq(net, init_net()))
+  return -EAFNOSUPPORT;
+
  if (!capable(CAP_NET_ADMIN))
   return -EPERM;
  if (sock->type != SOCK_RAW)
@@ -150,7 +153,7 @@ static int pfkey_create(struct socket *sock, int protocol)
   return -EPROTONOSUPPORT;

  err = -ENOMEM;
- sk = sk_alloc(PF_KEY, GFP_KERNEL, &key_proto, 1);
+ sk = sk_alloc(net, PF_KEY, GFP_KERNEL, &key_proto, 1);
  if (sk == NULL)
   goto out;

diff --git a/net/llc/af_llc.c b/net/llc/af_llc.c
index 190bb3e..6bc0fff 100644
--- a/net/llc/af_llc.c
+++ b/net/llc/af_llc.c
@@ -150,14 +150,17 @@ static struct proto llc_proto = {
  * socket type we have available.
  * Returns 0 upon success, negative upon failure.
  */
-static int llc_ui_create(struct socket *sock, int protocol)
+static int llc_ui_create(net_t net, struct socket *sock, int protocol)
```

```
 {
  struct sock *sk;
  int rc = -ESOCKTNOSUPPORT;

+ if (!net_eq(net, init_net()))
+  return -EAFNOSUPPORT;
+
  if (likely(sock->type == SOCK_DGRAM || sock->type == SOCK_STREAM)) {
   rc = -ENOMEM;
-  sk = llc_sk_alloc(PF_LLC, GFP_KERNEL, &llc_proto);
+  sk = llc_sk_alloc(net, PF_LLC, GFP_KERNEL, &llc_proto);
   if (sk) {
    rc = 0;
    llc_ui_sk_init(sock, sk);
diff --git a/net/llc/llc_conn.c b/net/llc/llc_conn.c
index c761c15..49f8703 100644
--- a/net/llc/llc_conn.c
+++ b/net/llc/llc_conn.c
@@ -700,7 +700,7 @@ static struct sock *llc_create_incoming_sock(struct sock *sk,
           struct llc_addr *saddr,
           struct llc_addr *daddr)
 {
- struct sock *newsk = llc_sk_alloc(sk->sk_family, GFP_ATOMIC,
+ struct sock *newsk = llc_sk_alloc(sk->sk_net, sk->sk_family, GFP_ATOMIC,
        sk->sk_prot);
  struct llc_sock *newllc, *llc = llc_sk(sk);

@@ -867,9 +867,9 @@ static void llc_sk_init(struct sock* sk)
  * Allocates a LLC sock and initializes it. Returns the new LLC sock
  * or %NULL if there's no memory available for one
  */
-struct sock *llc_sk_alloc(int family, gfp_t priority, struct proto *prot)
+struct sock *llc_sk_alloc(net_t net, int family, gfp_t priority, struct proto *prot)
 {
- struct sock *sk = sk_alloc(family, priority, prot, 1);
+ struct sock *sk = sk_alloc(net, family, priority, prot, 1);

  if (!sk)
   goto out;
diff --git a/net/netlink/af_netlink.c b/net/netlink/af_netlink.c
index 3c00f48..7433e71 100644
--- a/net/netlink/af_netlink.c
+++ b/net/netlink/af_netlink.c
@@ -371,14 +371,14 @@ static struct proto netlink_proto = {
  .obj_size = sizeof(struct netlink_sock),
 };

-static int __netlink_create(struct socket *sock, int protocol)
```

```
+static int __netlink_create(net_t net, struct socket *sock, int protocol)
 {
  struct sock *sk;
  struct netlink_sock *nlk;

  sock->ops = &netlink_ops;

- sk = sk_alloc(PF_NETLINK, GFP_KERNEL, &netlink_proto, 1);
+ sk = sk_alloc(net, PF_NETLINK, GFP_KERNEL, &netlink_proto, 1);
  if (!sk)
   return -ENOMEM;

@@ -393,13 +393,16 @@ static int __netlink_create(struct socket *sock, int protocol)
  return 0;
 }

-static int netlink_create(struct socket *sock, int protocol)
+static int netlink_create(net_t net, struct socket *sock, int protocol)
 {
  struct module *module = NULL;
  struct netlink_sock *nlk;
  unsigned int groups;
  int err = 0;

+ if (!net_eq(net, init_net()))
+  return -EAFNOSUPPORT;
+
  sock->state = SS_UNCONNECTED;

  if (sock->type != SOCK_RAW && sock->type != SOCK_DGRAM)
@@ -422,7 +425,7 @@ static int netlink_create(struct socket *sock, int protocol)
  groups = nl_table[protocol].groups;
  netlink_unlock_table();

- if ((err = __netlink_create(sock, protocol)) < 0)
+ if ((err = __netlink_create(net, sock, protocol)) < 0)
   goto out_module;

  nlk = nlk_sk(sock->sk);
@@ -1281,7 +1284,7 @@ netlink_kernel_create(int unit, unsigned int groups,
  if (sock_create_lite(PF_NETLINK, SOCK_DGRAM, unit, &sock))
   return NULL;

- if (__netlink_create(sock, unit) < 0)
+ if (__netlink_create(init_net(), sock, unit) < 0)
   goto out_sock_release;

  if (groups < 32)
```

```
diff --git a/net/netrom/af_netrom.c b/net/netrom/af_netrom.c
index 601d58c..3fa3f1a 100644
--- a/net/netrom/af_netrom.c
+++ b/net/netrom/af_netrom.c
@@ -409,15 +409,18 @@ static struct proto nr_proto = {
  .obj_size = sizeof(struct nr_sock),
 };

-static int nr_create(struct socket *sock, int protocol)
+static int nr_create(net_t net, struct socket *sock, int protocol)
 {
  struct sock *sk;
  struct nr_sock *nr;

+ if (!net_eq(net, init_net()))
+  return -EAFNOSUPPORT;
+
  if (sock->type != SOCK_SEQPACKET || protocol != 0)
   return -ESOCKTNOSUPPORT;

- if ((sk = sk_alloc(PF_NETROM, GFP_ATOMIC, &nr_proto, 1)) == NULL)
+ if ((sk = sk_alloc(net, PF_NETROM, GFP_ATOMIC, &nr_proto, 1)) == NULL)
   return -ENOMEM;

  nr = nr_sk(sk);
@@ -459,7 +462,7 @@ static struct sock *nr_make_new(struct sock *osk)
  if (osk->sk_type != SOCK_SEQPACKET)
   return NULL;

- if ((sk = sk_alloc(PF_NETROM, GFP_ATOMIC, osk->sk_prot, 1)) == NULL)
+ if ((sk = sk_alloc(osk->sk_net, PF_NETROM, GFP_ATOMIC, osk->sk_prot, 1)) == NULL)
   return NULL;

  nr = nr_sk(sk);
diff --git a/net/packet/af_packet.c b/net/packet/af_packet.c
index 04e295a..ca371ea 100644
--- a/net/packet/af_packet.c
+++ b/net/packet/af_packet.c
@@ -981,13 +981,16 @@ static struct proto packet_proto = {
  * Create a packet of type SOCK_PACKET.
  */

-static int packet_create(struct socket *sock, int protocol)
+static int packet_create(net_t net, struct socket *sock, int protocol)
 {
  struct sock *sk;
  struct packet_sock *po;
  __be16 proto = (__force __be16)protocol; /* weird, but documented */
```

```
  int err;

+ if (!net_eq(net, init_net()))
+  return -EAFNOSUPPORT;
+
  if (!capable(CAP_NET_RAW))
   return -EPERM;
  if (sock->type != SOCK_DGRAM && sock->type != SOCK_RAW
@@ -1000,7 +1003,7 @@ static int packet_create(struct socket *sock, int protocol)
  sock->state = SS_UNCONNECTED;

  err = -ENOBUFS;
- sk = sk_alloc(PF_PACKET, GFP_KERNEL, &packet_proto, 1);
+ sk = sk_alloc(net, PF_PACKET, GFP_KERNEL, &packet_proto, 1);
  if (sk == NULL)
   goto out;

diff --git a/net/rose/af_rose.c b/net/rose/af_rose.c
index 5532340..7d5e593 100644
--- a/net/rose/af_rose.c
+++ b/net/rose/af_rose.c
@@ -499,15 +499,18 @@ static struct proto rose_proto = {
  .obj_size = sizeof(struct rose_sock),
 };

-static int rose_create(struct socket *sock, int protocol)
+static int rose_create(net_t net, struct socket *sock, int protocol)
 {
  struct sock *sk;
  struct rose_sock *rose;

+ if (!net_eq(net, init_net()))
+  return -EAFNOSUPPORT;
+
  if (sock->type != SOCK_SEQPACKET || protocol != 0)
   return -ESOCKTNOSUPPORT;

- if ((sk = sk_alloc(PF_ROSE, GFP_ATOMIC, &rose_proto, 1)) == NULL)
+ if ((sk = sk_alloc(net, PF_ROSE, GFP_ATOMIC, &rose_proto, 1)) == NULL)
   return -ENOMEM;

  rose = rose_sk(sk);
@@ -545,7 +548,7 @@ static struct sock *rose_make_new(struct sock *osk)
  if (osk->sk_type != SOCK_SEQPACKET)
   return NULL;

- if ((sk = sk_alloc(PF_ROSE, GFP_ATOMIC, &rose_proto, 1)) == NULL)
+ if ((sk = sk_alloc(osk->sk_net, PF_ROSE, GFP_ATOMIC, &rose_proto, 1)) == NULL)
```

```
   return NULL;

  rose = rose_sk(sk);
diff --git a/net/sctp/ipv6.c b/net/sctp/ipv6.c
index ef36be0..0217546 100644
--- a/net/sctp/ipv6.c
+++ b/net/sctp/ipv6.c
@@ -622,7 +622,7 @@ static struct sock *sctp_v6_create_accept_sk(struct sock *sk,
  struct ipv6_pinfo *newnp, *np = inet6_sk(sk);
  struct sctp6_sock *newsctp6sk;

- newsk = sk_alloc(PF_INET6, GFP_KERNEL, sk->sk_prot, 1);
+ newsk = sk_alloc(sk->sk_net, PF_INET6, GFP_KERNEL, sk->sk_prot, 1);
  if (!newsk)
   goto out;

diff --git a/net/sctp/protocol.c b/net/sctp/protocol.c
index ea94951..9461a10 100644
--- a/net/sctp/protocol.c
+++ b/net/sctp/protocol.c
@@ -540,7 +540,7 @@ static struct sock *sctp_v4_create_accept_sk(struct sock *sk,
 {
  struct inet_sock *inet = inet_sk(sk);
  struct inet_sock *newinet;
- struct sock *newsk = sk_alloc(PF_INET, GFP_KERNEL, sk->sk_prot, 1);
+ struct sock *newsk = sk_alloc(sk->sk_net, PF_INET, GFP_KERNEL, sk->sk_prot, 1);

  if (!newsk)
   goto out;
diff --git a/net/socket.c b/net/socket.c
index 4e39631..0d0c92b 100644
--- a/net/socket.c
+++ b/net/socket.c
@@ -1053,7 +1053,7 @@ call_kill:
  return 0;
 }

-static int __sock_create(int family, int type, int protocol,
+static int __sock_create(net_t net, int family, int type, int protocol,
     struct socket **res, int kern)
 {
  int err;
@@ -1129,7 +1129,7 @@ static int __sock_create(int family, int type, int protocol,
  /* Now protected by module ref count */
  rcu_read_unlock();

- err = pf->create(sock, protocol);
+ err = pf->create(net, sock, protocol);
```

```
  if (err < 0)
   goto out_module_put;

@@ -1168,12 +1168,12 @@ out_release:

 int sock_create(int family, int type, int protocol, struct socket **res)
 {
- return __sock_create(family, type, protocol, res, 0);
+ return __sock_create(current->nsproxy->net_ns, family, type, protocol, res, 0);
 }

 int sock_create_kern(int family, int type, int protocol, struct socket **res)
 {
- return __sock_create(family, type, protocol, res, 1);
+ return __sock_create(init_net(), family, type, protocol, res, 1);
 }

 asmlinkage long sys_socket(int family, int type, int protocol)
diff --git a/net/tipc/socket.c b/net/tipc/socket.c
index 2a6a5a6..cf02a0c 100644
--- a/net/tipc/socket.c
+++ b/net/tipc/socket.c
@@ -162,13 +162,16 @@ static void advance_queue(struct tipc_sock *tsock)
  *
  * Returns 0 on success, errno otherwise
  */
-static int tipc_create(struct socket *sock, int protocol)
+static int tipc_create(net_t net, struct socket *sock, int protocol)
 {
  struct tipc_sock *tsock;
  struct tipc_port *port;
  struct sock *sk;
        u32 ref;

+ if (!net_eq(net, init_net()))
+  return -EAFNOSUPPORT;
+
  if (unlikely(protocol != 0))
   return -EPROTONOSUPPORT;

@@ -198,7 +201,7 @@ static int tipc_create(struct socket *sock, int protocol)
   return -EPROTOTYPE;
  }

- sk = sk_alloc(AF_TIPC, GFP_KERNEL, &tipc_proto, 1);
+ sk = sk_alloc(net, AF_TIPC, GFP_KERNEL, &tipc_proto, 1);
  if (!sk) {
   tipc_deleteport(ref);
```

```
   return -ENOMEM;
@@ -1363,7 +1366,7 @@ static int accept(struct socket *sock, struct socket *newsock, int flags)
  }
  buf = skb_peek(&sock->sk->sk_receive_queue);

- res = tipc_create(newsock, 0);
+ res = tipc_create(sock->sk->sk_net, newsock, 0);
  if (!res) {
   struct tipc_sock *new_tsock = tipc_sk(newsock->sk);
   struct tipc_portid id;
diff --git a/net/unix/af_unix.c b/net/unix/af_unix.c
index 30855e1..8015a03 100644
--- a/net/unix/af_unix.c
+++ b/net/unix/af_unix.c
@@ -569,7 +569,7 @@ static struct proto unix_proto = {
 */
 static struct lock_class_key af_unix_sk_receive_queue_lock_key;

-static struct sock * unix_create1(struct socket *sock)
+static struct sock * unix_create1(net_t net, struct socket *sock)
 {
  struct sock *sk = NULL;
  struct unix_sock *u;
@@ -577,7 +577,7 @@ static struct sock * unix_create1(struct socket *sock)
  if (atomic_read(&unix_nr_socks) >= 2*get_max_files())
   goto out;

- sk = sk_alloc(PF_UNIX, GFP_KERNEL, &unix_proto, 1);
+ sk = sk_alloc(net, PF_UNIX, GFP_KERNEL, &unix_proto, 1);
  if (!sk)
   goto out;

@@ -602,8 +602,11 @@ out:
  return sk;
 }

-static int unix_create(struct socket *sock, int protocol)
+static int unix_create(net_t net, struct socket *sock, int protocol)
 {
+ if (!net_eq(net, init_net()))
+  return -EAFNOSUPPORT;
+
  if (protocol && protocol != PF_UNIX)
   return -EPROTONOSUPPORT;

@@ -629,7 +632,7 @@ static int unix_create(struct socket *sock, int protocol)
   return -ESOCKTNOSUPPORT;
  }
```

```
- return unix_create1(sock) ? 0 : -ENOMEM;
+ return unix_create1(net, sock) ? 0 : -ENOMEM;
 }

 static int unix_release(struct socket *sock)
@@ -980,7 +983,7 @@ static int unix_stream_connect(struct socket *sock, struct sockaddr
*uaddr,
  err = -ENOMEM;

  /* create new sock for complete connection */
- newsk = unix_create1(NULL);
+ newsk = unix_create1(sk->sk_net, NULL);
  if (newsk == NULL)
   goto out;

diff --git a/net/wanrouter/af_wanpipe.c b/net/wanrouter/af_wanpipe.c
index c205973..542c737 100644
--- a/net/wanrouter/af_wanpipe.c
+++ b/net/wanrouter/af_wanpipe.c
@@ -191,7 +191,7 @@ struct net_device *wanpipe_find_free_dev(sdla_t *card);
 static void wanpipe_unlink_card (struct sock *);
 static int wanpipe_link_card (struct sock *);
 static struct sock *wanpipe_make_new(struct sock *);
-static struct sock *wanpipe_alloc_socket(void);
+static struct sock *wanpipe_alloc_socket(net_t net);
 static inline int get_atomic_device(struct net_device *dev);
 static int wanpipe_exec_cmd(struct sock *, int, unsigned int);
 static int get_ioctl_cmd (struct sock *, void *);
@@ -455,7 +455,7 @@ static struct sock *wanpipe_make_new(struct sock *osk)
  if (osk->sk_type != SOCK_RAW)
   return NULL;

- if ((sk = wanpipe_alloc_socket()) == NULL)
+ if ((sk = wanpipe_alloc_socket(osk->sk_net)) == NULL)
   return NULL;

  sk->sk_type = osk->sk_type;
@@ -498,12 +498,12 @@ static struct proto wanpipe_proto = {
  *
  *============================================================*/

-static struct sock *wanpipe_alloc_socket(void)
+static struct sock *wanpipe_alloc_socket(net_t net)
 {
  struct sock *sk;
  struct wanpipe_opt *wan_opt;
```

```
- if ((sk = sk_alloc(PF_WANPIPE, GFP_ATOMIC, &wanpipe_proto, 1)) == NULL)
+ if ((sk = sk_alloc(net, PF_WANPIPE, GFP_ATOMIC, &wanpipe_proto, 1)) == NULL)
   return NULL;

  if ((wan_opt = kzalloc(sizeof(struct wanpipe_opt), GFP_ATOMIC)) == NULL) {
@@ -1498,10 +1498,13 @@ struct net_device *wanpipe_find_free_dev(sdla_t *card)
  *      Crates AF_WANPIPE socket.
  *============================================================*/

-static int wanpipe_create(struct socket *sock, int protocol)
+static int wanpipe_create(net_t net, socket *sock, int protocol)
 {
  struct sock *sk;

+ if (!net_eq(net, init_net()))
+  return -EAFNOSUPPORT;
+
  //FIXME: This checks for root user, SECURITY ?
  //if (!capable(CAP_NET_RAW))
  // return -EPERM;
@@ -1511,7 +1514,7 @@ static int wanpipe_create(struct socket *sock, int protocol)

  sock->state = SS_UNCONNECTED;

- if ((sk = wanpipe_alloc_socket()) == NULL)
+ if ((sk = wanpipe_alloc_socket(net)) == NULL)
   return -ENOBUFS;

  sk->sk_reuse = 1;
diff --git a/net/x25/af_x25.c b/net/x25/af_x25.c
index b5c80b1..6602a34 100644
--- a/net/x25/af_x25.c
+++ b/net/x25/af_x25.c
@@ -465,10 +465,10 @@ static struct proto x25_proto = {
  .obj_size = sizeof(struct x25_sock),
 };

-static struct sock *x25_alloc_socket(void)
+static struct sock *x25_alloc_socket(net_t net)
 {
  struct x25_sock *x25;
- struct sock *sk = sk_alloc(AF_X25, GFP_ATOMIC, &x25_proto, 1);
+ struct sock *sk = sk_alloc(net, AF_X25, GFP_ATOMIC, &x25_proto, 1);

  if (!sk)
   goto out;
@@ -484,17 +484,20 @@ out:
  return sk;
```

```
 }

-static int x25_create(struct socket *sock, int protocol)
+static int x25_create(net_t net, struct socket *sock, int protocol)
 {
  struct sock *sk;
  struct x25_sock *x25;
  int rc = -ESOCKTNOSUPPORT;

+ if (!net_eq(net, init_net()))
+  return -EAFNOSUPPORT;
+
  if (sock->type != SOCK_SEQPACKET || protocol)
   goto out;

  rc = -ENOMEM;
- if ((sk = x25_alloc_socket()) == NULL)
+ if ((sk = x25_alloc_socket(net)) == NULL)
   goto out;

  x25 = x25_sk(sk);
@@ -542,7 +545,7 @@ static struct sock *x25_make_new(struct sock *osk)
  if (osk->sk_type != SOCK_SEQPACKET)
   goto out;

- if ((sk = x25_alloc_socket()) == NULL)
+ if ((sk = x25_alloc_socket(osk->sk_net)) == NULL)
   goto out;

  x25 = x25_sk(sk);
--
1.4.4.1.g278f


_____
Containers mailing list
Containers@lists.osdl.org
https://lists.osdl.org/mailman/listinfo/containers
```