Subject: [PATCH 4/4] net: Make the loopback device per network namespace
Posted by ebiederm on Thu, 27 Sep 2007 00:00:20 GMT
View Forum Message <> Reply to Message

This patch makes loopback_dev per network namespace. Adding
code to create a different loopback device for each network
namespace and adding the code to free a loopback device
when a network namespace exits.

This patch modifies all users the loopback_dev so they
access it as init_net.loopback_dev, keeping all of the
code compiling and working. A later pass will be needed to
update the users to use something other than the initial network
namespace.

Signed-off-by: Eric W. Biederman <ebiederm@xmission.com>
---
drivers/net/loopback.c | 26 ++++++++++++++++++++++-----
include/linux/netdevice.h | 1 -
include/net/net_namespace.h | 3 +++
net/core/dst.c | 5 +++--
net/decnet/dn_dev.c | 4 ++--
net/decnet/dn_route.c | 14 +++++++-------
net/ipv4/route.c | 18 +++++++++---------
net/ipv4/xfrm4_policy.c | 2 +-
net/ipv6/addrconf.c | 18 +++++++++---------
net/ipv6/netfilter/ip6t_REJECT.c | 2 +-
net/ipv6/route.c | 12 ++++++------
net/ipv6/xfrm6_policy.c | 2 +-
net/xfrm/xfrm_policy.c | 2 +-
13 files changed, 64 insertions(+), 45 deletions(-)

```
diff --git a/drivers/net/loopback.c b/drivers/net/loopback.c
index f3018bb..0f9d8c6 100644
--- a/drivers/net/loopback.c
+++ b/drivers/net/loopback.c
@@ -57,6 +57,7 @@
 #include <linux/ip.h>
 #include <linux/tcp.h>
 #include <linux/percpu.h>
+#include <net/net_namespace.h>

 struct pcpu_lstats {
  unsigned long packets;
@@ -252,7 +253,7 @@ static void loopback_setup(struct net_device *dev)
 }

 /* Setup and register the loopback device. */
```

```
-static int __init loopback_init(void)
+static int loopback_net_init(struct net *net)
 {
  struct net_device *dev;
  int err;
@@ -262,12 +263,13 @@ static int __init loopback_init(void)
  if (!dev)
   goto out;

+ dev->nd_net = net;
  err = register_netdev(dev);
  if (err)
   goto out_free_netdev;

  err = 0;
- loopback_dev = dev;
+ net->loopback_dev = dev;

 out:
  if (err)
@@ -279,7 +281,21 @@ out_free_netdev:
  goto out;
 }

-fs_initcall(loopback_init);
+static void loopback_net_exit(struct net *net)
+{
+ struct net_device *dev = net->loopback_dev;
+
+ unregister_netdev(dev);
+}
+
+static struct pernet_operations loopback_net_ops = {
+       .init = loopback_net_init,
+       .exit = loopback_net_exit,
+};
+
+static int __init loopback_init(void)
+{
+ return register_pernet_device(&loopback_net_ops);
+}

-struct net_device *loopback_dev;
-EXPORT_SYMBOL(loopback_dev);
+fs_initcall(loopback_init);
diff --git a/include/linux/netdevice.h b/include/linux/netdevice.h
index 2088097..71cf409 100644
--- a/include/linux/netdevice.h
```

```
+++ b/include/linux/netdevice.h
@@ -742,7 +742,6 @@ struct packet_type {
 #include <linux/interrupt.h>
 #include <linux/notifier.h>

-extern struct net_device  *loopback_dev;  /* The loopback */
 extern rwlock_t    dev_base_lock;  /* Device list lock */


diff --git a/include/net/net_namespace.h b/include/net/net_namespace.h
index 3ea4194..13b0e3b 100644
--- a/include/net/net_namespace.h
+++ b/include/net/net_namespace.h
@@ -9,6 +9,7 @@
 #include <linux/list.h>

 struct proc_dir_entry;
+struct net_device;
 struct net {
  atomic_t  count;  /* To decided when the network
       *  namespace should be freed.
@@ -23,6 +24,8 @@ struct net {
  struct proc_dir_entry  *proc_net_stat;
  struct proc_dir_entry  *proc_net_root;

+ struct net_device       *loopback_dev;          /* The loopback */
+
  struct list_head  dev_base_head;
  struct hlist_head  *dev_name_head;
  struct hlist_head *dev_index_head;
diff --git a/net/core/dst.c b/net/core/dst.c
index ad5ffa1..16958e6 100644
--- a/net/core/dst.c
+++ b/net/core/dst.c
@@ -18,6 +18,7 @@
 #include <linux/types.h>
 #include <net/net_namespace.h>

+#include <net/net_namespace.h>
 #include <net/dst.h>

 /*
@@ -278,11 +279,11 @@ static inline void dst_ifdown(struct dst_entry *dst, struct net_device
*dev,
  if (!unregister) {
   dst->input = dst->output = dst_discard;
  } else {
-  dst->dev = loopback_dev;
```

```
+  dst->dev = init_net.loopback_dev;
   dev_hold(dst->dev);
   dev_put(dev);
   if (dst->neighbour && dst->neighbour->dev == dev) {
-   dst->neighbour->dev = loopback_dev;
+   dst->neighbour->dev = init_net.loopback_dev;
    dev_put(dev);
    dev_hold(dst->neighbour->dev);
   }
diff --git a/net/decnet/dn_dev.c b/net/decnet/dn_dev.c
index bcaf4c5..26130af 100644
--- a/net/decnet/dn_dev.c
+++ b/net/decnet/dn_dev.c
@@ -869,10 +869,10 @@ last_chance:
   rv = dn_dev_get_first(dev, addr);
   read_unlock(&dev_base_lock);
   dev_put(dev);
-  if (rv == 0 || dev == loopback_dev)
+  if (rv == 0 || dev == init_net.loopback_dev)
    return rv;
  }
- dev = loopback_dev;
+ dev = init_net.loopback_dev;
  dev_hold(dev);
  goto last_chance;
 }
diff --git a/net/decnet/dn_route.c b/net/decnet/dn_route.c
index 96fe0aa..b7ebf99 100644
--- a/net/decnet/dn_route.c
+++ b/net/decnet/dn_route.c
@@ -887,7 +887,7 @@ static int dn_route_output_slow(struct dst_entry **pprt, const struct flowi
*old
      .scope = RT_SCOPE_UNIVERSE,
         } },
        .mark = oldflp->mark,
-       .iif = loopback_dev->ifindex,
+       .iif = init_net.loopback_dev->ifindex,
        .oif = oldflp->oif };
  struct dn_route *rt = NULL;
  struct net_device *dev_out = NULL, *dev;
@@ -904,7 +904,7 @@ static int dn_route_output_slow(struct dst_entry **pprt, const struct flowi
*old
          "dn_route_output_slow: dst=%04x src=%04x mark=%d"
          " iif=%d oif=%d\n", dn_ntohs(oldflp->fld_dst),
          dn_ntohs(oldflp->fld_src),
-         oldflp->mark, loopback_dev->ifindex, oldflp->oif);
+         oldflp->mark, init_net.loopback_dev->ifindex, oldflp->oif);
```

```
  /* If we have an output interface, verify its a DECnet device */
  if (oldflp->oif) {
@@ -957,7 +957,7 @@ source_ok:
   err = -EADDRNOTAVAIL;
   if (dev_out)
    dev_put(dev_out);
-  dev_out = loopback_dev;
+  dev_out = init_net.loopback_dev;
   dev_hold(dev_out);
   if (!fl.fld_dst) {
    fl.fld_dst =
@@ -966,7 +966,7 @@ source_ok:
    if (!fl.fld_dst)
     goto out;
   }
-  fl.oif = loopback_dev->ifindex;
+  fl.oif = init_net.loopback_dev->ifindex;
   res.type = RTN_LOCAL;
   goto make_route;
  }
@@ -1012,7 +1012,7 @@ source_ok:
      if (dev_out)
       dev_put(dev_out);
      if (dn_dev_islocal(neigh->dev, fl.fld_dst)) {
-      dev_out = loopback_dev;
+      dev_out = init_net.loopback_dev;
       res.type = RTN_LOCAL;
      } else {
       dev_out = neigh->dev;
@@ -1033,7 +1033,7 @@ source_ok:
   /* Possible improvement - check all devices for local addr */
   if (dn_dev_islocal(dev_out, fl.fld_dst)) {
    dev_put(dev_out);
-   dev_out = loopback_dev;
+   dev_out = init_net.loopback_dev;
    dev_hold(dev_out);
    res.type = RTN_LOCAL;
    goto select_source;
@@ -1069,7 +1069,7 @@ select_source:
   if (dev_out)
    dev_put(dev_out);
-  dev_out = loopback_dev;
+  dev_out = init_net.loopback_dev;
   dev_hold(dev_out);
   fl.oif = dev_out->ifindex;
   if (res.fi)
diff --git a/net/ipv4/route.c b/net/ipv4/route.c
```

```
index ca2878d..2a9b363 100644
--- a/net/ipv4/route.c
+++ b/net/ipv4/route.c
@@ -1402,8 +1402,8 @@ static void ipv4_dst_ifdown(struct dst_entry *dst, struct net_device
*dev,
 {
  struct rtable *rt = (struct rtable *) dst;
  struct in_device *idev = rt->idev;
- if (dev != loopback_dev && idev && idev->dev == dev) {
-  struct in_device *loopback_idev = in_dev_get(loopback_dev);
+ if (dev != init_net.loopback_dev && idev && idev->dev == dev) {
+  struct in_device *loopback_idev = in_dev_get(init_net.loopback_dev);
   if (loopback_idev) {
    rt->idev = loopback_idev;
    in_dev_put(idev);
@@ -1555,7 +1555,7 @@ static int ip_route_input_mc(struct sk_buff *skb, __be32 daddr,
__be32 saddr,
 #endif
  rth->rt_iif =
  rth->fl.iif = dev->ifindex;
- rth->u.dst.dev = loopback_dev;
+ rth->u.dst.dev = init_net.loopback_dev;
  dev_hold(rth->u.dst.dev);
  rth->idev = in_dev_get(rth->u.dst.dev);
  rth->fl.oif = 0;
@@ -1812,7 +1812,7 @@ static int ip_route_input_slow(struct sk_buff *skb, __be32 daddr,
__be32 saddr,
  if (res.type == RTN_LOCAL) {
   int result;
   result = fib_validate_source(saddr, daddr, tos,
-          loopback_dev->ifindex,
+          init_net.loopback_dev->ifindex,
           dev, &spec_dst, &itag);
   if (result < 0)
    goto martian_source;
@@ -1879,7 +1879,7 @@ local_input:
 #endif
  rth->rt_iif =
  rth->fl.iif = dev->ifindex;
- rth->u.dst.dev = loopback_dev;
+ rth->u.dst.dev = init_net.loopback_dev;
  dev_hold(rth->u.dst.dev);
  rth->idev = in_dev_get(rth->u.dst.dev);
  rth->rt_gateway = daddr;
@@ -2149,7 +2149,7 @@ static int ip_route_output_slow(struct rtable **rp, const struct flowi
*oldflp)
        RT_SCOPE_UNIVERSE),
          } },
```

```
        .mark = oldflp->mark,
-       .iif = loopback_dev->ifindex,
+       .iif = init_net.loopback_dev->ifindex,
        .oif = oldflp->oif };
  struct fib_result res;
  unsigned flags = 0;
@@ -2243,9 +2243,9 @@ static int ip_route_output_slow(struct rtable **rp, const struct flowi
*oldflp)
    fl.fl4_dst = fl.fl4_src = htonl(INADDR_LOOPBACK);
   if (dev_out)
    dev_put(dev_out);
-  dev_out = loopback_dev;
+  dev_out = init_net.loopback_dev;
   dev_hold(dev_out);
-  fl.oif = loopback_dev->ifindex;
+  fl.oif = init_net.loopback_dev->ifindex;
   res.type = RTN_LOCAL;
   flags |= RTCF_LOCAL;
   goto make_route;
@@ -2290,7 +2290,7 @@ static int ip_route_output_slow(struct rtable **rp, const struct flowi
*oldflp)
    fl.fl4_src = fl.fl4_dst;
   if (dev_out)
    dev_put(dev_out);
-  dev_out = loopback_dev;
+  dev_out = init_net.loopback_dev;
   dev_hold(dev_out);
   fl.oif = dev_out->ifindex;
   if (res.fi)
diff --git a/net/ipv4/xfrm4_policy.c b/net/ipv4/xfrm4_policy.c
index 29ab3de..329825c 100644
--- a/net/ipv4/xfrm4_policy.c
+++ b/net/ipv4/xfrm4_policy.c
@@ -306,7 +306,7 @@ static void xfrm4_dst_ifdown(struct dst_entry *dst, struct net_device
*dev,

  xdst = (struct xfrm_dst *)dst;
  if (xdst->u.rt.idev->dev == dev) {
-  struct in_device *loopback_idev = in_dev_get(loopback_dev);
+  struct in_device *loopback_idev = in_dev_get(init_net.loopback_dev);
   BUG_ON(!loopback_idev);

   do {
diff --git a/net/ipv6/addrconf.c b/net/ipv6/addrconf.c
index b43574f..6d5c3c2 100644
--- a/net/ipv6/addrconf.c
+++ b/net/ipv6/addrconf.c
@@ -2410,7 +2410,7 @@ static int addrconf_ifdown(struct net_device *dev, int how)
```

```
  ASSERT_RTNL();

- if (dev == loopback_dev && how == 1)
+ if (dev == init_net.loopback_dev && how == 1)
    how = 0;

  rt6_ifdown(dev);
@@ -4212,19 +4212,19 @@ int __init addrconf_init(void)
    * device and it being up should be removed.
    */
  rtnl_lock();
- if (!ipv6_add_dev(loopback_dev))
+ if (!ipv6_add_dev(init_net.loopback_dev))
    err = -ENOMEM;
  rtnl_unlock();
  if (err)
    return err;

- ip6_null_entry.u.dst.dev = loopback_dev;
- ip6_null_entry.rt6i_idev = in6_dev_get(loopback_dev);
+ ip6_null_entry.u.dst.dev = init_net.loopback_dev;
+ ip6_null_entry.rt6i_idev = in6_dev_get(init_net.loopback_dev);
 #ifdef CONFIG_IPV6_MULTIPLE_TABLES
- ip6_prohibit_entry.u.dst.dev = loopback_dev;
- ip6_prohibit_entry.rt6i_idev = in6_dev_get(loopback_dev);
- ip6_blk_hole_entry.u.dst.dev = loopback_dev;
- ip6_blk_hole_entry.rt6i_idev = in6_dev_get(loopback_dev);
+ ip6_prohibit_entry.u.dst.dev = init_net.loopback_dev;
+ ip6_prohibit_entry.rt6i_idev = in6_dev_get(init_net.loopback_dev);
+ ip6_blk_hole_entry.u.dst.dev = init_net.loopback_dev;
+ ip6_blk_hole_entry.rt6i_idev = in6_dev_get(init_net.loopback_dev);
 #endif

  register_netdevice_notifier(&ipv6_dev_notf);
@@ -4279,7 +4279,7 @@ void __exit addrconf_cleanup(void)
    continue;
   addrconf_ifdown(dev, 1);
  }
- addrconf_ifdown(loopback_dev, 2);
+ addrconf_ifdown(init_net.loopback_dev, 2);

  /*
   * Check hash table.
diff --git a/net/ipv6/netfilter/ip6t_REJECT.c b/net/ipv6/netfilter/ip6t_REJECT.c
index 5086053..3fd08d5 100644
--- a/net/ipv6/netfilter/ip6t_REJECT.c
+++ b/net/ipv6/netfilter/ip6t_REJECT.c
```

```
@@ -167,7 +167,7 @@ static inline void
 send_unreach(struct sk_buff *skb_in, unsigned char code, unsigned int hooknum)
 {
  if (hooknum == NF_IP6_LOCAL_OUT && skb_in->dev == NULL)
-  skb_in->dev = loopback_dev;
+  skb_in->dev = init_net.loopback_dev;

  icmpv6_send(skb_in, ICMPV6_DEST_UNREACH, code, 0, NULL);
 }
diff --git a/net/ipv6/route.c b/net/ipv6/route.c
index a7a21a7..6ff19f9 100644
--- a/net/ipv6/route.c
+++ b/net/ipv6/route.c
@@ -221,8 +221,8 @@ static void ip6_dst_ifdown(struct dst_entry *dst, struct net_device *dev,
  struct rt6_info *rt = (struct rt6_info *)dst;
  struct inet6_dev *idev = rt->rt6i_idev;

- if (dev != loopback_dev && idev != NULL && idev->dev == dev) {
-  struct inet6_dev *loopback_idev = in6_dev_get(loopback_dev);
+ if (dev != init_net.loopback_dev && idev != NULL && idev->dev == dev) {
+  struct inet6_dev *loopback_idev = in6_dev_get(init_net.loopback_dev);
   if (loopback_idev != NULL) {
    rt->rt6i_idev = loopback_idev;
    in6_dev_put(idev);
@@ -1185,12 +1185,12 @@ int ip6_route_add(struct fib6_config *cfg)
  if ((cfg->fc_flags & RTF_REJECT) ||
      (dev && (dev->flags&IFF_LOOPBACK) && !(addr_type&IPV6_ADDR_LOOPBACK))) {
   /* hold loopback dev/idev if we haven't done so. */
-  if (dev != loopback_dev) {
+  if (dev != init_net.loopback_dev) {
    if (dev) {
     dev_put(dev);
     in6_dev_put(idev);
    }
-   dev = loopback_dev;
+   dev = init_net.loopback_dev;
    dev_hold(dev);
    idev = in6_dev_get(dev);
    if (!idev) {
@@ -1894,13 +1894,13 @@ struct rt6_info *addrconf_dst_alloc(struct inet6_dev *idev,
  if (rt == NULL)
   return ERR_PTR(-ENOMEM);

- dev_hold(loopback_dev);
+ dev_hold(init_net.loopback_dev);
  in6_dev_hold(idev);

  rt->u.dst.flags = DST_HOST;
```

```
  rt->u.dst.input = ip6_input;
  rt->u.dst.output = ip6_output;
- rt->rt6i_dev = loopback_dev;
+ rt->rt6i_dev = init_net.loopback_dev;
  rt->rt6i_idev = idev;
  rt->u.dst.metrics[RTAX_MTU-1] = ipv6_get_mtu(rt->rt6i_dev);
  rt->u.dst.metrics[RTAX_ADVMSS-1] = ipv6_advmss(dst_mtu(&rt->u.dst));
diff --git a/net/ipv6/xfrm6_policy.c b/net/ipv6/xfrm6_policy.c
index cc07216..15aa4c5 100644
--- a/net/ipv6/xfrm6_policy.c
+++ b/net/ipv6/xfrm6_policy.c
@@ -375,7 +375,7 @@ static void xfrm6_dst_ifdown(struct dst_entry *dst, struct net_device
*dev,

  xdst = (struct xfrm_dst *)dst;
  if (xdst->u.rt6.rt6i_idev->dev == dev) {
-  struct inet6_dev *loopback_idev = in6_dev_get(loopback_dev);
+  struct inet6_dev *loopback_idev = in6_dev_get(init_net.loopback_dev);
   BUG_ON(!loopback_idev);

   do {
diff --git a/net/xfrm/xfrm_policy.c b/net/xfrm/xfrm_policy.c
index d6dfd7d..76f172f 100644
--- a/net/xfrm/xfrm_policy.c
+++ b/net/xfrm/xfrm_policy.c
@@ -1949,7 +1949,7 @@ static int stale_bundle(struct dst_entry *dst)
 void xfrm_dst_ifdown(struct dst_entry *dst, struct net_device *dev)
 {
  while ((dst = dst->child) && dst->xfrm && dst->dev == dev) {
-  dst->dev = loopback_dev;
+  dst->dev = init_net.loopback_dev;
   dev_hold(dst->dev);
   dev_put(dev);
  }
--
1.5.3.rc6.17.g1911
```