
Subject: Dynamically allocate the loopback device

Posted by [Daniel Lezcano](#) on Fri, 24 Aug 2007 12:50:25 GMT

[View Forum Message](#) <> [Reply to Message](#)

From: Daniel Lezcano <dlezcano@fr.ibm.com>

Doing this makes loopback.c a better example of how to do a simple network device, and it removes the special case single static allocation of a struct net_device, hopefully making maintenance easier.

Signed-off-by: Eric W. Biederman <ebiederm@xmission.com>

Signed-off-by: Daniel Lezcano <dlezcano@fr.ibm.com>

```
drivers/net/loopback.c      | 63 ++++++-----
include/linux/netdevice.h   |  2 +-
net/core/dst.c              |  8 ++--
net/decnet/dn_dev.c         |  4 +-
net/decnet/dn_route.c       | 14 ++++----
net/ipv4/devinet.c          |  6 ++--
net/ipv4/ipconfig.c         |  6 ++--
net/ipv4/ipvs/ip_vs_core.c  |  2 +-
net/ipv4/route.c            | 18 ++++----
net/ipv4/xfrm4_policy.c      |  2 +-
net/ipv6/addrconf.c         | 15 ++++---
net/ipv6/ip6_input.c        |  2 +-
net/ipv6/netfilter/ip6t_REJECT.c |  2 +-
net/ipv6/route.c            | 15 ++++----
net/ipv6/xfrm6_policy.c     |  2 +-
net/xfrm/xfrm_policy.c       |  4 +-
16 files changed, 89 insertions(+), 76 deletions(-)
```

diff --git a/drivers/net/loopback.c b/drivers/net/loopback.c

index 5106c23..3642aff 100644

--- a/drivers/net/loopback.c

+++ b/drivers/net/loopback.c

```
@@ -199,44 +199,57 @@ static const struct ethtool_ops loopback_ethtool_ops = {
    .get_rx_csum = always_on,
};
```

```
/*
```

```
- * The loopback device is special. There is only one instance and
```

```
- * it is statically allocated. Don't do this for other devices.
```

```
- */
```

```
- struct net_device loopback_dev = {
```

```
- .name = "lo",
```

```
- .get_stats = &get_stats,
```

```
- .mtu = (16 * 1024) + 20 + 20 + 12,
```

```

- .hard_start_xmit = loopback_xmit,
- .hard_header = eth_header,
- .hard_header_cache = eth_header_cache,
- .header_cache_update = eth_header_cache_update,
- .hard_header_len = ETH_HLEN, /* 14 */
- .addr_len = ETH_ALEN, /* 6 */
- .tx_queue_len = 0,
- .type = ARPHRD_LOOPBACK, /* 0x0001*/
- .rebuild_header = eth_rebuild_header,
- .flags = IFF_LOOPBACK,
- .features = NETIF_F_SG | NETIF_F_FRAGLIST
+static void loopback_setup(struct net_device *dev)
+{
+ dev->get_stats = &get_stats;
+ dev->mtu = (16 * 1024) + 20 + 20 + 12;
+ dev->hard_start_xmit = loopback_xmit;
+ dev->hard_header = eth_header;
+ dev->hard_header_cache = eth_header_cache;
+ dev->header_cache_update = eth_header_cache_update;
+ dev->hard_header_len = ETH_HLEN; /* 14 */
+ dev->addr_len = ETH_ALEN; /* 6 */
+ dev->tx_queue_len = 0;
+ dev->type = ARPHRD_LOOPBACK; /* 0x0001*/
+ dev->rebuild_header = eth_rebuild_header;
+ dev->flags = IFF_LOOPBACK;
+ dev->features = NETIF_F_SG | NETIF_F_FRAGLIST
#ifdef LOOPBACK_TSO
    | NETIF_F_TSO
#endif
    | NETIF_F_NO_CSUM | NETIF_F_HIGHDMA
-    | NETIF_F_LLTX,
- .ethtool_ops = &loopback_ethtool_ops,
-};
+    | NETIF_F_LLTX;
+ dev->ethtool_ops = &loopback_ethtool_ops;
+}

/* Setup and register the loopback device. */
static int __init loopback_init(void)
{
- int err = register_netdev(&loopback_dev);
+ struct net_device *dev;
+ int err;
+
+ err = -ENOMEM;
+ dev = alloc_netdev(0, "lo", loopback_setup);
+ if (!dev)
+ goto out;

```

```

+
+ err = register_netdev(dev);
+ if (err)
+ goto out_free_netdev;

+ err = 0;
+ loopback_dev = dev;
+
+out:
  if (err)
    panic("loopback: Failed to register netdevice: %d\n", err);
-
  return err;
+out_free_netdev:
+ free_netdev(dev);
+ goto out;
};

-module_init(loopback_init);
+fs_initcall(loopback_init);

+struct net_device *loopback_dev;
EXPORT_SYMBOL(loopback_dev);
diff --git a/include/linux/netdevice.h b/include/linux/netdevice.h
index 8d12f02..7cd0641 100644
--- a/include/linux/netdevice.h
+++ b/include/linux/netdevice.h
@@ -680,7 +680,7 @@ struct packet_type {
#include <linux/interrupt.h>
#include <linux/notifier.h>

-extern struct net_device loopback_dev; /* The loopback */
+extern struct net_device *loopback_dev; /* The loopback */
extern struct list_head dev_base_head; /* All devices */
extern rwlock_t dev_base_lock; /* Device list lock */

diff --git a/net/core/dst.c b/net/core/dst.c
index c6a0587..ad8549e 100644
--- a/net/core/dst.c
+++ b/net/core/dst.c
@@ -236,13 +236,13 @@ 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;
- dev_hold(&loopback_dev);
+ dst->dev = loopback_dev;

```

```

+ dev_hold(dst->dev);
  dev_put(dev);
  if (dst->neighbour && dst->neighbour->dev == dev) {
- dst->neighbour->dev = &loopback_dev;
+ dst->neighbour->dev = loopback_dev;
    dev_put(dev);
- dev_hold(&loopback_dev);
+ dev_hold(dst->neighbour->dev);
  }
}
}
diff --git a/net/decnet/dn_dev.c b/net/decnet/dn_dev.c
index fa6604f..9fea83e 100644
--- a/net/decnet/dn_dev.c
+++ b/net/decnet/dn_dev.c
@@ -868,10 +868,10 @@ 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 == loopback_dev)
    return rv;
  }
- dev = &loopback_dev;
+ dev = loopback_dev;
  dev_hold(dev);
  goto last_chance;
}
diff --git a/net/decnet/dn_route.c b/net/decnet/dn_route.c
index a4a6209..8c04ebc 100644
--- a/net/decnet/dn_route.c
+++ b/net/decnet/dn_route.c
@@ -883,7 +883,7 @@ 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 = loopback_dev->ifindex,
    .oif = oldflp->oif };
  struct dn_route *rt = NULL;
  struct net_device *dev_out = NULL, *dev;
@@ -900,7 +900,7 @@ 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, loopback_dev->ifindex, oldflp->oif);

/* If we have an output interface, verify its a DECnet device */
if (oldflp->oif) {
@@ -953,7 +953,7 @@ source_ok:
    err = -EADDRNOTAVAIL;
    if (dev_out)
        dev_put(dev_out);
-   dev_out = &loopback_dev;
+   dev_out = loopback_dev;
    dev_hold(dev_out);
    if (!fl.fld_dst) {
        fl.fld_dst =
@@ -962,7 +962,7 @@ source_ok:
        if (!fl.fld_dst)
            goto out;
    }
-   fl.oif = loopback_dev.ifindex;
+   fl.oif = loopback_dev->ifindex;
    res.type = RTN_LOCAL;
    goto make_route;
}
@@ -1008,7 +1008,7 @@ source_ok:
    if (dev_out)
        dev_put(dev_out);
    if (dn_dev_islocal(neigh->dev, fl.fld_dst)) {
-       dev_out = &loopback_dev;
+       dev_out = loopback_dev;
        res.type = RTN_LOCAL;
    } else {
        dev_out = neigh->dev;
@@ -1029,7 +1029,7 @@ 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 = loopback_dev;
    dev_hold(dev_out);
    res.type = RTN_LOCAL;
    goto select_source;
@@ -1065,7 +1065,7 @@ select_source:
    fl.fld_src = fl.fld_dst;
    if (dev_out)
        dev_put(dev_out);
-   dev_out = &loopback_dev;
+   dev_out = loopback_dev;
    dev_hold(dev_out);
    fl.oif = dev_out->ifindex;

```

```

    if (res.fi)
diff --git a/net/ipv4/devinet.c b/net/ipv4/devinet.c
index 5b77bda..808f529 100644
--- a/net/ipv4/devinet.c
+++ b/net/ipv4/devinet.c
@@ -203,7 +203,7 @@ static void inetdev_destroy(struct in_device *in_dev)
    ASSERT_RTNL();

    dev = in_dev->dev;
- if (dev == &loopback_dev)
+ if (dev == loopback_dev)
    return;

    in_dev->dead = 1;
@@ -1058,7 +1058,7 @@ static int inetdev_event(struct notifier_block *this, unsigned long event,
    in_dev = inetdev_init(dev);
    if (!in_dev)
        return notifier_from_errno(-ENOMEM);
- if (dev == &loopback_dev) {
+ if (dev == loopback_dev) {
    IN_DEV_CONF_SET(in_dev, NOXFRM, 1);
    IN_DEV_CONF_SET(in_dev, NOPOLICY, 1);
    }
@@ -1074,7 +1074,7 @@ static int inetdev_event(struct notifier_block *this, unsigned long event,
    case NETDEV_UP:
        if (dev->mtu < 68)
            break;
- if (dev == &loopback_dev) {
+ if (dev == loopback_dev) {
    struct in_ifaddr *ifa;
    if ((ifa = inet_alloc_ifa()) != NULL) {
        ifa->ifa_local =
diff --git a/net/ipv4/ipconfig.c b/net/ipv4/ipconfig.c
index c5b2470..3ec7690 100644
--- a/net/ipv4/ipconfig.c
+++ b/net/ipv4/ipconfig.c
@@ -189,11 +189,11 @@ static int __init ic_open_devs(void)
    rtnl_lock();

    /* bring loopback device up first */
- if (dev_change_flags(&loopback_dev, loopback_dev.flags | IFF_UP) < 0)
- printk(KERN_ERR "IP-Config: Failed to open %s\n", loopback_dev.name);
+ if (dev_change_flags(loopback_dev, loopback_dev->flags | IFF_UP) < 0)
+ printk(KERN_ERR "IP-Config: Failed to open %s\n", loopback_dev->name);

    for_each_netdev(dev) {
- if (dev == &loopback_dev)
+ if (dev == loopback_dev)

```

```

    continue;
    if (user_dev_name[0] ? !strcmp(dev->name, user_dev_name) :
        (!(dev->flags & IFF_LOOPBACK) &&
diff --git a/net/ipv4/ipvs/ip_vs_core.c b/net/ipv4/ipvs/ip_vs_core.c
index f005a2f..7450326 100644
--- a/net/ipv4/ipvs/ip_vs_core.c
+++ b/net/ipv4/ipvs/ip_vs_core.c
@@ -961,7 +961,7 @@ ip_vs_in(unsigned int hooknum, struct sk_buff **pskb,
 * ... don't know why 1st test DOES NOT include 2nd (?)
 */
if (unlikely(skb->pkt_type != PACKET_HOST
-    || skb->dev == &loopback_dev || skb->sk)) {
+    || skb->dev == loopback_dev || skb->sk)) {
    IP_VS_DBG(12, "packet type=%d proto=%d daddr=%d.%d.%d.%d ignored\n",
        skb->pkt_type,
        ip_hdr(skb)->protocol,
diff --git a/net/ipv4/route.c b/net/ipv4/route.c
index c7ca94b..4f13385 100644
--- a/net/ipv4/route.c
+++ b/net/ipv4/route.c
@@ -1404,8 +1404,8 @@ 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 != loopback_dev && idev && idev->dev == dev) {
+   struct in_device *loopback_idev = in_dev_get(loopback_dev);
    if (loopback_idev) {
        rt->idev = loopback_idev;
        in_dev_put(idev);
@@ -1557,7 +1557,7 @@ static int ip_route_input_mc(struct sk_buff *skb, __be32 daddr,
__be32 saddr,
#ifdef CONFIG_IP_ROUTE_MULTICAST
    rth->rt_iif =
    rth->fl.iif = dev->ifindex;
-   rth->u.dst.dev = &loopback_dev;
+   rth->u.dst.dev = loopback_dev;
    dev_hold(rth->u.dst.dev);
    rth->idev = in_dev_get(rth->u.dst.dev);
    rth->fl.oif = 0;
@@ -1814,7 +1814,7 @@ 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,

```

```

+     loopback_dev->ifindex,
        dev, &spec_dst, &itag);
    if (result < 0)
        goto martian_source;
@@ -1881,7 +1881,7 @@ local_input:
#endif
    rth->rt_iif =
    rth->fl.iif = dev->ifindex;
- rth->u.dst.dev = &loopback_dev;
+ rth->u.dst.dev = loopback_dev;
    dev_hold(rth->u.dst.dev);
    rth->idev = in_dev_get(rth->u.dst.dev);
    rth->rt_gateway = daddr;
@@ -2151,7 +2151,7 @@ static int ip_route_output_slow(struct rtable **rp, const struct flowi
*oldflp)
    RT_SCOPE_UNIVERSE),
    } },
    .mark = oldflp->mark,
-    .iif = loopback_dev.ifindex,
+    .iif = loopback_dev->ifindex,
    .oif = oldflp->oif };
    struct fib_result res;
    unsigned flags = 0;
@@ -2245,9 +2245,9 @@ 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 = loopback_dev;
    dev_hold(dev_out);
- fl.oif = loopback_dev.ifindex;
+ fl.oif = loopback_dev->ifindex;
    res.type = RTN_LOCAL;
    flags |= RTCF_LOCAL;
    goto make_route;
@@ -2292,7 +2292,7 @@ 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 = 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 4ff8ed3..29ab3de 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.iddev->dev == dev) {
- struct in_device *loopback_iddev = in_dev_get(&loopback_dev);
+ struct in_device *loopback_iddev = in_dev_get(loopback_dev);
    BUG_ON(!loopback_iddev);

    do {
diff --git a/net/ipv6/addrconf.c b/net/ipv6/addrconf.c
index 91ef3be..d806f89 100644
--- a/net/ipv6/addrconf.c
+++ b/net/ipv6/addrconf.c
@@ -2399,7 +2399,7 @@ static int addrconf_ifdown(struct net_device *dev, int how)

    ASSERT_RTNL();

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

    rt6_ifdown(dev);
@@ -4203,16 +4203,19 @@ 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(loopback_dev))
    err = -ENOMEM;
    rtnl_unlock();
    if (err)
    return err;

- ip6_null_entry.rt6i_iddev = in6_dev_get(&loopback_dev);
+ ip6_null_entry.u.dst.dev = loopback_dev;
+ ip6_null_entry.rt6i_iddev = in6_dev_get(loopback_dev);
#ifdef CONFIG_IPV6_MULTIPLE_TABLES
- ip6_prohibit_entry.rt6i_iddev = in6_dev_get(&loopback_dev);
- ip6_blk_hole_entry.rt6i_iddev = in6_dev_get(&loopback_dev);
+ ip6_prohibit_entry.u.dst.dev = loopback_dev;
+ ip6_prohibit_entry.rt6i_iddev = in6_dev_get(loopback_dev);
+ ip6_blk_hole_entry.u.dst.dev = loopback_dev;
+ ip6_blk_hole_entry.rt6i_iddev = in6_dev_get(loopback_dev);
#endif

```

```

    register_netdevice_notifier(&ipv6_dev_notf);
@@ -4267,7 +4270,7 @@ void __exit addrconf_cleanup(void)
    continue;
    addrconf_ifdown(dev, 1);
}
- addrconf_ifdown(&loopback_dev, 2);
+ addrconf_ifdown(loopback_dev, 2);

/*
 * Check hash table.
diff --git a/net/ipv6/ip6_input.c b/net/ipv6/ip6_input.c
index 30a5cb1..15d7910 100644
--- a/net/ipv6/ip6_input.c
+++ b/net/ipv6/ip6_input.c
@@ -86,7 +86,7 @@ int ipv6_rcv(struct sk_buff *skb, struct net_device *dev, struct packet_type
*pt
 *
 * BTW, when we send a packet for our own local address on a
 * non-loopback interface (e.g. ethX), it is being delivered
- * via the loopback interface (lo) here; skb->dev = &loopback_dev.
+ * via the loopback interface (lo) here; skb->dev = loopback_dev.
 * It, however, should be considered as if it is being
 * arrived via the sending interface (ethX), because of the
 * nature of scoping architecture. --yoshfujii
diff --git a/net/ipv6/netfilter/ip6t_REJECT.c b/net/ipv6/netfilter/ip6t_REJECT.c
index 2f487cd..5086053 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 = 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 55ea80f..3f5c65f 100644
--- a/net/ipv6/route.c
+++ b/net/ipv6/route.c
@@ -137,7 +137,6 @@ struct rt6_info ip6_null_entry = {
    .dst = {
        .__refcnt = ATOMIC_INIT(1),
        .__use = 1,
-    .dev = &loopback_dev,
        .obsolete = -1,
        .error = -ENETUNREACH,

```

```

        .metrics = { [RTAX_HOPLIMIT - 1] = 255, },
@@ -163,7 +162,6 @@ struct rt6_info ip6_prohibit_entry = {
    .dst = {
        .__refcnt = ATOMIC_INIT(1),
        .__use = 1,
-   .dev = &loopback_dev,
        .obsolete = -1,
        .error = -EACCES,
        .metrics = { [RTAX_HOPLIMIT - 1] = 255, },
@@ -183,7 +181,6 @@ struct rt6_info ip6_blk_hole_entry = {
    .dst = {
        .__refcnt = ATOMIC_INIT(1),
        .__use = 1,
-   .dev = &loopback_dev,
        .obsolete = -1,
        .error = -EINVAL,
        .metrics = { [RTAX_HOPLIMIT - 1] = 255, },
@@ -223,8 +220,8 @@ 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 != loopback_dev && idev != NULL && idev->dev == dev) {
+ struct inet6_dev *loopback_idev = in6_dev_get(loopback_dev);
    if (loopback_idev != NULL) {
        rt->rt6i_idev = loopback_idev;
        in6_dev_put(idev);
@@ -1187,12 +1184,12 @@ 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 != loopback_dev) {
    if (dev) {
        dev_put(dev);
        in6_dev_put(idev);
    }
- dev = &loopback_dev;
+ dev = loopback_dev;
    dev_hold(dev);
    idev = in6_dev_get(dev);
    if (!idev) {
@@ -1896,13 +1893,13 @@ struct rt6_info *addrconf_dst_alloc(struct inet6_dev *idev,
    if (rt == NULL)
        return ERR_PTR(-ENOMEM);

- dev_hold(&loopback_dev);

```

```

+ dev_hold(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 = 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 3ec0c47..cc07216 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.rt6i_idev->dev == dev) {
- struct inet6_dev *loopback_idev = in6_dev_get(&loopback_dev);
+ struct inet6_dev *loopback_idev = in6_dev_get(loopback_dev);
  BUG_ON(!loopback_idev);

  do {
diff --git a/net/xfrm/xfrm_policy.c b/net/xfrm/xfrm_policy.c
index c7b2503..ca37eec 100644
--- a/net/xfrm/xfrm_policy.c
+++ b/net/xfrm/xfrm_policy.c
@@ -1947,8 +1947,8 @@ 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;
- dev_hold(&loopback_dev);
+ dst->dev = loopback_dev;
+ dev_hold(dst->dev);
  dev_put(dev);
  }
 }
--
1.5.2.4

```

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

Subject: Re: Dynamically allocate the loopback device

Posted by [dev](#) on Fri, 24 Aug 2007 13:06:58 GMT

[View Forum Message](#) <> [Reply to Message](#)

Acked-By: Kirill Korotaev <dev@sw.ru>

dlezcano@fr.ibm.com wrote:

> From: Daniel Lezcano <dlezcano@fr.ibm.com>

>

> Doing this makes loopback.c a better example of how to do a

> simple network device, and it removes the special case

> single static allocation of a struct net_device, hopefully

> making maintenance easier.

>

> Signed-off-by: Eric W. Biederman <ebiederm@xmission.com>

> Signed-off-by: Daniel Lezcano <dlezcano@fr.ibm.com>

> ---

> drivers/net/loopback.c | 63 ++++++-----

> include/linux/netdevice.h | 2 +-

> net/core/dst.c | 8 ++--

> net/decnet/dn_dev.c | 4 +-

> net/decnet/dn_route.c | 14 +++++--

> net/ipv4/devinet.c | 6 ++--

> net/ipv4/ipconfig.c | 6 ++--

> net/ipv4/ipvs/ip_vs_core.c | 2 +-

> net/ipv4/route.c | 18 +++++--

> net/ipv4/xfrm4_policy.c | 2 +-

> net/ipv6/addrconf.c | 15 +++++--

> net/ipv6/ip6_input.c | 2 +-

> net/ipv6/netfilter/ip6t_REJECT.c | 2 +-

> net/ipv6/route.c | 15 +++++--

> net/ipv6/xfrm6_policy.c | 2 +-

> net/xfrm/xfrm_policy.c | 4 +-

> 16 files changed, 89 insertions(+), 76 deletions(-)

>

> diff --git a/drivers/net/loopback.c b/drivers/net/loopback.c

> index 5106c23..3642aff 100644

> --- a/drivers/net/loopback.c

> +++ b/drivers/net/loopback.c

> @@ -199,44 +199,57 @@ static const struct ethtool_ops loopback_ethtool_ops = {

> .get_rx_csum = always_on,

> };

>

> -/*

> - * The loopback device is special. There is only one instance and

> - * it is statically allocated. Don't do this for other devices.

> - */

> -struct net_device loopback_dev = {

> - .name = "lo",

```

> - .get_stats = &get_stats,
> - .mtu = (16 * 1024) + 20 + 20 + 12,
> - .hard_start_xmit = loopback_xmit,
> - .hard_header = eth_header,
> - .hard_header_cache = eth_header_cache,
> - .header_cache_update = eth_header_cache_update,
> - .hard_header_len = ETH_HLEN, /* 14 */
> - .addr_len = ETH_ALEN, /* 6 */
> - .tx_queue_len = 0,
> - .type = ARPHRD_LOOPBACK, /* 0x0001*/
> - .rebuild_header = eth_rebuild_header,
> - .flags = IFF_LOOPBACK,
> - .features = NETIF_F_SG | NETIF_F_FRAGLIST
> +static void loopback_setup(struct net_device *dev)
> +{
> + dev->get_stats = &get_stats;
> + dev->mtu = (16 * 1024) + 20 + 20 + 12;
> + dev->hard_start_xmit = loopback_xmit;
> + dev->hard_header = eth_header;
> + dev->hard_header_cache = eth_header_cache;
> + dev->header_cache_update = eth_header_cache_update;
> + dev->hard_header_len = ETH_HLEN; /* 14 */
> + dev->addr_len = ETH_ALEN; /* 6 */
> + dev->tx_queue_len = 0;
> + dev->type = ARPHRD_LOOPBACK; /* 0x0001*/
> + dev->rebuild_header = eth_rebuild_header;
> + dev->flags = IFF_LOOPBACK;
> + dev->features = NETIF_F_SG | NETIF_F_FRAGLIST
> #ifdef LOOPBACK_TSO
>     | NETIF_F_TSO
> #endif
>     | NETIF_F_NO_CSUM | NETIF_F_HIGHDMA
> -     | NETIF_F_LLTX,
> - .ethtool_ops = &loopback_ethtool_ops,
> -};
> +     | NETIF_F_LLTX;
> + dev->ethtool_ops = &loopback_ethtool_ops;
> +}
>
> /* Setup and register the loopback device. */
> static int __init loopback_init(void)
> {
> - int err = register_netdev(&loopback_dev);
> + struct net_device *dev;
> + int err;
> +
> + err = -ENOMEM;
> + dev = alloc_netdev(0, "lo", loopback_setup);

```

```

> + if (!dev)
> + goto out;
> +
> + err = register_netdev(dev);
> + if (err)
> + goto out_free_netdev;
>
> + err = 0;
> + loopback_dev = dev;
> +
> +out:
> if (err)
> panic("loopback: Failed to register netdevice: %d\n", err);
> -
> return err;
> +out_free_netdev:
> + free_netdev(dev);
> + goto out;
> };
>
> -module_init(loopback_init);
> +fs_initcall(loopback_init);
>
> +struct net_device *loopback_dev;
> EXPORT_SYMBOL(loopback_dev);
> diff --git a/include/linux/netdevice.h b/include/linux/netdevice.h
> index 8d12f02..7cd0641 100644
> --- a/include/linux/netdevice.h
> +++ b/include/linux/netdevice.h
> @@ -680,7 +680,7 @@ struct packet_type {
> #include <linux/interrupt.h>
> #include <linux/notifier.h>
>
> -extern struct net_device loopback_dev; /* The loopback */
> +extern struct net_device *loopback_dev; /* The loopback */
> extern struct list_head dev_base_head; /* All devices */
> extern rwlock_t dev_base_lock; /* Device list lock */
>
> diff --git a/net/core/dst.c b/net/core/dst.c
> index c6a0587..ad8549e 100644
> --- a/net/core/dst.c
> +++ b/net/core/dst.c
> @@ -236,13 +236,13 @@ 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;

```

```

> - dev_hold(&loopback_dev);
> + dst->dev = loopback_dev;
> + dev_hold(dst->dev);
> dev_put(dev);
> if (dst->neighbour && dst->neighbour->dev == dev) {
> - dst->neighbour->dev = &loopback_dev;
> + dst->neighbour->dev = loopback_dev;
> dev_put(dev);
> - dev_hold(&loopback_dev);
> + dev_hold(dst->neighbour->dev);
> }
> }
> }
> diff --git a/net/decnet/dn_dev.c b/net/decnet/dn_dev.c
> index fa6604f..9fea83e 100644
> --- a/net/decnet/dn_dev.c
> +++ b/net/decnet/dn_dev.c
> @@ -868,10 +868,10 @@ 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 == loopback_dev)
>   return rv;
> }
> - dev = &loopback_dev;
> + dev = loopback_dev;
> dev_hold(dev);
> goto last_chance;
> }
> diff --git a/net/decnet/dn_route.c b/net/decnet/dn_route.c
> index a4a6209..8c04ebc 100644
> --- a/net/decnet/dn_route.c
> +++ b/net/decnet/dn_route.c
> @@ -883,7 +883,7 @@ 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 = loopback_dev->ifindex,
>     .oif = oldflp->oif };
> struct dn_route *rt = NULL;
> struct net_device *dev_out = NULL, *dev;
> @@ -900,7 +900,7 @@ 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, loopback_dev->ifindex, oldflp->oif);
>
> /* If we have an output interface, verify its a DECnet device */
> if (oldflp->oif) {
> @@ -953,7 +953,7 @@ source_ok:
>     err = -EADDRNOTAVAIL;
>     if (dev_out)
>         dev_put(dev_out);
> - dev_out = &loopback_dev;
> + dev_out = loopback_dev;
>     dev_hold(dev_out);
>     if (!fl.fld_dst) {
>         fl.fld_dst =
> @@ -962,7 +962,7 @@ source_ok:
>         if (!fl.fld_dst)
>             goto out;
>     }
> - fl.oif = loopback_dev.ifindex;
> + fl.oif = loopback_dev->ifindex;
>     res.type = RTN_LOCAL;
>     goto make_route;
> }
> @@ -1008,7 +1008,7 @@ source_ok:
>     if (dev_out)
>         dev_put(dev_out);
>     if (dn_dev_islocal(neigh->dev, fl.fld_dst)) {
> - dev_out = &loopback_dev;
> + dev_out = loopback_dev;
>     res.type = RTN_LOCAL;
>     } else {
>         dev_out = neigh->dev;
> @@ -1029,7 +1029,7 @@ 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 = loopback_dev;
>     dev_hold(dev_out);
>     res.type = RTN_LOCAL;
>     goto select_source;
> @@ -1065,7 +1065,7 @@ select_source:
>     fl.fld_src = fl.fld_dst;
>     if (dev_out)
>         dev_put(dev_out);
> - dev_out = &loopback_dev;
> + dev_out = loopback_dev;

```

```

> dev_hold(dev_out);
> fl.oif = dev_out->ifindex;
> if (res.fi)
> diff --git a/net/ipv4/devinet.c b/net/ipv4/devinet.c
> index 5b77bda..808f529 100644
> --- a/net/ipv4/devinet.c
> +++ b/net/ipv4/devinet.c
> @@ -203,7 +203,7 @@ static void inetdev_destroy(struct in_device *in_dev)
> ASSERT_RTNL();
>
> dev = in_dev->dev;
> - if (dev == &loopback_dev)
> + if (dev == loopback_dev)
> return;
>
> in_dev->dead = 1;
> @@ -1058,7 +1058,7 @@ static int inetdev_event(struct notifier_block *this, unsigned long
event,
> in_dev = inetdev_init(dev);
> if (!in_dev)
> return notifier_from_errno(-ENOMEM);
> - if (dev == &loopback_dev) {
> + if (dev == loopback_dev) {
> IN_DEV_CONF_SET(in_dev, NOXFRM, 1);
> IN_DEV_CONF_SET(in_dev, NOPOLICY, 1);
> }
> @@ -1074,7 +1074,7 @@ static int inetdev_event(struct notifier_block *this, unsigned long
event,
> case NETDEV_UP:
> if (dev->mtu < 68)
> break;
> - if (dev == &loopback_dev) {
> + if (dev == loopback_dev) {
> struct in_ifaddr *ifa;
> if ((ifa = inet_alloc_ifa()) != NULL) {
> ifa->ifa_local =
> diff --git a/net/ipv4/ipconfig.c b/net/ipv4/ipconfig.c
> index c5b2470..3ec7690 100644
> --- a/net/ipv4/ipconfig.c
> +++ b/net/ipv4/ipconfig.c
> @@ -189,11 +189,11 @@ static int __init ic_open_devs(void)
> rtnl_lock();
>
> /* bring loopback device up first */
> - if (dev_change_flags(&loopback_dev, loopback_dev.flags | IFF_UP) < 0)
> - printk(KERN_ERR "IP-Config: Failed to open %s\n", loopback_dev.name);
> + if (dev_change_flags(loopback_dev, loopback_dev->flags | IFF_UP) < 0)
> + printk(KERN_ERR "IP-Config: Failed to open %s\n", loopback_dev->name);

```

```

>
> for_each_netdev(dev) {
> - if (dev == &loopback_dev)
> + if (dev == loopback_dev)
>     continue;
>     if (user_dev_name[0] ? !strcmp(dev->name, user_dev_name) :
>         (!(dev->flags & IFF_LOOPBACK) &&
> diff --git a/net/ipv4/ipvs/ip_vs_core.c b/net/ipv4/ipvs/ip_vs_core.c
> index f005a2f..7450326 100644
> --- a/net/ipv4/ipvs/ip_vs_core.c
> +++ b/net/ipv4/ipvs/ip_vs_core.c
> @@ -961,7 +961,7 @@ ip_vs_in(unsigned int hooknum, struct sk_buff **pskb,
>  * ... don't know why 1st test DOES NOT include 2nd (?)
>  */
>  if (unlikely(skb->pkt_type != PACKET_HOST
> -    || skb->dev == &loopback_dev || skb->sk)) {
> +    || skb->dev == loopback_dev || skb->sk)) {
>  IP_VS_DBG(12, "packet type=%d proto=%d daddr=%d.%d.%d.%d ignored\n",
>    skb->pkt_type,
>    ip_hdr(skb)->protocol,
> diff --git a/net/ipv4/route.c b/net/ipv4/route.c
> index c7ca94b..4f13385 100644
> --- a/net/ipv4/route.c
> +++ b/net/ipv4/route.c
> @@ -1404,8 +1404,8 @@ 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 != loopback_dev && idev && idev->dev == dev) {
> +     struct in_device *loopback_idev = in_dev_get(loopback_dev);
>     if (loopback_idev) {
>         rt->idev = loopback_idev;
>         in_dev_put(idev);
> @@ -1557,7 +1557,7 @@ 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 = loopback_dev;
>     dev_hold(rth->u.dst.dev);
>     rth->idev = in_dev_get(rth->u.dst.dev);
>     rth->fl.oif = 0;
> @@ -1814,7 +1814,7 @@ 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,
> +     loopback_dev->ifindex,
>     dev, &spec_dst, &itag);
>   if (result < 0)
>     goto martian_source;
> @@ -1881,7 +1881,7 @@ local_input:
> #endif
>   rth->rt_iif =
>   rth->fl.iif = dev->ifindex;
> - rth->u.dst.dev = &loopback_dev;
> + rth->u.dst.dev = loopback_dev;
>   dev_hold(rth->u.dst.dev);
>   rth->idev = in_dev_get(rth->u.dst.dev);
>   rth->rt_gateway = daddr;
> @@ -2151,7 +2151,7 @@ static int ip_route_output_slow(struct rtable **rp, const struct flowi
*oldflp)
>     RT_SCOPE_UNIVERSE),
>     } },
>     .mark = oldflp->mark,
> -     .iif = loopback_dev.ifindex,
> +     .iif = loopback_dev->ifindex,
>     .oif = oldflp->oif };
>   struct fib_result res;
>   unsigned flags = 0;
> @@ -2245,9 +2245,9 @@ 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 = loopback_dev;
>   dev_hold(dev_out);
> - fl.oif = loopback_dev.ifindex;
> + fl.oif = loopback_dev->ifindex;
>   res.type = RTN_LOCAL;
>   flags |= RTCF_LOCAL;
>   goto make_route;
> @@ -2292,7 +2292,7 @@ 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 = 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 4ff8ed3..29ab3de 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.iddev->dev == dev) {
> - struct in_device *loopback_iddev = in_dev_get(&loopback_dev);
> + struct in_device *loopback_iddev = in_dev_get(loopback_dev);
> BUG_ON(!loopback_iddev);
>
> do {
> diff --git a/net/ipv6/addrconf.c b/net/ipv6/addrconf.c
> index 91ef3be..d806f89 100644
> --- a/net/ipv6/addrconf.c
> +++ b/net/ipv6/addrconf.c
> @@ -2399,7 +2399,7 @@ static int addrconf_ifdown(struct net_device *dev, int how)
>
> ASSERT_RTNL();
>
> - if (dev == &loopback_dev && how == 1)
> + if (dev == loopback_dev && how == 1)
> how = 0;
>
> rt6_ifdown(dev);
> @@ -4203,16 +4203,19 @@ 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(loopback_dev))
> err = -ENOMEM;
> rtnl_unlock();
> if (err)
> return err;
>
> - ip6_null_entry.rt6i_iddev = in6_dev_get(&loopback_dev);
> + ip6_null_entry.u.dst.dev = loopback_dev;
> + ip6_null_entry.rt6i_iddev = in6_dev_get(loopback_dev);
> #ifdef CONFIG_IPV6_MULTIPLE_TABLES
> - ip6_prohibit_entry.rt6i_iddev = in6_dev_get(&loopback_dev);
> - ip6_blk_hole_entry.rt6i_iddev = in6_dev_get(&loopback_dev);
> + ip6_prohibit_entry.u.dst.dev = loopback_dev;
> + ip6_prohibit_entry.rt6i_iddev = in6_dev_get(loopback_dev);

```

```

> + ip6_blk_hole_entry.u.dst.dev = loopback_dev;
> + ip6_blk_hole_entry.rt6i_iddev = in6_dev_get(loopback_dev);
> #endif
>
> register_netdevice_notifier(&ipv6_dev_notf);
> @@ -4267,7 +4270,7 @@ void __exit addrconf_cleanup(void)
>     continue;
>     addrconf_ifdown(dev, 1);
> }
> - addrconf_ifdown(&loopback_dev, 2);
> + addrconf_ifdown(loopback_dev, 2);
>
> /*
>  * Check hash table.
> diff --git a/net/ipv6/ip6_input.c b/net/ipv6/ip6_input.c
> index 30a5cb1..15d7910 100644
> --- a/net/ipv6/ip6_input.c
> +++ b/net/ipv6/ip6_input.c
> @@ -86,7 +86,7 @@ int ipv6_rcv(struct sk_buff *skb, struct net_device *dev, struct
packet_type *pt
>  *
>  * BTW, when we send a packet for our own local address on a
>  * non-loopback interface (e.g. ethX), it is being delivered
> - * via the loopback interface (lo) here; skb->dev = &loopback_dev.
> + * via the loopback interface (lo) here; skb->dev = loopback_dev.
>  * It, however, should be considered as if it is being
>  * arrived via the sending interface (ethX), because of the
>  * nature of scoping architecture. --yoshfujii
> diff --git a/net/ipv6/netfilter/ip6t_REJECT.c b/net/ipv6/netfilter/ip6t_REJECT.c
> index 2f487cd..5086053 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 = 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 55ea80f..3f5c65f 100644
> --- a/net/ipv6/route.c
> +++ b/net/ipv6/route.c
> @@ -137,7 +137,6 @@ struct rt6_info ip6_null_entry = {
>     .dst = {
>     .__refcnt = ATOMIC_INIT(1),

```

```

> .__use = 1,
> - .dev = &loopback_dev,
> .obsolete = -1,
> .error = -ENETUNREACH,
> .metrics = { [RTAX_HOPLIMIT - 1] = 255, },
> @@ -163,7 +162,6 @@ struct rt6_info ip6_prohibit_entry = {
> .dst = {
> .__refcnt = ATOMIC_INIT(1),
> .__use = 1,
> - .dev = &loopback_dev,
> .obsolete = -1,
> .error = -EACCES,
> .metrics = { [RTAX_HOPLIMIT - 1] = 255, },
> @@ -183,7 +181,6 @@ struct rt6_info ip6_blk_hole_entry = {
> .dst = {
> .__refcnt = ATOMIC_INIT(1),
> .__use = 1,
> - .dev = &loopback_dev,
> .obsolete = -1,
> .error = -EINVAL,
> .metrics = { [RTAX_HOPLIMIT - 1] = 255, },
> @@ -223,8 +220,8 @@ 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 != loopback_dev && idev != NULL && idev->dev == dev) {
> + struct inet6_dev *loopback_idev = in6_dev_get(loopback_dev);
> if (loopback_idev != NULL) {
> rt->rt6i_idev = loopback_idev;
> in6_dev_put(idev);
> @@ -1187,12 +1184,12 @@ 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 != loopback_dev) {
> if (dev) {
> dev_put(dev);
> in6_dev_put(idev);
> }
> - dev = &loopback_dev;
> + dev = loopback_dev;
> dev_hold(dev);
> idev = in6_dev_get(dev);
> if (!idev) {
> @@ -1896,13 +1893,13 @@ struct rt6_info *addrconf_dst_alloc(struct inet6_dev *idev,

```

```

> if (rt == NULL)
> return ERR_PTR(-ENOMEM);
>
> - dev_hold(&loopback_dev);
> + dev_hold(loopback_dev);
> in6_dev_hold(iddev);
>
> 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 = loopback_dev;
> rt->rt6i_iddev = iddev;
> 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 3ec0c47..cc07216 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_iddev->dev == dev) {
> - struct inet6_dev *loopback_iddev = in6_dev_get(&loopback_dev);
> + struct inet6_dev *loopback_iddev = in6_dev_get(loopback_dev);
> BUG_ON(!loopback_iddev);
>
> do {
> diff --git a/net/xfrm/xfrm_policy.c b/net/xfrm/xfrm_policy.c
> index c7b2503..ca37eec 100644
> --- a/net/xfrm/xfrm_policy.c
> +++ b/net/xfrm/xfrm_policy.c
> @@ -1947,8 +1947,8 @@ 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;
> - dev_hold(&loopback_dev);
> + dst->dev = loopback_dev;
> + dev_hold(dst->dev);
> dev_put(dev);
> }
> }

```

Containers mailing list
Containers@lists.linux-foundation.org

Subject: Re: Dynamically allocate the loopback device
Posted by [Benjamin Thery](#) on Fri, 24 Aug 2007 13:22:27 GMT
[View Forum Message](#) <> [Reply to Message](#)

Kirill Korotaev wrote:

> Acked-By: Kirill Korotaev <dev@sw.ru>

Acked-by: Benjamin Thery <benjamin.thery@bull.net>

Looks good to me.

Compiled and tested on i386.

>

> dlezcano@fr.ibm.com wrote:

>> From: Daniel Lezcano <dlezcano@fr.ibm.com>

>>

>> Doing this makes loopback.c a better example of how to do a
>> simple network device, and it removes the special case
>> single static allocation of a struct net_device, hopefully
>> making maintenance easier.

>>

>> Signed-off-by: Eric W. Biederman <ebiederm@xmission.com>

>> Signed-off-by: Daniel Lezcano <dlezcano@fr.ibm.com>

>> ---

>> drivers/net/loopback.c | 63 ++++++-----
>> include/linux/netdevice.h | 2 +-
>> net/core/dst.c | 8 ++-
>> net/decnet/dn_dev.c | 4 +-
>> net/decnet/dn_route.c | 14 ++++----
>> net/ipv4/devinet.c | 6 ++-
>> net/ipv4/ipconfig.c | 6 ++-
>> net/ipv4/ipvs/ip_vs_core.c | 2 +-
>> net/ipv4/route.c | 18 ++++-----
>> net/ipv4/xfrm4_policy.c | 2 +-
>> net/ipv6/addrconf.c | 15 ++++---
>> net/ipv6/ip6_input.c | 2 +-
>> net/ipv6/netfilter/ip6t_REJECT.c | 2 +-
>> net/ipv6/route.c | 15 ++++----
>> net/ipv6/xfrm6_policy.c | 2 +-
>> net/xfrm/xfrm_policy.c | 4 +-
>> 16 files changed, 89 insertions(+), 76 deletions(-)
>>

>> diff --git a/drivers/net/loopback.c b/drivers/net/loopback.c

>> index 5106c23..3642aff 100644

>> --- a/drivers/net/loopback.c

```

>> +++ b/drivers/net/loopback.c
>> @@ -199,44 +199,57 @@ static const struct ethtool_ops loopback_ethtool_ops = {
>> .get_rx_csum = always_on,
>> };
>>
>> /*
>> - * The loopback device is special. There is only one instance and
>> - * it is statically allocated. Don't do this for other devices.
>> - */
>> -struct net_device loopback_dev = {
>> - .name = "lo",
>> - .get_stats = &get_stats,
>> - .mtu = (16 * 1024) + 20 + 20 + 12,
>> - .hard_start_xmit = loopback_xmit,
>> - .hard_header = eth_header,
>> - .hard_header_cache = eth_header_cache,
>> - .header_cache_update = eth_header_cache_update,
>> - .hard_header_len = ETH_HLEN, /* 14 */
>> - .addr_len = ETH_ALEN, /* 6 */
>> - .tx_queue_len = 0,
>> - .type = ARPHRD_LOOPBACK, /* 0x0001*/
>> - .rebuild_header = eth_rebuild_header,
>> - .flags = IFF_LOOPBACK,
>> - .features = NETIF_F_SG | NETIF_F_FRAGLIST
>> +static void loopback_setup(struct net_device *dev)
>> +{
>> + dev->get_stats = &get_stats;
>> + dev->mtu = (16 * 1024) + 20 + 20 + 12;
>> + dev->hard_start_xmit = loopback_xmit;
>> + dev->hard_header = eth_header;
>> + dev->hard_header_cache = eth_header_cache;
>> + dev->header_cache_update = eth_header_cache_update;
>> + dev->hard_header_len = ETH_HLEN; /* 14 */
>> + dev->addr_len = ETH_ALEN; /* 6 */
>> + dev->tx_queue_len = 0;
>> + dev->type = ARPHRD_LOOPBACK; /* 0x0001*/
>> + dev->rebuild_header = eth_rebuild_header;
>> + dev->flags = IFF_LOOPBACK;
>> + dev->features = NETIF_F_SG | NETIF_F_FRAGLIST
>> #ifdef LOOPBACK_TSO
>>     | NETIF_F_TSO
>> #endif
>>     | NETIF_F_NO_CSUM | NETIF_F_HIGHDMA
>> -     | NETIF_F_LLTX,
>> - .ethtool_ops = &loopback_ethtool_ops,
>> -};
>> +     | NETIF_F_LLTX;
>> + dev->ethtool_ops = &loopback_ethtool_ops;

```

```

>> +}
>>
>> /* Setup and register the loopback device. */
>> static int __init loopback_init(void)
>> {
>> - int err = register_netdev(&loopback_dev);
>> + struct net_device *dev;
>> + int err;
>> +
>> + err = -ENOMEM;
>> + dev = alloc_netdev(0, "lo", loopback_setup);
>> + if (!dev)
>> + goto out;
>> +
>> + err = register_netdev(dev);
>> + if (err)
>> + goto out_free_netdev;
>>
>> + err = 0;
>> + loopback_dev = dev;
>> +
>> +out:
>> if (err)
>>   panic("loopback: Failed to register netdevice: %d\n", err);
>> -
>> return err;
>> +out_free_netdev:
>> + free_netdev(dev);
>> + goto out;
>> };
>>
>> -module_init(loopback_init);
>> +fs_initcall(loopback_init);
>>
>> +struct net_device *loopback_dev;
>> EXPORT_SYMBOL(loopback_dev);
>> diff --git a/include/linux/netdevice.h b/include/linux/netdevice.h
>> index 8d12f02..7cd0641 100644
>> --- a/include/linux/netdevice.h
>> +++ b/include/linux/netdevice.h
>> @@ -680,7 +680,7 @@ struct packet_type {
>> #include <linux/interrupt.h>
>> #include <linux/notifier.h>
>>
>> -extern struct net_device loopback_dev; /* The loopback */
>> +extern struct net_device *loopback_dev; /* The loopback */
>> extern struct list_head dev_base_head; /* All devices */
>> extern rwlock_t dev_base_lock; /* Device list lock */

```

```

>>
>> diff --git a/net/core/dst.c b/net/core/dst.c
>> index c6a0587..ad8549e 100644
>> --- a/net/core/dst.c
>> +++ b/net/core/dst.c
>> @@ -236,13 +236,13 @@ 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;
>> - dev_hold(&loopback_dev);
>> + dst->dev = loopback_dev;
>> + dev_hold(dst->dev);
>> dev_put(dev);
>> if (dst->neighbour && dst->neighbour->dev == dev) {
>> - dst->neighbour->dev = &loopback_dev;
>> + dst->neighbour->dev = loopback_dev;
>> dev_put(dev);
>> - dev_hold(&loopback_dev);
>> + dev_hold(dst->neighbour->dev);
>> }
>> }
>> }
>> diff --git a/net/decnet/dn_dev.c b/net/decnet/dn_dev.c
>> index fa6604f..9fea83e 100644
>> --- a/net/decnet/dn_dev.c
>> +++ b/net/decnet/dn_dev.c
>> @@ -868,10 +868,10 @@ 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 == loopback_dev)
>> return rv;
>> }
>> - dev = &loopback_dev;
>> + dev = loopback_dev;
>> dev_hold(dev);
>> goto last_chance;
>> }
>> diff --git a/net/decnet/dn_route.c b/net/decnet/dn_route.c
>> index a4a6209..8c04ebc 100644
>> --- a/net/decnet/dn_route.c
>> +++ b/net/decnet/dn_route.c
>> @@ -883,7 +883,7 @@ 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 = loopback_dev->ifindex,
>>     .oif = oldflp->oif };
>> struct dn_route *rt = NULL;
>> struct net_device *dev_out = NULL, *dev;
>> @@ -900,7 +900,7 @@ 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, loopback_dev->ifindex, oldflp->oif);
>>
>> /* If we have an output interface, verify its a DECnet device */
>> if (oldflp->oif) {
>> @@ -953,7 +953,7 @@ source_ok:
>>     err = -EADDRNOTAVAIL;
>>     if (dev_out)
>>         dev_put(dev_out);
>> - dev_out = &loopback_dev;
>> + dev_out = loopback_dev;
>>     dev_hold(dev_out);
>>     if (!fl.fld_dst) {
>>         fl.fld_dst =
>> @@ -962,7 +962,7 @@ source_ok:
>>         if (!fl.fld_dst)
>>             goto out;
>>     }
>> - fl.oif = loopback_dev.ifindex;
>> + fl.oif = loopback_dev->ifindex;
>>     res.type = RTN_LOCAL;
>>     goto make_route;
>> }
>> @@ -1008,7 +1008,7 @@ source_ok:
>>     if (dev_out)
>>         dev_put(dev_out);
>>     if (dn_dev_islocal(neigh->dev, fl.fld_dst)) {
>> - dev_out = &loopback_dev;
>> + dev_out = loopback_dev;
>>         res.type = RTN_LOCAL;
>>     } else {
>>         dev_out = neigh->dev;
>> @@ -1029,7 +1029,7 @@ 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 = loopback_dev;
>> dev_hold(dev_out);
>> res.type = RTN_LOCAL;
>> goto select_source;
>> @@ -1065,7 +1065,7 @@ select_source:
>> fl.fld_src = fl.fld_dst;
>> if (dev_out)
>> dev_put(dev_out);
>> - dev_out = &loopback_dev;
>> + dev_out = loopback_dev;
>> dev_hold(dev_out);
>> fl.oif = dev_out->ifindex;
>> if (res.fi)
>> diff --git a/net/ipv4/devinet.c b/net/ipv4/devinet.c
>> index 5b77bda..808f529 100644
>> --- a/net/ipv4/devinet.c
>> +++ b/net/ipv4/devinet.c
>> @@ -203,7 +203,7 @@ static void inetdev_destroy(struct in_device *in_dev)
>> ASSERT_RTNL();
>>
>> dev = in_dev->dev;
>> - if (dev == &loopback_dev)
>> + if (dev == loopback_dev)
>> return;
>>
>> in_dev->dead = 1;
>> @@ -1058,7 +1058,7 @@ static int inetdev_event(struct notifier_block *this, unsigned long
event,
>> in_dev = inetdev_init(dev);
>> if (!in_dev)
>> return notifier_from_errno(-ENOMEM);
>> - if (dev == &loopback_dev) {
>> + if (dev == loopback_dev) {
>> IN_DEV_CONF_SET(in_dev, NOXFRM, 1);
>> IN_DEV_CONF_SET(in_dev, NOPOLICY, 1);
>> }
>> @@ -1074,7 +1074,7 @@ static int inetdev_event(struct notifier_block *this, unsigned long
event,
>> case NETDEV_UP:
>> if (dev->mtu < 68)
>> break;
>> - if (dev == &loopback_dev) {
>> + if (dev == loopback_dev) {
>> struct in_ifaddr *ifa;
>> if ((ifa = inet_alloc_ifa()) != NULL) {
>> ifa->ifa_local =
>> diff --git a/net/ipv4/ipconfig.c b/net/ipv4/ipconfig.c

```

```

>> index c5b2470..3ec7690 100644
>> --- a/net/ipv4/ipconfig.c
>> +++ b/net/ipv4/ipconfig.c
>> @@ -189,11 +189,11 @@ static int __init ic_open_devs(void)
>>  rtnl_lock();
>>
>>  /* bring loopback device up first */
>> - if (dev_change_flags(&loopback_dev, loopback_dev.flags | IFF_UP) < 0)
>> - printk(KERN_ERR "IP-Config: Failed to open %s\n", loopback_dev.name);
>> + if (dev_change_flags(loopback_dev, loopback_dev->flags | IFF_UP) < 0)
>> + printk(KERN_ERR "IP-Config: Failed to open %s\n", loopback_dev->name);
>>
>> for_each_netdev(dev) {
>> - if (dev == &loopback_dev)
>> + if (dev == loopback_dev)
>>     continue;
>>     if (user_dev_name[0] ? !strcmp(dev->name, user_dev_name) :
>>         (!(dev->flags & IFF_LOOPBACK)) &&
>> diff --git a/net/ipv4/ipvs/ip_vs_core.c b/net/ipv4/ipvs/ip_vs_core.c
>> index f005a2f..7450326 100644
>> --- a/net/ipv4/ipvs/ip_vs_core.c
>> +++ b/net/ipv4/ipvs/ip_vs_core.c
>> @@ -961,7 +961,7 @@ ip_vs_in(unsigned int hooknum, struct sk_buff **pskb,
>>  * ... don't know why 1st test DOES NOT include 2nd (?)
>>  */
>>  if (unlikely(skb->pkt_type != PACKET_HOST
>>      || skb->dev == &loopback_dev || skb->sk)) {
>> +    || skb->dev == loopback_dev || skb->sk)) {
>>  IP_VS_DBG(12, "packet type=%d proto=%d daddr=%d.%d.%d.%d ignored\n",
>>      skb->pkt_type,
>>      ip_hdr(skb)->protocol,
>> diff --git a/net/ipv4/route.c b/net/ipv4/route.c
>> index c7ca94b..4f13385 100644
>> --- a/net/ipv4/route.c
>> +++ b/net/ipv4/route.c
>> @@ -1404,8 +1404,8 @@ 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 != loopback_dev && idev && idev->dev == dev) {
>> + struct in_device *loopback_idev = in_dev_get(loopback_dev);
>>  if (loopback_idev) {
>>      rt->idev = loopback_idev;
>>      in_dev_put(idev);
>> @@ -1557,7 +1557,7 @@ 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 = loopback_dev;
>> dev_hold(rth->u.dst.dev);
>> rth->idev = in_dev_get(rth->u.dst.dev);
>> rth->fl.oif = 0;
>> @@ -1814,7 +1814,7 @@ 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,
>> + loopback_dev->ifindex,
>> dev, &spec_dst, &itag);
>> if (result < 0)
>> goto martian_source;
>> @@ -1881,7 +1881,7 @@ local_input:
>> #endif
>> rth->rt_iif =
>> rth->fl.iif = dev->ifindex;
>> - rth->u.dst.dev = &loopback_dev;
>> + rth->u.dst.dev = loopback_dev;
>> dev_hold(rth->u.dst.dev);
>> rth->idev = in_dev_get(rth->u.dst.dev);
>> rth->rt_gateway = daddr;
>> @@ -2151,7 +2151,7 @@ static int ip_route_output_slow(struct rtable **rp, const struct flowi
*oldflp)
>> RT_SCOPE_UNIVERSE),
>> },
>> .mark = oldflp->mark,
>> - .iif = loopback_dev.ifindex,
>> + .iif = loopback_dev->ifindex,
>> .oif = oldflp->oif };
>> struct fib_result res;
>> unsigned flags = 0;
>> @@ -2245,9 +2245,9 @@ 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 = loopback_dev;
>> dev_hold(dev_out);
>> - fl.oif = loopback_dev.ifindex;
>> + fl.oif = loopback_dev->ifindex;

```



```

>> res.type = RTN_LOCAL;
>> flags |= RTCF_LOCAL;
>> goto make_route;
>> @@ -2292,7 +2292,7 @@ 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 = 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 4ff8ed3..29ab3de 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(loopback_dev);
>> BUG_ON(!loopback_idev);
>>
>> do {
>> diff --git a/net/ipv6/addrconf.c b/net/ipv6/addrconf.c
>> index 91ef3be..d806f89 100644
>> --- a/net/ipv6/addrconf.c
>> +++ b/net/ipv6/addrconf.c
>> @@ -2399,7 +2399,7 @@ static int addrconf_ifdown(struct net_device *dev, int how)
>>
>> ASSERT_RTNL();
>>
>> - if (dev == &loopback_dev && how == 1)
>> + if (dev == loopback_dev && how == 1)
>> how = 0;
>>
>> rt6_ifdown(dev);
>> @@ -4203,16 +4203,19 @@ 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(loopback_dev))
>> err = -ENOMEM;
>> rtnl_unlock();

```

```

>> if (err)
>>     return err;
>>
>> - ip6_null_entry.rt6i_iddev = in6_dev_get(&loopback_dev);
>> + ip6_null_entry.u.dst.dev = loopback_dev;
>> + ip6_null_entry.rt6i_iddev = in6_dev_get(loopback_dev);
>> #ifdef CONFIG_IPV6_MULTIPLE_TABLES
>> - ip6_prohibit_entry.rt6i_iddev = in6_dev_get(&loopback_dev);
>> - ip6_blk_hole_entry.rt6i_iddev = in6_dev_get(&loopback_dev);
>> + ip6_prohibit_entry.u.dst.dev = loopback_dev;
>> + ip6_prohibit_entry.rt6i_iddev = in6_dev_get(loopback_dev);
>> + ip6_blk_hole_entry.u.dst.dev = loopback_dev;
>> + ip6_blk_hole_entry.rt6i_iddev = in6_dev_get(loopback_dev);
>> #endif
>>
>> register_netdevice_notifier(&ipv6_dev_notf);
>> @@ -4267,7 +4270,7 @@ void __exit addrconf_cleanup(void)
>>     continue;
>>     addrconf_ifdown(dev, 1);
>> }
>> - addrconf_ifdown(&loopback_dev, 2);
>> + addrconf_ifdown(loopback_dev, 2);
>>
>> /*
>>  * Check hash table.
>> diff --git a/net/ipv6/ip6_input.c b/net/ipv6/ip6_input.c
>> index 30a5cb1..15d7910 100644
>> --- a/net/ipv6/ip6_input.c
>> +++ b/net/ipv6/ip6_input.c
>> @@ -86,7 +86,7 @@ int ipv6_rcv(struct sk_buff *skb, struct net_device *dev, struct
packet_type *pt
>>  *
>>  * BTW, when we send a packet for our own local address on a
>>  * non-loopback interface (e.g. ethX), it is being delivered
>> - * via the loopback interface (lo) here; skb->dev = &loopback_dev.
>> + * via the loopback interface (lo) here; skb->dev = loopback_dev.
>>  * It, however, should be considered as if it is being
>>  * arrived via the sending interface (ethX), because of the
>>  * nature of scoping architecture. --yoshfujii
>> diff --git a/net/ipv6/netfilter/ip6t_REJECT.c b/net/ipv6/netfilter/ip6t_REJECT.c
>> index 2f487cd..5086053 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 = 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 55ea80f..3f5c65f 100644
>> --- a/net/ipv6/route.c
>> +++ b/net/ipv6/route.c
>> @@ -137,7 +137,6 @@ struct rt6_info ip6_null_entry = {
>> .dst = {
>> .__refcnt = ATOMIC_INIT(1),
>> .__use = 1,
>> - .dev = &loopback_dev,
>> .obsolete = -1,
>> .error = -ENETUNREACH,
>> .metrics = { [RTAX_HOPLIMIT - 1] = 255, },
>> @@ -163,7 +162,6 @@ struct rt6_info ip6_prohibit_entry = {
>> .dst = {
>> .__refcnt = ATOMIC_INIT(1),
>> .__use = 1,
>> - .dev = &loopback_dev,
>> .obsolete = -1,
>> .error = -EACCES,
>> .metrics = { [RTAX_HOPLIMIT - 1] = 255, },
>> @@ -183,7 +181,6 @@ struct rt6_info ip6_blk_hole_entry = {
>> .dst = {
>> .__refcnt = ATOMIC_INIT(1),
>> .__use = 1,
>> - .dev = &loopback_dev,
>> .obsolete = -1,
>> .error = -EINVAL,
>> .metrics = { [RTAX_HOPLIMIT - 1] = 255, },
>> @@ -223,8 +220,8 @@ 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 != loopback_dev && idev != NULL && idev->dev == dev) {
>> + struct inet6_dev *loopback_idev = in6_dev_get(loopback_dev);
>> if (loopback_idev != NULL) {
>> rt->rt6i_idev = loopback_idev;
>> in6_dev_put(idev);
>> @@ -1187,12 +1184,12 @@ 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 != loopback_dev) {
>>     if (dev) {
>>         dev_put(dev);
>>         in6_dev_put(idev);
>>     }
>> - dev = &loopback_dev;
>> + dev = loopback_dev;
>>     dev_hold(dev);
>>     idev = in6_dev_get(dev);
>>     if (!idev) {
>> @@ -1896,13 +1893,13 @@ struct rt6_info *addrconf_dst_alloc(struct inet6_dev *idev,
>>     if (rt == NULL)
>>         return ERR_PTR(-ENOMEM);
>>
>> - dev_hold(&loopback_dev);
>> + dev_hold(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 = 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 3ec0c47..cc07216 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(loopback_dev);
>>         BUG_ON(!loopback_idev);
>>
>>     do {
>> diff --git a/net/xfrm/xfrm_policy.c b/net/xfrm/xfrm_policy.c
>> index c7b2503..ca37eec 100644
>> --- a/net/xfrm/xfrm_policy.c
>> +++ b/net/xfrm/xfrm_policy.c
>> @@ -1947,8 +1947,8 @@ 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;  
>> - dev_hold(&loopback_dev);  
>> + dst->dev = loopback_dev;  
>> + dev_hold(dst->dev);  
>> dev_put(dev);  
>> }  
>> }  
>  
>  
> _____  
> Containers mailing list  
> Containers@lists.linux-foundation.org  
> https://lists.linux-foundation.org/mailman/listinfo/containers  
>
```

--

Benjamin Thery - BULL/DT/Open Software R&D

<http://www.bull.com>

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