Subject: [PATCH 9/12 net-2.6.25] [NETNS]: Add namespace parameter to ip_route_output_key.
Posted by den on Tue, 22 Jan 2008 15:59:26 GMT
View Forum Message <> Reply to Message

Needed to propagate it down to the ip_route_output_flow.

Signed-off-by: Denis V. Lunev <den@openvz.org>
---
 drivers/infiniband/core/addr.c | 4 ++--
 drivers/net/bonding/bond_main.c | 2 +-
 include/net/route.h | 2 +-
 net/atm/clip.c | 2 +-
 net/bridge/br_netfilter.c | 2 +-
 net/ipv4/arp.c | 6 +++---
 net/ipv4/icmp.c | 4 ++--
 net/ipv4/igmp.c | 6 +++---
 net/ipv4/ip_gre.c | 10 +++++++-----
 net/ipv4/ip_output.c | 2 +-
 net/ipv4/ipip.c | 8 ++++----
 net/ipv4/ipmr.c | 4 ++--
 net/ipv4/ipvs/ip_vs_xmit.c | 6 +++---
 net/ipv4/netfilter.c | 6 +++---
 net/ipv4/netfilter/nf_nat_rule.c | 2 +-
 net/ipv4/route.c | 6 +++---
 net/ipv4/syncookies.c | 2 +-
 net/ipv6/ip6_tunnel.c | 4 ++--
 net/ipv6/sit.c | 4 ++--
 net/rxrpc/ar-peer.c | 2 +-
 net/sctp/protocol.c | 4 ++--
 21 files changed, 44 insertions(+), 44 deletions(-)

diff --git a/drivers/infiniband/core/addr.c b/drivers/infiniband/core/addr.c
index 963177e..a58ad8a 100644
--- a/drivers/infiniband/core/addr.c
+++ b/drivers/infiniband/core/addr.c
@@ -158,7 +158,7 @@ static void addr_send_arp(struct sockaddr_in *dst_in)

  memset(&fl, 0, sizeof fl);
  fl.nl_u.ip4_u.daddr = dst_ip;
- if (ip_route_output_key(&rt, &fl))
+ if (ip_route_output_key(&init_net, &rt, &fl))
   return;

  neigh_event_send(rt->u.dst.neighbour, NULL);
@@ -179,7 +179,7 @@ static int addr_resolve_remote(struct sockaddr_in *src_in,
  memset(&fl, 0, sizeof fl);
  fl.nl_u.ip4_u.daddr = dst_ip;

```
  fl.nl_u.ip4_u.saddr = src_ip;
- ret = ip_route_output_key(&rt, &fl);
+ ret = ip_route_output_key(&init_net, &rt, &fl);
  if (ret)
   goto out;

diff --git a/drivers/net/bonding/bond_main.c b/drivers/net/bonding/bond_main.c
index b0b2603..7a7be20 100644
--- a/drivers/net/bonding/bond_main.c
+++ b/drivers/net/bonding/bond_main.c
@@ -2513,7 +2513,7 @@ static void bond_arp_send_all(struct bonding *bond, struct slave
*slave)
   fl.fl4_dst = targets[i];
   fl.fl4_tos = RTO_ONLINK;

-  rv = ip_route_output_key(&rt, &fl);
+  rv = ip_route_output_key(&init_net, &rt, &fl);
   if (rv) {
    if (net_ratelimit()) {
     printk(KERN_WARNING DRV_NAME
diff --git a/include/net/route.h b/include/net/route.h
index 6b970d7..d9b876a 100644
--- a/include/net/route.h
+++ b/include/net/route.h
@@ -111,7 +111,7 @@ extern void  ip_rt_redirect(__be32 old_gw, __be32 dst, __be32 new_gw,
           __be32 src, struct net_device *dev);
 extern void  rt_cache_flush(int how);
 extern int  __ip_route_output_key(struct net *, struct rtable **, const struct flowi *flp);
-extern int  ip_route_output_key(struct rtable **, struct flowi *flp);
+extern int  ip_route_output_key(struct net *, struct rtable **, struct flowi *flp);
 extern int  ip_route_output_flow(struct net *, struct rtable **rp, struct flowi *flp, struct sock *sk, int
flags);
 extern int  ip_route_input(struct sk_buff*, __be32 dst, __be32 src, u8 tos, struct net_device
*devin);
 extern unsigned short ip_rt_frag_needed(struct iphdr *iph, unsigned short new_mtu);
diff --git a/net/atm/clip.c b/net/atm/clip.c
index 45e0862..86b885e 100644
--- a/net/atm/clip.c
+++ b/net/atm/clip.c
@@ -534,7 +534,7 @@ static int clip_setentry(struct atm_vcc *vcc, __be32 ip)
   unlink_clip_vcc(clip_vcc);
   return 0;
  }
- error = ip_route_output_key(&rt, &fl);
+ error = ip_route_output_key(&init_net, &rt, &fl);
  if (error)
   return error;
  neigh = __neigh_lookup(&clip_tbl, &ip, rt->u.dst.dev, 1);
```

```
diff --git a/net/bridge/br_netfilter.c b/net/bridge/br_netfilter.c
index 0e884fe..d4579cf 100644
--- a/net/bridge/br_netfilter.c
+++ b/net/bridge/br_netfilter.c
@@ -336,7 +336,7 @@ static int br_nf_pre_routing_finish(struct sk_buff *skb)
    if (err != -EHOSTUNREACH || !in_dev || IN_DEV_FORWARD(in_dev))
     goto free_skb;

-   if (!ip_route_output_key(&rt, &fl)) {
+   if (!ip_route_output_key(&init_net, &rt, &fl)) {
     /* - Bridged-and-DNAT'ed traffic doesn't
      *   require ip_forwarding. */
     if (((struct dst_entry *)rt)->dev == dev) {
diff --git a/net/ipv4/arp.c b/net/ipv4/arp.c
index a44ff1a..a3cfd04 100644
--- a/net/ipv4/arp.c
+++ b/net/ipv4/arp.c
@@ -424,7 +424,7 @@ static int arp_filter(__be32 sip, __be32 tip, struct net_device *dev)
  int flag = 0;
  /*unsigned long now; */

- if (ip_route_output_key(&rt, &fl) < 0)
+ if (ip_route_output_key(&init_net, &rt, &fl) < 0)
   return 1;
  if (rt->u.dst.dev != dev) {
   NET_INC_STATS_BH(LINUX_MIB_ARPFILTER);
@@ -1002,7 +1002,7 @@ static int arp_req_set(struct net *net, struct arpreq *r,
   struct flowi fl = { .nl_u = { .ip4_u = { .daddr = ip,
         .tos = RTO_ONLINK } } };
   struct rtable * rt;
-  if ((err = ip_route_output_key(&rt, &fl)) != 0)
+  if ((err = ip_route_output_key(net, &rt, &fl)) != 0)
    return err;
   dev = rt->u.dst.dev;
   ip_rt_put(rt);
@@ -1109,7 +1109,7 @@ static int arp_req_delete(struct net *net, struct arpreq *r,
   struct flowi fl = { .nl_u = { .ip4_u = { .daddr = ip,
         .tos = RTO_ONLINK } } };
   struct rtable * rt;
-  if ((err = ip_route_output_key(&rt, &fl)) != 0)
+  if ((err = ip_route_output_key(net, &rt, &fl)) != 0)
    return err;
   dev = rt->u.dst.dev;
   ip_rt_put(rt);
diff --git a/net/ipv4/icmp.c b/net/ipv4/icmp.c
index 21422bf..c04aac5 100644
--- a/net/ipv4/icmp.c
+++ b/net/ipv4/icmp.c
```

```
@@ -404,7 +404,7 @@ static void icmp_reply(struct icmp_bxm *icmp_param, struct sk_buff
*skb)
       .tos = RT_TOS(ip_hdr(skb)->tos) } },
         .proto = IPPROTO_ICMP };
   security_skb_classify_flow(skb, &fl);
-  if (ip_route_output_key(&rt, &fl))
+  if (ip_route_output_key(&init_net, &rt, &fl))
    goto out_unlock;
  }
  if (icmpv4_xrlim_allow(rt, icmp_param->data.icmph.type,
@@ -598,7 +598,7 @@ void icmp_send(struct sk_buff *skb_in, int type, int code, __be32 info)
    struct dst_entry *odst;

    fl2.fl4_dst = fl.fl4_src;
-   if (ip_route_output_key(&rt2, &fl2))
+   if (ip_route_output_key(&init_net, &rt2, &fl2))
     goto out_unlock;

    /* Ugh! */
diff --git a/net/ipv4/igmp.c b/net/ipv4/igmp.c
index 1f5314c..994648b 100644
--- a/net/ipv4/igmp.c
+++ b/net/ipv4/igmp.c
@@ -301,7 +301,7 @@ static struct sk_buff *igmpv3_newpack(struct net_device *dev, int size)
        .nl_u = { .ip4_u = {
        .daddr = IGMPV3_ALL_MCR } },
        .proto = IPPROTO_IGMP };
-  if (ip_route_output_key(&rt, &fl)) {
+  if (ip_route_output_key(&init_net, &rt, &fl)) {
    kfree_skb(skb);
    return NULL;
   }
@@ -645,7 +645,7 @@ static int igmp_send_report(struct in_device *in_dev, struct ip_mc_list
*pmc,
   struct flowi fl = { .oif = dev->ifindex,
        .nl_u = { .ip4_u = { .daddr = dst } },
        .proto = IPPROTO_IGMP };
-  if (ip_route_output_key(&rt, &fl))
+  if (ip_route_output_key(&init_net, &rt, &fl))
    return -1;
  }
  if (rt->rt_src == 0) {
@@ -1401,7 +1401,7 @@ static struct in_device * ip_mc_find_dev(struct ip_mreqn *imr)
   dev_put(dev);
  }

- if (!dev && !ip_route_output_key(&rt, &fl)) {
+ if (!dev && !ip_route_output_key(&init_net, &rt, &fl)) {
```

```
   dev = rt->u.dst.dev;
   ip_rt_put(rt);
  }
diff --git a/net/ipv4/ip_gre.c b/net/ipv4/ip_gre.c
index a74983d..63f6917 100644
--- a/net/ipv4/ip_gre.c
+++ b/net/ipv4/ip_gre.c
@@ -480,7 +480,7 @@ out:
  fl.fl4_dst = eiph->saddr;
  fl.fl4_tos = RT_TOS(eiph->tos);
  fl.proto = IPPROTO_GRE;
- if (ip_route_output_key(&rt, &fl)) {
+ if (ip_route_output_key(&init_net, &rt, &fl)) {
   kfree_skb(skb2);
   return;
  }
@@ -493,7 +493,7 @@ out:
   fl.fl4_dst = eiph->daddr;
   fl.fl4_src = eiph->saddr;
   fl.fl4_tos = eiph->tos;
-  if (ip_route_output_key(&rt, &fl) ||
+  if (ip_route_output_key(&init_net, &rt, &fl) ||
       rt->u.dst.dev->type != ARPHRD_IPGRE) {
    ip_rt_put(rt);
    kfree_skb(skb2);
@@ -748,7 +748,7 @@ static int ipgre_tunnel_xmit(struct sk_buff *skb, struct net_device *dev)
       .saddr = tiph->saddr,
       .tos = RT_TOS(tos) } },
         .proto = IPPROTO_GRE };
-  if (ip_route_output_key(&rt, &fl)) {
+  if (ip_route_output_key(&init_net, &rt, &fl)) {
    tunnel->stat.tx_carrier_errors++;
    goto tx_error;
   }
@@ -921,7 +921,7 @@ static void ipgre_tunnel_bind_dev(struct net_device *dev)
       .tos = RT_TOS(iph->tos) } },
         .proto = IPPROTO_GRE };
   struct rtable *rt;
-  if (!ip_route_output_key(&rt, &fl)) {
+  if (!ip_route_output_key(&init_net, &rt, &fl)) {
    tdev = rt->u.dst.dev;
    ip_rt_put(rt);
   }
@@ -1177,7 +1177,7 @@ static int ipgre_open(struct net_device *dev)
       .tos = RT_TOS(t->parms.iph.tos) } },
         .proto = IPPROTO_GRE };
   struct rtable *rt;
-  if (ip_route_output_key(&rt, &fl))
```

```
+  if (ip_route_output_key(&init_net, &rt, &fl))
    return -EADDRNOTAVAIL;
   dev = rt->u.dst.dev;
   ip_rt_put(rt);
diff --git a/net/ipv4/ip_output.c b/net/ipv4/ip_output.c
index dc56e40..6a5b839 100644
--- a/net/ipv4/ip_output.c
+++ b/net/ipv4/ip_output.c
@@ -1377,7 +1377,7 @@ void ip_send_reply(struct sock *sk, struct sk_buff *skb, struct
ip_reply_arg *ar
         .dport = tcp_hdr(skb)->source } },
         .proto = sk->sk_protocol };
   security_skb_classify_flow(skb, &fl);
-  if (ip_route_output_key(&rt, &fl))
+  if (ip_route_output_key(&init_net, &rt, &fl))
    return;
  }

diff --git a/net/ipv4/ipip.c b/net/ipv4/ipip.c
index 160535b..da28158 100644
--- a/net/ipv4/ipip.c
+++ b/net/ipv4/ipip.c
@@ -405,7 +405,7 @@ out:
  fl.fl4_daddr = eiph->saddr;
  fl.fl4_tos = RT_TOS(eiph->tos);
  fl.proto = IPPROTO_IPIP;
- if (ip_route_output_key(&rt, &key)) {
+ if (ip_route_output_key(&init_net, &rt, &key)) {
   kfree_skb(skb2);
   return 0;
  }
@@ -418,7 +418,7 @@ out:
   fl.fl4_daddr = eiph->daddr;
   fl.fl4_src = eiph->saddr;
   fl.fl4_tos = eiph->tos;
-  if (ip_route_output_key(&rt, &fl) ||
+  if (ip_route_output_key(&init_net, &rt, &fl) ||
       rt->u.dst.dev->type != ARPHRD_TUNNEL) {
    ip_rt_put(rt);
    kfree_skb(skb2);
@@ -547,7 +547,7 @@ static int ipip_tunnel_xmit(struct sk_buff *skb, struct net_device *dev)
       .saddr = tiph->saddr,
       .tos = RT_TOS(tos) } },
         .proto = IPPROTO_IPIP };
-  if (ip_route_output_key(&rt, &fl)) {
+  if (ip_route_output_key(&init_net, &rt, &fl)) {
    tunnel->stat.tx_carrier_errors++;
    goto tx_error_icmp;
```

```
  }
@@ -668,7 +668,7 @@ static void ipip_tunnel_bind_dev(struct net_device *dev)
       .tos = RT_TOS(iph->tos) } },
         .proto = IPPROTO_IPIP };
   struct rtable *rt;
-  if (!ip_route_output_key(&rt, &fl)) {
+  if (!ip_route_output_key(&init_net, &rt, &fl)) {
    tdev = rt->u.dst.dev;
    ip_rt_put(rt);
   }
diff --git a/net/ipv4/ipmr.c b/net/ipv4/ipmr.c
index 2212717..a94f52c 100644
--- a/net/ipv4/ipmr.c
+++ b/net/ipv4/ipmr.c
@@ -1185,7 +1185,7 @@ static void ipmr_queue_xmit(struct sk_buff *skb, struct mfc_cache *c,
int vifi)
       .saddr = vif->local,
       .tos = RT_TOS(iph->tos) } },
         .proto = IPPROTO_IPIP };
-  if (ip_route_output_key(&rt, &fl))
+  if (ip_route_output_key(&init_net, &rt, &fl))
    goto out_free;
   encap = sizeof(struct iphdr);
  } else {
@@ -1194,7 +1194,7 @@ static void ipmr_queue_xmit(struct sk_buff *skb, struct mfc_cache *c,
int vifi)
           { .daddr = iph->daddr,
       .tos = RT_TOS(iph->tos) } },
         .proto = IPPROTO_IPIP };
-  if (ip_route_output_key(&rt, &fl))
+  if (ip_route_output_key(&init_net, &rt, &fl))
    goto out_free;
  }

diff --git a/net/ipv4/ipvs/ip_vs_xmit.c b/net/ipv4/ipvs/ip_vs_xmit.c
index 8436bf8..f63006c 100644
--- a/net/ipv4/ipvs/ip_vs_xmit.c
+++ b/net/ipv4/ipvs/ip_vs_xmit.c
@@ -78,7 +78,7 @@ __ip_vs_get_out_rt(struct ip_vs_conn *cp, u32 rtos)
       .tos = rtos, } },
    };

-   if (ip_route_output_key(&rt, &fl)) {
+   if (ip_route_output_key(&init_net, &rt, &fl)) {
     spin_unlock(&dest->dst_lock);
     IP_VS_DBG_RL("ip_route_output error, "
          "dest: %u.%u.%u.%u\n",
@@ -101,7 +101,7 @@ __ip_vs_get_out_rt(struct ip_vs_conn *cp, u32 rtos)
```

```
      .tos = rtos, } },
   };

-  if (ip_route_output_key(&rt, &fl)) {
+  if (ip_route_output_key(&init_net, &rt, &fl)) {
     IP_VS_DBG_RL("ip_route_output error, dest: "
          "%u.%u.%u.%u\n", NIPQUAD(cp->daddr));
     return NULL;
@@ -170,7 +170,7 @@ ip_vs_bypass_xmit(struct sk_buff *skb, struct ip_vs_conn *cp,

  EnterFunction(10);

- if (ip_route_output_key(&rt, &fl)) {
+ if (ip_route_output_key(&init_net, &rt, &fl)) {
    IP_VS_DBG_RL("ip_vs_bypass_xmit(): ip_route_output error, "
         "dest: %u.%u.%u.%u\n", NIPQUAD(iph->daddr));
    goto tx_error_icmp;
diff --git a/net/ipv4/netfilter.c b/net/ipv4/netfilter.c
index 6322155..9a904c6 100644
--- a/net/ipv4/netfilter.c
+++ b/net/ipv4/netfilter.c
@@ -33,7 +33,7 @@ int ip_route_me_harder(struct sk_buff *skb, unsigned addr_type)
    fl.nl_u.ip4_u.tos = RT_TOS(iph->tos);
    fl.oif = skb->sk ? skb->sk->sk_bound_dev_if : 0;
    fl.mark = skb->mark;
-   if (ip_route_output_key(&rt, &fl) != 0)
+   if (ip_route_output_key(&init_net, &rt, &fl) != 0)
     return -1;

    /* Drop old route. */
@@ -43,7 +43,7 @@ int ip_route_me_harder(struct sk_buff *skb, unsigned addr_type)
    /* non-local src, find valid iif to satisfy
     * rp-filter when calling ip_route_input. */
    fl.nl_u.ip4_u.daddr = iph->saddr;
-   if (ip_route_output_key(&rt, &fl) != 0)
+   if (ip_route_output_key(&init_net, &rt, &fl) != 0)
     return -1;

    odst = skb->dst;
@@ -187,7 +187,7 @@ EXPORT_SYMBOL(nf_ip_checksum);

 static int nf_ip_route(struct dst_entry **dst, struct flowi *fl)
 {
- return ip_route_output_key((struct rtable **)dst, fl);
+ return ip_route_output_key(&init_net, (struct rtable **)dst, fl);
 }

 static const struct nf_afinfo nf_ip_afinfo = {
```

```
diff --git a/net/ipv4/netfilter/nf_nat_rule.c b/net/ipv4/netfilter/nf_nat_rule.c
index 4391aec..5191822 100644
--- a/net/ipv4/netfilter/nf_nat_rule.c
+++ b/net/ipv4/netfilter/nf_nat_rule.c
@@ -97,7 +97,7 @@ static void warn_if_extra_mangle(__be32 dstip, __be32 srcip)
  struct flowi fl = { .nl_u = { .ip4_u = { .daddr = dstip } } };
  struct rtable *rt;

- if (ip_route_output_key(&rt, &fl) != 0)
+ if (ip_route_output_key(&init_net, &rt, &fl) != 0)
   return;

  if (rt->rt_src != srcip && !warned) {
diff --git a/net/ipv4/route.c b/net/ipv4/route.c
index 58ad12a..87076c6 100644
--- a/net/ipv4/route.c
+++ b/net/ipv4/route.c
@@ -2558,9 +2558,9 @@ int ip_route_output_flow(struct net *net, struct rtable **rp, struct flowi
*flp,

 EXPORT_SYMBOL_GPL(ip_route_output_flow);

-int ip_route_output_key(struct rtable **rp, struct flowi *flp)
+int ip_route_output_key(struct net *net, struct rtable **rp, struct flowi *flp)
 {
- return ip_route_output_flow(&init_net, rp, flp, NULL, 0);
+ return ip_route_output_flow(net, rp, flp, NULL, 0);
 }

 static int rt_fill_info(struct sk_buff *skb, u32 pid, u32 seq, int event,
@@ -2727,7 +2727,7 @@ static int inet_rtm_getroute(struct sk_buff *in_skb, struct nlmsghdr*
nlh, void
    },
    .oif = tb[RTA_OIF] ? nla_get_u32(tb[RTA_OIF]) : 0,
   };
-  err = ip_route_output_key(&rt, &fl);
+  err = ip_route_output_key(&init_net, &rt, &fl);
  }

  if (err)
diff --git a/net/ipv4/syncookies.c b/net/ipv4/syncookies.c
index 2da1be0..f470fe4 100644
--- a/net/ipv4/syncookies.c
+++ b/net/ipv4/syncookies.c
@@ -264,7 +264,7 @@ struct sock *cookie_v4_check(struct sock *sk, struct sk_buff *skb,
            { .sport = th->dest,
        .dport = th->source } } };
   security_req_classify_flow(req, &fl);
```

```
-  if (ip_route_output_key(&rt, &fl)) {
+  if (ip_route_output_key(&init_net, &rt, &fl)) {
     reqsk_free(req);
     goto out;
   }
diff --git a/net/ipv6/ip6_tunnel.c b/net/ipv6/ip6_tunnel.c
index 425c9ae..9031e52 100644
--- a/net/ipv6/ip6_tunnel.c
+++ b/net/ipv6/ip6_tunnel.c
@@ -533,7 +533,7 @@ ip4ip6_err(struct sk_buff *skb, struct inet6_skb_parm *opt,
  fl.fl4_dst = eiph->saddr;
  fl.fl4_tos = RT_TOS(eiph->tos);
  fl.proto = IPPROTO_IPIP;
- if (ip_route_output_key(&rt, &fl))
+ if (ip_route_output_key(&init_net, &rt, &fl))
   goto out;

  skb2->dev = rt->u.dst.dev;
@@ -545,7 +545,7 @@ ip4ip6_err(struct sk_buff *skb, struct inet6_skb_parm *opt,
   fl.fl4_dst = eiph->daddr;
   fl.fl4_src = eiph->saddr;
   fl.fl4_tos = eiph->tos;
-  if (ip_route_output_key(&rt, &fl) ||
+  if (ip_route_output_key(&init_net, &rt, &fl) ||
       rt->u.dst.dev->type != ARPHRD_TUNNEL) {
    ip_rt_put(rt);
    goto out;
diff --git a/net/ipv6/sit.c b/net/ipv6/sit.c
index 1c6fddb..e77239d 100644
--- a/net/ipv6/sit.c
+++ b/net/ipv6/sit.c
@@ -557,7 +557,7 @@ static int ipip6_tunnel_xmit(struct sk_buff *skb, struct net_device *dev)
       .tos = RT_TOS(tos) } },
         .oif = tunnel->parms.link,
         .proto = IPPROTO_IPV6 };
-  if (ip_route_output_key(&rt, &fl)) {
+  if (ip_route_output_key(&init_net, &rt, &fl)) {
    tunnel->stat.tx_carrier_errors++;
    goto tx_error_icmp;
   }
@@ -686,7 +686,7 @@ static void ipip6_tunnel_bind_dev(struct net_device *dev)
         .oif = tunnel->parms.link,
         .proto = IPPROTO_IPV6 };
   struct rtable *rt;
-  if (!ip_route_output_key(&rt, &fl)) {
+  if (!ip_route_output_key(&init_net, &rt, &fl)) {
    tdev = rt->u.dst.dev;
    ip_rt_put(rt);
```

```
   }
diff --git a/net/rxrpc/ar-peer.c b/net/rxrpc/ar-peer.c
index 90fa107..2abe208 100644
--- a/net/rxrpc/ar-peer.c
+++ b/net/rxrpc/ar-peer.c
@@ -57,7 +57,7 @@ static void rxrpc_assess_MTU_size(struct rxrpc_peer *peer)
   BUG();
  }

- ret = ip_route_output_key(&rt, &fl);
+ ret = ip_route_output_key(&init_net, &rt, &fl);
  if (ret < 0) {
   _leave(" [route err %d]", ret);
   return;
diff --git a/net/sctp/protocol.c b/net/sctp/protocol.c
index 3f7def2..1339742 100644
--- a/net/sctp/protocol.c
+++ b/net/sctp/protocol.c
@@ -454,7 +454,7 @@ static struct dst_entry *sctp_v4_get_dst(struct sctp_association *asoc,
      __FUNCTION__, NIPQUAD(fl.fl4_dst),
      NIPQUAD(fl.fl4_src));

- if (!ip_route_output_key(&rt, &fl)) {
+ if (!ip_route_output_key(&init_net, &rt, &fl)) {
   dst = &rt->u.dst;
  }

@@ -497,7 +497,7 @@ static struct dst_entry *sctp_v4_get_dst(struct sctp_association *asoc,
   if ((laddr->state == SCTP_ADDR_SRC) &&
       (AF_INET == laddr->a.sa.sa_family)) {
    fl.fl4_src = laddr->a.v4.sin_addr.s_addr;
-   if (!ip_route_output_key(&rt, &fl)) {
+   if (!ip_route_output_key(&init_net, &rt, &fl)) {
     dst = &rt->u.dst;
     goto out_unlock;
    }
--
1.5.3.rc5
```