Subject: [PATCH 1/11 2.6.26] [NETNS]: Process ARP in the context of the correct namespace.
Posted by den on Mon, 24 Mar 2008 14:36:02 GMT
View Forum Message <> Reply to Message

Get namespace from a device and pass it to the routing engine. Enable ARP
packet processing and device notifiers after that.

Signed-off-by: Denis V. Lunev <den@openvz.org>

```
---
 net/ipv4/arp.c |   23 +++++++++--------------
 1 files changed, 9 insertions(+), 14 deletions(-)

diff --git a/net/ipv4/arp.c b/net/ipv4/arp.c
index efe01df..6d90ec5 100644
--- a/net/ipv4/arp.c
+++ b/net/ipv4/arp.c
@@ -242,7 +242,7 @@ static int arp_constructor(struct neighbour *neigh)
   return -EINVAL;
  }

- neigh->type = inet_addr_type(&init_net, addr);
+ neigh->type = inet_addr_type(dev->nd_net, addr);

  parms = in_dev->arp_parms;
  __neigh_parms_put(neigh->parms);
@@ -341,14 +341,14 @@ static void arp_solicit(struct neighbour *neigh, struct sk_buff *skb)
  switch (IN_DEV_ARP_ANNOUNCE(in_dev)) {
  default:
  case 0:  /* By default announce any local IP */
-  if (skb && inet_addr_type(&init_net, ip_hdr(skb)->saddr) == RTN_LOCAL)
+  if (skb && inet_addr_type(dev->nd_net, ip_hdr(skb)->saddr) == RTN_LOCAL)
    saddr = ip_hdr(skb)->saddr;
   break;
  case 1:  /* Restrict announcements of saddr in same subnet */
   if (!skb)
    break;
   saddr = ip_hdr(skb)->saddr;
-  if (inet_addr_type(&init_net, saddr) == RTN_LOCAL) {
+  if (inet_addr_type(dev->nd_net, saddr) == RTN_LOCAL) {
    /* saddr should be known to target */
    if (inet_addr_onlink(in_dev, target, saddr))
     break;
@@ -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(&init_net, &rt, &fl) < 0)
```

```
+ if (ip_route_output_key(dev->nd_net, &rt, &fl) < 0)
   return 1;
  if (rt->u.dst.dev != dev) {
   NET_INC_STATS_BH(LINUX_MIB_ARPFILTER);
@@ -477,7 +477,7 @@ int arp_find(unsigned char *haddr, struct sk_buff *skb)

  paddr = skb->rtable->rt_gateway;

- if (arp_set_predefined(inet_addr_type(&init_net, paddr), haddr, paddr, dev))
+ if (arp_set_predefined(inet_addr_type(dev->nd_net, paddr), haddr, paddr, dev))
   return 0;

  n = __neigh_lookup(&arp_tbl, &paddr, dev, 1);
@@ -709,6 +709,7 @@ static int arp_process(struct sk_buff *skb)
  u16 dev_type = dev->type;
  int addr_type;
  struct neighbour *n;
+ struct net *net = dev->nd_net;

  /* arp_rcv below verifies the ARP header and verifies the device
   * is ARP'able.
@@ -804,7 +805,7 @@ static int arp_process(struct sk_buff *skb)
  /* Special case: IPv4 duplicate address detection packet (RFC2131) */
  if (sip == 0) {
   if (arp->ar_op == htons(ARPOP_REQUEST) &&
-      inet_addr_type(&init_net, tip) == RTN_LOCAL &&
+      inet_addr_type(net, tip) == RTN_LOCAL &&
       !arp_ignore(in_dev, sip, tip))
    arp_send(ARPOP_REPLY, ETH_P_ARP, sip, dev, tip, sha,
      dev->dev_addr, sha);
@@ -834,7 +835,7 @@ static int arp_process(struct sk_buff *skb)
    goto out;
   } else if (IN_DEV_FORWARD(in_dev)) {
       if (addr_type == RTN_UNICAST  && rt->u.dst.dev != dev &&
-       (arp_fwd_proxy(in_dev, rt) || pneigh_lookup(&arp_tbl, &init_net, &tip, dev, 0))) {
+       (arp_fwd_proxy(in_dev, rt) || pneigh_lookup(&arp_tbl, net, &tip, dev, 0))) {
     n = neigh_event_ns(&arp_tbl, sha, &sip, dev);
     if (n)
      neigh_release(n);
@@ -864,7 +865,7 @@ static int arp_process(struct sk_buff *skb)
    */
   if (n == NULL &&
       arp->ar_op == htons(ARPOP_REPLY) &&
-      inet_addr_type(&init_net, sip) == RTN_UNICAST)
+      inet_addr_type(net, sip) == RTN_UNICAST)
    n = __neigh_lookup(&arp_tbl, &sip, dev, 1);
  }
```

```
@@ -911,9 +912,6 @@ static int arp_rcv(struct sk_buff *skb, struct net_device *dev,
 {
  struct arphdr *arp;

- if (dev->nd_net != &init_net)
-  goto freeskb;
-
  /* ARP header, plus 2 device addresses, plus 2 IP addresses.  */
  if (!pskb_may_pull(skb, arp_hdr_len(dev)))
   goto freeskb;
@@ -1198,9 +1196,6 @@ static int arp_netdev_event(struct notifier_block *this, unsigned long
event, vo
 {
  struct net_device *dev = ptr;

- if (dev->nd_net != &init_net)
-  return NOTIFY_DONE;
-
  switch (event) {
  case NETDEV_CHANGEADDR:
   neigh_changeaddr(&arp_tbl, dev);
--
1.5.3.rc5
```