Subject: [patch 12/12] net namespace : Add broadcasting
Posted by Daniel Lezcano on Fri, 19 Jan 2007 15:47:26 GMT
View Forum Message <> Reply to Message

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

Broadcast packets should be delivered to l2 and all l3 childs

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

---
 include/linux/net_namespace.h | 11 ++++++++++
 net/core/net_namespace.c | 27 ++++++++++++++++++++++++++++
 net/ipv4/udp.c | 3 ++-
 3 files changed, 40 insertions(+), 1 deletion(-)

Index: 2.6.20-rc4-mm1/include/linux/net_namespace.h
===
--- 2.6.20-rc4-mm1.orig/include/linux/net_namespace.h
+++ 2.6.20-rc4-mm1/include/linux/net_namespace.h
@@ -9,6 +9,7 @@

 struct in_ifaddr;
 struct sk_buff;
+struct sock;

 struct net_namespace {
  struct kref kref;
@@ -109,6 +110,9 @@

 extern void net_ns_tag_sk_buff(struct sk_buff *skb);

+extern int net_ns_sock_is_visible(const struct sock *sk,
+   const struct net_namespace *net_ns);
+
 #define SELECT_SRC_ADDR net_ns_select_source_address

 #else /* CONFIG_NET_NS */
@@ -192,6 +196,13 @@
 {
  ;
 }
+
+static inline int net_ns_sock_is_visible(const struct sock *sk,
+   const struct net_namespace *net_ns)
+{
+ return 1;
+}

```
+
 #define SELECT_SRC_ADDR inet_select_addr

 #endif /* !CONFIG_NET_NS */
Index: 2.6.20-rc4-mm1/net/core/net_namespace.c
===================================================================
--- 2.6.20-rc4-mm1.orig/net/core/net_namespace.c
+++ 2.6.20-rc4-mm1/net/core/net_namespace.c
@@ -17,6 +17,7 @@
 #include <linux/ip.h>

 #include <net/ip_fib.h>
+#include <net/sock.h>

 struct net_namespace init_net_ns = {
  .kref = {
@@ -464,4 +465,30 @@
  struct net_namespace *net_ns = current_net_ns;
  skb->net_ns = net_ns;
 }
+
+/*
+ * This function checks if the socket is visible from the specified
+ * namespace. This is needed to ensure the broadcast and the multicast
+ * for multiple network namespace l2 and l3 to have the packets to be
+ * delivered. If we have a l3 namespace and its parent (l2 namespace)
+ * listening on a broadcast address, we should deliver the packet to
+ * both. That is done by the udp_v4_mcast_next function. But we should
+ * find a common point between sockets which are relatives to a
+ * namespace.  The common point is they have the same parent in case
+ * of l3 network namespace.
+ * @sk : the socket to be checked
+ * @net_ns : the receiving network namespace
+ * Returns: 1 if the socket is visible by the namespace, 0 otherwise.
+ */
+int net_ns_sock_is_visible(const struct sock *sk,
+      const struct net_namespace *net_ns)
+{
+ if (net_ns->level == NET_NS_LEVEL3)
+  net_ns = net_ns->parent;
+
+ if (sk->sk_net_ns->level == NET_NS_LEVEL3)
+  return sk->sk_net_ns->parent == net_ns;
+ else
+  return sk->sk_net_ns == net_ns;
+}
 #endif /* CONFIG_NET_NS */
Index: 2.6.20-rc4-mm1/net/ipv4/udp.c
```

```
===================================================================
--- 2.6.20-rc4-mm1.orig/net/ipv4/udp.c
+++ 2.6.20-rc4-mm1/net/ipv4/udp.c
@@ -309,9 +309,10 @@
        (inet->dport != rmt_port && inet->dport)  ||
        (inet->rcv_saddr && inet->rcv_saddr != loc_addr) ||
        ipv6_only_sock(s)     ||
-       !net_ns_match(sk->sk_net_ns, ns)   ||
        (s->sk_bound_dev_if && s->sk_bound_dev_if != dif))
     continue;
+  if (!net_ns_sock_is_visible(sk, ns))
+    continue;
   if (!ip_mc_sf_allow(s, loc_addr, rmt_addr, dif))
     continue;
   goto found;

--
```

_____

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