Subject: [patch 06/12] net namespace : check bind address
Posted by Daniel Lezcano on Fri, 19 Jan 2007 15:47:20 GMT
View Forum Message <> Reply to Message

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

Check the bind address is allowed. It must match ifaddr assigned to
the namespace and all derivative addresses.

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

---
 include/linux/net_namespace.h | 7 +++++
 net/core/net_namespace.c | 54 ++
 net/ipv4/af_inet.c | 2 +
 net/ipv4/raw.c | 3 ++
 4 files changed, 66 insertions(+)

Index: 2.6.20-rc4-mm1/net/ipv4/af_inet.c
===
--- 2.6.20-rc4-mm1.orig/net/ipv4/af_inet.c
+++ 2.6.20-rc4-mm1/net/ipv4/af_inet.c
@@ -433,6 +433,8 @@
   * is temporarily down)
   */
  err = -EADDRNOTAVAIL;
+ if (net_ns_check_bind(chk_addr_ret, addr->sin_addr.s_addr))
+ goto out;
  if (!sysctl_ip_nonlocal_bind &&
      !inet->freebind &&
      addr->sin_addr.s_addr != INADDR_ANY &&
Index: 2.6.20-rc4-mm1/net/ipv4/raw.c
===
--- 2.6.20-rc4-mm1.orig/net/ipv4/raw.c
+++ 2.6.20-rc4-mm1/net/ipv4/raw.c
@@ -559,7 +559,10 @@
  if (sk->sk_state != TCP_CLOSE || addr_len < sizeof(struct sockaddr_in))
   goto out;
  chk_addr_ret = inet_addr_type(addr->sin_addr.s_addr);
+
  ret = -EADDRNOTAVAIL;
+ if (net_ns_check_bind(chk_addr_ret, addr->sin_addr.s_addr))
+ goto out;
  if (addr->sin_addr.s_addr && chk_addr_ret != RTN_LOCAL &&
      chk_addr_ret != RTN_MULTICAST && chk_addr_ret != RTN_BROADCAST)
   goto out;
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
@@ -93,6 +93,8 @@

 extern int net_ns_ioctl(unsigned int cmd, void __user *arg);

+extern int net_ns_check_bind(int addr_type, u32 addr);
+
 #else /* CONFIG_NET_NS */

 #define INIT_NET_NS(net_ns)
@@ -148,6 +150,11 @@
  return -ENOSYS;
 }

+static inline int net_ns_check_bind(int addr_type, u32 addr)
+{
+ return 0;
+}
+
 #endif /* !CONFIG_NET_NS */

 #endif /* _LINUX_NET_NAMESPACE_H */
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
@@ -263,4 +263,58 @@
  return err;
 }

+/*
+ * This function check if the specified bind address is allowed.
+ * The bind is allowed if the address is:
+ * - 127.0.0.1
+ * - INADDR_ANY
+ * - INADDR_BROADCAST
+ * - a multicast address
+ * - the specified address match an ifaddr owned by the current
+ *   network namespace. That implies the local address and the
+ *   computed address from the netmask
+ * @addr_type : an addr type
+ * @addr : the requested bind address
+ * Returns: -EPERM on failure, 0 on success
+ */
+int net_ns_check_bind(int addr_type, u32 addr)
+{
+ int ret = -EPERM;
```

```
+       struct net_device *dev;
+       struct in_device *in_dev;
+ struct net_namespace *net_ns = current_net_ns;
+
+ if (LOOPBACK(addr) ||
+     MULTICAST(addr) ||
+     INADDR_ANY == addr ||
+     INADDR_BROADCAST == addr)
+  return 0;
+
+       read_lock(&dev_base_lock);
+       rcu_read_lock();
+       for (dev = dev_base; dev; dev = dev->next) {
+               in_dev = __in_dev_get_rcu(dev);
+               if (!in_dev)
+                       continue;
+
+               for_ifa(in_dev) {
+                       if (ifa->ifa_net_ns != net_ns)
+    continue;
+   if (addr == ifa->ifa_local ||
+       addr == ifa->ifa_broadcast ||
+       addr == (ifa->ifa_local & ifa->ifa_mask) ||
+       addr == ((ifa->ifa_address & ifa->ifa_mask)|
+         ~ifa->ifa_mask)) {
+    ret = 0;
+    goto out;
+   }
+               } endfor_ifa(in_dev);
+       }
+out:
+       read_unlock(&dev_base_lock);
+       rcu_read_unlock();
+
+ return ret;
+}
+
 #endif /* CONFIG_NET_NS */

--
```