Subject: [patch 10/12] net namespace : add the loopback isolation
Posted by Daniel Lezcano on Fri, 19 Jan 2007 15:47:24 GMT
View Forum Message <> Reply to Message

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

When a packet is outgoing, the namespace source is stored
into the skbuff. Because it is the loopback address, the
source == destination, so when the packet is incoming, it
has already the namespace destination set into the packet.

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

---
 include/linux/net_namespace.h | 13 ++++++++++--
 include/linux/skbuff.h | 5 ++++-
 net/core/net_namespace.c | 32 ++++++++++++++++++++++++++++++-----
 net/ipv4/ip_input.c | 2 +-
 net/ipv4/ip_output.c | 1 +
 5 files changed, 44 insertions(+), 9 deletions(-)

Index: 2.6.20-rc4-mm1/include/linux/skbuff.h
===
--- 2.6.20-rc4-mm1.orig/include/linux/skbuff.h
+++ 2.6.20-rc4-mm1/include/linux/skbuff.h
@@ -225,6 +225,7 @@
  * @dma_cookie: a cookie to one of several possible DMA operations
  * done by skb DMA functions
  * @secmark: security marking
+ *   @net_ns: namespace destination
  */

 struct sk_buff {
@@ -309,7 +310,9 @@
 #ifdef CONFIG_NETWORK_SECMARK
   __u32   secmark;
 #endif
-
+#ifdef CONFIG_NET_NS
+ struct net_namespace   *net_ns;
+#endif
   __u32   mark;

   /* These elements must be at the end, see alloc_skb() for details. */
Index: 2.6.20-rc4-mm1/net/ipv4/ip_input.c
===
--- 2.6.20-rc4-mm1.orig/net/ipv4/ip_input.c
+++ 2.6.20-rc4-mm1/net/ipv4/ip_input.c

```
@@ -396,7 +396,7 @@

  iph = skb->nh.iph;

- dst_net_ns = net_ns_find_from_dest_addr(iph->daddr);
+ dst_net_ns = net_ns_find_from_dest_addr(skb);
  if (dst_net_ns && !net_ns_match(net_ns, dst_net_ns))
   push_net_ns(dst_net_ns);
  /*
Index: 2.6.20-rc4-mm1/net/ipv4/ip_output.c
===================================================================
--- 2.6.20-rc4-mm1.orig/net/ipv4/ip_output.c
+++ 2.6.20-rc4-mm1/net/ipv4/ip_output.c
@@ -272,6 +272,7 @@

  IP_INC_STATS(IPSTATS_MIB_OUTREQUESTS);

+ net_ns_tag_sk_buff(skb);
  skb->dev = dev;
  skb->protocol = htons(ETH_P_IP);

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
@@ -8,6 +8,7 @@
 #include <linux/types.h>

 struct in_ifaddr;
+struct sk_buff;

 struct net_namespace {
  struct kref  kref;
@@ -101,10 +102,13 @@
 extern __be32 net_ns_select_source_address(const struct net_device *dev,
         u32 dst, int scope);

-extern struct net_namespace *net_ns_find_from_dest_addr(u32 daddr);
+extern struct net_namespace
+*net_ns_find_from_dest_addr(const struct sk_buff *skb);

 extern int net_ns_ifa_is_visible(const struct in_ifaddr *ifa);

+extern void net_ns_tag_sk_buff(struct sk_buff *skb);
+
 #define SELECT_SRC_ADDR net_ns_select_source_address

 #else /* CONFIG_NET_NS */
```

```
@@ -173,7 +177,8 @@
  return 0;
 }

-static inline struct net_namespace *net_ns_find_from_dest_addr(u32 daddr)
+static inline struct net_namespace
+*net_ns_find_from_dest_addr(const struct sk_buff *skb)
 {
  return NULL;
 }
@@ -183,6 +188,10 @@
  return 1;
 }

+static inline void net_ns_tag_sk_buff(struct sk_buff *skb)
+{
+ ;
+}
 #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
@@ -13,6 +13,9 @@
 #include <linux/in.h>
 #include <linux/netdevice.h>
 #include <linux/inetdevice.h>
+#include <linux/skbuff.h>
+#include <linux/ip.h>
+
 #include <net/ip_fib.h>

 struct net_namespace init_net_ns = {
@@ -389,18 +392,25 @@
 /*
  * This function finds the network namespace destination deduced from
  * the destination address. The network namespace is retrieved from
- * the ifaddr owned by a network namespace
- * @daddr  : destination
+ * the ifaddr owned by a network namespace. If the packet is for the
+ * loopback address so we assume the destination address is already filled
+ * by the sender which is the same as the receiver.
+ * @skb : the packet to be delivered
  * Returns : the network namespace destination or NULL if not found
  */
-struct net_namespace *net_ns_find_from_dest_addr(u32 daddr)
```

```
+struct net_namespace *net_ns_find_from_dest_addr(const struct sk_buff *skb)
 {
  struct net_namespace *net_ns = NULL;
  struct net_device *dev;
  struct in_device *in_dev;
+   struct iphdr *iph;
+   __be32 daddr;
+
+   iph = skb->nh.iph;
+   daddr = iph->daddr;

- if (LOOPBACK(daddr))
-  return current_net_ns;
+    if (LOOPBACK(daddr))
+    return skb->net_ns;

  read_lock(&dev_base_lock);
  rcu_read_lock();
@@ -442,4 +452,16 @@
   return 0;
 }

+/*
+ * This function allows to isolation the loopback. Because we are
+ * using the 127.0.0.1, we assume the source network namespace is
+ * the same as the destination network namespace. So setting the
+ * source, we have directly the destination when receiving the packet
+ * @skb : the packet to be tagged with the network namespace
+ */
+void net_ns_tag_sk_buff(struct sk_buff *skb)
+{
+ struct net_namespace *net_ns = current_net_ns;
+ skb->net_ns = net_ns;
+}
 #endif /* CONFIG_NET_NS */

--
```