Subject: [PATCH] NETFILTER: per-netns FILTER/MANGLE/RAW tables for real
Posted by Alexey Dobriyan on Mon, 03 Mar 2008 16:06:59 GMT
View Forum Message <> Reply to Message

Commit 9335f047fe61587ec82ff12fbb1220bcfdd32006 aka
"[NETFILTER]: ip_tables: per-netns FILTER, MANGLE, RAW"
added per-netns _view_ of iptables rules. They were shown to user, but
ignored by filtering code. Now that it's possible to at least ping loopback,
per-netns tables can affect filtering decisions.

netns is taken in case of
 PRE_ROUTING, LOCAL_IN -- from in device,
 POST_ROUTING, LOCAL_OUT -- from out device,
 FORWARD -- from in device which should be equal to out device's netns.
    This code is relatively new, so BUG_ON was plugged.

Wrappers were added to a) keep code the same from CONFIG_NET_NS=n users
(overwhelming majority), b) consolidate code in one place -- similar
changes will be done in ipv6 and arp netfilter code.

Signed-off-by: Alexey Dobriyan <adobriyan@sw.ru>
---

 include/linux/netfilter.h | 54 ++++++++++++++++++++++++++++++++++++-
 net/ipv4/netfilter/iptable_filter.c | 19 ++++++++++--
 net/ipv4/netfilter/iptable_mangle.c | 49 ++++++++++++++++++++++++++++++++----
 net/ipv4/netfilter/iptable_raw.c | 6 ++--
 4 files changed, 115 insertions(+), 13 deletions(-)

--- a/include/linux/netfilter.h
+++ b/include/linux/netfilter.h
@@ -6,11 +6,13 @@
 #include <linux/types.h>
 #include <linux/skbuff.h>
 #include <linux/net.h>
+#include <linux/netdevice.h>
 #include <linux/if.h>
 #include <linux/in.h>
 #include <linux/in6.h>
 #include <linux/wait.h>
 #include <linux/list.h>
+#include <net/net_namespace.h>
 #endif
 #include <linux/compiler.h>

@@ -67,7 +69,6 @@ extern void netfilter_init(void);
 #define NF_MAX_HOOKS 8

```
 struct sk_buff;
-struct net_device;

 typedef unsigned int nf_hookfn(unsigned int hooknum,
          struct sk_buff *skb,
@@ -311,5 +312,56 @@ extern void (*nf_ct_destroy)(struct nf_conntrack *);
 static inline void nf_ct_attach(struct sk_buff *new, struct sk_buff *skb) {}
 #endif

+static inline struct net *nf_pre_routing_net(const struct net_device *in,
+          const struct net_device *out)
+{
+#ifdef CONFIG_NET_NS
+ return in->nd_net;
+#else
+ return &init_net;
+#endif
+}
+
+static inline struct net *nf_local_in_net(const struct net_device *in,
+       const struct net_device *out)
+{
+#ifdef CONFIG_NET_NS
+ return in->nd_net;
+#else
+ return &init_net;
+#endif
+}
+
+static inline struct net *nf_forward_net(const struct net_device *in,
+      const struct net_device *out)
+{
+#ifdef CONFIG_NET_NS
+ BUG_ON(in->nd_net != out->nd_net);
+ return in->nd_net;
+#else
+ return &init_net;
+#endif
+}
+
+static inline struct net *nf_local_out_net(const struct net_device *in,
+        const struct net_device *out)
+{
+#ifdef CONFIG_NET_NS
+ return out->nd_net;
+#else
+ return &init_net;
+#endif
```

```
+}
+
+static inline struct net *nf_post_routing_net(const struct net_device *in,
+           const struct net_device *out)
+{
+#ifdef CONFIG_NET_NS
+ return out->nd_net;
+#else
+ return &init_net;
+#endif
+}
+
 #endif /*__KERNEL__*/
 #endif /*__LINUX_NETFILTER_H*/
--- a/net/ipv4/netfilter/iptable_filter.c
+++ b/net/ipv4/netfilter/iptable_filter.c
@@ -63,13 +63,25 @@ static struct xt_table packet_filter = {

 /* The work comes in here from netfilter.c. */
 static unsigned int
+ipt_local_in_hook(unsigned int hook,
+    struct sk_buff *skb,
+    const struct net_device *in,
+    const struct net_device *out,
+    int (*okfn)(struct sk_buff *))
+{
+ return ipt_do_table(skb, hook, in, out,
+       nf_local_in_net(in, out)->ipv4.iptable_filter);
+}
+
+static unsigned int
 ipt_hook(unsigned int hook,
   struct sk_buff *skb,
   const struct net_device *in,
   const struct net_device *out,
   int (*okfn)(struct sk_buff *))
 {
- return ipt_do_table(skb, hook, in, out, init_net.ipv4.iptable_filter);
+ return ipt_do_table(skb, hook, in, out,
+       nf_forward_net(in, out)->ipv4.iptable_filter);
 }

 static unsigned int
@@ -88,12 +100,13 @@ ipt_local_out_hook(unsigned int hook,
   return NF_ACCEPT;
  }

- return ipt_do_table(skb, hook, in, out, init_net.ipv4.iptable_filter);
```

```
+ return ipt_do_table(skb, hook, in, out,
+       nf_local_out_net(in, out)->ipv4.iptable_filter);
 }

 static struct nf_hook_ops ipt_ops[] __read_mostly = {
  {
-  .hook  = ipt_hook,
+  .hook  = ipt_local_in_hook,
   .owner  = THIS_MODULE,
   .pf  = PF_INET,
   .hooknum = NF_INET_LOCAL_IN,
--- a/net/ipv4/netfilter/iptable_mangle.c
+++ b/net/ipv4/netfilter/iptable_mangle.c
@@ -74,13 +74,47 @@ static struct xt_table packet_mangler = {

 /* The work comes in here from netfilter.c. */
 static unsigned int
-ipt_route_hook(unsigned int hook,
+ipt_pre_routing_hook(unsigned int hook,
+       struct sk_buff *skb,
+       const struct net_device *in,
+       const struct net_device *out,
+       int (*okfn)(struct sk_buff *))
+{
+ return ipt_do_table(skb, hook, in, out,
+       nf_pre_routing_net(in, out)->ipv4.iptable_mangle);
+}
+
+static unsigned int
+ipt_post_routing_hook(unsigned int hook,
+        struct sk_buff *skb,
+        const struct net_device *in,
+        const struct net_device *out,
+        int (*okfn)(struct sk_buff *))
+{
+ return ipt_do_table(skb, hook, in, out,
+       nf_post_routing_net(in, out)->ipv4.iptable_mangle);
+}
+
+static unsigned int
+ipt_local_in_hook(unsigned int hook,
+    struct sk_buff *skb,
+    const struct net_device *in,
+    const struct net_device *out,
+    int (*okfn)(struct sk_buff *))
+{
+ return ipt_do_table(skb, hook, in, out,
+       nf_local_in_net(in, out)->ipv4.iptable_mangle);
```

```
+}
+
+static unsigned int
+ipt_forward_hook(unsigned int hook,
   struct sk_buff *skb,
   const struct net_device *in,
   const struct net_device *out,
   int (*okfn)(struct sk_buff *))
 {
- return ipt_do_table(skb, hook, in, out, init_net.ipv4.iptable_mangle);
+ return ipt_do_table(skb, hook, in, out,
+        nf_forward_net(in, out)->ipv4.iptable_mangle);
 }

 static unsigned int
@@ -112,7 +146,8 @@ ipt_local_hook(unsigned int hook,
  daddr = iph->daddr;
  tos = iph->tos;

- ret = ipt_do_table(skb, hook, in, out, init_net.ipv4.iptable_mangle);
+ ret = ipt_do_table(skb, hook, in, out,
+       nf_local_out_net(in, out)->ipv4.iptable_mangle);
  /* Reroute for ANY change. */
  if (ret != NF_DROP && ret != NF_STOLEN && ret != NF_QUEUE) {
   iph = ip_hdr(skb);
@@ -130,21 +165,21 @@ ipt_local_hook(unsigned int hook,

 static struct nf_hook_ops ipt_ops[] __read_mostly = {
  {
-  .hook  = ipt_route_hook,
+  .hook  = ipt_pre_routing_hook,
   .owner  = THIS_MODULE,
   .pf  = PF_INET,
   .hooknum = NF_INET_PRE_ROUTING,
   .priority = NF_IP_PRI_MANGLE,
  },
  {
-  .hook  = ipt_route_hook,
+  .hook  = ipt_local_in_hook,
   .owner  = THIS_MODULE,
   .pf  = PF_INET,
   .hooknum = NF_INET_LOCAL_IN,
   .priority = NF_IP_PRI_MANGLE,
  },
  {
-  .hook  = ipt_route_hook,
+  .hook  = ipt_forward_hook,
   .owner  = THIS_MODULE,
```

```
   .pf  = PF_INET,
   .hooknum = NF_INET_FORWARD,
@@ -158,7 +193,7 @@ static struct nf_hook_ops ipt_ops[] __read_mostly = {
   .priority = NF_IP_PRI_MANGLE,
  },
  {
-  .hook  = ipt_route_hook,
+  .hook  = ipt_post_routing_hook,
   .owner  = THIS_MODULE,
   .pf  = PF_INET,
   .hooknum = NF_INET_POST_ROUTING,
--- a/net/ipv4/netfilter/iptable_raw.c
+++ b/net/ipv4/netfilter/iptable_raw.c
@@ -52,7 +52,8 @@ ipt_hook(unsigned int hook,
   const struct net_device *out,
   int (*okfn)(struct sk_buff *))
 {
- return ipt_do_table(skb, hook, in, out, init_net.ipv4.iptable_raw);
+ return ipt_do_table(skb, hook, in, out,
+       nf_pre_routing_net(in, out)->ipv4.iptable_raw);
 }

 static unsigned int
@@ -70,7 +71,8 @@ ipt_local_hook(unsigned int hook,
          "packet.\n");
   return NF_ACCEPT;
  }
- return ipt_do_table(skb, hook, in, out, init_net.ipv4.iptable_raw);
+ return ipt_do_table(skb, hook, in, out,
+       nf_local_out_net(in, out)->ipv4.iptable_raw);
 }

 /* 'raw' is the very first table. */
```