Subject: [PATCH 5/5] netns netfilter: per-netns FILTER, MANGLE, RAW
Posted by Alexey Dobriyan on Mon, 21 Jan 2008 14:55:09 GMT
View Forum Message <> Reply to Message

Now, iptables show and configure different set of rules in different
netnss'. Filtering decisions are still made by consulting only
init_net's set.

Changes are identical except naming so no splitting.

P.S.: one need to remove init_net checks in nf_sockopt.c and inet_create()
to see the effect.

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

```
 include/net/netns/ipv4.h           |    5 ++++
 net/ipv4/netfilter/iptable_filter.c |   41 ++++++++++++++++++++++++++-----------
 net/ipv4/netfilter/iptable_mangle.c |   41 ++++++++++++++++++++++++++-----------
 net/ipv4/netfilter/iptable_raw.c    |   41 ++++++++++++++++++++++++++-----------
 4 files changed, 92 insertions(+), 36 deletions(-)

--- a/include/net/netns/ipv4.h
+++ b/include/net/netns/ipv4.h
@@ -22,5 +22,10 @@ struct netns_ipv4 {
 #endif
  struct hlist_head *fib_table_hash;
  struct sock  *fibnl;
+#ifdef CONFIG_NETFILTER
+ struct xt_table  *iptable_filter;
+ struct xt_table  *iptable_mangle;
+ struct xt_table  *iptable_raw;
+#endif
 };
 #endif
--- a/net/ipv4/netfilter/iptable_filter.c
+++ b/net/ipv4/netfilter/iptable_filter.c
@@ -28,7 +28,7 @@ static struct
  struct ipt_replace repl;
  struct ipt_standard entries[3];
  struct ipt_error term;
-} initial_table __initdata = {
+} initial_table __net_initdata = {
  .repl = {
   .name = "filter",
   .valid_hooks = FILTER_VALID_HOOKS,
@@ -53,14 +53,13 @@ static struct
  .term = IPT_ERROR_INIT,   /* ERROR */
```

```
 };

-static struct xt_table __packet_filter = {
+static struct xt_table packet_filter = {
  .name  = "filter",
  .valid_hooks = FILTER_VALID_HOOKS,
  .lock  = RW_LOCK_UNLOCKED,
  .me  = THIS_MODULE,
  .af  = AF_INET,
 };
-static struct xt_table *packet_filter;

 /* The work comes in here from netfilter.c. */
 static unsigned int
@@ -70,7 +69,7 @@ ipt_hook(unsigned int hook,
   const struct net_device *out,
   int (*okfn)(struct sk_buff *))
 {
- return ipt_do_table(skb, hook, in, out, packet_filter);
+ return ipt_do_table(skb, hook, in, out, init_net.ipv4.iptable_filter);
 }

 static unsigned int
@@ -89,7 +88,7 @@ ipt_local_out_hook(unsigned int hook,
   return NF_ACCEPT;
  }

- return ipt_do_table(skb, hook, in, out, packet_filter);
+ return ipt_do_table(skb, hook, in, out, init_net.ipv4.iptable_filter);
 }

 static struct nf_hook_ops ipt_ops[] __read_mostly = {
@@ -120,6 +119,26 @@ static struct nf_hook_ops ipt_ops[] __read_mostly = {
 static int forward = NF_ACCEPT;
 module_param(forward, bool, 0000);

+static int __net_init iptable_filter_net_init(struct net *net)
+{
+ /* Register table */
+ net->ipv4.iptable_filter =
+  ipt_register_table(net, &packet_filter, &initial_table.repl);
+ if (IS_ERR(net->ipv4.iptable_filter))
+  return PTR_ERR(net->ipv4.iptable_filter);
+ return 0;
+}
+
+static void __net_exit iptable_filter_net_exit(struct net *net)
+{
```

```
+ ipt_unregister_table(net->ipv4.iptable_filter);
+}
+
+static struct pernet_operations iptable_filter_net_ops = {
+ .init = iptable_filter_net_init,
+ .exit = iptable_filter_net_exit,
+};
+
 static int __init iptable_filter_init(void)
 {
  int ret;
@@ -132,11 +151,9 @@ static int __init iptable_filter_init(void)
  /* Entry 1 is the FORWARD hook */
  initial_table.entries[1].target.verdict = -forward - 1;

- /* Register table */
- packet_filter = ipt_register_table(&init_net, &__packet_filter,
-        &initial_table.repl);
- if (IS_ERR(packet_filter))
-  return PTR_ERR(packet_filter);
+ ret = register_pernet_subsys(&iptable_filter_net_ops);
+ if (ret < 0)
+  return ret;

  /* Register hooks */
  ret = nf_register_hooks(ipt_ops, ARRAY_SIZE(ipt_ops));
@@ -146,14 +163,14 @@ static int __init iptable_filter_init(void)
  return ret;

  cleanup_table:
- ipt_unregister_table(packet_filter);
+ unregister_pernet_subsys(&iptable_filter_net_ops);
  return ret;
 }

 static void __exit iptable_filter_fini(void)
 {
  nf_unregister_hooks(ipt_ops, ARRAY_SIZE(ipt_ops));
- ipt_unregister_table(packet_filter);
+ unregister_pernet_subsys(&iptable_filter_net_ops);
 }

 module_init(iptable_filter_init);
--- a/net/ipv4/netfilter/iptable_mangle.c
+++ b/net/ipv4/netfilter/iptable_mangle.c
@@ -33,7 +33,7 @@ static struct
  struct ipt_replace repl;
  struct ipt_standard entries[5];
```

```
  struct ipt_error term;
-} initial_table __initdata = {
+} initial_table __net_initdata = {
  .repl = {
   .name = "mangle",
   .valid_hooks = MANGLE_VALID_HOOKS,
@@ -64,14 +64,13 @@ static struct
  .term = IPT_ERROR_INIT,   /* ERROR */
 };

-static struct xt_table __packet_mangler = {
+static struct xt_table packet_mangler = {
  .name  = "mangle",
  .valid_hooks = MANGLE_VALID_HOOKS,
  .lock  = RW_LOCK_UNLOCKED,
  .me  = THIS_MODULE,
  .af  = AF_INET,
 };
-static struct xt_table *packet_mangler;

 /* The work comes in here from netfilter.c. */
 static unsigned int
@@ -81,7 +80,7 @@ ipt_route_hook(unsigned int hook,
   const struct net_device *out,
   int (*okfn)(struct sk_buff *))
 {
- return ipt_do_table(skb, hook, in, out, packet_mangler);
+ return ipt_do_table(skb, hook, in, out, init_net.ipv4.iptable_mangle);
 }

 static unsigned int
@@ -113,7 +112,7 @@ ipt_local_hook(unsigned int hook,
  daddr = iph->daddr;
  tos = iph->tos;

- ret = ipt_do_table(skb, hook, in, out, packet_mangler);
+ ret = ipt_do_table(skb, hook, in, out, init_net.ipv4.iptable_mangle);
  /* Reroute for ANY change. */
  if (ret != NF_DROP && ret != NF_STOLEN && ret != NF_QUEUE) {
   iph = ip_hdr(skb);
@@ -167,15 +166,33 @@ static struct nf_hook_ops ipt_ops[] __read_mostly = {
 },
 };

+static int __net_init iptable_mangle_net_init(struct net *net)
+{
+ /* Register table */
+ net->ipv4.iptable_mangle =
```

```
+  ipt_register_table(net, &packet_mangler, &initial_table.repl);
+ if (IS_ERR(net->ipv4.iptable_mangle))
+  return PTR_ERR(net->ipv4.iptable_mangle);
+ return 0;
+}
+
+static void __net_exit iptable_mangle_net_exit(struct net *net)
+{
+ ipt_unregister_table(net->ipv4.iptable_mangle);
+}
+
+static struct pernet_operations iptable_mangle_net_ops = {
+ .init = iptable_mangle_net_init,
+ .exit = iptable_mangle_net_exit,
+};
+
 static int __init iptable_mangle_init(void)
 {
  int ret;

- /* Register table */
- packet_mangler = ipt_register_table(&init_net, &__packet_mangler,
-         &initial_table.repl);
- if (IS_ERR(packet_mangler))
-  return PTR_ERR(packet_mangler);
+ ret = register_pernet_subsys(&iptable_mangle_net_ops);
+ if (ret < 0)
+  return ret;

  /* Register hooks */
  ret = nf_register_hooks(ipt_ops, ARRAY_SIZE(ipt_ops));
@@ -185,14 +202,14 @@ static int __init iptable_mangle_init(void)
  return ret;

  cleanup_table:
- ipt_unregister_table(packet_mangler);
+ unregister_pernet_subsys(&iptable_mangle_net_ops);
  return ret;
 }

 static void __exit iptable_mangle_fini(void)
 {
  nf_unregister_hooks(ipt_ops, ARRAY_SIZE(ipt_ops));
- ipt_unregister_table(packet_mangler);
+ unregister_pernet_subsys(&iptable_mangle_net_ops);
 }

 module_init(iptable_mangle_init);
```

```
--- a/net/ipv4/netfilter/iptable_raw.c
+++ b/net/ipv4/netfilter/iptable_raw.c
@@ -14,7 +14,7 @@ static struct
  struct ipt_replace repl;
  struct ipt_standard entries[2];
  struct ipt_error term;
-} initial_table __initdata = {
+} initial_table __net_initdata = {
  .repl = {
   .name = "raw",
   .valid_hooks = RAW_VALID_HOOKS,
@@ -36,14 +36,13 @@ static struct
  .term = IPT_ERROR_INIT,   /* ERROR */
 };

-static struct xt_table __packet_raw = {
+static struct xt_table packet_raw = {
  .name = "raw",
  .valid_hooks =  RAW_VALID_HOOKS,
  .lock = RW_LOCK_UNLOCKED,
  .me = THIS_MODULE,
  .af = AF_INET,
 };
-static struct xt_table *packet_raw;

 /* The work comes in here from netfilter.c. */
 static unsigned int
@@ -53,7 +52,7 @@ ipt_hook(unsigned int hook,
   const struct net_device *out,
   int (*okfn)(struct sk_buff *))
 {
- return ipt_do_table(skb, hook, in, out, packet_raw);
+ return ipt_do_table(skb, hook, in, out, init_net.ipv4.iptable_raw);
 }

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

 /* 'raw' is the very first table. */
@@ -92,15 +91,33 @@ static struct nf_hook_ops ipt_ops[] __read_mostly = {
  },
 };
```

```
+static int __net_init iptable_raw_net_init(struct net *net)
+{
+ /* Register table */
+ net->ipv4.iptable_raw =
+  ipt_register_table(net, &packet_raw, &initial_table.repl);
+ if (IS_ERR(net->ipv4.iptable_raw))
+  return PTR_ERR(net->ipv4.iptable_raw);
+ return 0;
+}
+
+static void __net_exit iptable_raw_net_exit(struct net *net)
+{
+ ipt_unregister_table(net->ipv4.iptable_raw);
+}
+
+static struct pernet_operations iptable_raw_net_ops = {
+ .init = iptable_raw_net_init,
+ .exit = iptable_raw_net_exit,
+};
+
 static int __init iptable_raw_init(void)
 {
  int ret;

- /* Register table */
- packet_raw = ipt_register_table(&init_net, &__packet_raw,
-     &initial_table.repl);
- if (IS_ERR(packet_raw))
-  return PTR_ERR(packet_raw);
+ ret = register_pernet_subsys(&iptable_raw_net_ops);
+ if (ret < 0)
+  return ret;

  /* Register hooks */
  ret = nf_register_hooks(ipt_ops, ARRAY_SIZE(ipt_ops));
@@ -110,14 +127,14 @@ static int __init iptable_raw_init(void)
  return ret;

  cleanup_table:
- ipt_unregister_table(packet_raw);
+ unregister_pernet_subsys(&iptable_raw_net_ops);
  return ret;
 }

 static void __exit iptable_raw_fini(void)
 {
  nf_unregister_hooks(ipt_ops, ARRAY_SIZE(ipt_ops));
```

```
- ipt_unregister_table(packet_raw);
+ unregister_pernet_subsys(&iptable_raw_net_ops);
 }

 module_init(iptable_raw_init);
```

Subject: Re: [PATCH 5/5] netns netfilter: per-netns FILTER, MANGLE, RAW
Posted by Patrick McHardy on Tue, 22 Jan 2008 17:10:16 GMT
View Forum Message <> Reply to Message

Alexey Dobriyan wrote:
> Now, iptables show and configure different set of rules in different
> netnss'. Filtering decisions are still made by consulting only
> init_net's set.
>
> Changes are identical except naming so no splitting.
>
> P.S.: one need to remove init_net checks in nf_sockopt.c and inet_create()
> to see the effect.

Also applied, thanks.
