Subject: [PATCH 1/5] netns netfilter: per-netns ip6tables
Posted by Alexey Dobriyan on Thu, 24 Jan 2008 12:23:28 GMT
View Forum Message <> Reply to Message

* Propagate netns from userspace down to xt_find_table_lock()
* Register ip6 tables in netns (modules still use init_net)

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

 include/linux/netfilter_ipv6/ip6_tables.h | 3 +
 net/ipv6/netfilter/ip6_tables.c | 50 ++++++++++++++++++----------
 net/ipv6/netfilter/ip6table_filter.c | 2 -
 net/ipv6/netfilter/ip6table_mangle.c | 2 -
 net/ipv6/netfilter/ip6table_raw.c | 2 -
 5 files changed, 31 insertions(+), 28 deletions(-)

--- a/include/linux/netfilter_ipv6/ip6_tables.h
+++ b/include/linux/netfilter_ipv6/ip6_tables.h
@@ -305,7 +305,8 @@ ip6t_get_target(struct ip6t_entry *e)
 #include <linux/init.h>
 extern void ip6t_init(void) __init;

-extern struct xt_table *ip6t_register_table(struct xt_table *table,
+extern struct xt_table *ip6t_register_table(struct net *net,
+   struct xt_table *table,
                                             const struct ip6t_replace *repl);
 extern void ip6t_unregister_table(struct xt_table *table);
 extern unsigned int ip6t_do_table(struct sk_buff *skb,
--- a/net/ipv6/netfilter/ip6_tables.c
+++ b/net/ipv6/netfilter/ip6_tables.c
@@ -1118,7 +1118,7 @@ static int compat_table_info(const struct xt_table_info *info,
 }
 #endif

-static int get_info(void __user *user, int *len, int compat)
+static int get_info(struct net *net, void __user *user, int *len, int compat)
 {
  char name[IP6T_TABLE_MAXNAMELEN];
  struct xt_table *t;
@@ -1138,7 +1138,7 @@ static int get_info(void __user *user, int *len, int compat)
  if (compat)
   xt_compat_lock(AF_INET6);
 #endif
- t = try_then_request_module(xt_find_table_lock(&init_net, AF_INET6, name),
+ t = try_then_request_module(xt_find_table_lock(net, AF_INET6, name),
                              "ip6table_%s", name);
  if (t && !IS_ERR(t)) {

```
   struct ip6t_getinfo info;
@@ -1178,7 +1178,7 @@ static int get_info(void __user *user, int *len, int compat)
 }

 static int
-get_entries(struct ip6t_get_entries __user *uptr, int *len)
+get_entries(struct net *net, struct ip6t_get_entries __user *uptr, int *len)
 {
  int ret;
  struct ip6t_get_entries get;
@@ -1196,7 +1196,7 @@ get_entries(struct ip6t_get_entries __user *uptr, int *len)
   return -EINVAL;
  }

- t = xt_find_table_lock(&init_net, AF_INET6, get.name);
+ t = xt_find_table_lock(net, AF_INET6, get.name);
  if (t && !IS_ERR(t)) {
   struct xt_table_info *private = t->private;
   duprintf("t->private->number = %u\n", private->number);
@@ -1217,7 +1217,7 @@ get_entries(struct ip6t_get_entries __user *uptr, int *len)
 }

 static int
-__do_replace(const char *name, unsigned int valid_hooks,
+__do_replace(struct net *net, const char *name, unsigned int valid_hooks,
       struct xt_table_info *newinfo, unsigned int num_counters,
       void __user *counters_ptr)
 {
@@ -1235,7 +1235,7 @@ __do_replace(const char *name, unsigned int valid_hooks,
   goto out;
  }

- t = try_then_request_module(xt_find_table_lock(&init_net, AF_INET6, name),
+ t = try_then_request_module(xt_find_table_lock(net, AF_INET6, name),
         "ip6table_%s", name);
  if (!t || IS_ERR(t)) {
   ret = t ? PTR_ERR(t) : -ENOENT;
@@ -1288,7 +1288,7 @@ __do_replace(const char *name, unsigned int valid_hooks,
 }

 static int
-do_replace(void __user *user, unsigned int len)
+do_replace(struct net *net, void __user *user, unsigned int len)
 {
  int ret;
  struct ip6t_replace tmp;
@@ -1322,7 +1322,7 @@ do_replace(void __user *user, unsigned int len)
```

```
  duprintf("ip_tables: Translated table\n");

- ret = __do_replace(tmp.name, tmp.valid_hooks, newinfo,
+ ret = __do_replace(net, tmp.name, tmp.valid_hooks, newinfo,
       tmp.num_counters, tmp.counters);
  if (ret)
   goto free_newinfo_untrans;
@@ -1358,7 +1358,7 @@ add_counter_to_entry(struct ip6t_entry *e,
 }

 static int
-do_add_counters(void __user *user, unsigned int len, int compat)
+do_add_counters(struct net *net, void __user *user, unsigned int len, int compat)
 {
  unsigned int i;
  struct xt_counters_info tmp;
@@ -1410,7 +1410,7 @@ do_add_counters(void __user *user, unsigned int len, int compat)
   goto free;
  }

- t = xt_find_table_lock(&init_net, AF_INET6, name);
+ t = xt_find_table_lock(net, AF_INET6, name);
  if (!t || IS_ERR(t)) {
   ret = t ? PTR_ERR(t) : -ENOENT;
   goto free;
@@ -1815,7 +1815,7 @@ out_unlock:
 }

 static int
-compat_do_replace(void __user *user, unsigned int len)
+compat_do_replace(struct net *net, void __user *user, unsigned int len)
 {
  int ret;
  struct compat_ip6t_replace tmp;
@@ -1852,7 +1852,7 @@ compat_do_replace(void __user *user, unsigned int len)

  duprintf("compat_do_replace: Translated table\n");

- ret = __do_replace(tmp.name, tmp.valid_hooks, newinfo,
+ ret = __do_replace(net, tmp.name, tmp.valid_hooks, newinfo,
       tmp.num_counters, compat_ptr(tmp.counters));
  if (ret)
   goto free_newinfo_untrans;
@@ -1876,11 +1876,11 @@ compat_do_ip6t_set_ctl(struct sock *sk, int cmd, void __user *user,

  switch (cmd) {
  case IP6T_SO_SET_REPLACE:
-  ret = compat_do_replace(user, len);
```

```
+  ret = compat_do_replace(sk->sk_net, user, len);
   break;

  case IP6T_SO_SET_ADD_COUNTERS:
-  ret = do_add_counters(user, len, 1);
+  ret = do_add_counters(sk->sk_net, user, len, 1);
   break;

  default:
@@ -1929,7 +1929,8 @@ compat_copy_entries_to_user(unsigned int total_size, struct xt_table
*table,
 }

 static int
-compat_get_entries(struct compat_ip6t_get_entries __user *uptr, int *len)
+compat_get_entries(struct net *net, struct compat_ip6t_get_entries __user *uptr,
+      int *len)
 {
  int ret;
  struct compat_ip6t_get_entries get;
@@ -1950,7 +1951,7 @@ compat_get_entries(struct compat_ip6t_get_entries __user *uptr, int
*len)
  }

  xt_compat_lock(AF_INET6);
- t = xt_find_table_lock(&init_net, AF_INET6, get.name);
+ t = xt_find_table_lock(net, AF_INET6, get.name);
  if (t && !IS_ERR(t)) {
   struct xt_table_info *private = t->private;
   struct xt_table_info info;
@@ -1986,10 +1987,10 @@ compat_do_ip6t_get_ctl(struct sock *sk, int cmd, void __user *user,
int *len)

  switch (cmd) {
  case IP6T_SO_GET_INFO:
-  ret = get_info(user, len, 1);
+  ret = get_info(sk->sk_net, user, len, 1);
   break;
  case IP6T_SO_GET_ENTRIES:
-  ret = compat_get_entries(user, len);
+  ret = compat_get_entries(sk->sk_net, user, len);
   break;
  default:
   ret = do_ip6t_get_ctl(sk, cmd, user, len);
@@ -2008,11 +2009,11 @@ do_ip6t_set_ctl(struct sock *sk, int cmd, void __user *user,
unsigned int len)

  switch (cmd) {
```

```
  case IP6T_SO_SET_REPLACE:
-  ret = do_replace(user, len);
+  ret = do_replace(sk->sk_net, user, len);
   break;

  case IP6T_SO_SET_ADD_COUNTERS:
-  ret = do_add_counters(user, len, 0);
+  ret = do_add_counters(sk->sk_net, user, len, 0);
   break;

  default:
@@ -2033,11 +2034,11 @@ do_ip6t_get_ctl(struct sock *sk, int cmd, void __user *user, int *len)

  switch (cmd) {
  case IP6T_SO_GET_INFO:
-  ret = get_info(user, len, 0);
+  ret = get_info(sk->sk_net, user, len, 0);
   break;

  case IP6T_SO_GET_ENTRIES:
-  ret = get_entries(user, len);
+  ret = get_entries(sk->sk_net, user, len);
   break;

  case IP6T_SO_GET_REVISION_MATCH:
@@ -2074,7 +2075,8 @@ do_ip6t_get_ctl(struct sock *sk, int cmd, void __user *user, int *len)
  return ret;
 }

-struct xt_table *ip6t_register_table(struct xt_table *table, const struct ip6t_replace *repl)
+struct xt_table *ip6t_register_table(struct net *net, struct xt_table *table,
+         const struct ip6t_replace *repl)
 {
  int ret;
  struct xt_table_info *newinfo;
@@ -2101,7 +2103,7 @@ struct xt_table *ip6t_register_table(struct xt_table *table, const struct
ip6t_r
  if (ret != 0)
   goto out_free;

- new_table = xt_register_table(&init_net, table, &bootstrap, newinfo);
+ new_table = xt_register_table(net, table, &bootstrap, newinfo);
  if (IS_ERR(new_table)) {
   ret = PTR_ERR(new_table);
   goto out_free;
--- a/net/ipv6/netfilter/ip6table_filter.c
+++ b/net/ipv6/netfilter/ip6table_filter.c
@@ -132,7 +132,7 @@ static int __init ip6table_filter_init(void)
```

```
  initial_table.entries[1].target.verdict = -forward - 1;

  /* Register table */
- packet_filter = ip6t_register_table(&__packet_filter, &initial_table.repl);
+ packet_filter = ip6t_register_table(&init_net, &__packet_filter, &initial_table.repl);
  if (IS_ERR(packet_filter))
   return PTR_ERR(packet_filter);

--- a/net/ipv6/netfilter/ip6table_mangle.c
+++ b/net/ipv6/netfilter/ip6table_mangle.c
@@ -164,7 +164,7 @@ static int __init ip6table_mangle_init(void)
  int ret;

  /* Register table */
- packet_mangler = ip6t_register_table(&__packet_mangler, &initial_table.repl);
+ packet_mangler = ip6t_register_table(&init_net, &__packet_mangler, &initial_table.repl);
  if (IS_ERR(packet_mangler))
   return PTR_ERR(packet_mangler);

--- a/net/ipv6/netfilter/ip6table_raw.c
+++ b/net/ipv6/netfilter/ip6table_raw.c
@@ -77,7 +77,7 @@ static int __init ip6table_raw_init(void)
  int ret;

  /* Register table */
- packet_raw = ip6t_register_table(&__packet_raw, &initial_table.repl);
+ packet_raw = ip6t_register_table(&init_net, &__packet_raw, &initial_table.repl);
  if (IS_ERR(packet_raw))
   return PTR_ERR(packet_raw);
```