Subject: [PATCH 3/5] netns netfilter: per-netns arp_tables
Posted by Alexey Dobriyan on Thu, 24 Jan 2008 12:28:52 GMT
View Forum Message <> Reply to Message

* Propagate netns from userspace.
* arpt_register_table() registers table in supplied netns.

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

 include/linux/netfilter_arp/arp_tables.h | 3 +
 net/ipv4/netfilter/arp_tables.c | 55 ++++++++++++++++++++----------
 net/ipv4/netfilter/arptable_filter.c | 2 -
 3 files changed, 34 insertions(+), 26 deletions(-)

--- a/include/linux/netfilter_arp/arp_tables.h
+++ b/include/linux/netfilter_arp/arp_tables.h
@@ -271,7 +271,8 @@ struct arpt_error
   xt_register_target(tgt); })
 #define arpt_unregister_target(tgt) xt_unregister_target(tgt)

-extern struct arpt_table *arpt_register_table(struct arpt_table *table,
+extern struct arpt_table *arpt_register_table(struct net *net,
+   struct arpt_table *table,
           const struct arpt_replace *repl);
 extern void arpt_unregister_table(struct arpt_table *table);
 extern unsigned int arpt_do_table(struct sk_buff *skb,
--- a/net/ipv4/netfilter/arp_tables.c
+++ b/net/ipv4/netfilter/arp_tables.c
@@ -22,6 +22,7 @@
 #include <linux/mutex.h>
 #include <linux/err.h>
 #include <net/compat.h>
+#include <net/sock.h>
 #include <asm/uaccess.h>

 #include <linux/netfilter/x_tables.h>
@@ -850,7 +851,7 @@ 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[ARPT_TABLE_MAXNAMELEN];
  struct arpt_table *t;
@@ -870,7 +871,7 @@ static int get_info(void __user *user, int *len, int compat)
  if (compat)

```
   xt_compat_lock(NF_ARP);
 #endif
- t = try_then_request_module(xt_find_table_lock(&init_net, NF_ARP, name),
+ t = try_then_request_module(xt_find_table_lock(net, NF_ARP, name),
         "arptable_%s", name);
  if (t && !IS_ERR(t)) {
   struct arpt_getinfo info;
@@ -908,7 +909,8 @@ static int get_info(void __user *user, int *len, int compat)
  return ret;
 }

-static int get_entries(struct arpt_get_entries __user *uptr, int *len)
+static int get_entries(struct net *net, struct arpt_get_entries __user *uptr,
+         int *len)
 {
  int ret;
  struct arpt_get_entries get;
@@ -926,7 +928,7 @@ static int get_entries(struct arpt_get_entries __user *uptr, int *len)
   return -EINVAL;
  }

- t = xt_find_table_lock(&init_net, NF_ARP, get.name);
+ t = xt_find_table_lock(net, NF_ARP, get.name);
  if (t && !IS_ERR(t)) {
   struct xt_table_info *private = t->private;
   duprintf("t->private->number = %u\n",
@@ -947,7 +949,8 @@ static int get_entries(struct arpt_get_entries __user *uptr, int *len)
  return ret;
 }

-static int __do_replace(const char *name, unsigned int valid_hooks,
+static int __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)
@@ -966,7 +969,7 @@ static int __do_replace(const char *name, unsigned int valid_hooks,
   goto out;
  }

- t = try_then_request_module(xt_find_table_lock(&init_net, NF_ARP, name),
+ t = try_then_request_module(xt_find_table_lock(net, NF_ARP, name),
         "arptable_%s", name);
  if (!t || IS_ERR(t)) {
   ret = t ? PTR_ERR(t) : -ENOENT;
@@ -1019,7 +1022,7 @@ static int __do_replace(const char *name, unsigned int valid_hooks,
  return ret;
 }
```

```
-static int do_replace(void __user *user, unsigned int len)
+static int do_replace(struct net *net, void __user *user, unsigned int len)
 {
  int ret;
  struct arpt_replace tmp;
@@ -1053,7 +1056,7 @@ static int do_replace(void __user *user, unsigned int len)

  duprintf("arp_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;
@@ -1080,7 +1083,8 @@ static inline int add_counter_to_entry(struct arpt_entry *e,
  return 0;
 }

-static int do_add_counters(void __user *user, unsigned int len, int compat)
+static int do_add_counters(struct net *net, void __user *user, unsigned int len,
+       int compat)
 {
  unsigned int i;
  struct xt_counters_info tmp;
@@ -1132,7 +1136,7 @@ static int do_add_counters(void __user *user, unsigned int len, int
compat)
   goto free;
  }

- t = xt_find_table_lock(&init_net, NF_ARP, name);
+ t = xt_find_table_lock(net, NF_ARP, name);
  if (!t || IS_ERR(t)) {
   ret = t ? PTR_ERR(t) : -ENOENT;
   goto free;
@@ -1435,7 +1439,8 @@ struct compat_arpt_replace {
  struct compat_arpt_entry entries[0];
 };

-static int compat_do_replace(void __user *user, unsigned int len)
+static int compat_do_replace(struct net *net, void __user *user,
+        unsigned int len)
 {
  int ret;
  struct compat_arpt_replace tmp;
@@ -1471,7 +1476,7 @@ static int 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;
@@ -1494,11 +1499,11 @@ static int compat_do_arpt_set_ctl(struct sock *sk, int cmd, void
__user *user,

  switch (cmd) {
  case ARPT_SO_SET_REPLACE:
-  ret = compat_do_replace(user, len);
+  ret = compat_do_replace(sk->sk_net, user, len);
   break;

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

  default:
@@ -1584,7 +1589,8 @@ struct compat_arpt_get_entries {
  struct compat_arpt_entry entrytable[0];
 };

-static int compat_get_entries(struct compat_arpt_get_entries __user *uptr,
+static int compat_get_entries(struct net *net,
+         struct compat_arpt_get_entries __user *uptr,
          int *len)
 {
  int ret;
@@ -1604,7 +1610,7 @@ static int compat_get_entries(struct compat_arpt_get_entries __user
*uptr,
  }

  xt_compat_lock(NF_ARP);
- t = xt_find_table_lock(&init_net, NF_ARP, get.name);
+ t = xt_find_table_lock(net, NF_ARP, get.name);
  if (t && !IS_ERR(t)) {
   struct xt_table_info *private = t->private;
   struct xt_table_info info;
@@ -1641,10 +1647,10 @@ static int compat_do_arpt_get_ctl(struct sock *sk, int cmd, void
__user *user,

  switch (cmd) {
  case ARPT_SO_GET_INFO:
-  ret = get_info(user, len, 1);
+  ret = get_info(sk->sk_net, user, len, 1);
```

```
   break;
  case ARPT_SO_GET_ENTRIES:
-  ret = compat_get_entries(user, len);
+  ret = compat_get_entries(sk->sk_net, user, len);
   break;
  default:
   ret = do_arpt_get_ctl(sk, cmd, user, len);
@@ -1662,11 +1668,11 @@ static int do_arpt_set_ctl(struct sock *sk, int cmd, void __user *user,
unsigned

  switch (cmd) {
  case ARPT_SO_SET_REPLACE:
-  ret = do_replace(user, len);
+  ret = do_replace(sk->sk_net, user, len);
   break;

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

  default:
@@ -1686,11 +1692,11 @@ static int do_arpt_get_ctl(struct sock *sk, int cmd, void __user *user,
int *len

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

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

  case ARPT_SO_GET_REVISION_TARGET: {
@@ -1719,7 +1725,8 @@ static int do_arpt_get_ctl(struct sock *sk, int cmd, void __user *user,
int *len
  return ret;
 }

-struct arpt_table *arpt_register_table(struct arpt_table *table,
+struct arpt_table *arpt_register_table(struct net *net,
+           struct arpt_table *table,
           const struct arpt_replace *repl)
 {
  int ret;
```

```
@@ -1749,7 +1756,7 @@ struct arpt_table *arpt_register_table(struct arpt_table *table,
  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/ipv4/netfilter/arptable_filter.c
+++ b/net/ipv4/netfilter/arptable_filter.c
@@ -91,7 +91,7 @@ static int __init arptable_filter_init(void)
  int ret;

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

Subject: Re: [PATCH 3/5] netns netfilter: per-netns arp_tables
Posted by Patrick McHardy on Thu, 24 Jan 2008 17:41:56 GMT
View Forum Message <> Reply to Message

Alexey Dobriyan wrote:
> * Propagate netns from userspace.
> * arpt_register_table() registers table in supplied netns.

Applied.
