Subject: [PATCH net-2.6.25 13/19] [NETNS] Namespacing IPv4 fib rules.
Posted by den on Wed, 09 Jan 2008 18:01:02 GMT
View Forum Message <> Reply to Message

The final trick for rules: place fib4_rules_ops into struct net and modify
initialization path for this.

```
Acked-by: Benjamin Thery <benjamin.thery@bull.net>
Acked-by: Daniel Lezcano <dlezcano@fr.ibm.com>
Signed-off-by: Denis V. Lunev <den@openvz.org>
---
 include/net/netns/ipv4.h |    5 +++++
 net/ipv4/fib_rules.c     |   44 ++++++++++++++++++++++++--------------------
 2 files changed, 29 insertions(+), 20 deletions(-)

diff --git a/include/net/netns/ipv4.h b/include/net/netns/ipv4.h
index e06d7cf..299f8c6 100644
--- a/include/net/netns/ipv4.h
+++ b/include/net/netns/ipv4.h
@@ -4,14 +4,19 @@

 #ifndef __NETNS_IPV4_H__
 #define __NETNS_IPV4_H__
+
 struct ctl_table_header;
 struct ipv4_devconf;
+struct fib_rules_ops;

 struct netns_ipv4 {
 #ifdef CONFIG_SYSCTL
  struct ctl_table_header *forw_hdr;
 #endif
  struct ipv4_devconf *devconf_all;
  struct ipv4_devconf *devconf_dflt;
+#ifdef CONFIG_IP_MULTIPLE_TABLES
+ struct fib_rules_ops *rules_ops;
+#endif
 };
 #endif
diff --git a/net/ipv4/fib_rules.c b/net/ipv4/fib_rules.c
index 49819fe..72232ab 100644
--- a/net/ipv4/fib_rules.c
+++ b/net/ipv4/fib_rules.c
@@ -32,8 +32,6 @@
 #include <net/ip_fib.h>
 #include <net/fib_rules.h>

-static struct fib_rules_ops fib4_rules_ops;
```

```
-
 struct fib4_rule
 {
  struct fib_rule  common;
@@ -63,7 +61,7 @@ int fib_lookup(struct flowi *flp, struct fib_result *res)
  };
  int err;

- err = fib_rules_lookup(&fib4_rules_ops, flp, 0, &arg);
+ err = fib_rules_lookup(init_net.ipv4.rules_ops, flp, 0, &arg);
  res->r = arg.rule;

  return err;
@@ -149,6 +147,7 @@ static int fib4_rule_configure(struct fib_rule *rule, struct sk_buff *skb,
          struct nlmsghdr *nlh, struct fib_rule_hdr *frh,
          struct nlattr **tb)
 {
+ struct net *net = skb->sk->sk_net;
  int err = -EINVAL;
  struct fib4_rule *rule4 = (struct fib4_rule *) rule;

@@ -159,7 +158,7 @@ static int fib4_rule_configure(struct fib_rule *rule, struct sk_buff *skb,
   if (rule->action == FR_ACT_TO_TBL) {
    struct fib_table *table;

-   table = fib_empty_table(&init_net);
+   table = fib_empty_table(net);
    if (table == NULL) {
     err = -ENOBUFS;
     goto errout;
@@ -250,9 +249,9 @@ static u32 fib4_rule_default_pref(struct fib_rules_ops *ops)
  struct list_head *pos;
  struct fib_rule *rule;

- if (!list_empty(&fib4_rules_ops.rules_list)) {
-  pos = fib4_rules_ops.rules_list.next;
-  if (pos->next != &fib4_rules_ops.rules_list) {
+ if (!list_empty(&ops->rules_list)) {
+  pos = ops->rules_list.next;
+  if (pos->next != &ops->rules_list) {
    rule = list_entry(pos->next, struct fib_rule, list);
    if (rule->pref)
     return rule->pref - 1;
@@ -274,7 +273,7 @@ static void fib4_rule_flush_cache(void)
  rt_cache_flush(-1);
 }

-static struct fib_rules_ops fib4_rules_ops = {
```

```
+static struct fib_rules_ops fib4_rules_ops_template = {
  .family  = AF_INET,
  .rule_size = sizeof(struct fib4_rule),
  .addr_size = sizeof(u32),
@@ -288,24 +287,20 @@ static struct fib_rules_ops fib4_rules_ops = {
  .flush_cache = fib4_rule_flush_cache,
  .nlgroup = RTNLGRP_IPV4_RULE,
  .policy  = fib4_rule_policy,
- .rules_list = LIST_HEAD_INIT(fib4_rules_ops.rules_list),
  .owner  = THIS_MODULE,
 };

-static int __init fib_default_rules_init(void)
+static int fib_default_rules_init(struct fib_rules_ops *ops)
 {
  int err;

- err = fib_default_rule_add(&fib4_rules_ops, 0,
-       RT_TABLE_LOCAL, FIB_RULE_PERMANENT);
+ err = fib_default_rule_add(ops, 0, RT_TABLE_LOCAL, FIB_RULE_PERMANENT);
  if (err < 0)
   return err;
- err = fib_default_rule_add(&fib4_rules_ops, 0x7FFE,
-       RT_TABLE_MAIN, 0);
+ err = fib_default_rule_add(ops, 0x7FFE, RT_TABLE_MAIN, 0);
  if (err < 0)
   return err;
- err = fib_default_rule_add(&fib4_rules_ops, 0x7FFF,
-       RT_TABLE_DEFAULT, 0);
+ err = fib_default_rule_add(ops, 0x7FFF, RT_TABLE_DEFAULT, 0);
  if (err < 0)
   return err;
  return 0;
@@ -314,20 +309,29 @@ static int __init fib_default_rules_init(void)
 int __net_init fib4_rules_init(struct net *net)
 {
  int err;
+ struct fib_rules_ops *ops;
+
+ ops = kmemdup(&fib4_rules_ops_template, sizeof(*ops), GFP_KERNEL);
+ if (ops == NULL)
+  return -ENOMEM;
+ INIT_LIST_HEAD(&ops->rules_list);
+ fib_rules_register(net, ops);

- fib_rules_register(net, &fib4_rules_ops);
- err = fib_default_rules_init();
+ err = fib_default_rules_init(ops);
```

```
  if (err < 0)
   goto fail;
+ net->ipv4.rules_ops = ops;
  return 0;

 fail:
  /* also cleans all rules already added */
- fib_rules_unregister(net, &fib4_rules_ops);
+ fib_rules_unregister(net, ops);
+ kfree(ops);
  return err;
 }

 void __net_exit fib4_rules_exit(struct net *net)
 {
- fib_rules_unregister(net, &fib4_rules_ops);
+ fib_rules_unregister(net, net->ipv4.rules_ops);
+ kfree(net->ipv4.rules_ops);
 }
--
1.5.3.rc5
```