Subject: [patch 1/2][NETNS49][IPV4][IGMP] make igmp proc per namespace
Posted by Daniel Lezcano on Fri, 12 Oct 2007 17:10:14 GMT
View Forum Message <> Reply to Message

This patch makes the proc files: "igmp" and "mcfilter" per namespace.

Signed-off-by: Daniel Lezcano <dlezcano@fr.ibm.com>

---
 net/ipv4/igmp.c | 68 ++-------
 1 file changed, 60 insertions(+), 8 deletions(-)

Index: linux-2.6-netns/net/ipv4/igmp.c
===
--- linux-2.6-netns.orig/net/ipv4/igmp.c
+++ linux-2.6-netns/net/ipv4/igmp.c
@@ -2291,6 +2291,7 @@ int ip_check_mc(struct in_device *in_dev
 struct igmp_mc_iter_state {
  struct net_device *dev;
  struct in_device *in_dev;
+ struct net *net;
 };

 #define igmp_mc_seq_private(seq) ((struct igmp_mc_iter_state *)(seq)->private)
@@ -2299,9 +2300,10 @@ static inline struct ip_mc_list *igmp_mc
 {
  struct ip_mc_list *im = NULL;
  struct igmp_mc_iter_state *state = igmp_mc_seq_private(seq);
+ struct net *net = state->net;

  state->in_dev = NULL;
- for_each_netdev(&init_net, state->dev) {
+ for_each_netdev(net, state->dev) {
   struct in_device *in_dev;
   in_dev = in_dev_get(state->dev);
   if (!in_dev)
@@ -2426,31 +2428,47 @@ static int igmp_mc_seq_open(struct inode

  if (!s)
   goto out;
+
+ s->net = get_proc_net(inode);
+ if (!s->net)
+ goto out_kfree;
+
  rc = seq_open(file, &igmp_mc_seq_ops);
  if (rc)
- goto out_kfree;

```
+  goto out_put_net;

  seq = file->private_data;
  seq->private = s;
 out:
  return rc;
+out_put_net:
+ put_net(s->net);
 out_kfree:
  kfree(s);
  goto out;
 }

+static int igmp_mc_seq_release(struct inode *inode, struct file *file)
+{
+ struct seq_file *seq = file->private_data;
+ struct igmp_mc_iter_state *state = seq->private;
+ put_net(state->net);
+ return seq_release_private(inode, file);
+}
+
 static const struct file_operations igmp_mc_seq_fops = {
  .owner  = THIS_MODULE,
  .open  = igmp_mc_seq_open,
  .read  = seq_read,
  .llseek  = seq_lseek,
- .release = seq_release_private,
+ .release = igmp_mc_seq_release,
 };

 struct igmp_mcf_iter_state {
  struct net_device *dev;
  struct in_device *idev;
  struct ip_mc_list *im;
+ struct net *net;
 };

 #define igmp_mcf_seq_private(seq) ((struct igmp_mcf_iter_state *)(seq)->private)
@@ -2460,10 +2478,11 @@ static inline struct ip_sf_list *igmp_mc
  struct ip_sf_list *psf = NULL;
  struct ip_mc_list *im = NULL;
  struct igmp_mcf_iter_state *state = igmp_mcf_seq_private(seq);
+ struct net *net = state->net;

  state->idev = NULL;
  state->im = NULL;
- for_each_netdev(&init_net, state->dev) {
+ for_each_netdev(net, state->dev) {
```

```
   struct in_device *idev;
   idev = in_dev_get(state->dev);
   if (unlikely(idev == NULL))
@@ -2600,31 +2619,64 @@ static int igmp_mcf_seq_open(struct inod

  if (!s)
   goto out;
+
+ s->net = get_proc_net(inode);
+ if (!s->net)
+  goto out_kfree;
+
  rc = seq_open(file, &igmp_mcf_seq_ops);
  if (rc)
-  goto out_kfree;
+  goto out_put_net;

  seq = file->private_data;
  seq->private = s;
 out:
  return rc;
+out_put_net:
+ put_net(s->net);
 out_kfree:
  kfree(s);
  goto out;
 }

+static int igmp_mcf_seq_release(struct inode *inode, struct file *file)
+{
+ struct seq_file *seq = file->private_data;
+ struct igmp_mcf_iter_state *state = seq->private;
+ put_net(state->net);
+ return seq_release_private(inode, file);
+
+}
+
 static const struct file_operations igmp_mcf_seq_fops = {
  .owner  = THIS_MODULE,
  .open  = igmp_mcf_seq_open,
  .read  = seq_read,
  .llseek  = seq_lseek,
- .release = seq_release_private,
+ .release = igmp_mcf_seq_release,
+};
+
+static int igmp_mc_net_init(struct net *net)
+{
```

```
+ proc_net_fops_create(net, "igmp", S_IRUGO, &igmp_mc_seq_fops);
+ proc_net_fops_create(net, "mcfilter", S_IRUGO, &igmp_mcf_seq_fops);
+ return 0;
+}
+
+static void igmp_mc_net_exit(struct net *net)
+{
+ proc_net_remove(net, "igmp");
+ proc_net_remove(net, "mcfilter");
+}
+
+struct pernet_operations igmp_mc_net_ops = {
+ .init = igmp_mc_net_init,
+ .exit = igmp_mc_net_exit,
 };

 int __init igmp_mc_proc_init(void)
 {
- proc_net_fops_create(&init_net, "igmp", S_IRUGO, &igmp_mc_seq_fops);
- proc_net_fops_create(&init_net, "mcfilter", S_IRUGO, &igmp_mcf_seq_fops);
+ register_pernet_subsys(&igmp_mc_net_ops);
  return 0;
 }
 #endif

--
```