
Subject: [PATCH net-next 4/9] netns: register net.ipv4.route.flush in each namespace

Posted by [den](#) on Fri, 04 Jul 2008 13:17:07 GMT

[View Forum Message](#) <> [Reply to Message](#)

Signed-off-by: Denis V. Lunev <den@openvz.org>

```
include/net/netns/ipv4.h | 1 +
net/ipv4/route.c          | 79 ++++++-----
2 files changed, 70 insertions(+), 10 deletions(-)
```

diff --git a/include/net/netns/ipv4.h b/include/net/netns/ipv4.h

index 6ef90b5..a29adf1 100644

--- a/include/net/netns/ipv4.h

+++ b/include/net/netns/ipv4.h

@@ -18,6 +18,7 @@ struct netns_ipv4 {

struct ctl_table_header *forw_hdr;

struct ctl_table_header *frags_hdr;

struct ctl_table_header *ipv4_hdr;

+ struct ctl_table_header *route_hdr;

#endif

struct ipv4_devconf *devconf_all;

struct ipv4_devconf *devconf_dflt;

diff --git a/net/ipv4/route.c b/net/ipv4/route.c

index 790de32..6fe799d 100644

--- a/net/ipv4/route.c

+++ b/net/ipv4/route.c

@@ -2835,6 +2835,7 @@ static int ipv4_sysctl_rtcache_flush(ctl_table *ctl, int write,

{

if (write) {

int flush_delay;

+ struct net *net;

static DEFINE_MUTEX(flush_mutex);

mutex_lock(&flush_mutex);

@@ -2843,7 +2844,8 @@ static int ipv4_sysctl_rtcache_flush(ctl_table *ctl, int write,

ctl->data = NULL;

mutex_unlock(&flush_mutex);

- rt_cache_flush(&init_net, flush_delay);

+ net = (struct net *)ctl->extra1;

+ rt_cache_flush(net, flush_delay);

return 0;

}

@@ -2859,24 +2861,18 @@ static int ipv4_sysctl_rtcache_flush_strategy(ctl_table *table,

size_t newlen)

{

```

    int delay;
+ struct net *net;
    if (newlen != sizeof(int))
        return -EINVAL;
    if (get_user(delay, (int __user *)newval))
        return -EFAULT;
- rt_cache_flush(&init_net, delay);
+ net = (struct net *)table->extra1;
+ rt_cache_flush(net, delay);
    return 0;
}

ctl_table ipv4_route_table[] = {
{
- .ctl_name = NET_IPV4_ROUTE_FLUSH,
- .procname = "flush",
- .maxlen = sizeof(int),
- .mode = 0200,
- .proc_handler = &ipv4_sysctl_rtcache_flush,
- .strategy = &ipv4_sysctl_rtcache_flush_strategy,
- },
- {
    .ctl_name = NET_IPV4_ROUTE_GC_THRESH,
    .procname = "gc_thresh",
    .data = &ipv4_dst_ops.gc_thresh,
@@ -3014,6 +3010,66 @@ ctl_table ipv4_route_table[] = {
    },
    { .ctl_name = 0 }
};
+
+static __net_initdata struct ctl_path ipv4_route_path[] = {
+ { .procname = "net", .ctl_name = CTL_NET, },
+ { .procname = "ipv4", .ctl_name = NET_IPV4, },
+ { .procname = "route", .ctl_name = NET_IPV4_ROUTE, },
+ { },
+};
+
+
+static struct ctl_table ipv4_route_flush_table[] = {
+ {
+ .ctl_name = NET_IPV4_ROUTE_FLUSH,
+ .procname = "flush",
+ .maxlen = sizeof(int),
+ .mode = 0200,
+ .proc_handler = &ipv4_sysctl_rtcache_flush,
+ .strategy = &ipv4_sysctl_rtcache_flush_strategy,
+ },
+ { .ctl_name = 0 },

```

```

+};
+
+static __net_init int sysctl_route_net_init(struct net *net)
+{
+ struct ctl_table *tbl;
+
+ tbl = ipv4_route_flush_table;
+ if (net != &init_net) {
+  tbl = kmemdup(tbl, sizeof(ipv4_route_flush_table), GFP_KERNEL);
+  if (tbl == NULL)
+   goto err_dup;
+ }
+ tbl[0].extra1 = net;
+
+ net->ipv4.route_hdr =
+ register_net_sysctl_table(net, ipv4_route_path, tbl);
+ if (net->ipv4.route_hdr == NULL)
+  goto err_reg;
+ return 0;
+
+err_reg:
+ if (tbl != ipv4_route_flush_table)
+  kfree(tbl);
+err_dup:
+ return -ENOMEM;
+}
+
+static __net_exit void sysctl_route_net_exit(struct net *net)
+{
+ struct ctl_table *tbl;
+
+ tbl = net->ipv4.route_hdr->ctl_table_arg;
+ unregister_net_sysctl_table(net->ipv4.route_hdr);
+ BUG_ON(tbl == ipv4_route_flush_table);
+ kfree(tbl);
+}
+
+static __net_initdata struct pernet_operations sysctl_route_ops = {
+ .init = sysctl_route_net_init,
+ .exit = sysctl_route_net_exit,
+};
+
+#endif
+
+#ifdef CONFIG_NET_CLS_ROUTE
@@ -3090,6 +3146,9 @@ int __init ip_rt_init(void)
+
+ rtnl_register(PF_INET, RTM_GETROUTE, inet_rtm_getroute, NULL);

```

```
+#ifdef CONFIG_SYSCTL
+ register_pernet_subsys(&sysctl_route_ops);
+#endif
    return rc;
}
```

--
1.5.3.rc5

Containers mailing list
Containers@lists.linux-foundation.org
<https://lists.linux-foundation.org/mailman/listinfo/containers>
