Subject: [patch 24/38][IPV6] rt6_stats - make rt6_stats per namespace
Posted by Daniel Lezcano on Mon, 03 Dec 2007 16:17:00 GMT
View Forum Message <> Reply to Message

The rt6_stats is now per namespace with this patch. It is
allocated when a network namespace is created and freed
when the network namespace exits and references are relative
to the network namespace.

Signed-off-by: Benjamin Thery <benjamin.thery@bull.net>
Signed-off-by: Daniel Lezcano <dlezcano@fr.ibm.com>
---
 net/ipv6/ip6_fib.c | 26 ++++++++++++++++++----------
 1 file changed, 15 insertions(+), 11 deletions(-)

Index: linux-2.6-netns/net/ipv6/ip6_fib.c
===
--- linux-2.6-netns.orig/net/ipv6/ip6_fib.c
+++ linux-2.6-netns/net/ipv6/ip6_fib.c
@@ -601,6 +601,7 @@ static int fib6_add_rt2node(struct fib6_
 {
  struct rt6_info *iter = NULL;
  struct rt6_info **ins;
+ struct net *net = &init_net;

  ins = &fn->leaf;

@@ -648,10 +649,10 @@ static int fib6_add_rt2node(struct fib6_
  rt->rt6i_node = fn;
  atomic_inc(&rt->rt6i_ref);
  inet6_rt_notify(RTM_NEWROUTE, rt, info);
- init_net.rt6_stats->fib_rt_entries++;
+ net->rt6_stats->fib_rt_entries++;

  if ((fn->fn_flags & RTN_RTINFO) == 0) {
- init_net.rt6_stats->fib_route_nodes++;
+ net->rt6_stats->fib_route_nodes++;
   fn->fn_flags |= RTN_RTINFO;
  }

@@ -1080,14 +1081,15 @@ static void fib6_del_route(struct fib6_n
 {
  struct fib6_walker_t *w;
  struct rt6_info *rt = *rtp;
+ struct net *net = &init_net;

  RT6_TRACE("fib6_del_route\n");

```
  /* Unlink it */
  *rtp = rt->u.dst.rt6_next;
  rt->rt6i_node = NULL;
- init_net.rt6_stats->fib_rt_entries--;
- init_net.rt6_stats->fib_discarded_routes++;
+ net->rt6_stats->fib_rt_entries--;
+ net->rt6_stats->fib_discarded_routes++;

  /* Reset round-robin state, if necessary */
  if (fn->rr_ptr == rt)
@@ -1113,7 +1115,7 @@ static void fib6_del_route(struct fib6_n
  /* If it was last route, expunge its radix tree node */
  if (fn->leaf == NULL) {
   fn->fn_flags &= ~RTN_RTINFO;
-  init_net.rt6_stats->fib_route_nodes--;
+  net->rt6_stats->fib_route_nodes--;
   fn = fib6_repair_tree(fn);
  }

@@ -1482,10 +1484,14 @@ static int fib6_net_init(struct net *net
  timer->base = &boot_tvec_bases;
   net->ip6_fib_timer = timer;

+  net->rt6_stats = kzalloc(sizeof(*net->rt6_stats), GFP_KERNEL);
+  if (!net->rt6_stats)
+   goto out_timer;
+
  net->fib_table_hash = kzalloc(sizeof(*net->fib_table_hash)*FIB_TABLE_HASHSZ,
          GFP_KERNEL);
  if (!net->fib_table_hash)
-  goto out_timer;
+  goto out_rt6_stats;

  net->fib6_main_tbl = kzalloc(sizeof(*net->fib6_main_tbl), GFP_KERNEL);
  if (!net->fib6_main_tbl)
@@ -1512,6 +1518,8 @@ static int fib6_net_init(struct net *net

 out_fib6_main_tbl:
  kfree(net->fib_table_hash);
+out_rt6_stats:
+ kfree(net->rt6_stats);
 out_timer:
  kfree(timer);
 out:
@@ -1528,6 +1536,7 @@ static void fib6_net_exit(struct net *ne
 #endif
  kfree(net->fib6_main_tbl);
  kfree(net->fib_table_hash);
```

```
+ kfree(net->rt6_stats);
 }

 static struct pernet_operations fib6_net_ops = {
@@ -1542,10 +1551,6 @@ void __init fib6_init(void)
         0, SLAB_HWCACHE_ALIGN|SLAB_PANIC,
         NULL);

-   init_net.rt6_stats = kzalloc(sizeof(*init_net.rt6_stats), GFP_KERNEL);
-   if (!init_net.rt6_stats)
-    panic("IPV6: failed to allocate rt6_stats.\n");
-
  register_pernet_subsys(&fib6_net_ops);
         __rtnl_register(PF_INET6, RTM_GETROUTE, NULL, inet6_dump_fib);
 }
@@ -1553,6 +1558,5 @@ void __init fib6_init(void)
 void fib6_gc_cleanup(void)
 {
  unregister_pernet_subsys(&fib6_net_ops);
- kfree(init_net.rt6_stats);
  kmem_cache_destroy(fib6_node_kmem);
 }

--
```