Subject: [PATCH 2/3] [NETNS49] Add struct net to all calls for rt_cache_flush
Posted by den on Thu, 18 Oct 2007 10:01:27 GMT
View Forum Message <> Reply to Message

commit 628d4768843fc6d572596b91e29f8ce6506c276a
Author: Denis V. Lunev <den@openvz.org>
Date: Thu Oct 18 12:48:38 2007 +0400

    Add struct net to all calls for rt_cache_flush. Additionally, the function
    manipulating with timer should not actually call the garbage collector as
    the namespace is not known in the global rt_secret_rebuild.

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

```
diff --git a/include/net/route.h b/include/net/route.h
index 688c750..3372dce 100644
--- a/include/net/route.h
+++ b/include/net/route.h
@@ -111,7 +111,7 @@ extern int  ip_rt_init(void);
 extern void  ip_rt_redirect(__be32 old_gw, __be32 dst, __be32 new_gw,
          __be32 src, struct net_device *dev);
 extern void  ip_rt_advice(struct rtable **rp, int advice);
-extern void  rt_cache_flush(int how);
+extern void  rt_cache_flush(struct net *net, int how);
 extern int  __ip_route_output_key(struct rtable **, const struct flowi *flp);
 extern int  ip_route_output_key(struct rtable **, struct flowi *flp);
 extern int  ip_route_output_flow(struct rtable **rp, struct flowi *flp, struct sock *sk, int flags);
diff --git a/net/ipv4/arp.c b/net/ipv4/arp.c
index 71c4d6d..7d9fb24 100644
--- a/net/ipv4/arp.c
+++ b/net/ipv4/arp.c
@@ -1222,7 +1222,7 @@ static int arp_netdev_event(struct notifier_block *this, unsigned long
event, vo
  switch (event) {
  case NETDEV_CHANGEADDR:
   neigh_changeaddr(&arp_tbl, dev);
-  rt_cache_flush(0);
+  rt_cache_flush(dev->nd_net, 0);
   break;
  default:
   break;
diff --git a/net/ipv4/devinet.c b/net/ipv4/devinet.c
index 32f52d4..ab196a1 100644
--- a/net/ipv4/devinet.c
+++ b/net/ipv4/devinet.c
@@ -56,6 +56,7 @@
 #include <linux/sysctl.h>
 #endif
```

```
 #include <linux/kmod.h>
+#include <linux/nsproxy.h>

 #include <net/arp.h>
 #include <net/ip.h>
@@ -1348,7 +1349,7 @@ void inet_forward_change(struct net *net)
  }
  read_unlock(&dev_base_lock);

- rt_cache_flush(0);
+ rt_cache_flush(net, 0);
 }

 static int devinet_sysctl_forward(struct ctl_table *ctl, int write,
@@ -1364,7 +1365,7 @@ static int devinet_sysctl_forward(struct ctl_table *ctl, int write,
   if (valp == &IPV4_DEVCONF_ALL(net, FORWARDING))
    inet_forward_change(net);
   else if (valp != &IPV4_DEVCONF_DFLT(net, FORWARDING))
-   rt_cache_flush(0);
+   rt_cache_flush(current->nsproxy->net_ns, 0);
  }

  return ret;
@@ -1379,7 +1380,7 @@ int ipv4_doint_and_flush(struct ctl_table *ctl, int write,
  int ret = proc_dointvec(ctl, write, filp, buffer, lenp, ppos);

  if (write && *valp != val)
-  rt_cache_flush(0);
+  rt_cache_flush(current->nsproxy->net_ns, 0);

  return ret;
 }
@@ -1392,7 +1393,7 @@ int ipv4_doint_and_flush_strategy(struct ctl_table *table, int __user
*name, int
          newval, newlen);

  if (ret == 1)
-  rt_cache_flush(0);
+  rt_cache_flush(current->nsproxy->net_ns, 0);

  return ret;
 }
diff --git a/net/ipv4/fib_frontend.c b/net/ipv4/fib_frontend.c
index ad6bfce..835a8b7 100644
--- a/net/ipv4/fib_frontend.c
+++ b/net/ipv4/fib_frontend.c
@@ -113,7 +113,7 @@ static void fib_flush(struct net *net)
  }
```

```
  if (flushed)
-  rt_cache_flush(-1);
+  rt_cache_flush(net, -1);
 }

 /*
@@ -866,21 +866,22 @@ static void fib_disable_ip(struct net_device *dev, int force)
  struct net *net = dev->nd_net;
  if (fib_sync_down(net, 0, dev, force))
   fib_flush(net);
- rt_cache_flush(0);
+ rt_cache_flush(dev->nd_net, 0);
  arp_ifdown(dev);
 }

 static int fib_inetaddr_event(struct notifier_block *this, unsigned long event, void *ptr)
 {
  struct in_ifaddr *ifa = (struct in_ifaddr*)ptr;
+ struct net_device *dev = ifa->ifa_dev->dev;

  switch (event) {
  case NETDEV_UP:
   fib_add_ifaddr(ifa);
 #ifdef CONFIG_IP_ROUTE_MULTIPATH
-  fib_sync_up(ifa->ifa_dev->dev);
+  fib_sync_up(dev);
 #endif
-  rt_cache_flush(-1);
+  rt_cache_flush(dev->nd_net, -1);
   break;
  case NETDEV_DOWN:
   fib_del_ifaddr(ifa);
@@ -888,9 +889,9 @@ static int fib_inetaddr_event(struct notifier_block *this, unsigned long
event,
     /* Last address was deleted from this interface.
        Disable IP.
     */
-   fib_disable_ip(ifa->ifa_dev->dev, 1);
+   fib_disable_ip(dev, 1);
   } else {
-   rt_cache_flush(-1);
+   rt_cache_flush(dev->nd_net, -1);
   }
   break;
  }
@@ -921,14 +922,14 @@ static int fib_netdev_event(struct notifier_block *this, unsigned long
event, vo
```

```
 #ifdef CONFIG_IP_ROUTE_MULTIPATH
   fib_sync_up(dev);
 #endif
-  rt_cache_flush(-1);
+  rt_cache_flush(dev->nd_net, -1);
    break;
   case NETDEV_DOWN:
    fib_disable_ip(dev, 0);
    break;
   case NETDEV_CHANGEMTU:
   case NETDEV_CHANGE:
-  rt_cache_flush(0);
+  rt_cache_flush(dev->nd_net, 0);
    break;
   }
   return NOTIFY_DONE;
diff --git a/net/ipv4/fib_hash.c b/net/ipv4/fib_hash.c
index 5d28d44..c32bd0a 100644
--- a/net/ipv4/fib_hash.c
+++ b/net/ipv4/fib_hash.c
@@ -455,7 +455,7 @@ static int fn_hash_insert(struct fib_table *tb, struct fib_config *cfg)

    fib_release_info(fi_drop);
    if (state & FA_S_ACCESSED)
-    rt_cache_flush(-1);
+    rt_cache_flush(cfg->fc_nlinfo.net, -1);
    rtmsg_fib(RTM_NEWROUTE, key, fa, cfg->fc_dst_len, tb->tb_id,
       &cfg->fc_nlinfo, NLM_F_REPLACE);
    return 0;
@@ -522,7 +522,7 @@ static int fn_hash_insert(struct fib_table *tb, struct fib_config *cfg)

   if (new_f)
    fz->fz_nent++;
- rt_cache_flush(-1);
+ rt_cache_flush(cfg->fc_nlinfo.net, -1);

   rtmsg_fib(RTM_NEWROUTE, key, new_fa, cfg->fc_dst_len, tb->tb_id,
      &cfg->fc_nlinfo, 0);
@@ -604,7 +604,7 @@ static int fn_hash_delete(struct fib_table *tb, struct fib_config *cfg)
   write_unlock_bh(&fib_hash_lock);

   if (fa->fa_state & FA_S_ACCESSED)
-    rt_cache_flush(-1);
+    rt_cache_flush(cfg->fc_nlinfo.net, -1);
   fn_free_alias(fa);
   if (kill_fn) {
    fn_free_node(f);
diff --git a/net/ipv4/fib_rules.c b/net/ipv4/fib_rules.c
```

```
index c93b278..63df6f9 100644
--- a/net/ipv4/fib_rules.c
+++ b/net/ipv4/fib_rules.c
@@ -281,7 +281,7 @@ static size_t fib4_rule_nlmsg_payload(struct fib_rule *rule)

 static void fib4_rule_flush_cache(struct net *net, struct fib_rules_ops *ops)
 {
- rt_cache_flush(-1);
+ rt_cache_flush(net, -1);
 }

 static struct fib4_rule_table fib4_rule_table = {
diff --git a/net/ipv4/fib_trie.c b/net/ipv4/fib_trie.c
index 1415ad4..8a24f21 100644
--- a/net/ipv4/fib_trie.c
+++ b/net/ipv4/fib_trie.c
@@ -1230,7 +1230,7 @@ static int fn_trie_insert(struct fib_table *tb, struct fib_config *cfg)

    fib_release_info(fi_drop);
    if (state & FA_S_ACCESSED)
-    rt_cache_flush(-1);
+    rt_cache_flush(cfg->fc_nlinfo.net, -1);
    rtmsg_fib(RTM_NEWROUTE, htonl(key), new_fa, plen,
     tb->tb_id, &cfg->fc_nlinfo, NLM_F_REPLACE);

@@ -1283,7 +1283,7 @@ static int fn_trie_insert(struct fib_table *tb, struct fib_config *cfg)
  list_add_tail_rcu(&new_fa->fa_list,
     (fa ? &fa->fa_list : fa_head));

- rt_cache_flush(-1);
+ rt_cache_flush(cfg->fc_nlinfo.net, -1);
  rtmsg_fib(RTM_NEWROUTE, htonl(key), new_fa, plen, tb->tb_id,
     &cfg->fc_nlinfo, 0);
 succeeded:
@@ -1649,7 +1649,7 @@ static int fn_trie_delete(struct fib_table *tb, struct fib_config *cfg)
   trie_leaf_remove(t, key);

  if (fa->fa_state & FA_S_ACCESSED)
-  rt_cache_flush(-1);
+  rt_cache_flush(cfg->fc_nlinfo.net, -1);

  fib_release_info(fa->fa_info);
  alias_free_mem_rcu(fa);
diff --git a/net/ipv4/route.c b/net/ipv4/route.c
index f9a59ff..ed1842b 100644
--- a/net/ipv4/route.c
+++ b/net/ipv4/route.c
@@ -91,6 +91,7 @@
```

```
 #include <linux/jhash.h>
 #include <linux/rcupdate.h>
 #include <linux/times.h>
+#include <linux/nsproxy.h>
 #include <net/net_namespace.h>
 #include <net/protocol.h>
 #include <net/ip.h>
@@ -670,7 +671,7 @@ static void rt_run_flush(unsigned long dummy)

 static DEFINE_SPINLOCK(rt_flush_lock);

-void rt_cache_flush(int delay)
+static int __rt_cache_flush(int delay)
 {
  unsigned long now = jiffies;
  int user_mode = !in_softirq();
@@ -699,8 +700,7 @@ void rt_cache_flush(int delay)

  if (delay <= 0) {
   spin_unlock_bh(&rt_flush_lock);
-  rt_run_flush(0);
-  return;
+  return 1;
  }

  if (rt_deadline == 0)
@@ -708,13 +708,21 @@ void rt_cache_flush(int delay)

  mod_timer(&rt_flush_timer, now+delay);
  spin_unlock_bh(&rt_flush_lock);
+ return 0;
+}
+
+void rt_cache_flush(struct net *net, int delay)
+{
+ if (__rt_cache_flush(delay))
+  rt_run_flush(0);
 }

 static void rt_secret_rebuild(unsigned long dummy)
 {
  unsigned long now = jiffies;

- rt_cache_flush(0);
+ __rt_cache_flush(0);
+ rt_run_flush(0);
  mod_timer(&rt_secret_timer, now + ip_rt_secret_interval);
 }
```

```
@@ -2716,7 +2724,7 @@ done:

 void ip_rt_multicast_event(struct in_device *in_dev)
 {
- rt_cache_flush(0);
+ rt_cache_flush(in_dev->dev->nd_net, 0);
 }

 #ifdef CONFIG_SYSCTL
@@ -2728,7 +2736,7 @@ static int ipv4_sysctl_rtcache_flush(struct ctl_table *ctl, int write,
 {
  if (write) {
   proc_dointvec(ctl, write, filp, buffer, lenp, ppos);
-  rt_cache_flush(flush_delay);
+  rt_cache_flush(current->nsproxy->net_ns, flush_delay);
   return 0;
  }

@@ -2748,7 +2756,7 @@ static int ipv4_sysctl_rtcache_flush_strategy(struct ctl_table *table,
   return -EINVAL;
  if (get_user(delay, (int __user *)newval))
   return -EFAULT;
- rt_cache_flush(delay);
+ rt_cache_flush(current->nsproxy->net_ns, delay);
  return 0;
 }
```