Subject: [PATCH v4 2/3] SUNRPC: optimize net_ns dereferencing in rpcbind creation calls
Posted by Stanislav Kinsbursky on Tue, 08 Nov 2011 10:43:17 GMT
View Forum Message <> Reply to Message

Static rpcbind creation functions can be parametrized by network namespace
pointer, calculated only once, instead of using init_net pointer (or taking it
from current when virtualization will be comleted) in many places.

Signed-off-by: Stanislav Kinsbursky <skinsbursky@parallels.com>

---
 net/sunrpc/rpcb_clnt.c | 35 ++++++++++++++++++++----------------
 1 files changed, 19 insertions(+), 16 deletions(-)

```
diff --git a/net/sunrpc/rpcb_clnt.c b/net/sunrpc/rpcb_clnt.c
index 7d32f19..3df276a 100644
--- a/net/sunrpc/rpcb_clnt.c
+++ b/net/sunrpc/rpcb_clnt.c
@@ -161,10 +161,10 @@ static void rpcb_map_release(void *data)
  kfree(map);
 }

-static int rpcb_get_local(void)
+static int rpcb_get_local(struct net *net)
 {
  int cnt;
- struct sunrpc_net *sn = net_generic(&init_net, sunrpc_net_id);
+ struct sunrpc_net *sn = net_generic(net, sunrpc_net_id);

  spin_lock(&sn->rpcb_clnt_lock);
  if (sn->rpcb_users)
@@ -201,9 +201,10 @@ void rpcb_put_local(void)
  }
 }

-static void rpcb_set_local(struct rpc_clnt *clnt, struct rpc_clnt *clnt4)
+static void rpcb_set_local(struct net *net, struct rpc_clnt *clnt,
+   struct rpc_clnt *clnt4)
 {
- struct sunrpc_net *sn = net_generic(&init_net, sunrpc_net_id);
+ struct sunrpc_net *sn = net_generic(net, sunrpc_net_id);

  /* Protected by rpcb_create_local_mutex */
  sn->rpcb_local_clnt = clnt;
@@ -211,22 +212,23 @@ static void rpcb_set_local(struct rpc_clnt *clnt, struct rpc_clnt *clnt4)
  smp_wmb();
  sn->rpcb_users = 1;
```

```
  dprintk("RPC:       created new rpcb local clients (rpcb_local_clnt: "
-   "%p, rpcb_local_clnt4: %p)\n", sn->rpcb_local_clnt,
-   sn->rpcb_local_clnt4);
+   "%p, rpcb_local_clnt4: %p) for net %p%s\n",
+   sn->rpcb_local_clnt, sn->rpcb_local_clnt4,
+   net, (net == &init_net) ? " (init_net)" : "");
 }

 /*
  * Returns zero on success, otherwise a negative errno value
  * is returned.
  */
-static int rpcb_create_local_unix(void)
+static int rpcb_create_local_unix(struct net *net)
 {
  static const struct sockaddr_un rpcb_localaddr_rpcbind = {
   .sun_family  = AF_LOCAL,
   .sun_path  = RPCBIND_SOCK_PATHNAME,
  };
  struct rpc_create_args args = {
-  .net  = &init_net,
+  .net  = net,
   .protocol = XPRT_TRANSPORT_LOCAL,
   .address = (struct sockaddr *)&rpcb_localaddr_rpcbind,
   .addrsize = sizeof(rpcb_localaddr_rpcbind),
@@ -259,7 +261,7 @@ static int rpcb_create_local_unix(void)
   clnt4 = NULL;
  }

- rpcb_set_local(clnt, clnt4);
+ rpcb_set_local(net, clnt, clnt4);

 out:
  return result;
@@ -269,7 +271,7 @@ out:
  * Returns zero on success, otherwise a negative errno value
  * is returned.
  */
-static int rpcb_create_local_net(void)
+static int rpcb_create_local_net(struct net *net)
 {
  static const struct sockaddr_in rpcb_inaddr_loopback = {
   .sin_family  = AF_INET,
@@ -277,7 +279,7 @@ static int rpcb_create_local_net(void)
   .sin_port  = htons(RPCBIND_PORT),
  };
  struct rpc_create_args args = {
-  .net  = &init_net,
```

```
+  .net  = net,
   .protocol = XPRT_TRANSPORT_TCP,
   .address = (struct sockaddr *)&rpcb_inaddr_loopback,
   .addrsize = sizeof(rpcb_inaddr_loopback),
@@ -311,7 +313,7 @@ static int rpcb_create_local_net(void)
   clnt4 = NULL;
  }

- rpcb_set_local(clnt, clnt4);
+ rpcb_set_local(net, clnt, clnt4);

 out:
  return result;
@@ -325,16 +327,17 @@ int rpcb_create_local(void)
 {
  static DEFINE_MUTEX(rpcb_create_local_mutex);
  int result = 0;
+ struct net *net = &init_net;

- if (rpcb_get_local())
+ if (rpcb_get_local(net))
   return result;

  mutex_lock(&rpcb_create_local_mutex);
- if (rpcb_get_local())
+ if (rpcb_get_local(net))
   goto out;

- if (rpcb_create_local_unix() != 0)
-  result = rpcb_create_local_net();
+ if (rpcb_create_local_unix(net) != 0)
+  result = rpcb_create_local_net(net);

 out:
  mutex_unlock(&rpcb_create_local_mutex);
```