Subject: [PATCH 05/11] SUNRPC: register debug sysctl table per network namespace
Posted by Stanislav Kinsbursky on Wed, 14 Dec 2011 10:47:39 GMT
View Forum Message <> Reply to Message

This patch introduces per-net sysctl header for debug table and moves debug
table registration to per-net opreations.
Also, this patch moves SUNRPC sysctl root registration prior to per-net
operations, since now they depends on it.

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

---
 net/sunrpc/netns.h | 1 +
 net/sunrpc/sunrpc_syms.c | 9 ++++++---
 net/sunrpc/sysctl.c | 17 +++++++++----------
 3 files changed, 14 insertions(+), 13 deletions(-)

diff --git a/net/sunrpc/netns.h b/net/sunrpc/netns.h
index a949ddb..05ce3b8 100644
--- a/net/sunrpc/netns.h
+++ b/net/sunrpc/netns.h
@@ -14,6 +14,7 @@ struct sunrpc_net {
  spinlock_t rpc_client_lock;
 #ifdef RPC_DEBUG
  struct ctl_table_set sysctls;
+ struct ctl_table_header *debug_ctl_header;
 #endif
 };

diff --git a/net/sunrpc/sunrpc_syms.c b/net/sunrpc/sunrpc_syms.c
index 44759e9..0fd156a 100644
--- a/net/sunrpc/sunrpc_syms.c
+++ b/net/sunrpc/sunrpc_syms.c
@@ -90,18 +90,21 @@ init_sunrpc(void)

  cache_initialize();

- err = register_pernet_subsys(&sunrpc_net_ops);
- if (err)
- goto out4;
 #ifdef RPC_DEBUG
  rpc_register_sysctl();
 #endif
+ err = register_pernet_subsys(&sunrpc_net_ops);
+ if (err)
+ goto out4;
  cache_register(&unix_gid_cache);

```
  svc_init_xprt_sock(); /* svc sock transport */
  init_socket_xprt(); /* clnt sock transport */
  return 0;

 out4:
+#ifdef RPC_DEBUG
+ rpc_unregister_sysctl();
+#endif
  rpcauth_remove_module();
 out3:
  rpc_destroy_mempool();
diff --git a/net/sunrpc/sysctl.c b/net/sunrpc/sysctl.c
index adebf0a..eda80cf 100644
--- a/net/sunrpc/sysctl.c
+++ b/net/sunrpc/sysctl.c
@@ -42,7 +42,6 @@ EXPORT_SYMBOL_GPL(nlm_debug);
 #include <linux/nsproxy.h>
 #include "netns.h"

-static struct ctl_table_header *sunrpc_table_header;
 static ctl_table  debug_table[];

 struct ctl_path sunrpc_path[] = {
@@ -79,6 +78,9 @@ int debug_sysctl_init(struct net *net)
  struct sunrpc_net *sn = net_generic(net, sunrpc_net_id);

  setup_sysctl_set(&sn->sysctls, NULL, NULL);
+ sn->debug_ctl_header = register_sunrpc_sysctl(net, debug_table);
+ if (sn->debug_ctl_header == NULL)
+  return -ENOMEM;
  return 0;
 }

@@ -86,26 +88,21 @@ void debug_sysctl_exit(struct net *net)
 {
  struct sunrpc_net *sn = net_generic(net, sunrpc_net_id);

+ unregister_sysctl_table(sn->debug_ctl_header);
+ sn->debug_ctl_header = NULL;
  WARN_ON(!list_empty(&sn->sysctls.list));
 }

 void
 rpc_register_sysctl(void)
 {
- if (!sunrpc_table_header) {
-  register_sysctl_root(&sunrpc_debug_root);
-  sunrpc_table_header = register_sunrpc_sysctl(&init_net, debug_table);
```

```
- }
+ register_sysctl_root(&sunrpc_debug_root);
 }

 void
 rpc_unregister_sysctl(void)
 {
- if (sunrpc_table_header) {
-  unregister_sysctl_table(sunrpc_table_header);
-  unregister_sysctl_root(&sunrpc_debug_root);
-  sunrpc_table_header = NULL;
- }
+ unregister_sysctl_root(&sunrpc_debug_root);
 }

 static int proc_do_xprt(ctl_table *table, int write,
```