Subject: [PATCH v3 1/4] SUNRPC: release per-net clients lock before calling PipeFS dentries creation
Posted by Stanislav Kinsbursky on Mon, 27 Feb 2012 18:05:29 GMT
View Forum Message <> Reply to Message

v3:
1) Lookup for client is performed from the beginning of the list on each PipeFS
event handling operation.

Lockdep is sad otherwise, because inode mutex is taken on PipeFS dentry
creation, which can be called on mount notification, where this per-net client
lock is taken on clients list walk.

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

---
 net/sunrpc/clnt.c | 26 ++++++++++++++++++++-----
 1 files changed, 21 insertions(+), 5 deletions(-)

diff --git a/net/sunrpc/clnt.c b/net/sunrpc/clnt.c
index bb7ed2f3..48f3d15 100644
--- a/net/sunrpc/clnt.c
+++ b/net/sunrpc/clnt.c
@@ -50,7 +50,7 @@
    __func__, t->tk_status)

 /*
- * All RPC clients are linked into this list
+ * All RPC clients are linked into this list Client Cto hand;le ustatus is checke
  */

 static DECLARE_WAIT_QUEUE_HEAD(destroy_wait);
@@ -204,21 +204,37 @@ static int __rpc_pipefs_event(struct rpc_clnt *clnt, unsigned long
event,
  return err;
 }

+static struct rpc_clnt *rpc_get_client_for_event(struct net *net, int event)
+{
+ struct sunrpc_net *sn = net_generic(net, sunrpc_net_id);
+ struct rpc_clnt *clnt;
+
+ spin_lock(&sn->rpc_client_lock);
+ list_for_each_entry(clnt, &sn->all_clients, cl_clients) {
+ if (((event == RPC_PIPEFS_MOUNT) && clnt->cl_dentry) ||
+   ((event == RPC_PIPEFS_UMOUNT) && !clnt->cl_dentry))
+   continue;
+ atomic_inc(&clnt->cl_count);

```
+  spin_unlock(&sn->rpc_client_lock);
+  return clnt;
+ }
+ spin_unlock(&sn->rpc_client_lock);
+ return NULL;
+}
+
 static int rpc_pipefs_event(struct notifier_block *nb, unsigned long event,
        void *ptr)
 {
  struct super_block *sb = ptr;
  struct rpc_clnt *clnt;
  int error = 0;
- struct sunrpc_net *sn = net_generic(sb->s_fs_info, sunrpc_net_id);

- spin_lock(&sn->rpc_client_lock);
- list_for_each_entry(clnt, &sn->all_clients, cl_clients) {
+ while ((clnt = rpc_get_client_for_event(sb->s_fs_info, event))) {
   error = __rpc_pipefs_event(clnt, event, sb);
+  rpc_release_client(clnt);
   if (error)
    break;
  }
- spin_unlock(&sn->rpc_client_lock);
  return error;
 }
```