Subject: [PATCH v2 06/12] NFS: callback up - users counting cleanup
Posted by Stanislav Kinsbursky on Tue, 22 May 2012 07:36:31 GMT
View Forum Message <> Reply to Message

Usage coutner now increased only is the service was started sccessfully.
Even if service is running already, then goto is not required anymore, because
service creation and start will be skipped.
With this patch code looks clearer.

Signed-off-by: Stanislav Kinsbursky <skinsbursky@parallels.com>
---
 fs/nfs/callback.c | 22 ++++++++++------------
 1 files changed, 10 insertions(+), 12 deletions(-)

diff --git a/fs/nfs/callback.c b/fs/nfs/callback.c
index 8643ff7..11201dd 100644
--- a/fs/nfs/callback.c
+++ b/fs/nfs/callback.c
@@ -254,6 +254,9 @@ static int nfs_callback_start_svc(int minorversion, struct rpc_xprt *xprt,

  nfs_callback_bc_serv(minorversion, xprt, serv);

+ if (cb_info->task)
+ return 0;
+
  minorversion_setup = nfs_minorversion_callback_svc_setup(minorversion,
      serv, &rqstp, &callback_svc);
  if (!minorversion_setup) {
@@ -309,6 +312,8 @@ static int nfs_callback_up_net(int minorversion, struct svc_serv *serv,
struct n
 err_socks:
  svc_rpcb_cleanup(serv, net);
 err_bind:
+ dprintk("NFS: Couldn't create callback socket: err = %d; "
+   "net = %p\n", ret, net);
  return ret;
 }

@@ -353,7 +358,7 @@ int nfs_callback_up(u32 minorversion, struct rpc_xprt *xprt)
 {
  struct svc_serv *serv;
  struct nfs_callback_data *cb_info = &nfs_callback_info[minorversion];
- int ret = 0;
+ int ret;
  struct net *net = xprt->xprt_net;

  mutex_lock(&nfs_callback_mutex);
@@ -364,11 +369,6 @@ int nfs_callback_up(u32 minorversion, struct rpc_xprt *xprt)

```
   goto err_create;
  }

- if (cb_info->users++ || cb_info->task != NULL) {
-  nfs_callback_bc_serv(minorversion, xprt, serv);
-  goto out;
- }
-
  ret = nfs_callback_up_net(minorversion, serv, net);
  if (ret < 0)
   goto err_net;
@@ -377,13 +377,14 @@ int nfs_callback_up(u32 minorversion, struct rpc_xprt *xprt)
  if (ret < 0)
   goto err_start;

-out:
+ cb_info->users++;
  /*
   * svc_create creates the svc_serv with sv_nrthreads == 1, and then
   * svc_prepare_thread increments that. So we need to call svc_destroy
   * on both success and failure so that the refcount is 1 when the
   * thread exits.
   */
+err_net:
  svc_destroy(serv);
 err_create:
  mutex_unlock(&nfs_callback_mutex);
@@ -391,11 +392,8 @@ err_create:

 err_start:
  svc_shutdown_net(serv, net);
-err_net:
- dprintk("NFS: Couldn't create callback socket or server thread; "
-  "err = %d\n", ret);
- cb_info->users--;
- goto out;
+ dprintk("NFS: Couldn't create server thread; err = %d\n", ret);
+ goto err_net;
 }

 /*
```