
Subject: [PATCH 04/10] NFS: callback up - transport backchannel cleanup

Posted by [Stanislav Kinsbursky](#) on Thu, 26 Apr 2012 15:55:20 GMT

[View Forum Message](#) <> [Reply to Message](#)

No need to assign transports backchannel server explicitly in
nfs41_callback_up() - there is nfs_callback_bc_serv() function for this.
By using it, nfs4_callback_up() and nfs41_callback_up() can be called without
transport argument.

Note: service have to be passed to nfs_callback_bc_serv() instead of callback,
since callback link can be uninitialized.

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

fs/nfs/callback.c | 34 ++++++-----
1 files changed, 17 insertions(+), 17 deletions(-)

diff --git a/fs/nfs/callback.c b/fs/nfs/callback.c

index 9f5e790..92ce1eb 100644

--- a/fs/nfs/callback.c

+++ b/fs/nfs/callback.c

@@ -129,7 +129,7 @@ nfs4_callback_svc(void *vrqstp)

* Prepare to bring up the NFSv4 callback service

*/

static struct svc_rqst *

-nfs4_callback_up(struct svc_serv *serv, struct rpc_xprt *xprt)

+nfs4_callback_up(struct svc_serv *serv)

{

return svc_prepare_thread(serv, &serv->sv_pools[0], NUMA_NO_NODE);

}

@@ -185,16 +185,10 @@ nfs41_callback_svc(void *vrqstp)

* Bring up the NFSv4.1 callback service

*/

static struct svc_rqst *

-nfs41_callback_up(struct svc_serv *serv, struct rpc_xprt *xprt)

+nfs41_callback_up(struct svc_serv *serv)

{

struct svc_rqst *rqstp;

- /*

- * Save the svc_serv in the transport so that it can

- * be referenced when the session backchannel is initialized

- */

- xprt->bc_serv = serv;

-

INIT_LIST_HEAD(&serv->sv_cb_list);

spin_lock_init(&serv->sv_cb_lock);

```

    init_waitqueue_head(&serv->sv_cb_waitq);
@@ -209,21 +203,25 @@ nfs41_callback_up(struct svc_serv *serv, struct rpc_xprt *xprt)
}

static inline int nfs_minorversion_callback_svc_setup(u32 minorversion,
- struct svc_serv *serv, struct rpc_xprt *xprt,
+ struct svc_serv *serv,
    struct svc_rqst **rqstpp, int (**callback_svc)(void *vrqstp))
{
    if (minorversion) {
- *rqstpp = nfs41_callback_up(serv, xprt);
+ *rqstpp = nfs41_callback_up(serv);
        *callback_svc = nfs41_callback_svc;
    }
    return minorversion;
}

static inline void nfs_callback_bc_serv(u32 minorversion, struct rpc_xprt *xprt,
- struct nfs_callback_data *cb_info)
+ struct svc_serv *serv)
{
    if (minorversion)
- xprt->bc_serv = cb_info->serv;
+ /*
+  * Save the svc_serv in the transport so that it can
+  * be referenced when the session backchannel is initialized
+  */
+ xprt->bc_serv = serv;
}
#else
static int nfs41_callback_up_net(struct svc_serv *serv, struct net *net)
@@ -232,14 +230,14 @@ static int nfs41_callback_up_net(struct svc_serv *serv, struct net *net)
}

static inline int nfs_minorversion_callback_svc_setup(u32 minorversion,
- struct svc_serv *serv, struct rpc_xprt *xprt,
+ struct svc_serv *serv,
    struct svc_rqst **rqstpp, int (**callback_svc)(void *vrqstp))
{
    return 0;
}

static inline void nfs_callback_bc_serv(u32 minorversion, struct rpc_xprt *xprt,
- struct nfs_callback_data *cb_info)
+ struct svc_serv *serv)
{
}
#endif /* CONFIG_NFS_V4_1 */

```

```

@@ -331,7 +329,7 @@ int nfs_callback_up(u32 minorversion, struct rpc_xprt *xprt)
}

if (cb_info->users++ || cb_info->task != NULL) {
- nfs_callback_bc_serv(minorversion, xprt, cb_info);
+ nfs_callback_bc_serv(minorversion, xprt, serv);
goto out;
}

@@ -339,11 +337,13 @@ int nfs_callback_up(u32 minorversion, struct rpc_xprt *xprt)
if (ret < 0)
goto err_net;

+ nfs_callback_bc_serv(minorversion, xprt, serv);
+
minorversion_setup = nfs_minorversion_callback_svc_setup(minorversion,
- serv, xprt, &rqstp, &callback_svc);
+ serv, &rqstp, &callback_svc);
if (!minorversion_setup) {
/* v4.0 callback setup */
- rqstp = nfs4_callback_up(serv, xprt);
+ rqstp = nfs4_callback_up(serv);
callback_svc = nfs4_callback_svc;
}

```
