
Subject: [PATCH v4 03/10] NFS: move per-net callback thread initialization to
nfs_callback_up_net()

Posted by [Stanislav Kinsbursky](#) on Mon, 20 Aug 2012 14:00:16 GMT

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

v4:

1) Callback transport creation routine selection by version simplified.

This new function is now called before `nfs_minorversion_callback_svc_setup()`.

Also few small changes:

1) current network namespace in `nfs_callback_up()` was replaced by transport net.
2) `svc_shutdown_net()` was moved prior to callback usage counter decrement
(because in case of per-net data allocation failure `svc_shutdown_net()` have to
be skipped).

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

```
fs/nfs/callback.c | 133 ++++++-----  
fs/nfs/nfs4client.c | 2 -  
2 files changed, 87 insertions(+), 48 deletions(-)
```

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

```
index 18efeb5..a53b4e5 100644
```

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

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

```
@@ -39,6 +39,32 @@ static struct svc_program nfs4_callback_program;
```

```
unsigned short nfs_callback_tcpport6;
```

```
+static int nfs4_callback_up_net(struct svc_serv *serv, struct net *net)
```

```
+{
```

```
+ int ret;
```

```
+
```

```
+ ret = svc_create_xprt(serv, "tcp", net, PF_INET,
```

```
+ nfs_callback_set_tcpport, SVC_SOCKET_ANONYMOUS);
```

```
+ if (ret <= 0)
```

```
+ goto out_err;
```

```
+ nfs_callback_tcpport = ret;
```

```
+ dprintk("NFS: Callback listener port = %u (af %u, net %p)\n",
```

```
+ nfs_callback_tcpport, PF_INET, net);
```

```
+
```

```
+ ret = svc_create_xprt(serv, "tcp", net, PF_INET6,
```

```
+ nfs_callback_set_tcpport, SVC_SOCKET_ANONYMOUS);
```

```
+ if (ret > 0) {
```

```
+ nfs_callback_tcpport6 = ret;
```

```
+ dprintk("NFS: Callback listener port = %u (af %u, net %p)\n",
```

```
+ nfs_callback_tcpport6, PF_INET6, net);
```

```

+ } else if (ret != -EAFNOSUPPORT)
+ goto out_err;
+ return 0;
+
+out_err:
+ return (ret) ? ret : -ENOMEM;
+}
+
+/*
+ * This is the NFSv4 callback kernel thread.
+ */
@@ -80,36 +106,21 @@ nfs4_callback_svc(void *vrqstp)
static struct svc_rqst *
nfs4_callback_up(struct svc_serv *serv, struct rpc_xprt *xprt)
{
- int ret;
-
- ret = svc_create_xprt(serv, "tcp", &init_net, PF_INET,
- nfs_callback_set_tcpport, SVC_SOCKET_ANONYMOUS);
- if (ret <= 0)
- goto out_err;
- nfs_callback_tcpport = ret;
- dprintk("NFS: Callback listener port = %u (af %u)\n",
- nfs_callback_tcpport, PF_INET);
-
- ret = svc_create_xprt(serv, "tcp", &init_net, PF_INET6,
- nfs_callback_set_tcpport, SVC_SOCKET_ANONYMOUS);
- if (ret > 0) {
- nfs_callback_tcpport6 = ret;
- dprintk("NFS: Callback listener port = %u (af %u)\n",
- nfs_callback_tcpport6, PF_INET6);
- } else if (ret == -EAFNOSUPPORT)
- ret = 0;
- else
- goto out_err;
-
- return svc_prepare_thread(serv, &serv->sv_pools[0], NUMA_NO_NODE);
-
-out_err:
- if (ret == 0)
- ret = -ENOMEM;
- return ERR_PTR(ret);
}

#ifdef CONFIG_NFS_V4_1
+static int nfs41_callback_up_net(struct svc_serv *serv, struct net *net)
+{
+ /*

```

```

+ * Create an svc_sock for the back channel service that shares the
+ * fore channel connection.
+ * Returns the input port (0) and sets the svc_serv bc_xprt on success
+ */
+ return svc_create_xprt(serv, "tcp-bc", net, PF_INET, 0,
+     SVC_SOCKET_ANONYMOUS);
+}
+
+/*
+ * The callback service for NFSv4.1 callbacks
+ */
@@ -152,19 +163,6 @@ static struct svc_rqst *
nfs41_callback_up(struct svc_serv *serv, struct rpc_xprt *xprt)
{
    struct svc_rqst *rqstp;
- int ret;
-
- /*
- * Create an svc_sock for the back channel service that shares the
- * fore channel connection.
- * Returns the input port (0) and sets the svc_serv bc_xprt on success
- */
- ret = svc_create_xprt(serv, "tcp-bc", &init_net, PF_INET, 0,
-     SVC_SOCKET_ANONYMOUS);
- if (ret < 0) {
-     rqstp = ERR_PTR(ret);
-     goto out;
- }

    /*
     * Save the svc_serv in the transport so that it can
@@ -180,7 +178,6 @@ nfs41_callback_up(struct svc_serv *serv, struct rpc_xprt *xprt)
    svc_xprt_put(serv->sv_bc_xprt);
    serv->sv_bc_xprt = NULL;
}
-out:
    dprintk("--> %s return %ld\n", __func__,
        IS_ERR(rqstp) ? PTR_ERR(rqstp) : 0);
    return rqstp;
@@ -204,6 +201,11 @@ static inline void nfs_callback_bc_serv(u32 minorversion, struct rpc_xprt
*xprt,
    xprt->bc_serv = cb_info->serv;
}
#else
+static int nfs41_callback_up_net(struct svc_serv *serv, struct net *net)
+{
+ return 0;
+}

```

```

+
static inline int nfs_minorversion_callback_svc_setup(u32 minorversion,
    struct svc_serv *serv, struct rpc_xprt *xprt,
    struct svc_rqst **rqstpp, int (**callback_svc)(void *vrqstp))
@@ -217,6 +219,44 @@ static inline void nfs_callback_bc_serv(u32 minorversion, struct rpc_xprt
*xprt,
}
#endif /* CONFIG_NFS_V4_1 */

+static int nfs_callback_up_net(int minorversion, struct svc_serv *serv, struct net *net)
+{
+ int ret;
+
+ dprintk("NFS: create per-net callback data; net=%p\n", net);
+
+ ret = svc_bind(serv, net);
+ if (ret < 0) {
+ printk(KERN_WARNING "NFS: bind callback service failed\n");
+ goto err_bind;
+ }
+
+ switch (minorversion) {
+ case 0:
+ ret = nfs4_callback_up_net(serv, net);
+ break;
+ case 1:
+ ret = nfs41_callback_up_net(serv, net);
+ break;
+ default:
+ printk(KERN_ERR "NFS: unknown callback version: %d\n",
+ minorversion);
+ ret = -EINVAL;
+ break;
+ }
+
+ if (ret < 0) {
+ printk(KERN_ERR "NFS: callback service start failed\n");
+ goto err_socks;
+ }
+ return 0;
+
+err_socks:
+ svc_rpcb_cleanup(serv, net);
+err_bind:
+ return ret;
+}
+
static struct svc_serv *nfs_callback_create_svc(int minorversion)

```

```

{
    struct nfs_callback_data *cb_info = &nfs_callback_info[minorversion];
@@ -267,7 +307,7 @@ int nfs_callback_up(u32 minorversion, struct rpc_xprt *xprt)
    char svc_name[12];
    int ret = 0;
    int minorversion_setup;
- struct net *net = &init_net;
+ struct net *net = xprt->xprt_net;

    mutex_lock(&nfs_callback_mutex);

@@ -282,11 +322,9 @@ int nfs_callback_up(u32 minorversion, struct rpc_xprt *xprt)
    goto out;
}

- ret = svc_bind(serv, net);
- if (ret < 0) {
-     printk(KERN_WARNING "NFS: bind callback service failed\n");
-     goto out_err;
- }
+ ret = nfs_callback_up_net(minorversion, serv, net);
+ if (ret < 0)
+     goto err_net;

    minorversion_setup = nfs_minorversion_callback_svc_setup(minorversion,
        serv, xprt, &rqstp, &callback_svc);
@@ -326,10 +364,11 @@ err_create:
    mutex_unlock(&nfs_callback_mutex);
    return ret;
out_err:
+ svc_shutdown_net(serv, net);
+err_net:
    dprintk("NFS: Couldn't create callback socket or server thread; "
        "err = %d\n", ret);
    cb_info->users--;
- svc_shutdown_net(serv, net);
    goto out;
}

diff --git a/fs/nfs/nfs4client.c b/fs/nfs/nfs4client.c
index 1dfaa13..a89aebb 100644
--- a/fs/nfs/nfs4client.c
+++ b/fs/nfs/nfs4client.c
@@ -85,7 +85,7 @@ error:
static void nfs4_destroy_callback(struct nfs_client *clp)
{
    if (__test_and_clear_bit(NFS_CS_CALLBACK, &clp->cl_res_state))
-     nfs_callback_down(clp->cl_mvops->minor_version, &init_net);

```

```
+ nfs_callback_down(clp->cl_mvops->minor_version, clp->cl_net);  
}
```

```
static void nfs4_shutdown_client(struct nfs_client *clp)
```
