
Subject: Re: [PATCH v3 03/11] NFS: move per-net callback thread initialization to
nfs_callback_up_net()

Posted by [bfields](#) on Tue, 24 Jul 2012 21:47:31 GMT

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

On Tue, Jul 03, 2012 at 08:19:46PM +0400, Stanislav Kinsbursky wrote:

> 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 | 125 ++++++-----

> fs/nfs/client.c | 2 -

> 2 files changed, 79 insertions(+), 48 deletions(-)

>

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

> index f8d0c21..b61ed61 100644

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

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

> @@ -63,6 +63,32 @@ static struct kernel_param_ops param_ops_portnr = {

>

> module_param_named(callback_tcpport, nfs_callback_set_tcpport, portnr, 0644);

>

> +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.
>  */
> @@ -104,36 +130,21 @@ 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);
> }
>
> #if defined(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
>  */
> @@ -176,19 +187,6 @@ 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
> @@ -204,7 +202,6 @@ 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;
> @@ -228,6 +225,11 @@ 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))
> @@ -241,6 +243,36 @@ 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;
> + }
> +
> + if (minorversion) {
> + ret = nfs41_callback_up_net(serv, net);
> + if (ret < 0)
> + goto err_socks;
> + }
> +
> + if (ret == 0)
> + ret = nfs4_callback_up_net(serv, net);

```

So in the minorversion 1 case, and in the absence of errors, you set up both the 4.1 and 4.0 callback xprts?

That doesn't look right. You should need only one or the other.

--b.

```

> + if (ret < 0)
> + 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];

```

```

> @@ -287,7 +319,7 @@ 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);
>
> @@ -302,11 +334,9 @@ 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);
> @@ -346,10 +376,11 @@ 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/client.c b/fs/nfs/client.c
> index 28bc770..d8c918b 100644
> --- a/fs/nfs/client.c
> +++ b/fs/nfs/client.c
> @@ -225,7 +225,7 @@ static void nfs4_shutdown_session(struct nfs_client *clp)
> 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)  
>
```
