
Subject: Re: [PATCH v2 1/4] SUNRPC: release per-net clients lock before calling PipeFS dentries creation

Posted by [Myklebust, Trond](#) on Mon, 27 Feb 2012 17:37:13 GMT

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

On Mon, 2012-02-27 at 20:55 +0400, Stanislav Kinsbursky wrote:

> > On Mon, 2012-02-27 at 19:50 +0400, Stanislav Kinsbursky wrote:

> >> 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 | 10 ++++++---

> >> 1 files changed, 7 insertions(+), 3 deletions(-)

> >>

> >> diff --git a/net/sunrpc/clnt.c b/net/sunrpc/clnt.c

> >> index bb7ed2f3..ddb5741 100644

> >> --- a/net/sunrpc/clnt.c

> >> +++ b/net/sunrpc/clnt.c

> >> @@ -84,7 +84,7 @@ static void rpc_register_client(struct rpc_clnt *clnt)

> >> struct sunrpc_net *sn = net_generic(clnt->cl_xprt->xprt_net, sunrpc_net_id);

> >>

> >> spin_lock(&sn->rpc_client_lock);

> >> - list_add(&clnt->cl_clients,&sn->all_clients);

> >> + list_add_tail(&clnt->cl_clients,&sn->all_clients);

> >> spin_unlock(&sn->rpc_client_lock);

> >> }

> >>

> >> @@ -208,15 +208,19 @@ static int rpc_pipefs_event(struct notifier_block *nb, unsigned long event,

> >> void *ptr)

> >> {

> >> struct super_block *sb = ptr;

> >> - struct rpc_clnt *clnt;

> >> + struct rpc_clnt *clnt, *tmp;

> >> 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) {

> >> + list_for_each_entry_safe(clnt, tmp,&sn->all_clients, cl_clients) {

> >> + atomic_inc(&clnt->cl_count);

> >> + spin_unlock(&sn->rpc_client_lock);

> >> error = __rpc_pipefs_event(clnt, event, sb);

> >> + rpc_release_client(clnt);

```

>>> if (error)
>>> break;
>>> + spin_lock(&sn->rpc_client_lock);
>>> }
>>> spin_unlock(&sn->rpc_client_lock);
>>> return error;
>>>
>>
>> This won't be safe. Nothing guarantees that 'tmp' remains valid after
>> you drop the spin_lock.
>>
>> I think you rather need to add a check for whether clnt->cl_dentry is in
>> the right state (NULL if RPC_PIPEFS_UMOUNT or non-NULL if
>> RPC_PIPEFS_MOUNT) before deciding whether or not to atomic_inc() and
>> drop the lock, so that you can restart the loop after calling
>> __rpc_pipefs_event().
>>
>
> Gmmm.
> Please, correct me, if I'm wrong, that you are proposing something like this:
>
> spin_lock(&sn->rpc_client_lock);
> again:
> 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);
> error = __rpc_pipefs_event(clnt, event, sb);
> rpc_release_client(clnt);
> spin_lock(&sn->rpc_client_lock);
> if (error)
> break;
> goto again;
> }
> spin_unlock(&sn->rpc_client_lock);

```

Something along those lines, yes... Alternatively, you could do as David proposes, and grab a reference to the next rpc_client before calling rpc_release_client. Either works for me...

--

Trond Myklebust
Linux NFS client maintainer

NetApp
Trond.Myklebust@netapp.com

