Subject: Re: [PATCH 03/16] net: Basic network namespace infrastructure.
Posted by ebiederm on Mon, 10 Sep 2007 06:32:43 GMT
View Forum Message <> Reply to Message

"Paul E. McKenney" <paulmck@linux.vnet.ibm.com> writes:

>> I know I cannot use get_net for the reference in in /proc because
>> otherwise I could not release the network namespace unless I was to
>> unmount the filesystem, which is not a desirable property.
>>
>> I think I can change the idiom to:
>>
>> struct net *maybe_get_net(struct net *net)
>> {
>>   if (!atomic_inc_not_zero(&net->count))
>>   net = NULL;
>> return net;
>> }
>>
>> Which would make dev_seq_open be:
>>
>> static int dev_seq_open(struct inode *inode, struct file *file)
>> {
>> struct seq_file *seq;
>> int res;
>> res = seq_open(file, &dev_seq_ops);
>> if (!res) {
>>   seq = file->private_data;
>>   seq->private = maybe_get_net(PROC_NET(inode));
>>   if (!seq->private) {
>>   res = -ENOENT;
>>   seq_release(inode, file);
>>   }
>> }
>> return res;
>> }
>>
>> I'm still asking myself if I need any kind of locking to ensure
>> struct net does not go away in the mean time, if so rcu_read_lock()
>> should be sufficient.
>
> Agreed -- and it might be possible to leverage the existing locking
> in the /proc code.

Yes. The generic /proc code takes care of this. It appears
to ensure that any ongoing operations will be waited for and
no more operations will be started once remove_proc_entry
is called. So I just need the maybe_get_net thing to have

safe ref counting.

That is what I thought but I figured I would review that part while
I was looking at everything.

Eric

_____
Containers mailing list
Containers@lists.linux-foundation.org
https://lists.linux-foundation.org/mailman/listinfo/containers