
Subject: Re: [PATCH 3/4] SUNRPC: service destruction in network namespace context

Posted by [bfields](#) on Fri, 27 Jan 2012 14:33:35 GMT

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

On Fri, Jan 27, 2012 at 01:08:06PM +0400, Stanislav Kinsbursky wrote:

```
> >On Wed, Jan 25, 2012 at 05:47:26PM +0400, Stanislav Kinsbursky wrote:
> >>This patch introduces network namespace filter for service destruction
> >>function.
> >>Nothing special here - just do exactly the same operations, but only for
> >>transports in passed networks namespace context.
> >>BTW, BUG_ON() checks for empty service transports lists were returned into
> >>svc_destroy() function. This is because of swithing generic svc_close_all() to
> >>networks namespace dependable svc_close_net().
> >>
> >>Signed-off-by: Stanislav Kinsbursky<skinsbursky@parallels.com>
> >>
> >>---
> >> include/linux/sunrpc/svcsock.h | 2 +-
> >> net/sunrpc/svc.c | 9 ++++++---
> >> net/sunrpc/svc_xprt.c | 27 ++++++++-----
> >> 3 files changed, 25 insertions(+), 13 deletions(-)
> >>
> >>diff --git a/include/linux/sunrpc/svcsock.h b/include/linux/sunrpc/svcsock.h
> >>index c84e974..cb4ac69 100644
> >>--- a/include/linux/sunrpc/svcsock.h
> >>+++ b/include/linux/sunrpc/svcsock.h
> >>@@ -34,7 +34,7 @@ struct svc_sock {
> >> /*
> >>  * Function prototypes.
> >>  */
> >>-void svc_close_all(struct svc_serv *);
> >>+void svc_close_net(struct svc_serv *, struct net *);
> >> int svc_recv(struct svc_rqst *, long);
> >> int svc_send(struct svc_rqst *);
> >> void svc_drop(struct svc_rqst *);
> >>diff --git a/net/sunrpc/svc.c b/net/sunrpc/svc.c
> >>index a8b49a0..279bbd8 100644
> >>--- a/net/sunrpc/svc.c
> >>+++ b/net/sunrpc/svc.c
> >>@@ -517,6 +517,8 @@ EXPORT_SYMBOL_GPL(svc_create_pooled);
> >> void
> >> svc_destroy(struct svc_serv *serv)
> >> {
> >>+ struct net *net = current->nsproxy->net_ns;
> >>+
> >> dprintk("svc: svc_destroy(%s, %d)\n",
```

```

> >> serv->sv_program->pg_name,
> >> serv->sv_nthreads);
> >>@@ -539,10 +541,13 @@ svc_destroy(struct svc_serv *serv)
> >> * caller is using--nfsd_mutex in the case of nfsd). So it's
> >> * safe to traverse those lists and shut everything down:
> >> */
> >>- svc_close_all(serv);
> >>+ svc_close_net(serv, net);
> >>+
> >>+ BUG_ON(!list_empty(&serv->sv_permsocks));
> >>+ BUG_ON(!list_empty(&serv->sv_tempsocks));
> >
> >I'm confused--what guarantees this is true, at this point?
> >
>
> Hi, Bruce.
> I'm confused with your question. IOW, this must be true, because
> this code is executed only in case of last service thread is
> exiting, doesn't it?
>
> >There are two ways I could imagine containerizing svc_serv: either we
> >create a new one for each namespace, or we share a single global one
> >between them.
> >
>
> This is done for the second one.
>
> >If the former, then something that takes a "serv" argument shouldn't
> >also need a "net" argument--the serv should already know which namespace
> >it belongs to.
> >
> >If the latter, then these lists could have sockets from multiple
> >namespaces, and they aren't guaranteed to be empty here.
> >
> >?
> >
>
> I'll explain it on Lockd example (this code is done already - I just
> haven't sent it yet).
> Lockd is still only one thread and can handle lock requests from
> different network namespaces:
> 1) Introduced per-net lockd users counter and resources.
> 2) nlmsvc_users counter become global one. I.e. it's equal to sum of
> all per-net lockd users counters.
> 3) For each lockd_up() call global and current net lockd users
> counters are increased by one.
> 3) On lockd_up() call: if nlmsvc_users if equal to 0, then lockd thread is started.
> 4) On lockd_up() call: if current network context lockd users

```

> counter equal to 0, then resources for Lockd service are allocated
> in current network context.
> 5) On lockd_down() call: if current network context lockd users
> counter equal to 0, then resources for Lockd service are released in
> current network context (svc_shutdown_net() introduced in this
> series).
> 6) On lockd_down() call: if nlmsvc_users if equal to 0, then lockd
> thread is stopped and svc_destroy is called. And herewe can expect,
> that no service transports left.

OK, so at this point svc_close_net(serv, current->nsproxy->net_ns) is
enough to clear out sv_permsocks and sv_temp_socks because we know that
the only sockets left are in that namespace. Got it.

> I've just realized, that probably it's possible to implement some
> more generic helpers in SUNRPC code to make the code looks clearer.
> I would appreciate for any advices how to do so.

I'm not sure. The one thing that might have helped mere here would be
a comment to explain what's going on, maybe something like:

```
svc_close_net(serv, net);  
+ /*  
+  * The last user is gone, so the only sockets left belonged its  
+  * network namespace:  
+  */  
BUG_ON(!list_empty(&serv->sv_permsocks));  
BUG_ON(!list_empty(&serv->sv_tempsocks));
```

--b.
