Subject: Re: [PATCH 3/7] uts namespaces: use init_utsname when appropriate
Posted by ebiederm on Sun, 09 Apr 2006 09:44:19 GMT
View Forum Message <> Reply to Message

"Serge E. Hallyn" <serue@us.ibm.com> writes:

>> This also probably makes sense as utsname(). It doesn't
>> really matter as this is before init is executed. But logically
>> this is a user space or per namespace action.
>
> Right, I was kind of favoring using init_utsname() for anything
> __init. But utsname() will of course work just as well there.

Basically anything that should move to klibc I favor using
utsname() for. That tends to make it clear it follows
the usual user space rules.

With a little luck HPA might actually have this code deleted
in -mm before we get to far.

>> > diff --git a/net/sunrpc/clnt.c b/net/sunrpc/clnt.c
>> > index aa8965e..97c8439 100644
>> > --- a/net/sunrpc/clnt.c
>> > +++ b/net/sunrpc/clnt.c
>> > @@ -176,10 +176,10 @@ rpc_new_client(struct rpc_xprt *xprt, ch
>> > }
>> >
>> > /* save the nodename */
>> > - clnt->cl_nodelen = strlen(system_utsname.nodename);
>> > + clnt->cl_nodelen = strlen(init_utsname()->nodename);
>> > if (clnt->cl_nodelen > UNX_MAXNODENAME)
>> > clnt->cl_nodelen = UNX_MAXNODENAME;
>> > - memcpy(clnt->cl_nodename, system_utsname.nodename, clnt->cl_nodelen);
>> > + memcpy(clnt->cl_nodename, init_utsname()->nodename, clnt->cl_nodelen);
>> > return clnt;
>> >
>> > out_no_auth:
>>
>> Using nodename is practically the definition of something
>> that should per namespace I think. Plus it would be really inconsistent
>> to use utsname() and the init_utsname for the nfs rpc calls.
>>
>> Unless I am missing something.
>
> It seemed like this would be happening in any old context, so that
> current->uts_ns could be any process'. Tracing it back further,
> it seems like nfs+lockd should have the context available. So I'll
> switch this as well.

I have not traced that path recently. So I don't remember.
This is one of those odd cases that makes a real difference.

This reminds me of another piece of the conversation.
kernel_thread vs. kthread, and the oddities of daemonize.

In general user space cannot kill kernel threads, so having
a kernel thread inside a namespace is dangerous because it
means the namespace can never exit.

There are two ways to avoid the associated problems.
- modify daemonize to always use the instance of that
  namespace associated with init_task.
- modify all interesting kernel threads to use the
  kthread api instead of kernel_thread. Using kthread
  makes the kernel threads children of keventd and always
  in the initial namespace instance. As such we know
  we aren't inside of any user space namespace instance.

Eric
