Subject: Re: [PATCH 1/12] L2 network namespace: current network namespace operations
Posted by ebiederm on Sat, 09 Dec 2006 07:33:14 GMT
View Forum Message <> Reply to Message

Herbert Poetzl <herbert@13thfloor.at> writes:

> On Fri, Dec 08, 2006 at 01:03:29PM -0700, Eric W. Biederman wrote:

> it should not be necessary to do that, and IMHO
> changing the namespace temporarily is not such
> a good idea, as that might cause all kinds of
> ugly races, when other parts of the OS (from
> other CPUs) access process relevant information
> (utilizing the namespaces)

Consider a dedicated variable not current->nsproxy.

>> Assuming we are doing it then we should do it for every path both
>> socket and network device and do the lookup once and the cache it
>> globally in the current execution context.
>>
>> We should not change current->nsproxy. I don't think for packet
>> processing we need to change every namespace do we? The uid namespace
>> and the like should be irrelevant correct?
>
> hmm, wouldn't it be better to pass the relevant
> information (network context) within the network
> stack where needed, instead of changing the
> network assignment of 'current' for processing
> network packets?

Changing current is clearly a no-go but I'm not convinced it rules
out a global variable. Having a global variable seems to solve the
incremental merge and regression testing problems. I see a lot of
value in that proposition.

> I remeber from a prototype Linux-VServer implementation
> that this wasn't that complicated to do ...

Ok. Here are is the situation as a I see it. Just a little more
explicitly.

- Changing any of the normal namespace pointers is wrong.
- We need to lookup the network namespace at some point during
  packet processing.
- We want to do performance testing, with and without our changes.
- The network stack is big, constantly moving target with that has a

nontrivial number of global variables.
- We want a minimal disruption of the network stack.

Therefore I believe it makes sense to access network stack global variables as:

```
__get_net_var(some_variable_name);
```

Because it is trivial to compile the change out, because it imposes no apparent inefficiencies and from the per cpu code we already know exactly how to support variables of the above form.

The definition and declarations would be of the form:
```
DEFINE_PER_CPU(type, name) = ?;
DECLARE_PER_CPU(type, name);
```

To support that there needs to be a consistent way to get the appropriate network namespace.   The easiest way I can imagine to do that is to have a global variable either per task or per cpu so it doesn't need locking that caches the current L2 network namespace.

This variable very much should not be current->nsproxy. But something dedicated to the network code.

If we didn't have the constraint of needing to compile the changes out. I would think this is a rather silly form.

But the above form keeps it explicit when you reference a variable that is in the network namespace, it allows for architecture specific optimizations and allows us to do a head to head comparison with the existing network code.

In addition it keeps the perturbations of the network stack to an extreme minimum.

So long as the assertion holds that we rarely need to modify which network namespace we are working this sounds like a reasonable solution.

OpenVZ has been doing roughly this for a while so I don't expect we will find it a great difficulty keeping the variable appropriately updated.

Eric