
Subject: [PATCH 1/12] L2 network namespace: current network namespace operations

Posted by [Mishin Dmitry](#) on Wed, 06 Dec 2006 22:24:16 GMT

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

Added functions and macros required to operate with current network namespaces. They are required in order to switch network namespace for incoming packets and to not extend current network interface by additional network namespace argue.

Signed-off-by: Dmitry Mishin <dim@openvz.org>

include/linux/net_namespace.h | 41 ++++++
kernel/nsproxy.c | 1 +
net/core/net_namespace.c | 2 ++
3 files changed, 42 insertions(+), 2 deletions(-)

--- linux-2.6.19-rc6-mm2.orig/include/linux/net_namespace.h

+++ linux-2.6.19-rc6-mm2/include/linux/net_namespace.h

@ @ -7,6 +7,7 @ @

```
struct net_namespace {  
    struct kref kref;  
+ struct nsproxy *ns;  
};
```

```
extern struct net_namespace init_net_ns;  
@@ -32,7 +33,27 @@ static inline void put_net_ns(struct net  
    kref_put(&ns->kref, free_net_ns);  
}
```

-#else

+#define current_net_ns (current->nsproxy->net_ns)

+

+static inline struct net_namespace *push_net_ns(struct net_namespace *to)

+{

+ struct nsproxy *orig;

+

+ orig = current->nsproxy;

+ current->nsproxy = to->ns;

+ return orig->net_ns;

+}

+

+static inline void pop_net_ns(struct net_namespace *orig)

+{

+ current->nsproxy = orig->ns;

+}

+

```

+#define net_ns_match(target, ns) ((target) == (ns))
+
+#define net_ns_hash(ns) ((ns)->hash)
+
+#else /* CONFIG_NET_NS */
+
+#define INIT_NET_NS(net_ns)
+
@@ -57,6 +78,22 @@ static inline int copy_net_ns(int flags,
static inline void put_net_ns(struct net_namespace *ns)
{
}
-#endif
+
+#define current_net_ns NULL
+
+static inline struct net_namespace *push_net_ns(struct net_namespace *ns)
+{
+ return NULL;
+}
+
+static inline void pop_net_ns(struct net_namespace *ns)
+{
+}
+
+#define net_ns_match(target, ns) ((void)(ns), 1)
+
+#define net_ns_hash(ns) (0)
+
+#endif /* !CONFIG_NET_NS */
+
+#endif /* _LINUX_NET_NAMESPACE_H */
--- linux-2.6.19-rc6-mm2.orig/kernel/nsproxy.c
+++ linux-2.6.19-rc6-mm2/kernel/nsproxy.c
@@ -534,6 +534,7 @@ asmlinkage long sys_unshare_ns(unsigned
}

if (new_net) {
+ new_net->ns = current->nsproxy;
net = current->nsproxy->net_ns;
current->nsproxy->net_ns = new_net;
new_net = net;
--- linux-2.6.19-rc6-mm2.orig/net/core/net_namespace.c
+++ linux-2.6.19-rc6-mm2/net/core/net_namespace.c
@@ -14,6 +14,7 @@ struct net_namespace init_net_ns = {
.kref = {
.refcount = ATOMIC_INIT(2),
},

```

```

+ .ns = &init_nsproxy,
};

#ifdef CONFIG_NET_NS
@@ -32,6 +33,7 @@ static struct net_namespace *clone_net_n
return NULL;

kref_init(&ns->kref);
+ ns->ns = old_ns->ns;
return ns;
}

```

Containers mailing list
Containers@lists.osdl.org
<https://lists.osdl.org/mailman/listinfo/containers>

Subject: Re: [PATCH 1/12] L2 network namespace: current network namespace operations

Posted by [Cedric Le Goater](#) on Thu, 07 Dec 2006 09:37:18 GMT

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

Dmitry Mishin wrote:

```

> Added functions and macros required to operate with current network namespaces.
> They are required in order to switch network namespace for incoming packets and
> to not extend current network interface by addtional network namespace argue.
>
> Signed-off-by: Dmitry Mishin <dim@openvz.org>
>
> ---
> include/linux/net_namespace.h | 41 ++++++
> kernel/nsproxy.c              | 1 +
> net/core/net_namespace.c      | 2 ++
> 3 files changed, 42 insertions(+), 2 deletions(-)
>
> --- linux-2.6.19-rc6-mm2.orig/include/linux/net_namespace.h
> +++ linux-2.6.19-rc6-mm2/include/linux/net_namespace.h
> @@ -7,6 +7,7 @@
>
> struct net_namespace {
>     struct kref kref;
> + struct nsproxy *ns;
> };
>

```

why do you need that back pointer ?

(The answer must be in the following patches but I'm being lazy and asking the author :)

thanks,

C.

Containers mailing list
Containers@lists.osdl.org
<https://lists.osdl.org/mailman/listinfo/containers>

Subject: Re: [PATCH 1/12] L2 network namespace: current network namespace operations

Posted by [Mishin Dmitry](#) on Thu, 07 Dec 2006 11:23:20 GMT

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

On Thursday 07 December 2006 12:37, Cedric Le Goater wrote:

> Dmitry Mishin wrote:

> > Added functions and macros required to operate with current network namespaces.

> > They are required in order to switch network namespace for incoming packets and

> > to not extend current network interface by addtional network namespace argue.

> >

> > Signed-off-by: Dmitry Mishin <dim@openvz.org>

> >

> > ---

> > include/linux/net_namespace.h | 41 ++++++

> > kernel/nsproxy.c | 1 +

> > net/core/net_namespace.c | 2 ++

> > 3 files changed, 42 insertions(+), 2 deletions(-)

> >

> > --- linux-2.6.19-rc6-mm2.orig/include/linux/net_namespace.h

> > +++ linux-2.6.19-rc6-mm2/include/linux/net_namespace.h

> > @@ -7,6 +7,7 @@

> >

> > struct net_namespace {

> > struct kref kref;

> > + struct nsproxy *ns;

> > };

> >

>

> why do you need that back pointer ?

>

> (The answer must be in the following patches but I'm being lazy and

> asking the author :)

Because for the incoming packets, I need to switch networking namespace per-task and not per-nsproxy. If I switch it just in current->nsproxy, it means that all tasks shring this nsproxy will switch network context.

It is one of the reasons, why we need per-task exec_context pointer.

--

Thanks,
Dmitry.

Containers mailing list
Containers@lists.osdl.org
<https://lists.osdl.org/mailman/listinfo/containers>

Subject: Re: [PATCH 1/12] L2 network namespace: current network namespace operations

Posted by [ebiederm](#) on Fri, 08 Dec 2006 20:03:29 GMT

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

Dmitry Mishin <dim@openvz.org> writes:

```
> On Thursday 07 December 2006 12:37, Cedric Le Goater wrote:
>> Dmitry Mishin wrote:
>> > Added functions and macros required to operate with current network
> namespaces.
>> > They are required in order to switch network namespace for incoming packets
> and
>> > to not extend current network interface by addtional network namespace
> argue.
>> >
>> > Signed-off-by: Dmitry Mishin <dim@openvz.org>
>> >
>> > ---
>> > include/linux/net_namespace.h | 41 ++++++
>> > kernel/nsproxy.c             | 1 +
>> > net/core/net_namespace.c     | 2 ++
>> > 3 files changed, 42 insertions(+), 2 deletions(-)
>> >
>> > --- linux-2.6.19-rc6-mm2.orig/include/linux/net_namespace.h
>> > +++ linux-2.6.19-rc6-mm2/include/linux/net_namespace.h
>> > @@ -7,6 +7,7 @@
>> >
>> > struct net_namespace {
>> >     struct kref kref;
>> > + struct nsproxy *ns;
>> > };
>> >
>> >
>> why do you need that back pointer ?
```

>>
>> (The answer must be in the following patches but I'm being lazy and
>> asking the author :)
> Because for the incoming packets, I need to switch networking namespace
> per-task and not per-nsproxy. If I switch it just in current->nsproxy, it means
> that all tasks sharing this nsproxy will switch network context.
>
> It is one of the reasons, why we need per-task exec_context pointer.

Ugh.

If necessary push_net_ns and pop_net_ns should be implemented like get_fs
and set_fs. Using a sane variable in thread_info, or per cpu.

I'm not at all certain I'm comfortable doing this in interrupt context.

Assuming we are doing it then we should do it for every path both socket
and network device and do the lookup once and then 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?

Eric

Containers mailing list
Containers@lists.osdl.org
<https://lists.osdl.org/mailman/listinfo/containers>

Subject: Re: [PATCH 1/12] L2 network namespace: current network namespace
operations

Posted by [ebiederm](#) on Fri, 08 Dec 2006 20:50:51 GMT

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

Dmitry Mishin <dim@openvz.org> writes:

> Added functions and macros required to operate with current network namespaces.
> They are required in order to switch network namespace for incoming packets and
> to not extend current network interface by additional network namespace arg.
>
> Signed-off-by: Dmitry Mishin <dim@openvz.org>
>
> -#else
> +#define current_net_ns (current->nsproxy->net_ns)
> +#else /* CONFIG_NET_NS */
>

```
> #define INIT_NET_NS(net_ns)
>
> @@ -57,6 +78,22 @@ static inline int copy_net_ns(int flags,
> static inline void put_net_ns(struct net_namespace *ns)
> {
> }
> -#endif
> +
> +#define current_net_ns NULL
>
> +#endif /* !CONFIG_NET_NS */
```

Ouch! NULL is not a good default.

Can we please pick an idiom for referencing global network stack variables that works if we are compiled in or not. At least if we are going to offer the option.

That way we can merge the changes for looking up all of the globals before merging the network namespace support.

Doing it this way seems to imply we will need context support to implement this.

My initial suggestion is to base the work on the per cpu variable support.

Using `__get_net_var(variable)`. To reference the global variable. And the variables marked as `__per_net` in their declaration so we know the variables are per network namespace.

This allows us to handle ipv6 and other modules that only have their variables present when they are loaded the same way per cpu variables are treated. And it ensures that the form used when everything is

Eric

Containers mailing list
Containers@lists.osdl.org
<https://lists.osdl.org/mailman/listinfo/containers>

Subject: Re: [PATCH 1/12] L2 network namespace: current network namespace operations

Posted by [Herbert Poetzl](#) on Sat, 09 Dec 2006 04:24:10 GMT

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

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

> Dmitry Mishin <dim@openvz.org> writes:

>

> > On Thursday 07 December 2006 12:37, Cedric Le Goater wrote:

> >> Dmitry Mishin wrote:

> >> > Added functions and macros required to operate with current network namespaces.

> >> > They are required in order to switch network namespace for incoming packets

> > and

> >> > to not extend current network interface by additional network namespace

> > argue.

> >> >

> >> > Signed-off-by: Dmitry Mishin <dim@openvz.org>

> >> >

> >> > ---

> >> > include/linux/net_namespace.h | 41 ++++++

> >> > kernel/nsproxy.c | 1 +

> >> > net/core/net_namespace.c | 2 ++

> >> > 3 files changed, 42 insertions(+), 2 deletions(-)

> >> >

> >> > --- linux-2.6.19-rc6-mm2.orig/include/linux/net_namespace.h

> >> > +++ linux-2.6.19-rc6-mm2/include/linux/net_namespace.h

> >> > @@ -7,6 +7,7 @@

> >> >

> >> > struct net_namespace {

> >> > struct kref kref;

> >> > + struct nsproxy *ns;

> >> > };

> >> >

> >>

> >> why do you need that back pointer ?

> >>

> >> (The answer must be in the following patches but I'm being lazy and

> >> asking the author :)

> > Because for the incoming packets, I need to switch networking

> > namespace per-task and not per-nsproxy. If I switch it just in

> > current->nsproxy, it means that all tasks shring this nsproxy will

> > switch network context.

> >

> > It is one of the reasons, why we need per-task exec_context pointer.

>

> Ugh.

>

> If necessary push_net_ns and pop_net_ns should be implemented like

> get_fs and set_fs. Using a sane variable in thread_info, or per cpu.

>

> I'm not at all certain I'm comfortable doing this in interrupt

> context.

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)

- > 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?

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

best,
Herbert

- > Eric
- > _____
- > Containers mailing list
- > Containers@lists.osdl.org
- > <https://lists.osdl.org/mailman/listinfo/containers>

Containers mailing list
Containers@lists.osdl.org
<https://lists.osdl.org/mailman/listinfo/containers>

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

Containers mailing list
Containers@lists.osdl.org
<https://lists.osdl.org/mailman/listinfo/containers>

Subject: Re: [PATCH 1/12] L2 network namespace: current network namespace operations

Posted by [Mishin Dmitry](#) on Thu, 21 Dec 2006 13:26:37 GMT

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

On Friday 08 December 2006 23:50, Eric W. Biederman wrote:

> Dmitry Mishin <dim@openvz.org> writes:

>

> > Added functions and macros required to operate with current network namespaces.
> > They are required in order to switch network namespace for incoming packets and
> > to not extend current network interface by additional network namespace argue.

> >

> > Signed-off-by: Dmitry Mishin <dim@openvz.org>

> >

> > -#else

> > +#define current_net_ns (current->nsproxy->net_ns)

> > +#else /* CONFIG_NET_NS */

> >

> > #define INIT_NET_NS(net_ns)

> >

> > @@ -57,6 +78,22 @@ static inline int copy_net_ns(int flags,

> > static inline void put_net_ns(struct net_namespace *ns)

> > {

> > }

> > -#endif

> > +

> > +#define current_net_ns NULL

>

> > +#endif /* !CONFIG_NET_NS */

>

> Ouch! NULL is not a good default.

>

> Can we please pick an idiom for referencing global network stack
> variables that works if we are compiled in or not. At least if
> we are going to offer the option.

>

> That way we can merge the changes for looking up all of the globals
> before merging the network namespace support.

>

> Doing it this way seems to imply we will need context support to
> implement this.

>

> My initial suggestion is to base the work on the per cpu variable
> support.

>

> Using __get_net_var(variable). To reference the global variable.

> And the variables marked as __per_net in their declaration so

> we know the variables are per network namespace.

>

> This allows us to handle ipv6 and other modules that only have their
> variables present when they are loaded the same way per cpu variables
> are treated. And it ensures that the form used when everything is

Eric,

please, clarify, what you mean. For example, what we have to do with dev_base,
dev_tail variables?

>
> Eric
>

--
Thanks,
Dmitry.

Containers mailing list
Containers@lists.osdl.org
<https://lists.osdl.org/mailman/listinfo/containers>
