

---

Subject: [PATCH 4/4] net: Implement the per network namespace sysctl infrastructure

Posted by [ebiederm](#) on Thu, 29 Nov 2007 17:53:29 GMT

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

---

The user interface is: register\_net\_sysctl\_table and unregister\_net\_sysctl\_table. Very much like the current interface except there is a network namespace parameter.

With this any sysctl registered with register\_net\_sysctl\_table will only show up to tasks in the same network namespace.

All other sysctls continue to be globally visible.

Signed-off-by: Eric W. Biederman <[ebiederm@xmission.com](mailto:ebiederm@xmission.com)>

---

```
include/net/net_namespace.h | 9 +++++++
net/sysctl_net.c            | 57 ++++++++++++++++++++++++++++++++++++++
2 files changed, 66 insertions(+), 0 deletions(-)
```

```
diff --git a/include/net/net_namespace.h b/include/net/net_namespace.h
```

```
index 4d0d634..235214c 100644
```

```
--- a/include/net/net_namespace.h
```

```
+++ b/include/net/net_namespace.h
```

```
@@ -25,6 +25,8 @@ struct net {
    struct proc_dir_entry *proc_net_stat;
    struct proc_dir_entry *proc_net_root;
```

```
+ struct list_head sysctl_table_headers;
```

```
+
```

```
    struct net_device    *loopback_dev;    /* The loopback */
```

```
    struct list_head dev_base_head;
```

```
@@ -144,4 +146,11 @@ extern void unregister_pernet_subsys(struct pernet_operations *);
extern int register_pernet_device(struct pernet_operations *);
extern void unregister_pernet_device(struct pernet_operations *);
```

```
+struct ctl_path;
```

```
+struct ctl_table;
```

```
+struct ctl_table_header;
```

```
+extern struct ctl_table_header *register_net_sysctl_table(struct net *net,
```

```
+ const struct ctl_path *path, struct ctl_table *table);
```

```
+extern void unregister_net_sysctl_table(struct ctl_table_header *header);
```

```
+
```

```
#endif /* __NET_NET_NAMESPACE_H */
```

```
diff --git a/net/sysctl_net.c b/net/sysctl_net.c
```

```
index cd4eafb..c50c793 100644
```

```
--- a/net/sysctl_net.c
```

```

+++ b/net/sysctl_net.c
@@ -14,6 +14,7 @@

#include <linux/mm.h>
#include <linux/sysctl.h>
#include <linux/nsproxy.h>

#include <net/sock.h>

@@ -54,3 +55,59 @@ struct ctl_table net_table[] = {
    #endif
    { 0 },
};
+
+static struct list_head *
+net_ctl_header_lookup(struct ctl_table_root *root, struct nsproxy *namespaces)
+{
+    return &namespaces->net_ns->sysctl_table_headers;
+}
+
+static struct ctl_table_root net_sysctl_root = {
+    .lookup = net_ctl_header_lookup,
+};
+
+static int sysctl_net_init(struct net *net)
+{
+    INIT_LIST_HEAD(&net->sysctl_table_headers);
+    return 0;
+}
+
+static void sysctl_net_exit(struct net *net)
+{
+    WARN_ON(!list_empty(&net->sysctl_table_headers));
+    return;
+}
+
+static struct pernet_operations sysctl_pernet_ops = {
+    .init = sysctl_net_init,
+    .exit = sysctl_net_exit,
+};
+
+static __init int sysctl_init(void)
+{
+    int ret;
+    ret = register_pernet_subsys(&sysctl_pernet_ops);
+    if (ret)
+        goto out;
+    register_sysctl_root(&net_sysctl_root);

```

```

+out:
+ return ret;
+}
+subsys_initcall(sysctl_init);
+
+struct ctl_table_header *register_net_sysctl_table(struct net *net,
+ const struct ctl_path *path, struct ctl_table *table)
+{
+ struct nsproxy namespaces;
+ namespaces = *current->nsproxy;
+ namespaces.net_ns = net;
+ return __register_sysctl_paths(&net_sysctl_root,
+ &namespaces, path, table);
+}
+EXPORT_SYMBOL_GPL(register_net_sysctl_table);
+
+void unregister_net_sysctl_table(struct ctl_table_header *header)
+{
+ return unregister_sysctl_table(header);
+}
+EXPORT_SYMBOL_GPL(unregister_net_sysctl_table);
--
1.5.3.rc6.17.g1911

```

---

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

---



---

Subject: Re: [PATCH 4/4] net: Implement the per network namespace sysctl infrastructure

Posted by [serue](#) on Fri, 30 Nov 2007 16:18:56 GMT

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

---

Quoting Eric W. Biederman (ebiederm@xmission.com):

```

>
> The user interface is: register_net_sysctl_table and
> unregister_net_sysctl_table. Very much like the current
> interface except there is a network namespace parameter.
>
> With this any sysctl registered with register_net_sysctl_table
> will only show up to tasks in the same network namespace.
>
> All other sysctls continue to be globally visible.
>
> Signed-off-by: Eric W. Biederman <ebiederm@xmission.com>
> ---

```

```

> include/net/net_namespace.h | 9 ++++++
> net/sysctl_net.c | 57 ++++++
> 2 files changed, 66 insertions(+), 0 deletions(-)
>
> diff --git a/include/net/net_namespace.h b/include/net/net_namespace.h
> index 4d0d634..235214c 100644
> --- a/include/net/net_namespace.h
> +++ b/include/net/net_namespace.h
> @@ -25,6 +25,8 @@ struct net {
> struct proc_dir_entry *proc_net_stat;
> struct proc_dir_entry *proc_net_root;
>
> + struct list_head sysctl_table_headers;
> +
> struct net_device *loopback_dev; /* The loopback */
>
> struct list_head dev_base_head;
> @@ -144,4 +146,11 @@ extern void unregister_pernet_subsys(struct pernet_operations *);
> extern int register_pernet_device(struct pernet_operations *);
> extern void unregister_pernet_device(struct pernet_operations *);
>
> +struct ctl_path;
> +struct ctl_table;
> +struct ctl_table_header;
> +extern struct ctl_table_header *register_net_sysctl_table(struct net *net,
> + const struct ctl_path *path, struct ctl_table *table);
> +extern void unregister_net_sysctl_table(struct ctl_table_header *header);
> +
> #endif /* __NET_NET_NAMESPACE_H */
> diff --git a/net/sysctl_net.c b/net/sysctl_net.c
> index cd4eafb..c50c793 100644
> --- a/net/sysctl_net.c
> +++ b/net/sysctl_net.c
> @@ -14,6 +14,7 @@
>
> #include <linux/mm.h>
> #include <linux/sysctl.h>
> +#include <linux/nsproxy.h>
>
> #include <net/sock.h>
>
> @@ -54,3 +55,59 @@ struct ctl_table net_table[] = {
> #endif
> { 0 },
> };
> +
> +static struct list_head *
> +net_ctl_header_lookup(struct ctl_table_root *root, struct nsproxy *namespaces)

```

```

> +{
> + return &namespaces->net_ns->sysctl_table_headers;
> +}
> +
> +static struct ctl_table_root net_sysctl_root = {
> + .lookup = net_ctl_header_lookup,
> +};
> +
> +static int sysctl_net_init(struct net *net)
> +{
> + INIT_LIST_HEAD(&net->sysctl_table_headers);
> + return 0;
> +}
> +
> +static void sysctl_net_exit(struct net *net)
> +{
> + WARN_ON(!list_empty(&net->sysctl_table_headers));
> + return;
> +}
> +
> +static struct pernet_operations sysctl_pernet_ops = {
> + .init = sysctl_net_init,
> + .exit = sysctl_net_exit,
> +};
> +
> +static __init int sysctl_init(void)
> +{
> + int ret;
> + ret = register_pernet_subsys(&sysctl_pernet_ops);
> + if (ret)
> + goto out;
> + register_sysctl_root(&net_sysctl_root);
> +out:
> + return ret;
> +}
> +subsys_initcall(sysctl_init);
> +
> +struct ctl_table_header *register_net_sysctl_table(struct net *net,
> + const struct ctl_path *path, struct ctl_table *table)
> +{
> + struct nsproxy namespaces;
> + namespaces = *current->nsproxy;
> + namespaces.net_ns = net;
> + return __register_sysctl_paths(&net_sysctl_root,
> + &namespaces, path, table);

```

Hey Eric,

the patches look nice.

The hand-forcing of the passed-in net\_ns into a copy of current->nsproxy does make it seem like nsproxy may not be the best choice of what to pass in. Doesn't only net\_sysctl\_root->lookup() look at the argument?

But I assume you don't want to be more general than sending in a nsproxy so as to dissuade abuse of this interface for needlessly complex sysctl interfaces?

(Well I expect that'll become clear once the the patches using this come out.)

Are you planning to use this infrastructure for the uts and ipc sysctls as well?

thanks,  
-serge

```
> +}  
> +EXPORT_SYMBOL_GPL(register_net_sysctl_table);  
> +  
> +void unregister_net_sysctl_table(struct ctl_table_header *header)  
> +{  
> + return unregister_sysctl_table(header);  
> +}  
> +EXPORT_SYMBOL_GPL(unregister_net_sysctl_table);  
> --  
> 1.5.3.rc6.17.g1911
```

---

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

---

---

Subject: Re: [PATCH 4/4] net: Implement the per network namespace sysctl infrastructure  
Posted by [Pavel Emelianov](#) on Fri, 30 Nov 2007 16:23:15 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

[snip]

```
>> +    &namespaces, path, table);  
>  
> Hey Eric,  
>  
> the patches look nice.
```

Agree ;)

> The hand-forcing of the passed-in net\_ns into a copy of current->nsproxy  
> does make it seem like nsproxy may not be the best choice of what to  
> pass in. Doesn't only net\_sysctl\_root->lookup() look at the argument?  
>  
> But I assume you don't want to be more general than sending in a  
> nsproxy so as to dissuade abuse of this interface for needlessly complex  
> sysctl interfaces?  
>  
> (Well I expect that'll become clear once the the patches using this  
> come out.)  
>  
> Are you planning to use this infrastructure for the uts and ipc  
> sysctls as well?

I have sent some patches concerning uts and ipc already.  
I'd appreciate any feedback on it :)

> thanks,  
> -serge

Thanks,  
Pavel

---

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

---

---

Subject: Re: [PATCH 4/4] net: Implement the per network namespace sysctl  
infrastructure

Posted by [ebiederm](#) on Fri, 30 Nov 2007 21:49:25 GMT

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

---

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

>  
> Hey Eric,  
>  
> the patches look nice.  
>  
> The hand-forcing of the passed-in net\_ns into a copy of current->nsproxy  
> does make it seem like nsproxy may not be the best choice of what to  
> pass in. Doesn't only net\_sysctl\_root->lookup() look at the argument?

Yes. Although I call it from \_\_register\_sysctl\_paths.

> But I assume you don't want to be more general than sending in a  
> nsproxy so as to dissuade abuse of this interface for needlessly complex  
> sysctl interfaces?

A bit of that. I would love to pass in a task\_struct so you can use anything from a task. The trouble is I don't have any task\_structs or nsproxys with the proper value at the point where I am first setting this up. Further I have to have the full sysctl lookup working or I could not call sysctl\_check.

> (Well I expect that'll become clear once the the patches using this  
> come out.)  
>  
> Are you planning to use this infrastructure for the uts and ipc  
> sysctls as well?

Yes. Where it comes in especially useful, is I can move /proc/sys to /proc/sys/<tgid>/task/<pid>/sys. And get a particular processes view of sysctl.

We also get a little more reuse of common functions.

Otherwise Pavel does have a point that using this for uts and ipc is not a savings lines of code wise.

After having seen Pavel changes I am asking myself if there is a sane way to remove the ctl\_name argument from the ctl\_path.

Anyway where I am with the nsproxy question was that I don't see anything easily better. What I have works and gets the job done, and doesn't have any module unload races or holes where a sloppy programmer can mess up the sysctl tree. We needed a solution. Trying any harder to find something better would take ages. So I figured this implementation was good enough.

Eric

---

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

---

---

Subject: Re: [PATCH 4/4] net: Implement the per network namespace sysctl infrastructure

Posted by [serue](#) on Sat, 01 Dec 2007 00:01:50 GMT

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

---

Quoting Eric W. Biederman (ebiederm@xmission.com):

> "Serge E. Hallyn" <serue@us.ibm.com> writes:  
>  
> >  
> > Hey Eric,  
> >  
> > the patches look nice.  
> >  
> > The hand-forcing of the passed-in net\_ns into a copy of current->nsproxy  
> > does make it seem like nsproxy may not be the best choice of what to  
> > pass in. Doesn't only net\_sysctl\_root->lookup() look at the argument?  
>  
> Yes. Although I call it from \_\_register\_sysctl\_paths.  
>  
> > But I assume you don't want to be more general than sending in a  
> > nsproxy so as to dissuade abuse of this interface for needlessly complex  
> > sysctl interfaces?  
>  
> A bit of that. I would love to pass in a task\_struct so you can use  
> anything from a task. The trouble is I don't have any task\_structs or  
> nsproxys with the proper value at the point where I am first setting  
> this up. Further I have to have the full sysctl lookup working or I  
> could not call sysctl\_check.  
>  
> > (Well I expect that'll become clear once the the patches using this  
> > come out.)  
> >  
> > Are you planning to use this infrastructure for the uts and ipc  
> > sysctls as well?  
>  
> Yes. Where it comes in especially useful, is I can move /proc/sys  
> to /proc/sys/<tgid>/task/<pid>/sys. And get a particular processes  
> view of sysctl.  
>  
> We also get a little more reuse of common functions.  
>  
> Otherwise Pavel does have a point that using this for uts and ipc  
> is not a savings lines of code wise.  
>  
> After having seen Pavel changes I am asking myself if there is a sane  
> way to remove the ctl\_name argument from the ctl\_path.  
>  
> Anyway where I am with the nsproxy question was that I don't  
> see anything easily better. What I have works and gets the job  
> done, and doesn't have any module unload races or holes where a sloppy  
> programmer can mess up the sysctl tree. We needed a solution.  
> Trying any harder to find something better would take ages. So  
> I figured this implementation was good enough.

I agree. So it's already in -mm but still

Acked-by: Serge Hallyn <serue@us.ibm.com>

thanks,  
-serge

---

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

---