
Subject: [PATCH][NETNS] Move some code into __init section when CONFIG_NET_NS=n

Posted by [Pavel Emelianov](#) on Thu, 04 Oct 2007 13:54:11 GMT

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

With the net namespaces many code leaved the __init section, thus making the kernel occupy more memory than it did before. Since we have a config option that prohibits the namespace creation, the functions that initialize/finalize some netns stuff are simply not needed and can be freed after the boot.

Currently, this is almost not noticeable, since few calls are no longer in __init, but when the namespaces will be merged it will be possible to free more code. I propose to use the __net_init, __net_exit and __net_initdata "attributes" for functions/variables that are not used if the CONFIG_NET_NS is not set to save more space in memory.

Signed-off-by: Pavel Emelyanov <xemul@openvz.org>

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

```
index 934c840..747170f 100644
```

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

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

```
@@ -99,6 +99,15 @@ static inline void release_net(struct ne
```

```
#define for_each_net(VAR) \
```

```
list_for_each_entry(VAR, &net_namespace_list, list)
```

```
+#ifdef CONFIG_NET_NS
```

```
+#define __net_init
```

```
+#define __net_exit
```

```
+#define __net_initdata
```

```
+#else
```

```
+#define __net_init __init
```

```
+#define __net_exit __exit
```

```
+#define __net_initdata __initdata
```

```
+#endif
```

```
struct pernet_operations {
```

```
struct list_head list;
```

```
diff --git a/drivers/net/loopback.c b/drivers/net/loopback.c
```

```
index d6997ae..be25aa3 100644
```

```
--- a/drivers/net/loopback.c
```

```
+++ b/drivers/net/loopback.c
```

```
@@ -250,7 +250,7 @@ static void loopback_setup(struct net_de
```

```
}
```

```

/* Setup and register the loopback device. */
-static int loopback_net_init(struct net *net)
+static __net_init int loopback_net_init(struct net *net)
{
    struct net_device *dev;
    int err;
@@ -278,14 +278,14 @@ out_free_netdev:
    goto out;
}

-static void loopback_net_exit(struct net *net)
+static __net_exit void loopback_net_exit(struct net *net)
{
    struct net_device *dev = net->loopback_dev;

    unregister_netdev(dev);
}

-static struct pernet_operations loopback_net_ops = {
+static struct pernet_operations __net_initdata loopback_net_ops = {
    .init = loopback_net_init,
    .exit = loopback_net_exit,
};
diff --git a/fs/proc/proc_net.c b/fs/proc/proc_net.c
index 85cc8e8..2e91fb7 100644
--- a/fs/proc/proc_net.c
+++ b/fs/proc/proc_net.c
@@ -140,7 +140,7 @@ static struct inode_operations proc_net_
    .setattr = proc_net_setattr,
};

-static int proc_net_ns_init(struct net *net)
+static __net_init int proc_net_ns_init(struct net *net)
{
    struct proc_dir_entry *root, *netd, *net_statd;
    int err;
@@ -178,19 +178,19 @@ free_root:
    goto out;
}

-static void proc_net_ns_exit(struct net *net)
+static __net_exit void proc_net_ns_exit(struct net *net)
{
    remove_proc_entry("stat", net->proc_net);
    remove_proc_entry("net", net->proc_net_root);
    kfree(net->proc_net_root);
}

```

```

-struct pernet_operations proc_net_ns_ops = {
+struct pernet_operations __net_initdata proc_net_ns_ops = {
    .init = proc_net_ns_init,
    .exit = proc_net_ns_exit,
};

-int proc_net_init(void)
+int __init proc_net_init(void)
{
    proc_net_shadow = proc_mkdir("net", NULL);
    proc_net_shadow->proc_iops = &proc_net_dir_inode_operations;
diff --git a/net/core/dev.c b/net/core/dev.c
index d998646..37f8858 100644
--- a/net/core/dev.c
+++ b/net/core/dev.c
@@ -2611,7 +2611,7 @@ static const struct file_operations ptyp
};

-static int dev_proc_net_init(struct net *net)
+static int __net_init dev_proc_net_init(struct net *net)
{
    int rc = -ENOMEM;

@@ -2636,7 +2636,7 @@ out_dev:
    goto out;
}

-static void dev_proc_net_exit(struct net *net)
+static void __net_exit dev_proc_net_exit(struct net *net)
{
    wext_proc_exit(net);

@@ -2645,7 +2645,7 @@ static void dev_proc_net_exit(struct net
    proc_net_remove(net, "dev");
}

-static struct pernet_operations dev_proc_ops = {
+static struct pernet_operations __net_initdata dev_proc_ops = {
    .init = dev_proc_net_init,
    .exit = dev_proc_net_exit,
};
@@ -4250,7 +4250,7 @@ static struct hlist_head *netdev_create_
}

/* Initialize per network namespace state */
-static int netdev_init(struct net *net)

```

```

+static int __net_init netdev_init(struct net *net)
{
    INIT_LIST_HEAD(&net->dev_base_head);
    rwlock_init(&dev_base_lock);
@@ -4271,18 +4271,18 @@ err_name:
    return -ENOMEM;
}

-static void netdev_exit(struct net *net)
+static void __net_exit netdev_exit(struct net *net)
{
    kfree(net->dev_name_head);
    kfree(net->dev_index_head);
}

-static struct pernet_operations netdev_net_ops = {
+static struct pernet_operations __net_initdata netdev_net_ops = {
    .init = netdev_init,
    .exit = netdev_exit,
};

-static void default_device_exit(struct net *net)
+static void __net_exit default_device_exit(struct net *net)
{
    struct net_device *dev, *next;
    /*
@@ -4308,7 +4308,7 @@ static void default_device_exit(struct n
    rtnl_unlock();
}

-static struct pernet_operations default_device_ops = {
+static struct pernet_operations __net_initdata default_device_ops = {
    .exit = default_device_exit,
};

diff --git a/net/core/dev_mcast.c b/net/core/dev_mcast.c
index 896b0ca..15241cf 100644
--- a/net/core/dev_mcast.c
+++ b/net/core/dev_mcast.c
@@ -273,19 +273,19 @@ static const struct file_operations dev_

#endif

-static int dev_mc_net_init(struct net *net)
+static int __net_init dev_mc_net_init(struct net *net)
{
    if (!proc_net_fops_create(net, "dev_mcast", 0, &dev_mc_seq_fops))
        return -ENOMEM;
}

```

```

return 0;
}

-static void dev_mc_net_exit(struct net *net)
+static void __net_exit dev_mc_net_exit(struct net *net)
{
    proc_net_remove(net, "dev_mcast");
}

-static struct pernet_operations dev_mc_net_ops = {
+static struct pernet_operations __net_initdata dev_mc_net_ops = {
    .init = dev_mc_net_init,
    .exit = dev_mc_net_exit,
};
diff --git a/net/netlink/af_netlink.c b/net/netlink/af_netlink.c
index 46eb5ea..3ef3282 100644
--- a/net/netlink/af_netlink.c
+++ b/net/netlink/af_netlink.c
@@ -1924,7 +1924,7 @@ static struct net_proto_family netlink_f
    .owner = THIS_MODULE, /* for consistency 8) */
};

-static int netlink_net_init(struct net *net)
+static int __net_init netlink_net_init(struct net *net)
{
#ifdef CONFIG_PROC_FS
    if (!proc_net_fops_create(net, "netlink", 0, &netlink_seq_fops))
@@ -1933,14 +1933,14 @@ static int netlink_net_init(struct net *
    return 0;
}

-static void netlink_net_exit(struct net *net)
+static void __net_exit netlink_net_exit(struct net *net)
{
#ifdef CONFIG_PROC_FS
    proc_net_remove(net, "netlink");
#endif
}

-static struct pernet_operations netlink_net_ops = {
+static struct pernet_operations __net_initdata netlink_net_ops = {
    .init = netlink_net_init,
    .exit = netlink_net_exit,
};

```

Subject: Re: [PATCH][NETNS] Move some code into __init section when

CONFIG_NET_NS=n

Posted by [Alexey Dobriyan](#) on Thu, 04 Oct 2007 14:02:02 GMT

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

On Thu, Oct 04, 2007 at 05:54:11PM +0400, Pavel Emelyanov wrote:

> With the net namespaces many code leaved the __init section,
> thus making the kernel occupy more memory than it did before.
> Since we have a config option that prohibits the namespace
> creation, the functions that initialize/finalize some netns
> stuff are simply not needed and can be freed after the boot.
>
> Currently, this is almost not noticeable, since few calls
> are no longer in __init, but when the namespaces will be
> merged it will be possible to free more code. I propose to
> use the __net_init, __net_exit and __net_initdata "attributes"
> for functions/variables that are not used if the CONFIG_NET_NS
> is not set to save more space in memory.

```
> #ifdef CONFIG_NET_NS  
> #define __net_init  
> #define __net_exit  
> #define __net_initdata  
> #else  
> #define __net_init __init  
> #define __net_exit __exit  
> #define __net_initdata __initdata  
> #endif
```

Yet another set of double-underscored section annotations is the last thing that is needed, methinks. :)

Subject: Re: [PATCH][NETNS] Move some code into __init section when CONFIG_NET_NS=n

Posted by [davem](#) on Mon, 08 Oct 2007 07:17:16 GMT

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

From: Alexey Dobriyan <adobriyan@sw.ru>

Date: Thu, 4 Oct 2007 18:02:02 +0400

> On Thu, Oct 04, 2007 at 05:54:11PM +0400, Pavel Emelyanov wrote:
> > With the net namespaces many code leaved the __init section,
> > thus making the kernel occupy more memory than it did before.
> > Since we have a config option that prohibits the namespace
> > creation, the functions that initialize/finalize some netns
> > stuff are simply not needed and can be freed after the boot.
> >
> > Currently, this is almost not noticeable, since few calls

> > are no longer in __init, but when the namespaces will be
> > merged it will be possible to free more code. I propose to
> > use the __net_init, __net_exit and __net_initdata "attributes"
> > for functions/variables that are not used if the CONFIG_NET_NS
> > is not set to save more space in memory.
>
> > +#ifdef CONFIG_NET_NS
> > +#define __net_init
> > +#define __net_exit
> > +#define __net_initdata
> > +#else
> > +#define __net_init __init
> > +#define __net_exit __exit
> > +#define __net_initdata __initdata
> > +#endif
>
> Yet another set of double-underscored section annotations is the last thing
> that is needed, methinks. :)

I'm not so sure. I think double-underscores are a powerful deterrent for developers. Over time folks have learned that this prefix in an interface name means "semantics are non-trivial, use with care" and that definitely applies to init sections.

I'm going to apply Pavel's patch, if you want to touch it up do so as a followon patch, thanks!

Subject: Re: [PATCH][NETNS] Move some code into __init section when CONFIG_NET_NS=n

Posted by [davem](#) on Mon, 08 Oct 2007 07:18:07 GMT

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

From: Pavel Emelyanov <xemul@openvz.org>

Date: Thu, 04 Oct 2007 17:54:11 +0400

> With the net namespaces many code leaved the __init section,
> thus making the kernel occupy more memory than it did before.
> Since we have a config option that prohibits the namespace
> creation, the functions that initialize/finalize some netns
> stuff are simply not needed and can be freed after the boot.
>
> Currently, this is almost not noticeable, since few calls
> are no longer in __init, but when the namespaces will be
> merged it will be possible to free more code. I propose to
> use the __net_init, __net_exit and __net_initdata "attributes"
> for functions/variables that are not used if the CONFIG_NET_NS
> is not set to save more space in memory.

>
> Signed-off-by: Pavel Emelyanov <xemul@openvz.org>

Applied, thanks Pavel.

Subject: Re: [PATCH][NETNS] Move some code into __init section when
CONFIG_NET_NS=n

Posted by [davem](#) on Mon, 08 Oct 2007 07:30:41 GMT

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

From: David Miller <davem@davemloft.net>
Date: Mon, 08 Oct 2007 00:17:16 -0700 (PDT)

> I'm going to apply Pavel's patch, if you want to touch it up do so
> as a followon patch, thanks!

Actually, I take that back, it doesn't even build, I'm therefore
removing the patch:

```
LD    .tmp_vmlinux1
`.exit.text' referenced in section `.init.data' of fs/built-in.o: defined in discarded section `.exit.text' of
fs/built-in.o
`.exit.text' referenced in section `.init.data' of drivers/built-in.o: defined in discarded section
`.exit.text' of drivers/built-in.o
`.exit.text' referenced in section `.init.data' of net/built-in.o: defined in discarded section `.exit.text'
of net/built-in.o
`.exit.text' referenced in section `.init.data' of net/built-in.o: defined in discarded section `.exit.text'
of net/built-in.o
`.exit.text' referenced in section `.init.data' of net/built-in.o: defined in discarded section `.exit.text'
of net/built-in.o
`.exit.text' referenced in section `.init.data' of net/built-in.o: defined in discarded section `.exit.text'
of net/built-in.o
`.exit.text' referenced in section `.init.data' of net/built-in.o: defined in discarded section `.exit.text'
of net/built-in.o
make: *** [.tmp_vmlinux1] Error 1
```

Please fix this up and resubmit if you like.
