
Subject: Re: [PATCH 3/5] Move the IPC namespace under the option
Posted by [Cedric Le Goater](#) on Thu, 27 Sep 2007 12:38:51 GMT
[View Forum Message](#) <> [Reply to Message](#)

Pavel Emelyanov wrote:

> Currently all the IPC namespace management code is in
> ipc/util.c. I moved this code into ipc/namespace.c file
> which is compiled out when needed.
>
> The linux/ipc_namespace.h file is used to store the
> prototypes of the functions in namespace.c and the stubs
> for NAMESPACES=n case. This is done so, because the stub
> for copy_ipc_namespace requires the knowledge of the
> CLONE_NEWIPC flag, which is in sched.h. But the linux/ipc.h
> file itself is included into many many .c files via the
> sys.h->sem.h sequence so adding the sched.h into it will
> make all these .c depend on sched.h which is not that good.
> On the other hand the knowledge about the namespaces stuff
> is required in 4 .c files only.
>
> Besides, this patch compiles out some auxiliary functions
> from ipc/sem.c, msg.c and shm.c files.
>
> Signed-off-by: Pavel Emelyanov <xemul@openvz.org>

^
that sounds french :)

Also, for the code aesthetic,

I think that ipc/ipc_namespace.c is a better name which fits
the header file ipc/ipc_namespace.h.

I'm wondering if it possible to move the init routines :

```
#ifdef CONFIG_NAMESPACES
int {msg,sem,shm}_init_ns(struct ipc_namespace *ns)
{
    ...
}
#endif
```

from the ipc/{msg,sem,shm}.c file to the ipc/ipc_namespace.c file.
It would look better in the code than using ugly #ifdef.

Thanks !

C.

```

>
> ---
>
> diff --git a/include/linux/ipc.h b/include/linux/ipc.h
> index 96988d1..b882610 100644
> --- a/include/linux/ipc.h
> +++ b/include/linux/ipc.h
> @@ -100,56 +100,6 @@ struct kern_ipc_perm
> void *security;
> };
>
> -struct ipc_ids;
> -struct ipc_namespace {
> - struct kref kref;
> - struct ipc_ids *ids[3];
> -
> - int sem_ctls[4];
> - int used_sems;
> -
> - int msg_ctlmax;
> - int msg_ctlmnb;
> - int msg_ctlmni;
> -
> - size_t shm_ctlmax;
> - size_t shm_ctlall;
> - int shm_ctlmni;
> - int shm_tot;
> -};
> -
> -extern struct ipc_namespace init_ipc_ns;
> -
> -#ifdef CONFIG_SYSVIPC
> -#define INIT_IPC_NS(ns) .ns = &init_ipc_ns,
> -extern void free_ipc_ns(struct kref *kref);
> -extern struct ipc_namespace *copy_ipcs(unsigned long flags,
> - struct ipc_namespace *ns);
> -#else
> -#define INIT_IPC_NS(ns)
> -static inline struct ipc_namespace *copy_ipcs(unsigned long flags,
> - struct ipc_namespace *ns)
> -{
> - return ns;
> -}
> -#endif
> -
> -static inline struct ipc_namespace *get_ipc_ns(struct ipc_namespace *ns)
> -{

```

```

> -#ifdef CONFIG_SYSVIPC
> - if (ns)
> - kref_get(&ns->kref);
> -#endif
> - return ns;
> -}
> -
> -static inline void put_ipc_ns(struct ipc_namespace *ns)
> -{
> -#ifdef CONFIG_SYSVIPC
> - kref_put(&ns->kref, free_ipc_ns);
> -#endif
> -}
> -
> #endif /* __KERNEL__ */
>
> #endif /* _LINUX_IPC_H */
> diff --git a/include/linux/ipc_namespace.h b/include/linux/ipc_namespace.h
> new file mode 100644
> index 0000000..3d8a516
> --- /dev/null
> +++ b/include/linux/ipc_namespace.h
> @@ -0,0 +1,67 @@
> +#ifndef __IPC_NAMESPACE_H__
> +#define __IPC_NAMESPACE_H__
> +
> +#include <linux/err.h>
> +
> +struct ipc_ids;
> +struct ipc_namespace {
> + struct kref kref;
> + struct ipc_ids *ids[3];
> +
> + int sem_ctls[4];
> + int used_sems;
> +
> + int msg_ctlmax;
> + int msg_ctlmnb;
> + int msg_ctlmni;
> +
> + size_t shm_ctlmax;
> + size_t shm_ctlall;
> + int shm_ctlmni;
> + int shm_tot;
> +};
> +
> +extern struct ipc_namespace init_ipc_ns;
> +

```

```

> + #ifdef CONFIG_SYSVIPC
> + #define INIT_IPC_NS(ns) .ns = &init_ipc_ns,
> + #else
> + #define INIT_IPC_NS(ns)
> + #endif
> +
> + #if defined(CONFIG_SYSVIPC) && defined(CONFIG_NAMESPACES)
> + extern void free_ipc_ns(struct kref *kref);
> + extern struct ipc_namespace *copy_ipcs(unsigned long flags,
> +     struct ipc_namespace *ns);
> +
> + static inline struct ipc_namespace *get_ipc_ns(struct ipc_namespace *ns)
> + {
> +     if (ns)
> +         kref_get(&ns->kref);
> +     return ns;
> + }
> +
> + static inline void put_ipc_ns(struct ipc_namespace *ns)
> + {
> +     kref_put(&ns->kref, free_ipc_ns);
> + }
> + #else
> + static inline struct ipc_namespace *copy_ipcs(unsigned long flags,
> +     struct ipc_namespace *ns)
> + {
> +     if (flags & CLONE_NEWIPC)
> +         return ERR_PTR(-EINVAL);
> +
> +     return ns;
> + }
> +
> + static inline struct ipc_namespace *get_ipc_ns(struct ipc_namespace *ns)
> + {
> +     return ns;
> + }
> +
> + static inline void put_ipc_ns(struct ipc_namespace *ns)
> + {
> + }
> + #endif
> + #endif
> diff --git a/ipc/util.c b/ipc/util.c
> index fd29246..44fb843 100644
> --- a/ipc/util.c
> +++ b/ipc/util.c
> @@ -32,6 +32,7 @@
> #include <linux/proc_fs.h>

```

```

> #include <linux/audit.h>
> #include <linux/nsproxy.h>
> +#include <linux/ipc_namespace.h>
>
> #include <asm/unistd.h>
>
> @@ -50,66 +51,6 @@ struct ipc_namespace init_ipc_ns = {
> },
> };
>
> -static struct ipc_namespace *clone_ipc_ns(struct ipc_namespace *old_ns)
> -{
> - int err;
> - struct ipc_namespace *ns;
> -
> - err = -ENOMEM;
> - ns = kmalloc(sizeof(struct ipc_namespace), GFP_KERNEL);
> - if (ns == NULL)
> - goto err_mem;
> -
> - err = sem_init_ns(ns);
> - if (err)
> - goto err_sem;
> - err = msg_init_ns(ns);
> - if (err)
> - goto err_msg;
> - err = shm_init_ns(ns);
> - if (err)
> - goto err_shm;
> -
> - kref_init(&ns->kref);
> - return ns;
> -
> -err_shm:
> - msg_exit_ns(ns);
> -err_msg:
> - sem_exit_ns(ns);
> -err_sem:
> - kfree(ns);
> -err_mem:
> - return ERR_PTR(err);
> -}
> -
> -struct ipc_namespace *copy_ipcs(unsigned long flags, struct ipc_namespace *ns)
> -{
> - struct ipc_namespace *new_ns;
> -
> - BUG_ON(!ns);

```

```

> - get_ipc_ns(ns);
> -
> - if (!(flags & CLONE_NEWIPC))
> - return ns;
> -
> - new_ns = clone_ipc_ns(ns);
> -
> - put_ipc_ns(ns);
> - return new_ns;
> -}
> -
> -void free_ipc_ns(struct kref *kref)
> -{
> - struct ipc_namespace *ns;
> -
> - ns = container_of(kref, struct ipc_namespace, kref);
> - sem_exit_ns(ns);
> - msg_exit_ns(ns);
> - shm_exit_ns(ns);
> - kfree(ns);
> -}
> -
> /**
>  * ipc_init - initialise IPC subsystem
>  *
> diff --git a/ipc/namespace.c b/ipc/namespace.c
> new file mode 100644
> index 0000000..cef1139
> --- /dev/null
> +++ b/ipc/namespace.c
> @@ -0,0 +1,73 @@
> +/*
> + * linux/ipc/namespace.c
> + * Copyright (C) 2006 Pavel Emelyanov <xemul@openvz.org> OpenVZ, SWsoft Inc.
> + */
> +
> +#include <linux/ipc.h>
> +#include <linux/msg.h>
> +#include <linux/ipc_namespace.h>
> +#include <linux/rcupdate.h>
> +#include <linux/nsproxy.h>
> +#include <linux/slab.h>
> +
> +#include "util.h"
> +
> +static struct ipc_namespace *clone_ipc_ns(struct ipc_namespace *old_ns)
> +{
> + int err;

```

```

> + struct ipc_namespace *ns;
> +
> + err = -ENOMEM;
> + ns = kmalloc(sizeof(struct ipc_namespace), GFP_KERNEL);
> + if (ns == NULL)
> +     goto err_mem;
> +
> + err = sem_init_ns(ns);
> + if (err)
> +     goto err_sem;
> + err = msg_init_ns(ns);
> + if (err)
> +     goto err_msg;
> + err = shm_init_ns(ns);
> + if (err)
> +     goto err_shm;
> +
> + kref_init(&ns->kref);
> + return ns;
> +
> +err_shm:
> + msg_exit_ns(ns);
> +err_msg:
> + sem_exit_ns(ns);
> +err_sem:
> + kfree(ns);
> +err_mem:
> + return ERR_PTR(err);
> +}
> +
> +struct ipc_namespace *copy_ipcs(unsigned long flags, struct ipc_namespace *ns)
> +{
> + struct ipc_namespace *new_ns;
> +
> + BUG_ON(!ns);
> + get_ipc_ns(ns);
> +
> + if (!(flags & CLONE_NEWIPC))
> +     return ns;
> +
> + new_ns = clone_ipc_ns(ns);
> +
> + put_ipc_ns(ns);
> + return new_ns;
> +}
> +
> +void free_ipc_ns(struct kref *kref)
> +{

```

```

> + struct ipc_namespace *ns;
> +
> + ns = container_of(kref, struct ipc_namespace, kref);
> + sem_exit_ns(ns);
> + msg_exit_ns(ns);
> + shm_exit_ns(ns);
> + kfree(ns);
> +}
> diff --git a/ipc/Makefile b/ipc/Makefile
> index b93bba6..d81fb35 100644
> --- a/ipc/Makefile
> +++ b/ipc/Makefile
> @@ -7,4 +7,5 @@ obj-$(CONFIG_SYSVIPC) += util.o msgutil.
> obj-$(CONFIG_SYSVIPC_SYSCTL) += ipc_sysctl.o
> obj_mq-$(CONFIG_COMPAT) += compat_mq.o
> obj-$(CONFIG_POSIX_QUEUE) += mqueue.o msgutil.o $(obj_mq-y)
> +obj-$(CONFIG_NAMESPACES) += namespace.o
>
> diff --git a/ipc/ipc_sysctl.c b/ipc/ipc_sysctl.c
> index 79e24e8..7f4235b 100644
> --- a/ipc/ipc_sysctl.c
> +++ b/ipc/ipc_sysctl.c
> @@ -14,6 +14,7 @@
> #include <linux/nsproxy.h>
> #include <linux/sysctl.h>
> #include <linux/uaccess.h>
> +#include <linux/ipc_namespace.h>
>
> static void *get_ipc(ctl_table *table)
> {
> diff --git a/ipc/msg.c b/ipc/msg.c
> index b7274db..eb74965 100644
> --- a/ipc/msg.c
> +++ b/ipc/msg.c
> @@ -36,6 +36,7 @@
> #include <linux/seq_file.h>
> #include <linux/mutex.h>
> #include <linux/nsproxy.h>
> +#include <linux/ipc_namespace.h>
>
> #include <asm/current.h>
> #include <asm/uaccess.h>
> @@ -92,6 +93,7 @@ static void __msg_init_ns(struct ipc_nam
> ipc_init_ids(ids);
> }
>
> +#ifdef CONFIG_NAMESPACES
> int msg_init_ns(struct ipc_namespace *ns)

```

```

> {
> struct ipc_ids *ids;
> @@ -127,6 +129,7 @@ void msg_exit_ns(struct ipc_namespace *n
> kfree(ns->ids[IPC_MSG_IDS]);
> ns->ids[IPC_MSG_IDS] = NULL;
> }
> +#endif
>
> void __init msg_init(void)
> {
> diff --git a/ipc/sem.c b/ipc/sem.c
> index 45c7e57..2e9f449 100644
> --- a/ipc/sem.c
> +++ b/ipc/sem.c
> @@ -82,6 +82,7 @@
> #include <linux/seq_file.h>
> #include <linux/mutex.h>
> #include <linux/nsproxy.h>
> +#include <linux/ipc_namespace.h>
>
> #include <asm/uaccess.h>
> #include "util.h"
> @@ -130,6 +131,7 @@ static void __sem_init_ns(struct ipc_nam
> ipc_init_ids(ids);
> }
>
> +#ifdef CONFIG_NAMESPACES
> int sem_init_ns(struct ipc_namespace *ns)
> {
> struct ipc_ids *ids;
> @@ -165,6 +167,7 @@ void sem_exit_ns(struct ipc_namespace *n
> kfree(ns->ids[IPC_SEM_IDS]);
> ns->ids[IPC_SEM_IDS] = NULL;
> }
> +#endif
>
> void __init sem_init (void)
> {
> diff --git a/ipc/shm.c b/ipc/shm.c
> index f28f2a3..2717cbc 100644
> --- a/ipc/shm.c
> +++ b/ipc/shm.c
> @@ -38,6 +38,7 @@
> #include <linux/mutex.h>
> #include <linux/nsproxy.h>
> #include <linux/mount.h>
> +#include <linux/ipc_namespace.h>
>

```

```

> #include <asm/uaccess.h>
>
> @@ -97,6 +98,7 @@ static void do_shm_rmid(struct ipc_names
>   shm_destroy(ns, shp);
> }
>
> +#ifdef CONFIG_NAMESPACES
> int shm_init_ns(struct ipc_namespace *ns)
> {
>   struct ipc_ids *ids;
> @@ -132,6 +134,7 @@ void shm_exit_ns(struct ipc_namespace *n
>   kfree(ns->ids[IPC_SHM_IDS]);
>   ns->ids[IPC_SHM_IDS] = NULL;
> }
> +#endif
>
> void __init shm_init (void)
> {
> diff --git a/ipc/util.h b/ipc/util.h
> index 99414a3..8972402 100644
> --- a/ipc/util.h
> +++ b/ipc/util.h
> @@ -20,6 +20,8 @@ void sem_init (void);
> void msg_init (void);
> void shm_init (void);
>
> +struct ipc_namespace;
> +
> int sem_init_ns(struct ipc_namespace *ns);
> int msg_init_ns(struct ipc_namespace *ns);
> int shm_init_ns(struct ipc_namespace *ns);
> diff --git a/kernel/nsproxy.c b/kernel/nsproxy.c
> index ee68964..5a27426 100644
> --- a/kernel/nsproxy.c
> +++ b/kernel/nsproxy.c
> @@ -20,6 +20,7 @@
> #include <linux/mnt_namespace.h>
> #include <linux/utsname.h>
> #include <linux/pid_namespace.h>
> +#include <linux/ipc_namespace.h>
>
> static struct kmem_cache *nsproxy_cachep;
>
>
>

```

Containers mailing list

