
Subject: [PATCH 3/3] Switch IPC namespace to use sysctl shadows

Posted by [Pavel Emelianov](#) on Tue, 20 Nov 2007 11:47:41 GMT

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

The same as with UTS - use the shadows. With this there's no longer need in having kludged ->proc_handler and ->strategy for ctl_tables.

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

diff --git a/include/linux/ipc.h b/include/linux/ipc.h

index 408696e..a9fef83 100644

--- a/include/linux/ipc.h

+++ b/include/linux/ipc.h

@@ -118,6 +118,8 @@ struct ipc_namespace {

size_t shm_ctlall;

int shm_ctlmni;

int shm_tot;

+

+ struct ctl_table_header *ctl_head;

};

extern struct ipc_namespace init_ipc_ns;

diff --git a/ipc/ipc_sysctl.c b/ipc/ipc_sysctl.c

index 79e24e8..5affdfe 100644

--- a/ipc/ipc_sysctl.c

+++ b/ipc/ipc_sysctl.c

@@ -15,84 +15,6 @@

#include <linux/sysctl.h>

#include <linux/uaccess.h>

-static void *get_ipc(ctl_table *table)

{

- char *which = table->data;

- struct ipc_namespace *ipc_ns = current->nsproxy->ipc_ns;

- which = (which - (char *)&init_ipc_ns) + (char *)ipc_ns;

- return which;

}

-

-#ifdef CONFIG_PROC_FS

-static int proc_ipc_dointvec(ctl_table *table, int write, struct file *filp,

- void __user *buffer, size_t *lenp, loff_t *ppos)

{

- struct ctl_table ipc_table;

- memcpy(&ipc_table, table, sizeof(ipc_table));

- ipc_table.data = get_ipc(table);

```

-
- return proc_dointvec(&ipc_table, write, filp, buffer, lenp, ppos);
-}
-
-static int proc_ipc_doulongvec_minmax(ctl_table *table, int write,
- struct file *filp, void __user *buffer, size_t *lenp, loff_t *ppos)
-{
- struct ctl_table ipc_table;
- memcpy(&ipc_table, table, sizeof(ipc_table));
- ipc_table.data = get_ipc(table);
-
- return proc_doulongvec_minmax(&ipc_table, write, filp, buffer,
- lenp, ppos);
-}
-
-#else
-#define proc_ipc_doulongvec_minmax NULL
-#define proc_ipc_dointvec NULL
-#endif
-
-#ifdef CONFIG_SYSCTL_SYSCALL
-/* The generic sysctl ipc data routine. */
-static int sysctl_ipc_data(ctl_table *table, int __user *name, int nlen,
- void __user *oldval, size_t __user *oldlenp,
- void __user *newval, size_t newlen)
-{
- size_t len;
- void *data;
-
- /* Get out of I don't have a variable */
- if (!table->data || !table->maxlen)
- return -ENOTDIR;
-
- data = get_ipc(table);
- if (!data)
- return -ENOTDIR;
-
- if (oldval && oldlenp) {
- if (get_user(len, oldlenp))
- return -EFAULT;
- if (len) {
- if (len > table->maxlen)
- len = table->maxlen;
- if (copy_to_user(oldval, data, len))
- return -EFAULT;
- if (put_user(len, oldlenp))
- return -EFAULT;
- }
- }
- }

```

```

- }
-
- if (newval && newlen) {
-   if (newlen > table->maxlen)
-     newlen = table->maxlen;
-
-   if (copy_from_user(data, newval, newlen))
-     return -EFAULT;
- }
- return 1;
-}
-#else
-#define sysctl_ipc_data NULL
-#endif
-
static struct ctl_table ipc_kern_table[] = {
{
    .ctl_name = KERN_SHMMAX,
@@ -100,8 +22,8 @@ static struct ctl_table ipc_kern_table[] = {
    .data = &init_ipc_ns.shm_ctlmax,
    .maxlen = sizeof (init_ipc_ns.shm_ctlmax),
    .mode = 0644,
-   .proc_handler = proc_ipc_doulongvec_minmax,
-   .strategy = sysctl_ipc_data,
+   .proc_handler = proc_doulongvec_minmax,
+   .strategy = sysctl_data,
},
{
    .ctl_name = KERN_SHMALL,
@@ -109,8 +31,8 @@ static struct ctl_table ipc_kern_table[] = {
    .data = &init_ipc_ns.shm_ctlall,
    .maxlen = sizeof (init_ipc_ns.shm_ctlall),
    .mode = 0644,
-   .proc_handler = proc_ipc_doulongvec_minmax,
-   .strategy = sysctl_ipc_data,
+   .proc_handler = proc_doulongvec_minmax,
+   .strategy = sysctl_data,
},
{
    .ctl_name = KERN_SHMMNI,
@@ -118,8 +40,8 @@ static struct ctl_table ipc_kern_table[] = {
    .data = &init_ipc_ns.shm_ctlmni,
    .maxlen = sizeof (init_ipc_ns.shm_ctlmni),
    .mode = 0644,
-   .proc_handler = proc_ipc_dointvec,
-   .strategy = sysctl_ipc_data,
+   .proc_handler = proc_dointvec,
+   .strategy = sysctl_data,
},

```

```

},
{
    .ctl_name = KERN_MSGMAX,
@@ -127,8 +49,8 @@ static struct ctl_table ipc_kern_table[] = {
    .data = &init_ipc_ns.msg_ctlmax,
    .maxlen = sizeof (init_ipc_ns.msg_ctlmax),
    .mode = 0644,
-   .proc_handler = proc_ipc_dointvec,
-   .strategy = sysctl_ipc_data,
+   .proc_handler = proc_dointvec,
+   .strategy = sysctl_data,
},
{
    .ctl_name = KERN_MSGMNI,
@@ -136,8 +58,8 @@ static struct ctl_table ipc_kern_table[] = {
    .data = &init_ipc_ns.msg_ctlmni,
    .maxlen = sizeof (init_ipc_ns.msg_ctlmni),
    .mode = 0644,
-   .proc_handler = proc_ipc_dointvec,
-   .strategy = sysctl_ipc_data,
+   .proc_handler = proc_dointvec,
+   .strategy = sysctl_data,
},
{
    .ctl_name = KERN_MSGMNB,
@@ -145,8 +67,8 @@ static struct ctl_table ipc_kern_table[] = {
    .data = &init_ipc_ns.msg_ctlmnb,
    .maxlen = sizeof (init_ipc_ns.msg_ctlmnb),
    .mode = 0644,
-   .proc_handler = proc_ipc_dointvec,
-   .strategy = sysctl_ipc_data,
+   .proc_handler = proc_dointvec,
+   .strategy = sysctl_data,
},
{
    .ctl_name = KERN_SEM,
@@ -154,8 +76,8 @@ static struct ctl_table ipc_kern_table[] = {
    .data = &init_ipc_ns.sem_ctls,
    .maxlen = 4*sizeof (int),
    .mode = 0644,
-   .proc_handler = proc_ipc_dointvec,
-   .strategy = sysctl_ipc_data,
+   .proc_handler = proc_dointvec,
+   .strategy = sysctl_data,
},
};
@@ -170,9 +92,43 @@ static struct ctl_table ipc_root_table[] = {

```

```

    {}
};

+int ipc_clone_sysctl(struct ipc_namespace *ns)
+{
+ struct ctl_table_header *h;
+ struct ctl_table *t;
+
+ h = create_sysctl_shadow(init_ipc_ns.ctl_head);
+ if (h == NULL)
+ return -ENOMEM;
+
+ t = h->ctl_table->child;
+
+ t[0].data = &ns->shm_ctlmax;
+ t[1].data = &ns->shm_ctlall;
+ t[2].data = &ns->shm_ctlmni;
+ t[3].data = &ns->msg_ctlmax;
+ t[4].data = &ns->msg_ctlmni;
+ t[5].data = &ns->msg_ctlmnb;
+ t[6].data = &ns->sem_ctls;
+
+ ns->ctl_head = h;
+ return 0;
+}
+
+void ipc_free_sysctl(struct ipc_namespace *ns)
+{
+ free_sysctl_shadow(ns->ctl_head);
+}
+
+static struct ctl_table_header *ipc_sysctl_shadow(struct ctl_table_header *h)
+{
+ return current->nsproxy->ipc_ns->ctl_head;
+}
+
+static int __init ipc_sysctl_init(void)
+{
+ register_sysctl_table(ipc_root_table);
+ init_ipc_ns.ctl_head = register_sysctl_table_shadow(ipc_root_table,
+ ipc_sysctl_shadow);
+ return 0;
+}

diff --git a/ipc/util.c b/ipc/util.c
index 76c1f34..a8cb99d 100644
--- a/ipc/util.c
+++ b/ipc/util.c

```

```

@@ -61,6 +61,9 @@ static struct ipc_namespace *clone_ipc_ns(struct ipc_namespace *old_ns)
    if (ns == NULL)
        goto err_mem;

+ if (ipc_clone_sysctl(ns))
+ goto err_ctl;
+
    err = sem_init_ns(ns);
    if (err)
        goto err_sem;
@@ -79,6 +82,8 @@ err_shm:
err_msg:
    sem_exit_ns(ns);
err_sem:
+ ipc_free_sysctl(ns);
+err_ctl:
    kfree(ns);
err_mem:
    return ERR_PTR(err);
@@ -108,6 +113,7 @@ void free_ipc_ns(struct kref *kref)
    sem_exit_ns(ns);
    msg_exit_ns(ns);
    shm_exit_ns(ns);
+ ipc_free_sysctl(ns);
    kfree(ns);
}

```

```

diff --git a/ipc/util.h b/ipc/util.h
index 9ffea40..dfd53c6 100644

```

```

--- a/ipc/util.h
+++ b/ipc/util.h
@@ -28,6 +28,9 @@ void sem_exit_ns(struct ipc_namespace *ns);
void msg_exit_ns(struct ipc_namespace *ns);
void shm_exit_ns(struct ipc_namespace *ns);

+int ipc_clone_sysctl(struct ipc_namespace *ns);
+void ipc_free_sysctl(struct ipc_namespace *ns);
+
struct ipc_ids {
    int in_use;
    unsigned short seq;

```

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