

---

Subject: Re: PATCH -mm] fix create\_new\_namespaces() return value  
Posted by [Cedric Le Goater](#) on Tue, 12 Jun 2007 12:19:42 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

Badari Pulavarty wrote:

>  
>  
> Cedric Le Goater wrote:  
>  
>> The following patch modifies create\_new\_namespaces() to also use the  
>> errors returned by the copy\_\*\_ns routines and not to systematically  
>> return ENOMEM.  
>>  
>  
> In my initial version, I did same. It doesn't work :(  
>  
> copy\_\*\_ns() routines doesn't return any errors. All they return is NULL  
> in case of a  
> failure + with the exception of copy\_mnt\_ns, there are no other failure  
> cases.  
> So, there is no way to find out why the copy\_\*\_ns() routines failed from  
> create\_new\_namespaces().  
> If you really really want to do this, change all copy\_\*\_ns() routines to  
> returns meaningful  
> errors instead of NULL.

Here's a second try to apply on top of -mm branch,

Thanks for the review Badari.

C.

The following patch modifies create\_new\_namespaces() to also use the  
errors returned by the copy\_\*\_ns routines and not to systematically  
return ENOMEM.

Changes since [try #1]:

- fixed return values of clone\_\*\_ns() routines

Signed-off-by: Cedric Le Goater <clg@fr.ibm.com>

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

Cc: Badari Pulavarty <pbadari@us.ibm.com>

Cc: Pavel Emelianov <xemul@openvz.org>

Cc: Herbert Poetzl <herbert@13thfloor.at>

Cc: Eric W. Biederman <ebiederm@xmission.com>

---

fs/namespace.c | 4 ++--

```
kernel/nsproxy.c      | 23 ++++++-----
kernel/user_namespace.c | 6 +++--
kernel/utsname.c      | 10 +++++--
4 files changed, 28 insertions(+), 15 deletions(-)
```

Index: 2.6.22-rc4-mm2/kernel/nsproxy.c

=====

--- 2.6.22-rc4-mm2.orig/kernel/nsproxy.c

+++ 2.6.22-rc4-mm2/kernel/nsproxy.c

@@ -58,30 +58,41 @@ static struct nsproxy \*create\_new\_namesp

struct fs\_struct \*new\_fs)

{  
struct nsproxy \*new\_nsp;

+ int err;

new\_nsp = clone\_nsproxy(tsk->nsproxy);

if (!new\_nsp)  
return ERR\_PTR(-ENOMEM);

new\_nsp->mnt\_ns = copy\_mnt\_ns(flags, tsk->nsproxy->mnt\_ns, new\_fs);

- if (IS\_ERR(new\_nsp->mnt\_ns))

+ if (IS\_ERR(new\_nsp->mnt\_ns)) {

+ err = PTR\_ERR(new\_nsp->mnt\_ns);

goto out\_ns;

+ }

new\_nsp->uts\_ns = copy\_utsname(flags, tsk->nsproxy->uts\_ns);

- if (IS\_ERR(new\_nsp->uts\_ns))

+ if (IS\_ERR(new\_nsp->uts\_ns)) {

+ err = PTR\_ERR(new\_nsp->uts\_ns);

goto out\_uts;

+ }

new\_nsp->ipc\_ns = copy\_ipcs(flags, tsk->nsproxy->ipc\_ns);

- if (IS\_ERR(new\_nsp->ipc\_ns))

+ if (IS\_ERR(new\_nsp->ipc\_ns)) {

+ err = PTR\_ERR(new\_nsp->ipc\_ns);

goto out\_ipc;

+ }

new\_nsp->pid\_ns = copy\_pid\_ns(flags, tsk->nsproxy->pid\_ns);

- if (IS\_ERR(new\_nsp->pid\_ns))

+ if (IS\_ERR(new\_nsp->pid\_ns)) {

+ err = PTR\_ERR(new\_nsp->pid\_ns);

goto out\_pid;

+ }

new\_nsp->user\_ns = copy\_user\_ns(flags, tsk->nsproxy->user\_ns);

```

- if (IS_ERR(new_nsp->user_ns))
+ if (IS_ERR(new_nsp->user_ns)) {
+ err = PTR_ERR(new_nsp->user_ns);
+ goto out_user;
+ }

```

```

return new_nsp;

```

```

@@ -99,7 +110,7 @@ out_uts:
    put_mnt_ns(new_nsp->mnt_ns);
out_ns:
    kfree(new_nsp);
- return ERR_PTR(-ENOMEM);
+ return ERR_PTR(err);
}

```

```

/*

```

```

Index: 2.6.22-rc4-mm2/fs/namespace.c

```

```

--- 2.6.22-rc4-mm2.orig/fs/namespace.c

```

```

+++ 2.6.22-rc4-mm2/fs/namespace.c

```

```

@@ -1599,7 +1599,7 @@ static struct mnt_namespace *dup_mnt_ns(

```

```

    new_ns = kmalloc(sizeof(struct mnt_namespace), GFP_KERNEL);
    if (!new_ns)
- return NULL;
+ return ERR_PTR(-ENOMEM);

```

```

    atomic_set(&new_ns->count, 1);
    INIT_LIST_HEAD(&new_ns->list);

```

```

@@ -1613,7 +1613,7 @@ static struct mnt_namespace *dup_mnt_ns(

```

```

    if (IS_ERR(new_ns->root)) {
        up_write(&namespace_sem);
        kfree(new_ns);

```

```

- return NULL;
+ return ERR_PTR(-ENOMEM);
}

```

```

    spin_lock(&vfsmount_lock);
    list_add_tail(&new_ns->list, &new_ns->root->mnt_list);

```

```

Index: 2.6.22-rc4-mm2/kernel/user_namespace.c

```

```

--- 2.6.22-rc4-mm2.orig/kernel/user_namespace.c

```

```

+++ 2.6.22-rc4-mm2/kernel/user_namespace.c

```

```

@@ -34,7 +34,7 @@ static struct user_namespace *clone_user

```

```

    ns = kmalloc(sizeof(struct user_namespace), GFP_KERNEL);
    if (!ns)
- return NULL;

```

```

+ return ERR_PTR(-ENOMEM);

    kref_init(&ns->kref);

@@ -45,7 +45,7 @@ static struct user_namespace *clone_user
    ns->root_user = alloc_uid(ns, 0);
    if (!ns->root_user) {
        kfree(ns);
- return NULL;
+ return ERR_PTR(-ENOMEM);
    }

    /* Reset current->user with a new one */
@@ -53,7 +53,7 @@ static struct user_namespace *clone_user
    if (!new_user) {
        free_uid(ns->root_user);
        kfree(ns);
- return NULL;
+ return ERR_PTR(-ENOMEM);
    }

```

```

    switch_uid(new_user);

```

Index: 2.6.22-rc4-mm2/kernel/utsname.c

```

=====
--- 2.6.22-rc4-mm2.orig/kernel/utsname.c
+++ 2.6.22-rc4-mm2/kernel/utsname.c
@@ -13,6 +13,7 @@
#include <linux/uts.h>
#include <linux/utsname.h>
#include <linux/version.h>
+#include <linux/err.h>

/*
 * Clone a new ns copying an original utsname, setting refcount to 1
@@ -24,10 +25,11 @@ static struct uts_namespace *clone_uts_n
    struct uts_namespace *ns;

    ns = kmalloc(sizeof(struct uts_namespace), GFP_KERNEL);
- if (ns) {
-     memcpy(&ns->name, &old_ns->name, sizeof(ns->name));
-     kref_init(&ns->kref);
- }
+ if (!ns)
+     return ERR_PTR(-ENOMEM);
+
+     memcpy(&ns->name, &old_ns->name, sizeof(ns->name));
+     kref_init(&ns->kref);
    return ns;

```

}

---

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

---