

---

Subject: Re: PATCH -mm] fix create\_new\_namespaces() return value

Posted by [Badari Pulavarty](#) on Mon, 11 Jun 2007 16:22:52 GMT

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

---

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.

>  
>  
>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>  
>---  
> kernel/nsproxy.c | 23 ++++++-----  
> 1 file changed, 17 insertions(+), 6 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);
>
Hmm.. copy_user_ns() ? I don't see this in rc4-mm2.

```

Thanks,  
Badari

---

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

---



---

Subject: Re: PATCH -mm] fix create\_new\_namespaces() return value

Posted by [Cedric Le Goater](#) on Tue, 12 Jun 2007 07:49:10 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.
```

mnt ant uts namespaces return NULL right but this is not true for ipc namespace which returns -ENOMEM. user namespace returns -EINVAL if CONFIG\_USER\_NS is not selected. I expect pid namespace to do same and net namespace to do even more.

```
> 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.
```

I will certainly do and change mnt ant uts namespaces to return -ENOMEM but we will still require the patch below.

```
>> 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>
>> ---
>> kernel/nsproxy.c | 23 ++++++-----+
>> 1 file changed, 17 insertions(+), 6 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);
>>
>>     }
>>
> Hmm.. copy_user_ns() ? I don't see this in rc4-mm2.

```

it's in the -mm stack.

C.

---

Containers mailing list  
 Containers@lists.linux-foundation.org

---

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](mailto:clg@fr.ibm.com)>

Cc: Serge E. Hallyn <[serue@us.ibm.com](mailto:serue@us.ibm.com)>

Cc: Badari Pulavarty <[pbadari@us.ibm.com](mailto:pbadari@us.ibm.com)>

Cc: Pavel Emelianov <[xemul@openvz.org](mailto:xemul@openvz.org)>

Cc: Herbert Poetzl <[herbert@13thfloor.at](mailto: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>

---