
Subject: Re: [RFC][PATCH 4/4]: Enable cloning PTY namespaces
Posted by [Pavel Emelianov](#) on Wed, 06 Feb 2008 16:35:18 GMT
[View Forum Message](#) <> [Reply to Message](#)

Serge E. Hallyn wrote:

> Quoting Pavel Emelianov (xemul@openvz.org):

>> Serge E. Hallyn wrote:

>>> Quoting Pavel Emelianov (xemul@openvz.org):

>>>> sukadev@us.ibm.com wrote:

>>>>> From: Sukadev Bhattiprolu <sukadev@us.ibm.com>

>>>>> Subject: [RFC][PATCH 4/4]: Enable cloning PTY namespaces

>>>>>

>>>>> Enable cloning PTY namespaces.

>>>>>

>>>>> TODO:

>>>>> This version temporarily uses the clone flag '0x80000000' which

>>>>> is unused in mainline atm, but used for CLONE_IO in -mm.

>>>>> While we must extend clone() (urgently) to solve this, it hopefully

>>>>> does not affect review of the rest of this patchset.

>>>>>

>>>>> Changelog:

>>>>> - Version 0: Based on earlier versions from Serge Hallyn and

>>>>> Matt Helsley.

>>>>>

>>>>> Signed-off-by: Sukadev Bhattiprolu <sukadev@us.ibm.com>

>>>>> ---

>>>>> fs/devpts/inode.c | 84 ++++++

>>>>> include/linux/devpts_fs.h | 52 ++++++

>>>>> include/linux/init_task.h | 1

>>>>> include/linux/nsproxy.h | 2 +

>>>>> include/linux/sched.h | 2 +

>>>>> kernel/fork.c | 2 -

>>>>> kernel/nsproxy.c | 17 ++++++-

>>>>> 7 files changed, 146 insertions(+), 14 deletions(-)

>>>>>

>>>>> Index: linux-2.6.24/fs/devpts/inode.c

>>>>> =====

>>>>> --- linux-2.6.24.orig/fs/devpts/inode.c 2008-02-05 19:16:39.000000000 -0800

>>>>> +++ linux-2.6.24/fs/devpts/inode.c 2008-02-05 20:27:41.000000000 -0800

>>>>> @@ -25,18 +25,25 @@

>>>>> #define DEVPTS_SUPER_MAGIC 0x1cd1

>>>>>

>>>>> extern int pty_limit; /* Config limit on Unix98 ptys */

>>>>> -static DEFINE_IDR(allocated_ptys);

>>>>> static DECLARE_MUTEX(allocated_ptys_lock);

>>>>> +static struct file_system_type devpts_fs_type;

>>>>> +

>>>>> +struct pts_namespace init_pts_ns = {

```

>>>>> + .kref = {
>>>>> + .refcount = ATOMIC_INIT(2),
>>>>> + },
>>>>> + .allocated_ptys = IDR_INIT(init_pts_ns.allocated_ptys),
>>>>> + .mnt = NULL,
>>>>> +};
>>>>>
>>>>> static inline struct idr *current_pts_ns_allocated_ptys(void)
>>>>> {
>>>>> - return &allocated_ptys;
>>>>> + return &current->nsproxy->pts_ns->allocated_ptys;
>>>>> }
>>>>>
>>>>> -static struct vfsmount *devpts_mnt;
>>>>> static inline struct vfsmount *current_pts_ns_mnt(void)
>>>>> {
>>>>> - return devpts_mnt;
>>>>> + return current->nsproxy->pts_ns->mnt;
>>>>> }
>>>>>
>>>>> static struct {
>>>>> @@ -59,6 +66,42 @@ static match_table_t tokens = {
>>>>> {Opt_err, NULL}
>>>>> };
>>>>>
>>>>> +struct pts_namespace *new_pts_ns(void)
>>>>> +{
>>>>> + struct pts_namespace *ns;
>>>>> +
>>>>> + ns = kmalloc(sizeof(*ns), GFP_KERNEL);
>>>>> + if (!ns)
>>>>> + return ERR_PTR(-ENOMEM);
>>>>> +
>>>>> + ns->mnt = kern_mount_data(&devpts_fs_type, ns);
>>>>> You create a circular references here - the namespace
>>>>> holds the vfsmnt, the vfsmnt holds a superblock, a superblock
>>>>> holds the namespace.
>>>>> Hmm, yeah, good point. That was probably in my original version last
>>>>> year, so my fault not Suka's. Suka, would it work to have the
>>>>> sb->s_info point to the namespace but not grab a reference, than have
>>>>> If you don't then you may be in situation, when this devpts
>>>>> is mounted from userspace and in case the namespace is dead
>>>>> superblock will point to garbage... Superblock MUST hold the
>>>>> namespace :)
>>>>>
>>>>> > But when the ns is freed sb->s_info would be NULL. Surely the helpers
>>>>> > can be made to handle that safely?

```

Hm... How do we find the proper superblock? Have a reference on it from the namespace? I'm afraid it will be easy to resolve the locking issues here.

I propose another scheme - we simply don't have ANY references from namespace to superblock/vfsmount, but get the current namespace in `devpts_get_sb()` and put in `devpts_free_sb()`.

```
>>> free_pts_ns() null out its sb->s_info, i.e. something like
>>>
>>> void free_pts_ns(struct kref *ns_kref)
>>> {
>>>     struct pts_namespace *ns;
>>>     struct super_block *sb;
>>>
>>>     ns = container_of(ns_kref, struct pts_namespace, kref);
>>>     BUG_ON(ns == &init_pts_ns);
>>>     sb = ns->mnt->mnt_sb;
>>>
>>>     mntput(ns->mnt);
>>>     sb->s_info = NULL;
>>>
>>>     /*
>>>      * TODO:
>>>      *     idr_remove_all(&ns->allocated_ptys); introduced in
>>>      *     .6.23
>>>      */
>>>     idr_destroy(&ns->allocated_ptys);
>>>     kfree(ns);
>>> }
>>>
>>>
>
```

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