

```
>>>>>
>>>>> +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 vfmnt, the vfmnt 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().
```

I've choosen another path in mq_ns.

I also don't take any refcount on superblock/vfsmount of the new mq_ns bc of the circular ref. I've considered that namespaces only apply to processes : the refcount of a namespace is incremented each time a new task is cloned and the namespace (in my case mq_ns) is released when the last tasks exists. But this becomes an issue with user mounts which survives task death. you end up having a user mount pointing to a bogus mq_ns.

unless you require to have CLONE_NEWNS at the sametime.

Now, this CLONE_NEWNS enforcement seems to be an issue with bind mount.

... jumping to the other thread :)

C.

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