Subject: Re: [RFC][PATCH 3/4]: Enable multiple mounts of /dev/pts
Posted by serue on Thu, 07 Feb 2008 14:22:35 GMT
View Forum Message <> Reply to Message

```
Quoting Pavel Emelyanov (xemul@openvz.org):
> Serge E. Hallyn wrote:
> > Quoting Pavel Emelyanov (xemul@openvz.org):
> >> [snip]
> >>
> >>>> Mmm. I wanted to send one small objection to Cedric's patches with mqns,
> >>>> but the thread was abandoned by the time I decided to do-it-right-now.
> >>>>
> >>>> So I can put it here: forcing the CLONE_NEWNS is not very good, since
> >>>> this makes impossible to push a bind mount inside a new namespace, which
> >>>> may operate in some chroot environment. But this ability is heavily
> >>> Which direction do you want to go?  I'm wondering whether mounts
> >>> propagation can address it.
> >> Hardly. AFAIS there's no way to let the chroot-ed tasks see parts of
> >> vfs tree, that left behind them after chroot, unless they are in the
> >> same mntns as you, and you bind mount this parts to their tree. No?
> >
> > Well no, but I suspect I'm just not understanding what you want to do.
> > But if the chroot is under /jail1, and you've done, say,
> >
> >  mkdir -p /share/pts
> >  mkdir -p /jail1/share
> >  mount --bind /share /share
> >  mount --make-shared /share
> >  mount --bind /share /jail1/share
> >  mount --make-slave /jail1/share
> >
> > before the chroot-ed tasks were cloned with CLONE_NEWNS, then when you
> > do
> >
> >  mount --bind /dev/pts /share/pts
> >
> > from the parent mntns (not that I know why you'd want to do *that* :)
> > then the chroot'ed tasks will see the original mntns's /dev/pts under
> > /jail1/share.
>
> I haven't yet tried that, but :( this function
>
>  static inline int check_mnt(struct vfsmount *mnt)
>  {
>         return mnt->mnt_ns == current->nsproxy->mnt_ns;
>  }
>
> and this code in do_loopback
```

>
>   if (!check_mnt(nd->mnt) || !check_mnt(old_nd.mnt))
>   goto out;
>
> makes me think that trying to bind a mount from another mntns
> ot _to_ another is prohibited... Do I miss something?

That's used at the top of explicit mounting paths, so if you found a way
to access a nameidata in the other mnt_ns and tried to mount /dev/pts
straight onto that nd this check would cause it to fail. But what I
described above mounts onto /share/pts, which is in the same ns. Then
the mouts propagation code in fs/pnode.c forwards the mount into the
other namespace.

Still I suspect I wasn't quite thinking right. If the target task had
already umounted /dev/pts and remounted it, there would be nothing to
forward your bind mount to and so nothing would happen.

Still that's moot :) Either we should find a way to get rid of the
CLONE_NEWNS requirement, or we should provide a cgroup to access
/dev/pts under the cgroup file tree.

> >>> Though really, I think you're right - we shouldn't break the kernel
> >>> doing CLONE_NEWMQ or CLONE_NEWPTS without CLONE_NEWNS, so we shouldn't
> >>> force the combination.
> >>>
> >>>> exploited in OpenVZ, so if we can somehow avoid forcing the NEWNS flag
> >>>> that would be very very good :) See my next comment about this issue.
> >>>>
> >>>>> Pavel, not long ago you said you were starting to look at tty and pty
> >>>>> stuff - did you have any different ideas on devpts virtualization, or
> >>>>> are you ok with this minus your comments thus far?
> >>>> I have a similar idea of how to implement this, but I didn't thought
> >>>> about the details. As far as this issue is concerned, I see no reasons
> >>>> why we need a kern_mount-ed devtpsfs instance. If we don't make such,
> >>>> we may safely hold the ptsns from the superblock and be happy. The
> >>>> same seems applicable to the mqns, no?
> >>> But the current->nsproxy->devpts->mnt is used in several functions in
> >>> patch 3.
> >> Indeed. I overlooked this. Then we're in a deep ... problem here.
> >>
> >> Breaking this circle was not that easy with pid namespaces, so
> >> I put the strut in proc_flush_task - when the last task from the
> >> namespace exits the kern-mount-ed vfsmnt is dropped, but we can't
> >> do the same stuff with devpts.
> >
> > But I still don't see what the problem is with my proposal? So long as
> > you agree that if there are no tasks remaining in the devptsns,

> > then any task which has its devpts mounted should see an empty directory
> > (due to sb->s_info being NULL), I think it works.
>
> Well, if we _do_ can handle the races with ns->devpts_mnt switch
> from not NULL to NULL, then I'm fine with this approach.
>
> I just remember, that with pid namespaces this caused a complicated
> locking and performance degradation. This is the problem I couldn't
> remember yesterday.

Yeah it sure seems like there must be some gotcha in there somewhere...

> >> I do not remember now what the problem was and it's already quite
> >> late in Moscow, so if you don't mind I'll revisit the issue tomorrow.
> >
> > Ok, that's fine. I'll let it sit until then too :) Good night.
> >
> >> Off-topic: does any of you know whether Andrew is willing to accept
> >> new features in the nearest future? The problem is that I have a
> >> device visibility controller fixed and pending to send, but I can't
> >> guess a good time for it :)
> >
> > Well even if Andrew won't take it I'd like to see it, so I'd appreciate
> > a resend.
> >
> >>>> The reason I have the kern_mount-ed instance of proc for pid namespaces
> >>>> is that I need a vfsmount to flush task entries from, but allowing
> >>>> it to be NULL (i.e. no kern_mount, but optional user mounts) means
> >>>> handing all the possible races, which is too heavy. But do we actually
> >>>> need the vfsmount for devpts and mqns if no user-space mounts exist?
> >>>>
> >>>> Besides, I planned to include legacy ptys virtualization and console
> >>>> virtualizatin in this namespace, but it seems, that it is not present
> >>>> in this particular one.
> >>> I had been thinking the consoles would have their own ns, since there's
> >>> really nothing linking them, but there really is no good reason why
> >>> userspace should ever want them separate. So I'm fine with combining
> >>> them.
> >> OK.
> >