
Subject: Re: Re: [patch 05/10] add "permit user mounts in new namespace" clone flag

Posted by [Ram Pai](#) on Wed, 18 Apr 2007 18:35:19 GMT

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

On Wed, 2007-04-18 at 11:19 +0200, Miklos Szeredi wrote:

> > > Allowing this and other flags to NOT be propagated just makes it
> > > possible to have a set of shared mounts with asymmetric properties,
> > > which may actually be desirable.
> >
> > The shared mount feature was designed to ensure that the mount remained
> > identical at all the locations.
>
> OK, so remount not propagating mount flags is a bug then?

As I said earlier, are there any flags currently that if not propagated can lead to conflicts with the shared subtree semantics? I am not aware of any. If you did notice a case, than according to me its a bug.

But the new proposed 'allow unprivileged mounts' flag; if not propagated among peers (and slaves) of a shared mount can lead to conflicts with shared subtree semantics. Since mount in one shared-mount; when propagated to its peer fails to mount and hence lead to un-identical peers.

>
> > Now designing features to make it un-identical but still naming it
> > shared, will break its original purpose. Slave mounts were designed
> > to make it asymmetric.
>
> What if I want to modify flags in a master mount, but not the slave
> mount? Would I be screwed? For example: mount is read-only in both
> master and slave. I want to mark it read-write in master but not in
> slave. What do I do?

Making mounts read-only or read-write -- will that effect mount propagation in such a way that future mounts in any one of the peers will not be able to propagate that mount to its peers or slaves?

I don't think it will. Hence its ok to selectively mark some mounts read-only and some mounts read-write.

However with the introduction of unprivileged mount semantics, there can be cases where a user has privileges to mount at one location but not at a different location. if these two location happen to share a peer-relationship than I see a case of interference of read-write

flag semantics with shared subtree semantics. And hence we will end up propagating the read-write flag too or have to craft a different semantics that stays consistent.

- >
- > > Whatever feature that is desired to be exploited; can that be exploited
- > > with the current set of semantics that we have? Is there a real need to
- > > make the mounts asymmetric but at the same time name them as shared?
- > > Maybe I dont understand what the desired application is?
- >
- > I do think this question of propagating mount flags is totally
- > independent of user mounts.
- >
- > As it stands, currently remount doesn't propagate mount flags, and I
- > don't see any compelling reasons why it should.
- >
- > The patchset introduces a new mount flag "allowusermnt", but I don't
- > see any compelling reason to propagate this flag `_either_`.
- >
- > Please say so if you do have such a reason. As I've explained, having
- > this flag set differently in parts of a propagation tree does not
- > interfere with or break propagation in any way.

As I said earlier, I see a case where two mounts that are peers of each other can become un-identical if we dont propagate the "allowusermnt".

As a practical example.

`/tmp` and `/mnt` are peers of each other.
`/tmp` has its "allowusermnt" flag set, which has not been propagated to `/mnt`.

now a normal-user mounts an ext2 file system under `/tmp` at `/tmp/1`

unfortunately the mount wont appear under `/mnt/1`

and this breaks the shared-subtree semantics which promises: whatever is mounted under `/tmp` will also be visible under `/mnt`

and in case if you allow the mount to appear under `/mnt/1`, you will break unpriviledge mounts semantics which promises: a normal user will not be able to mount at a location that does not allow user-mounts.

RP

>
> Miklos
>
