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

Posted by [ebiederm](#) on Wed, 18 Apr 2007 17:14:57 GMT

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

Miklos Szeredi <miklos@szeredi.hu> writes:

>> > I've tried to make this unprivileged mount thing as simple as
>> > possible, and no simpler. If we can make it even simpler, all the
>> > better.
>>
>> We are certainly much more complex than the code in plan9 (just
>> read through it) so I think we have room for improvement.
>>
>> Just for reference what I saw in plan 9 was:
>> - No super user checks in its mount, unmount, or namespace creation paths.
>> - A flag to deny new mounts but not new bind mounts (for administrative
> purposes
>> the comment said).
>>
>> Our differences from plan9.
>> - suid capable binaries. (SUID please go away).
>> - A history of programs assuming only root could call mount/unmount.
>
> I hate suid as well. _The_ motivation behind this patchset was to get
> rid of "fusermount", a suid mount helper for fuse.
>
> But I don't think suid is going away, and definitely not overnight.

Agreed, unless it just happens that killing is equally as hard as doing non-privileged mounts. Which I really don't think will be the case. suid executables can always be replaced by a non-privileged part that connect to a daemon on localhost.

> Also I don't think we want to require auditing userspace before
> enabling user mounts.

Given the complexity of the code to avoid audits adds, and especially the uncertainty of how all of the pieces add up together I really think we do need to audit user space (but only in a limited way).

This is a paradigm shift in how mounts are managed and to get the full benefit of it we need to be prepared to deal with the paradigm shift.

Essentially what the audit must ensure is that

(a) non-root users are always run in a non-default namespace so mounts and unmounts they generate will not have global effect.

(b) If we setup something like the proposed /share that administrative tools know how to treat it properly.

That is not an especially hard audit of user space and it noticeably reduces the complexity of what we must implement.

> If I understand correctly, your proposal is to get rid of MNT_USER and
> MNT_ALLOWUSERMNT and allow/deny unprivileged mounts and umounts based
> on a boolean sysctl flag and on a check if the target namespace is the
> initial namespace or not.

No. I really do not want to treat the initial namespace in a special way from an implementation point of view.

>From a distro point of view I don't think we should ever allow a user into the initial mount namespace, and that is what we should audit. All of the ways a user can login to a machine.

We need to audit the login paths anyway if we are going to do anything interesting with the mount namespace. So I see this as no real additional overhead to make things usable.

> And maybe add some extra checks which
> prevent ugliness from happening with suid programs. Is this correct?

Definitely add some extra permission checks to prevent ugliness from happening with suid executables. In the general case the problem with suid executables is that they expect parts of the mount namespace that a user cannot modify not to be modified.

Ensuring that our permission checks keep that promise for mount and umount is going to be a bit challenging, because we need to think through a lot of scenarios. But it is not that hard.

> If so, how are we going to make sure this won't break existing
> userspace without doing a full audit of all suid programs in every
> distro that wants this feature?

The permission checks to ensure you can not modify a filesystem with mounts in a way that you could not modify it by creating or deleting files should be enough.

That probably means that we are restricted to mounting filesystems with the equivalent of `uid=$(id -u) gid=$(id -g)`. That is if you aren't root all files in the filesystem you cause to be mounted must be owned by you. That way you have permission to unmount or delete them.

At the very least the user who causes a mount to be created should be the owner and have complete control over the directory mount point. So that they can at least theoretically remove all of the files and directories at the mount point (which would be equivalent to unmounting it).

> Also how are we going to prevent the user from creating millions of
> mounts, and using up all the kernel memory for vfsmounts?

I definitely agree that we need some kind of accounting. I don't think we can do this per user (which would be nice) and I don't believe your code does attempts to limit mounts by user either. A simple global limit on the number of mounts should be sufficient. If necessary we can allow the limit to be ignored if you have the appropriate capability.

Eric

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