Subject: Re: [patch 05/10] add "permit user mounts in new namespace" clone flag
Posted by serue on Thu, 12 Apr 2007 20:32:08 GMT
View Forum Message <> Reply to Message

Quoting Miklos Szeredi (miklos@szeredi.hu):

```
> From: Miklos Szeredi <mszeredi@suse.cz>
> 
> If CLONE_NEWNS and CLONE_NEWNS_USERMNT are given to clone(2) or
> unshare(2), then allow user mounts within the new namespace.
> 
> This is not flexible enough, because user mounts can't be enabled for
> the initial namespace.
> 
> The remaining clone bits also getting dangerously few...
> 
> Alternatives are:
> 
>   - prctl() flag
>   - setting through the containers filesystem
```

Sorry, I know I had mentioned it, but this is definately my least
favorite approach.

Curious whether are any other suggestions/opinions from the containers
list?

thanks,
-serge

```
> Signed-off-by: Miklos Szeredi <mszeredi@suse.cz>
> ---
> 
> Index: linux/fs/namespace.c
> ===================================================================
> --- linux.orig/fs/namespace.c 2007-04-12 13:46:19.000000000 +0200
> +++ linux/fs/namespace.c 2007-04-12 13:54:36.000000000 +0200
> @@ -1617,6 +1617,8 @@ struct mnt_namespace *copy_mnt_ns(int fl
>    return ns;
> 
>   new_ns = dup_mnt_ns(ns, new_fs);
> + if (new_ns && (flags & CLONE_NEWNS_USERMNT))
> +  new_ns->flags |= MNT_NS_PERMIT_USERMOUNTS;
> 
>   put_mnt_ns(ns);
>   return new_ns;
> Index: linux/include/linux/sched.h
> ===================================================================
> --- linux.orig/include/linux/sched.h 2007-04-12 13:26:48.000000000 +0200
> +++ linux/include/linux/sched.h 2007-04-12 13:54:36.000000000 +0200
> @@ -26,6 +26,7 @@
>  #define CLONE_STOPPED  0x02000000 /* Start in stopped state */
>  #define CLONE_NEWUTS  0x04000000 /* New utsname group? */
>  #define CLONE_NEWIPC  0x08000000 /* New ipcs */
> +#define CLONE_NEWNS_USERMNT 0x10000000 /* Allow user mounts in ns? */
>
>  /*
>   * Scheduling policies
> Index: linux/kernel/fork.c
> ===================================================================
> --- linux.orig/kernel/fork.c 2007-04-11 18:27:46.000000000 +0200
> +++ linux/kernel/fork.c 2007-04-12 13:59:10.000000000 +0200
> @@ -1586,7 +1586,7 @@ asmlinkage long sys_unshare(unsigned lon
>   err = -EINVAL;
>   if (unshare_flags & ~(CLONE_THREAD|CLONE_FS|CLONE_NEWNS|CLONE_SIGHAND|
>      CLONE_VM|CLONE_FILES|CLONE_SYSVSEM|
> -    CLONE_NEWUTS|CLONE_NEWIPC))
> +    CLONE_NEWUTS|CLONE_NEWIPC|CLONE_NEWNS_USERMNT))
>    goto bad_unshare_out;
>
>   if ((err = unshare_thread(unshare_flags)))
>
> --
> -
```

> To unsubscribe from this list: send the line "unsubscribe linux-fsdevel" in
> the body of a message to majordomo@vger.kernel.org
> More majordomo info at http://vger.kernel.org/majordomo-info.html

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

Subject: Re: [patch 05/10] add "permit user mounts in new namespace" clone flag
Posted by Herbert Poetzl on Fri, 13 Apr 2007 04:16:12 GMT

View Forum Message <> Reply to Message

On Thu, Apr 12, 2007 at 03:32:08PM -0500, Serge E. Hallyn wrote:

> Quoting Miklos Szeredi (miklos@szeredi.hu):
> > From: Miklos Szeredi <mszeredi@suse.cz>
> >
> > If CLONE_NEWNS and CLONE_NEWNS_USERMNT are given to clone(2) or
> > unshare(2), then allow user mounts within the new namespace.
>
> > This is not flexible enough, because user mounts can't be enabled for
> > the initial namespace.
> >

> > The remaining clone bits also getting dangerously few...

ATM I think we do not have that many CLONE flags
available, so that this feature will have to wait
for a clone2/64 or similar ...

> > Alternatives are:
> >
> >   - prctl() flag
> >   - setting through the containers filesystem

> Sorry, I know I had mentioned it, but this is definately my least
> favorite approach.
>
> Curious whether are any other suggestions/opinions from the containers
> list?

question: how is mounting filesystems (loopback,
fuse, etc) secured in such way that the user
cannot 'create' device nodes with 'unfortunate'
permissions?

TIA,
Herbert

> thanks,
> -serge
>
```
> > Signed-off-by: Miklos Szeredi <mszeredi@suse.cz>
> > ---
> >
> > Index: linux/fs/namespace.c
> > ===================================================================
> > --- linux.orig/fs/namespace.c 2007-04-12 13:46:19.000000000 +0200
> > +++ linux/fs/namespace.c 2007-04-12 13:54:36.000000000 +0200
> > @@ -1617,6 +1617,8 @@ struct mnt_namespace *copy_mnt_ns(int fl
> >    return ns;
> >
> >   new_ns = dup_mnt_ns(ns, new_fs);
> > + if (new_ns && (flags & CLONE_NEWNS_USERMNT))
> > +  new_ns->flags |= MNT_NS_PERMIT_USERMOUNTS;
> >
> >   put_mnt_ns(ns);
> >   return new_ns;
> > Index: linux/include/linux/sched.h
> > ===================================================================
> > --- linux.orig/include/linux/sched.h 2007-04-12 13:26:48.000000000 +0200
> > +++ linux/include/linux/sched.h 2007-04-12 13:54:36.000000000 +0200
> > @@ -26,6 +26,7 @@
> >  #define CLONE_STOPPED  0x02000000 /* Start in stopped state */
> >  #define CLONE_NEWUTS  0x04000000 /* New utsname group? */
> >  #define CLONE_NEWIPC  0x08000000 /* New ipcs */
> > +#define CLONE_NEWNS_USERMNT 0x10000000 /* Allow user mounts in ns? */
> >
> >  /*
> >   * Scheduling policies
> > Index: linux/kernel/fork.c
> > ===================================================================
> > --- linux.orig/kernel/fork.c 2007-04-11 18:27:46.000000000 +0200
> > +++ linux/kernel/fork.c 2007-04-12 13:59:10.000000000 +0200
> > @@ -1586,7 +1586,7 @@ asmlinkage long sys_unshare(unsigned lon
> >   err = -EINVAL;
> >   if (unshare_flags & ~(CLONE_THREAD|CLONE_FS|CLONE_NEWNS|CLONE_SIGHAND|
> >      CLONE_VM|CLONE_FILES|CLONE_SYSVSEM|
> > -    CLONE_NEWUTS|CLONE_NEWIPC))
> > +    CLONE_NEWUTS|CLONE_NEWIPC|CLONE_NEWNS_USERMNT))
> >    goto bad_unshare_out;
> >
> >   if ((err = unshare_thread(unshare_flags)))
> >
> > --
> > -
> > To unsubscribe from this list: send the line "unsubscribe linux-fsdevel" in
> > the body of a message to majordomo@vger.kernel.org
> > More majordomo info at  http://vger.kernel.org/majordomo-info.html
> _______________________________________________
> Containers mailing list
> Containers@lists.linux-foundation.org
> https://lists.linux-foundation.org/mailman/listinfo/containers
```

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

Subject: Re: [patch 05/10] add "permit user mounts in new namespace" clone flag
Posted by ebiederm on Fri, 13 Apr 2007 04:45:33 GMT

View Forum Message <> Reply to Message

"Serge E. Hallyn" <serue@us.ibm.com> writes:

```
> Quoting Miklos Szeredi (miklos@szeredi.hu):
>> From: Miklos Szeredi <mszeredi@suse.cz>
>>
>> If CLONE_NEWNS and CLONE_NEWNS_USERMNT are given to clone(2) or
>> unshare(2), then allow user mounts within the new namespace.
```

>>
>> This is not flexible enough, because user mounts can't be enabled for
>> the initial namespace.
>>
>> The remaining clone bits also getting dangerously few...
>>
>> Alternatives are:
>>
>>   - prctl() flag
>>   - setting through the containers filesystem
>
> Sorry, I know I had mentioned it, but this is definately my least
> favorite approach.
>
> Curious whether are any other suggestions/opinions from the containers
> list?

Given the existence of shared subtrees allowing/denying this at the mount
namespace level is silly and wrong.

If we need more than just the filesystem permission checks can we
make it a mount flag settable with mount and remount that allows
non-privileged users the ability to create mount points under it
in directories they have full read/write access to.

I don't like the use of clone flags for this purpose but in this
case the shared subtress are a much more fundamental reasons for not
doing this at the namespace level.

Eric

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

Subject: Re: [patch 05/10] add "permit user mounts in new namespace" clone flag
Posted by Miklos Szeredi on Fri, 13 Apr 2007 07:09:54 GMT
View Forum Message <> Reply to Message

> question: how is mounting filesystems (loopback,
> fuse, etc) secured in such way that the user
> cannot 'create' device nodes with 'unfortunate'

> permissions?

All unprivileged mounts have "nosuid,nodev" added to their options.

Miklos

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

Subject: Re: [patch 05/10] add "permit user mounts in new namespace" clone flag
Posted by Miklos Szeredi on Fri, 13 Apr 2007 07:12:07 GMT
View Forum Message <> Reply to Message

> Given the existence of shared subtrees allowing/denying this at the mount
> namespace level is silly and wrong.
>
> If we need more than just the filesystem permission checks can we
> make it a mount flag settable with mount and remount that allows
> non-privileged users the ability to create mount points under it
> in directories they have full read/write access to.

OK, that makes sense.

> I don't like the use of clone flags for this purpose but in this
> case the shared subtress are a much more fundamental reasons for not
> doing this at the namespace level.

I'll drop the clone flag, and add a mount flag instead.

Thanks,
Miklos

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

Subject: Re: Re: [patch 05/10] add "permit user mounts in new namespace" clone flag
Posted by ebiederm on Mon, 16 Apr 2007 19:16:48 GMT
View Forum Message <> Reply to Message

Miklos Szeredi <miklos@szeredi.hu> writes:

>> > That depends. Current patches check the "unprivileged submounts

>> > allowed under this mount" flag only on the requested mount and not on
>> > the propagated mounts. Do you see a problem with this?
>>
>> I think privileges of this sort should propagate. If I read what you
>> just said correctly if I have a private mount namespace I won't be able
>> to mount anything unless when it was setup the unprivileged submount
>> command was explicitly set.
>
> By design yes. Why is that a problem?

It certainly doesn't match my intuition.

Why are directory permissions not sufficient to allow/deny non-priveleged mounts?
I don't understand that contention yet.

I should probably go back and look and see how plan9 handles mount/unmount
permissions. Plan9 gets away with a lot more because it doesn't have
a suid bit and mount namespaces were always present, so they don't have
backwards compatibility problems.

My best guess at the moment is that plan9 treated mount/unmount as
completely unprivileged and used the mount namespaces to limit the
scope of what would be affected by a mount/unmount operation. I think
that may be reasonable in linux as well but it will require the
presence of a mount namespace to limit the affects of what a user can
do.

So short of a more thorough audit I believe the final semantics should
be:
- mount/unmount for non-priveleged processes should only be limited
  by the mount namespace and directory permissions.
- CLONE_NEWNS should not be a privileged operation.

What prevents us from allowing these things?

- Unprivileged CLONE_NEWNS and unprivileged mounts needs resource
  accounting so we don't have a denial of service attack.

- Unprivileged mounts must be limited to directories that we have
  permission to modify in a way that we could get the same effect
  as the mount or unmount operation in terms of what files are visible
  otherwise we can mess up SUID executables.

- Anything else?

There are user space issues such as a reasonable pam module and how
to do backups. However those are user space issues.

What am I missing that requires us to add MNT_USER and MNT_USERMNT?

Eric

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

Subject: Re: Re: [patch 05/10] add "permit user mounts in new namespace" clone flag
Posted by serue on Mon, 16 Apr 2007 19:56:52 GMT

View Forum Message <> Reply to Message

Quoting Eric W. Biederman (ebiederm@xmission.com):
> Miklos Szeredi <miklos@szeredi.hu> writes:
>
> >> > That depends. Current patches check the "unprivileged submounts
> >> > allowed under this mount" flag only on the requested mount and not on
> >> > the propagated mounts. Do you see a problem with this?
> >>
> >> I think privileges of this sort should propagate. If I read what you
> >> just said correctly if I have a private mount namespace I won't be able
> >> to mount anything unless when it was setup the unprivileged submount
> >> command was explicitly set.
> >
> > By design yes. Why is that a problem?
>
> It certainly doesn't match my intuition.
>
> Why are directory permissions not sufficient to allow/deny non-priveleged mounts?
> I don't understand that contention yet.

The same scenarios laid out previously in this thread. I.e.

1. user hallyn does mount --bind / /home/hallyn/root
2. (...)
3. admin does "deluser hallyn"

and deluser starts wiping out root

Or,

1. user hallyn does mount --bind / /home/hallyn/root
2. backup daemon starts backing up /home/hallyn/root/home/hallyn/root/home...

So we started down the path of forcing users to clone a new namespace
before doing users mounts, which is what the clone flag was about. Using

per-mount flags also suffices as you had pointed out, which is being
done here. But directory permissions are inadequate.

(Unless you want to tackle each problem legacy tool one at a time to
remove problems - i.e. deluser should umount everything under
/home/hallyn before deleting, backup should be spawned from it's own
namespace cloned right after boot or just back up on one filesystem,
etc.)

-serge

> I should probably go back and look and see how plan9 handles mount/unmount
> permissions. Plan9 gets away with a lot more because it doesn't have
> a suid bit and mount namespaces were always present, so they don't have
> backwards compatibility problems.
>
> My best guess at the moment is that plan9 treated mount/unmount as
> completely unprivileged and used the mount namespaces to limit the
> scope of what would be affected by a mount/unmount operation. I think
> that may be reasonable in linux as well but it will require the
> presence of a mount namespace to limit the affects of what a user can
> do.
>
> So short of a more thorough audit I believe the final semantics should
> be:
> - mount/unmount for non-priveleged processes should only be limited
>   by the mount namespace and directory permissions.
> - CLONE_NEWNS should not be a privileged operation.
>
> What prevents us from allowing these things?
>
> - Unprivileged CLONE_NEWNS and unprivileged mounts needs resource
>   accounting so we don't have a denial of service attack.
>
> - Unprivileged mounts must be limited to directories that we have
>   permission to modify in a way that we could get the same effect
>   as the mount or unmount operation in terms of what files are visible
>   otherwise we can mess up SUID executables.
>
> - Anything else?
>
> There are user space issues such as a reasonable pam module and how
> to do backups. However those are user space issues.
>
> What am I missing that requires us to add MNT_USER and MNT_USERMNT?
>
> Eric
> -

> To unsubscribe from this list: send the line "unsubscribe linux-fsdevel" in
> the body of a message to majordomo@vger.kernel.org
> More majordomo info at http://vger.kernel.org/majordomo-info.html

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

Subject: Re: Re: [patch 05/10] add "permit user mounts in new namespace" clone flag
Posted by ebiederm on Tue, 17 Apr 2007 09:04:14 GMT

View Forum Message <> Reply to Message

"Serge E. Hallyn" <serue@us.ibm.com> writes:
>>
>> Why are directory permissions not sufficient to allow/deny non-priveleged
> mounts?
>> I don't understand that contention yet.
>
> The same scenarios laid out previously in this thread. I.e.
>
> 1. user hallyn does mount --bind / /home/hallyn/root
> 2. (...)
> 3. admin does "deluser hallyn"
>
> and deluser starts wiping out root
>
> Or,
>
> 1. user hallyn does mount --bind / /home/hallyn/root
> 2. backup daemon starts backing up /home/hallyn/root/home/hallyn/root/home...
>
> So we started down the path of forcing users to clone a new namespace
> before doing user mounts, which is what the clone flag was about. Using
> per-mount flags also suffices as you had pointed out, which is being
> done here. But directory permissions are inadequate.

Interesting....

So far even today these things can happen, however they are sufficiently
unlikely the tools don't account for them.

Once a hostile user can cause them things are more of a problem.

> (Unless you want to tackle each problem legacy tool one at a time to
> remove problems - i.e. deluser should umount everything under
> /home/hallyn before deleting, backup should be spawned from it's own

> namespace cloned right after boot or just back up on one filesystem,
> etc.)

I don't see a way that backup and deluser won't need to be modified to work properly in a system where non-priveleged mounts are allowed, at least they will need to account for /share.

That said it is clearly a hazard if we enable this functionality by default.

If we setup a pam module that triggers on login and perhaps when cron and at jobs run to setup an additional mount namespace I think keeping applications locked away in their own mount namespace is sufficient to avoid hostile users from doing unexpected things to the initial mount namespace. So unless I am mistake it should be relatively simple to prevent user space from encountering problems.

That still leaves the question of how we handle systems with an old user space that is insufficiently robust to deal with mounts occurring at unexpected locations.

  I think a simple sysctl to enable/disable of non-priveleged mounts
  defaulting to disabled is enough.

Am I correct or will it be more difficult than just a little pam module to ensure non-trusted users never run in the initial mount namespace?

Eric

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

---

Subject: Re: Re: [patch 05/10] add "permit user mounts in new namespace" clone flag
Posted by Miklos Szeredi on Tue, 17 Apr 2007 11:09:25 GMT
View Forum Message <> Reply to Message

> Interesting....
>
> So far even today these things can happen, however they are sufficiently
> unlikely the tools don't account for them.
>
> Once a hostile user can cause them things are more of a problem.
>

> > (Unless you want to tackle each problem legacy tool one at a time to
> > remove problems - i.e. deluser should umount everything under
> > /home/hallyn before deleting, backup should be spawned from it's own
> > namespace cloned right after boot or just back up on one filesystem,
> > etc.)
>
> I don't see a way that backup and deluser won't need to be modified
> to work properly in a system where non-priveleged mounts are allowed,
> at least they will need to account for /share.
>
> That said it is clearly a hazard if we enable this functionality by
> default.
>
> If we setup a pam module that triggers on login and perhaps when
> cron and at jobs run to setup an additional mount namespace I think
> keeping applications locked away in their own mount namespace is
> sufficient to avoid hostile users from doing unexpected things to
> the initial mount namespace. So unless I am mistake it should be
> relatively simple to prevent user space from encountering problems.
>
> That still leaves the question of how we handle systems with an old
> user space that is insufficiently robust to deal with mounts occurring
> at unexpected locations.
>
>
>   I think a simple sysctl to enable/disable of non-priveleged mounts
>   defaulting to disabled is enough.
>
> Am I correct or will it be more difficult than just a little pam
> module to ensure non-trusted users never run in the initial mount
> namespace?

I'm still not sure, what your problem is.

With the v3 of the usermounts patchset, by default, user mounts are disabled, because the "allow unpriv submounts" flag is cleared on all mounts.

There are several options available to sysadmins and distro builders to enable user mounts in a secure way:

- pam module, which creates a private namespace, and sets "allow unpriv submounts" on the mounts within the namespace

- pam module, which rbinds / onto /mnt/ns/$USER, and chroots into /mnt/ns/$USER, then sets the "allow unpriv submounts" on the mounts under /mnt/ns/$USER.

- sysadmin creates /mnt/usermounts writable to all users, with
  sticky bit (same as /tmp), does "mount --bind /mnt/usermounts
  /mnt/usermounts" and sets the "allow unpriv submounts" on
  /mnt/usermounts.

All of these are perfectly safe wrt userdel and backup (assuming it
doesn't try back up /mnt).

Miklos

Subject: Re: Re: [patch 05/10] add "permit user mounts in new namespace" clone flag
Posted by serue on Tue, 17 Apr 2007 14:25:19 GMT

View Forum Message <> Reply to Message

Quoting Eric W. Biederman (ebiederm@xmission.com):
> "Serge E. Hallyn" <serue@us.ibm.com> writes:
> >>
> >> Why are directory permissions not sufficient to allow/deny non-priveleged
> > > mounts?
> >> I don't understand that contention yet.
> >
> > The same scenarios laid out previously in this thread. I.e.
> >
> > 1. user hallyn does mount --bind / /home/hallyn/root
> > 2. (...)
> > 3. admin does "deluser hallyn"
> >
> > and deluser starts wiping out root
> >
> > Or,
> >
> > 1. user hallyn does mount --bind / /home/hallyn/root
> > 2. backup daemon starts backing up /home/hallyn/root/home/hallyn/root/home...
> >
> > So we started down the path of forcing users to clone a new namespace
> > before doing user mounts, which is what the clone flag was about. Using
> > per-mount flags also suffices as you had pointed out, which is being
> > done here. But directory permissions are inadequate.
>
> Interesting....
>
> So far even today these things can happen, however they are sufficiently

> unlikely the tools don't account for them.
>
> Once a hostile user can cause them things are more of a problem.
>
> > (Unless you want to tackle each problem legacy tool one at a time to
> > remove problems - i.e. deluser should umount everything under
> > /home/hallyn before deleting, backup should be spawned from it's own
> > namespace cloned right after boot or just back up on one filesystem,
> > etc.)
>
> I don't see a way that backup and deluser won't need to be modified
> to work properly in a system where non-priveleged mounts are allowed,
> at least they will need to account for /share.

Yes, all the tools need to avoid /share. Though at least it's a single
location we can avoid, and it is purely a system configuration issue,
whereas fixing deluser to watch for user mounts under /home involves (I
assume) rewriting a part of it.

> That said it is clearly a hazard if we enable this functionality by
> default.
>
> If we setup a pam module that triggers on login and perhaps when
> cron and at jobs run to setup an additional mount namespace I think
> keeping applications locked away in their own mount namespace is
> sufficient to avoid hostile users from doing unexpected things to
> the initial mount namespace. So unless I am mistake it should be
> relatively simple to prevent user space from encountering problems.
>
> That still leaves the question of how we handle systems with an old
> user space that is insufficiently robust to deal with mounts occurring
> at unexpected locations.
>
>
>   I think a simple sysctl to enable/disable of non-priveleged mounts
>   defaulting to disabled is enough.
>
> Am I correct or will it be more difficult than just a little pam
> module to ensure non-trusted users never run in the initial mount
> namespace?

The danger with relying on the pam module is that you have to plug it in
all the right places. For instance, if we're talking about malicious
users, now we have to start worrying about an ftp daemon with user login
that isn't using pam, and happens to have an exploitable bug.

So it seems to me the per-mount flag you suggested really is the best
solution. Now the pam module is still needed, but only to set things up

so that the user *can* do user mounts. If there's a way to login
bypassing the pam module, then the user simply won't be able to do user
mounts anywhere but under /share, and as Miklos suggested the perms on
share can probably be set to 000.

-serge

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

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

Posted by serue on Tue, 17 Apr 2007 14:28:45 GMT

View Forum Message <> Reply to Message

Quoting Eric W. Biederman (ebiederm@xmission.com):
> "Serge E. Hallyn" <serue@us.ibm.com> writes:
> >>
> >> Why are directory permissions not sufficient to allow/deny non-priveleged
> > mounts?
> >> I don't understand that contention yet.
> >
> > The same scenarios laid out previously in this thread. I.e.
> >
> > 1. user hallyn does mount --bind / /home/hallyn/root
> > 2. (...)
> > 3. admin does "deluser hallyn"
> >
> > and deluser starts wiping out root
> >
> > Or,
> >
> > 1. user hallyn does mount --bind / /home/hallyn/root
> > 2. backup daemon starts backing up /home/hallyn/root/home/hallyn/root/home...
> >
> > So we started down the path of forcing users to clone a new namespace
> > before doing user mounts, which is what the clone flag was about. Using
> > per-mount flags also suffices as you had pointed out, which is being
> > done here. But directory permissions are inadequate.
>
> Interesting....
>
> So far even today these things can happen, however they are sufficiently
> unlikely the tools don't account for them.
>
> Once a hostile user can cause them things are more of a problem.

>
> > (Unless you want to tackle each problem legacy tool one at a time to
> > remove problems - i.e. deluser should umount everything under
> > /home/hallyn before deleting, backup should be spawned from it's own
> > namespace cloned right after boot or just back up on one filesystem,
> > etc.)
>
> I don't see a way that backup and deluser won't need to be modified
> to work properly in a system where non-priveleged mounts are allowed,
> at least they will need to account for /share.
>
> That said it is clearly a hazard if we enable this functionality by
> default.
>
> If we setup a pam module that triggers on login and perhaps when
> cron and at jobs run to setup an additional mount namespace I think
> keeping applications locked away in their own mount namespace is
> sufficient to avoid hostile users from doing unexpected things to
> the initial mount namespace. So unless I am mistake it should be
> relatively simple to prevent user space from encountering problems.
>
> That still leaves the question of how we handle systems with an old
> user space that is insufficiently robust to deal with mounts occurring
> at unexpected locations.
>
>
>   I think a simple sysctl to enable/disable of non-priveleged mounts
>   defaulting to disabled is enough.

There is a sysctl for max_user_mounts which can be set to 0.

So a simple on/off sysctl is unnecessary, but given that admins might
wonder whether 0 means infinite :), and I agree on/off is important, a
second one wouldn't hurt.

> Am I correct or will it be more difficult than just a little pam
> module to ensure non-trusted users never run in the initial mount
> namespace?
>
> Eric

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

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

flag
Posted by ebiederm on Tue, 17 Apr 2007 18:16:45 GMT
View Forum Message <> Reply to Message

Miklos Szeredi <miklos@szeredi.hu> writes:

> I'm still not sure, what your problem is.

My problem right now is that I see a serious complexity escalation in
the user interface that we must support indefinitely.

I see us taking a nice powerful concept and seriously watering it down.
To some extent we have to avoid confusing suid applications. (I would
so love to remove the SUID bit...).

I'm being contrary to ensure we have a good code review.

I have heard it said that there are two kinds of design. Something
so simple it obviously has no deficiencies. Something so complex it has
no obvious deficiencies. I am very much afraid we are slipping the
mount namespace into the latter category of work. Which is a bad
bad thing for core OS feature.

> With the v3 of the usermounts patchset, by default, user mounts are
> disabled, because the "allow unpriv submounts" flag is cleared on all
> mounts.
>
> There are several options available to sysadmins and distro builders
> to enable user mounts in a secure way:
>
>   - pam module, which creates a private namespace, and sets "allow
>   unpriv submounts" on the mounts within the namespace
>
>   - pam module, which rbinds / onto /mnt/ns/$USER, and chroots into
>   /mnt/ns/$USER, then sets the "allow unpriv submounts" on the
>   mounts under /mnt/ns/$USER.

In part this really disturbs me because we now have two mechanisms for
controlling the scope of what a user can do.

A flag or a new namespace. Two mechanisms to accomplish the same
thing sound wrong, and hard to manage.

>   - sysadmin creates /mnt/usermounts writable to all users, with
>   sticky bit (same as /tmp), does "mount --bind /mnt/usermounts
>   /mnt/usermounts" and sets the "allow unpriv submounts" on
>   /mnt/usermounts.
>
> All of these are perfectly safe wrt userdel and backup (assuming it

> doesn't try back up /mnt).

I also don't understand at all the user= mount flag and options.
All it seemed to be used for was adding permissions to unmount. In user
space to deal with the lack of any form of untrusted mounts I can understand
this. In kernel space this seems to be more of a problem.

Eric

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

Subject: Re: Re: [patch 05/10] add "permit user mounts in new namespace" clone flag
Posted by Miklos Szeredi on Tue, 17 Apr 2007 18:36:04 GMT
View Forum Message <> Reply to Message

> > I'm still not sure, what your problem is.
>
> My problem right now is that I see a serious complexity escalation in
> the user interface that we must support indefinitely.
>
> I see us taking a nice powerful concept and seriously watering it down.
> To some extent we have to avoid confusing suid applications. (I would
> so love to remove the SUID bit...).
>
> I'm being contrary to ensure we have a good code review.

OK. And it's very much appreciated :)

> I have heard it said that there are two kinds of design. Something
> so simple it obviously has no deficiencies. Something so complex it has
> no obvious deficiencies. I am very much afraid we are slipping the
> mount namespace into the latter category of work. Which is a bad
> bad thing for core OS feature.

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.

> > With the v3 of the usermounts patchset, by default, user mounts are
> > disabled, because the "allow unpriv submounts" flag is cleared on all
> > mounts.
> >
> > There are several options available to sysadmins and distro builders
> > to enable user mounts in a secure way:

> >
> >   - pam module, which creates a private namespace, and sets "allow
> >   unpriv submounts" on the mounts within the namespace
> >
> >   - pam module, which rbinds / onto /mnt/ns/$USER, and chroots into
> >   /mnt/ns/$USER, then sets the "allow unpriv submounts" on the
> >   mounts under /mnt/ns/$USER.
>
> In part this really disturbs me because we now have two mechanisms for
> controlling the scope of what a user can do.

You mean rbind+chroot and clone(CLONE_NS)? Yes, those are two
different mechanisms achieving very similar results. But what has
this to do with unprivileged mounts?

> A flag or a new namespace. Two mechanisms to accomplish the same
> thing sound wrong, and hard to manage.

The flag permitting the unprivileged mounts (which we now agreed to
name "allowusermnt") is used in both cases.

Just creating a new namespace doesn't always imply that you want to
allow user mounts inside, does it? These are orthogonal features.

> >   - sysadmin creates /mnt/usermounts writable to all users, with
> >   sticky bit (same as /tmp), does "mount --bind /mnt/usermounts
> >   /mnt/usermounts" and sets the "allow unpriv submounts" on
> >   /mnt/usermounts.
> >
> > All of these are perfectly safe wrt userdel and backup (assuming it
> > doesn't try back up /mnt).
>
> I also don't understand at all the user= mount flag and options.

The "user=UID" or (or MNT_USER flag) serves multiple purposes:

  - help mount(8) move away from /etc/mtab
  - allow unprivileged umounts
  - account user mounts

> All it seemed to be used for was adding permissions to unmount. In user
> space to deal with the lack of any form of untrusted mounts I can understand
> this. In kernel space this seems to be more of a problem.

Why is handling unprivileged mounts in kernel different from handling
them in userspace in this respect?

Miklos

Subject: Re: Re: [patch 05/10] add "permit user mounts in new namespace" clone flag
Posted by ebiederm on Tue, 17 Apr 2007 19:54:14 GMT
View Forum Message <> Reply to Message

Miklos Szeredi <miklos@szeredi.hu> writes:

```
>> > I'm still not sure, what your problem is.
>>
>> My problem right now is that I see a serious complexity escalation in
>> the user interface that we must support indefinitely.
>>
>> I see us taking a nice powerful concept and seriously watering it down.
>> To some extent we have to avoid confusing suid applications.  (I would
>> so love to remove the SUID bit...).
>>
>> I'm being contrary to ensure we have a good code review.
>
> OK.  And it's very much appreciated :)
>
>> I have heard it said that there are two kinds of design.  Something
>> so simple it obviously has no deficiencies.  Something so complex it has
>> no obvious deficiencies.  I am very much afraid we are slipping the
>> mount namespace into the latter category of work.  Which is a bad
>> bad thing for core OS feature.
>
> 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 then 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 it's 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.

>> In part this really disturbs me because we now have two mechanisms for
>> controlling the scope of what a user can do.
>
> You mean rbind+chroot and clone(CLONE_NS)? Yes, those are two
> different mechanisms achieving very similar results. But what has
> this to do with unprivileged mounts?

The practical question is how do we limit what a user can mount and
unmount.

I would contend that at first glance stuffing a user in their own
mount namespace is sufficient, on a system with utilities aware
of the consequences of mount/unmount.

So we may not need a unprivileged mount disable except as a way
to allow an old user space to run a new kernel.

>> A flag or a new namespace. Two mechanisms to accomplish the same
>> thing sound wrong, and hard to manage.
>
> The flag permitting the unprivileged mounts (which we now agreed to
> name "allowusermnt") is used in both cases.
>
> Just creating a new namespace doesn't always imply that you want to
> allow user mounts inside, does it? These are orthogonal features.

After user space has been updated we always want to allow unprivileged
mounts.

If I get pushed I will say that we need to remove suid exec capability
from user space as well. At which point we don't even need directory
security checks, there is enough benefit there I certainly think
it is worth considering having an entire NOSUID user space.

Removing suid is probably excessive but if it isn't much harder
then sane mount namespace support we should probably consider it.

>> >   - sysadmin creates /mnt/usermounts writable to all users, with
>> >   sticky bit (same as /tmp), does "mount --bind /mnt/usermounts
>> >   /mnt/usermounts" and sets the "allow unpriv submounts" on
>> >   /mnt/usermounts.
>> >
>> > All of these are perfectly safe wrt userdel and backup (assuming it
>> > doesn't try back up /mnt).
>>
>> I also don't understand at all the user= mount flag and options.

>
> The "user=UID" or (or MNT_USER flag) serves multiple purposes:
>
>   - help mount(8) move away from /etc/mtab
>   - allow unprivileged umounts
>   - account user mounts
>
>> All it seemed to be used for was adding permissions to unmount. In user
>> space to deal with the lack of any form of untrusted mounts I can understand
>> this. In kernel space this seems to be more of a problem.
>
> Why is handling unprivileged mounts in kernel different from handling
> them in userspace in this respect?

Ok. I just looked at what user space is doing. The difference is that
what user space is doing predates mount namespaces, and was there as
far as I can tell to keep one user from causing problems for
another user.   If we choose to make mount namespaces to be
the unit of granularity we don't need this capability.

All we have to do is deny unmounts that would confuse a suid
executable. Which mounts are those?

Eric

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

Subject: Re: Re: [patch 05/10] add "permit user mounts in new namespace" clone flag
Posted by Miklos Szeredi on Wed, 18 Apr 2007 09:11:19 GMT

View Forum Message <> Reply to Message

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

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

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?

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

Miklos

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

---

Subject: Re: Re: [patch 05/10] add "permit user mounts in new namespace" clone flag
Posted by Trond Myklebust on Wed, 18 Apr 2007 13:55:05 GMT
View Forum Message <> Reply to Message

On Wed, 2007-04-18 at 11:11 +0200, Miklos Szeredi wrote:
> > > 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 then 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 it's 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.
> Also I don't think we want to require auditing userspace before
> enabling user mounts.
>
> 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. And maybe add some extra checks which
> prevent ugliness from happening with suid programs. Is this correct?
>
> 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?
>
> Also how are we going to prevent the user from creating millions of
> mounts, and using up all the kernel memory for vfsmounts?

Don't forget that almost all mount flags are per-superblock. How are you
planning on dealing with the case that one user mounts a filesystem
read-only, while another is trying to mount the same one read-write?

Trond

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

Subject: Re: Re: [patch 05/10] add "permit user mounts in new namespace" clone flag
Posted by Miklos Szeredi on Wed, 18 Apr 2007 14:03:23 GMT

View Forum Message <> Reply to Message

> > > > 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 then 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 it's 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.
> > Also I don't think we want to require auditing userspace before
> > enabling user mounts.
> >
> > 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. And maybe add some extra checks which
> > prevent ugliness from happening with suid programs. Is this correct?
> >
> > 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?
> >
> > Also how are we going to prevent the user from creating millions of
> > mounts, and using up all the kernel memory for vfsmounts?
>
> Don't forget that almost all mount flags are per-superblock. How are you
> planning on dealing with the case that one user mounts a filesystem
> read-only, while another is trying to mount the same one read-write?

Yeah, I forgot, the per-mount read-only patches are not yet in.

That doesn't really change my agrument though. _If_ the flag is per
mount, then it makes sense to be able to change it on a master and not
on a slave. If mount flags are propagated, this is not possible.

Miklos

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

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

flag
Posted by Trond Myklebust on Wed, 18 Apr 2007 14:26:29 GMT
View Forum Message <> Reply to Message

On Wed, 2007-04-18 at 16:03 +0200, Miklos Szeredi wrote:
> > Don't forget that almost all mount flags are per-superblock. How are you
> > planning on dealing with the case that one user mounts a filesystem
> > read-only, while another is trying to mount the same one read-write?
>
> Yeah, I forgot, the per-mount read-only patches are not yet in.
>
> That doesn't really change my agrument though. _If_ the flag is per
> mount, then it makes sense to be able to change it on a master and not
> on a slave. If mount flags are propagated, this is not possible.

Read-only isn't the only issue. On something like NFS, there are flags
to set the security flavour, turn on/off encryption etc.

If I mount your home directory using no encryption in my namespace, for
instance, then neither you nor the administrator will be able to turn it
on afterwards in yours without first unmounting it from mine so that the
superblock is destroyed.

Cheers
  Trond

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

Subject: Re: Re: [patch 05/10] add "permit user mounts in new namespace" clone flag
Posted by Christoph Hellwig on Wed, 18 Apr 2007 15:01:10 GMT
View Forum Message <> Reply to Message

On Wed, Apr 18, 2007 at 10:26:29AM -0400, Trond Myklebust wrote:
> > That doesn't really change my agrument though. _If_ the flag is per
> > mount, then it makes sense to be able to change it on a master and not
> > on a slave. If mount flags are propagated, this is not possible.
>
> Read-only isn't the only issue. On something like NFS, there are flags
> to set the security flavour, turn on/off encryption etc.
>
> If I mount your home directory using no encryption in my namespace, for
> instance, then neither you nor the administrator will be able to turn it
> on afterwards in yours without first unmounting it from mine so that the

> superblock is destroyed.

I suspect the right answer here is to make nfs mount handling smarter.
The way mounting works the filesystem is allowed to choose whether it
can re-used a superblock or needs a new one. In the NFS case we probably
want to allow multiple superblocks for the same export if important
paramaters like the security model mismatch.

This is however only the technical part of the answer. There's a more
important one, and that's the user experience - we need a much better
way to document which flags are per-superblock and which are per-mount.

Due to this and some other issues we'll probably need a new mount API
in some time.

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

Subject: Re: Re: [patch 05/10] add "permit user mounts in new namespace" clone flag
Posted by Miklos Szeredi on Wed, 18 Apr 2007 15:06:33 GMT

View Forum Message <> Reply to Message

> > > Don't forget that almost all mount flags are per-superblock. How are you
> > > planning on dealing with the case that one user mounts a filesystem
> > > read-only, while another is trying to mount the same one read-write?
> >
> > Yeah, I forgot, the per-mount read-only patches are not yet in.
> >
> > That doesn't really change my agrument though. _If_ the flag is per
> > mount, then it makes sense to be able to change it on a master and not
> > on a slave. If mount flags are propagated, this is not possible.
>
> Read-only isn't the only issue. On something like NFS, there are flags
> to set the security flavour, turn on/off encryption etc.
>
> If I mount your home directory using no encryption in my namespace, for
> instance, then neither you nor the administrator will be able to turn it
> on afterwards in yours without first unmounting it from mine so that the
> superblock is destroyed.

OK, that's interesting, but I fail to grasp the relevance of this to
unprivileged mounts.

Or are you thinking of unprivileged NFS mounts? Well, think again,
because that involves _much_ more than it seems at first glance.

Miklos

____

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

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 then 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 it's 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 avoids 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'm 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

Subject: Re: Re: [patch 05/10] add "permit user mounts in new namespace" clone flag
Posted by Miklos Szeredi on Wed, 18 Apr 2007 18:05:34 GMT

View Forum Message <> Reply to Message

> >> > 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 then 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 it's 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 avoids 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.

But unprivileged proceses are not only created through login. What
happens, when some process does setuid(NONZERO). With your proposal
it now gained the privilege of mounting in the initial namespace. IOW
every program calling setuid() now has to be audited for any problems
this could cause.

> (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.

Ah.

> >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.

Umm, well. What if user mounts a filesystem read-only. Then wants to
unmount, but hey, it's read-only, no permission to write, no
unmount...

That's just a stupid example, but it shows, that file permissions are
not always useful for determining what you can mount, and especially
what you can unmount.

> 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.

That rules out unprivileged bind mounts.

> 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).

Checking the permissions on the mountpoint to allow unmounting is

- rather inelegant: user can't see those permissions, can only determine if umount is allowed by trial and error

- may be a security hole, e.g.:

```
    sysadmin:

        mkdir -m 777 /mnt/disk
        mount /dev/hda2 /mnt/disk
```

Of course the user doesn't have the right to delete the contents of the mount, yet the permissions on the mountpoint would imply that s/he has permission to umount the disk.

> > 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'm 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.

Yup, that would work. Accounting only unprivileged mounts may be more intuitive though:

  http://lkml.org/lkml/2005/5/11/38

Miklos

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

Subject: Re: Re: [patch 05/10] add "permit user mounts in new namespace" clone flag
Posted by Trond Myklebust on Wed, 18 Apr 2007 19:00:47 GMT

View Forum Message <> Reply to Message

On Wed, 2007-04-18 at 16:01 +0100, Christoph Hellwig wrote:
> I suspect the right answer here is to make nfs mount handling smarter.
> The way mounting works the filesystem is allowed to choose whether it
> can re-used a superblock or needs a new one. In the NFS case we probably
> want to allow multiple superblocks for the same export if important
> paramaters like the security model mismatch.

It might also be another application for layering. If I were able to
share enough of the inode and dentry information between superblocks,
then all sorts of interesting possibilities arise...

Cheers
  Trond

Subject: Re: Re: [patch 05/10] add "permit user mounts in new namespace" clone flag
Posted by Miklos Szeredi on Thu, 19 Apr 2007 09:02:10 GMT

View Forum Message <> Reply to Message

> Checking the permissions on the mountpoint to allow unmounting is
>
>   - rather inelegant: user can't see those permissions, can only
>   determine if umount is allowed by trial and error
>
>   - may be a security hole, e.g.:
>
>   sysadmin:
>
>   mkdir -m 777 /mnt/disk
>   mount /dev/hda2 /mnt/disk
>
> Of course the user doesn't have the right to delete the contents of
> the mount, yet the permissions on the mountpoint would imply that s/he
> has permission to umount the disk.

It is becoming increasingly apparent, that mount/umount permission
based on file permissions is inherently broken:

1) there are user-writable files under /proc/$PID/, which definitely
   shouldn't be allowed to be overmounted

2) if user mounts an fs read-only, then wants to create a submount of
   this, it will fail with the current patchset

Solving 2) should be trivial: submounting a mount owned by the user should be always allowed regardless of the file permissions.

Maybe this could be generaized to say, that a mount can be submounted by an unprivileged user IFF parent mount is owned by said user.

This would get rid of some of the complications in the current patchset, namely the functionality of MNT_ALLOWUSERMNT and MNT_USER flags would be merged, and the permission checking would be removed.

For example on login, the user could get a private namespace set up some that the home directory is owned by the user, and hence can be freely submounted:

```
  clone(CLONE_NEWNS)
  mount --bind /home/$USER /home/$USER
  mount --remount -ouser=$USER /home/$USER
```

This is of course more limiting than allowing mounts based on file permissions, but it's also a lot cleaner.

Hmm?

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