Subject: Re: [PATCH] Send quota messages via netlink
Posted by serge on Thu, 30 Aug 2007 22:14:47 GMT
View Forum Message <> Reply to Message

Quoting Jan Kara (jack@suse.cz):
> On Thu 30-08-07 14:10:10, Serge E. Hallyn wrote:
> > Quoting Jan Kara (jack@suse.cz):
> > > On Wed 29-08-07 15:06:43, Eric W. Biederman wrote:
> > > > Jan Kara <jack@suse.cz> writes:
> > > > >> However I'm still confused about the use of current->user. If that
> > > > >> is what we really want and not the user who's quota will be charged
> > > > >> it gets to be a really trick business, because potentially the uid
> > > > >> we want to deliver varies depending on who opened the netlink socket.
> > > > >   I see it's a complicated matter :). What I need to somehow pass to
> > > > > userspace is something (and I don't really care whether it will be number,
> > > > > string or whatever) that userspace can read and e.g. find a terminal
> > > > > window or desktop the affected user has open and also translate the
> > > > > identity to some user-understandable name (average user Joe has to
> > > > > understand that he should quickly cleanup his home directory ;).
> > > > >   Thinking more about it, we could probably pass a string to userspace in
> > > > > the format:
> > > > >   <namespace type>:<user identification>
> > > > >
> > > > > So for example we can have something like:
> > > > >   unix:1000 (traditional unix UIDs)
> > > > >   nfs4:joe@machine
> > > > >
> > > > > The problem is: Are we able to find out in which "namespace type" we are
> > > > > and send enough identifying information from a context of unpriviledged
> > > > > user?
> > > >
> > > > Ok. This provides enough context to understand what you are trying to do.
> > > > You do want the unix user id, not the filesystem notion. Because you
> > > > are looking for the user.
> > > >
> > > > So we have to figure out how to do the hard thing which is look at
> > > > who opened our netlink broadcast see if they are in the same user
> > > > namespace as current->user. Which is a pain and we don't currently
> > > > have the infrastructure for.
> > >   There can be arbitrary number of listeners (potentially from different
> > > namespaces if I understand it correctly) listening to broadcasts. So I
> >
> > Currently that is true, but i think isolating netlink sockets is going
> > to have to be done pretty soon.
> >
> > On the one hand cloning a new netlink socket ns when you unshare
> > CLONE_NEWNET may seem 'obvious', but I think doing so when you unshare
> > CLONE_NEWUSER make much more sense considering netlink's use for audit

> > and now for quota.
> >
> > > think we should pass some universal identifier rather than try to find out
> >
> > Even with isolating netlink we still may want to send out an identifier.
> > However, just as with mounts extensions we're printing out the memory
> > address of vfsmounts, we might just want to print out the memory address
> > of the userns. It's not universal, but should be good enough.
>   Maybe before proceeding further with the discussion I'd like to
> understand following: What are these user namespaces supposed to be good
> for?

(Please skip to the message end first, as I think you may not care about
the next bit of my blathering)

Right now they are only good for providing some separate accounting for
uid 1000 in one user namespace versus uid 1000 in another namespace.
All security enforcement must be done by actually providing separate
filesystems and separate pid namespaces and, hopefully, with a selinux
policy.

Eventually the idea will be that uid 1000 in one user namespace and uid
1000 in another namespace will be completely separate entities. A
mounted filesystem will be tied to a particuler user namespace, and
the kernel will provide any cross-userns access perhaps the way I
described, with uid equivalence implemented through the keyring.

But note that this isn't really relevant when we get to NFS. Two user
namespaces on one machine should have different network namespaces and
network addresses as well, and so should look to the NFS server like two
separate machines.

So the user namespaces are only really relevant when talking about local
filesystems.

>   I imagine it so that you have a machine and on it several virtual
> machines which are sharing a filesystem (or it could be a cluster). Now you
> want UIDs to be independent between these virtual machines. That's it,
> right?
>   Now to continue the example: Alice has UID 100 on machineA, Bob has
> UID 100 on machineB. These translate to UIDs 1000 and 1001 on the common
> filesystem. Process of Alice writes to a file and Bob becomes to be over
> quota. In this situation, there would be probably two processes (from
> machineA and machineB) listening on the netlink socket. We want to send a
> message so that on Alice's desktop we can show a message: "You caused
> Bob to exceed his quotas" and of Bob's desktop: "Alice has caused that you
> are over quota.".

Since this is over NFS, you handle it the way you would any other time
that user Alice on some other machine managed to do this.

> Because there may be is not a notion of Bob on machineA or of Alice on
> machineB, we are in trouble, right? What I like the most is to use the
> filesystem identities (as you suggested in some other email). I. e. because
> both Alice and Bob share a filesystem, identities of both have to make sense
> to it (for example for purposes of permission checking). So we can probably

Right, so long as we're talking about local filesystems that's the way
to go. If a file write was allowed which brought bob over quota,
clearly the person responsible had some uid valid on the filesystem to
allow him to do so.

> send via netlink these (in our example ids 1000 and 1001) and hope that
> inside machineA and machineB there will be a way to translate these
> identities to names "Alice" and "Bob". So that user can understand what
> is happenning. Does this sound plausible?
> If we go this route, then we only need a kernel function, that will
> for a pair ($filesystem, $task) return indentity of that $task used
> for operations on $filesystem...

Ok, now I see. This is again unrelated to user namespaces, it's an
issue regardless.

Is there no way to just report Alice as the guilty party to Bob on his
machine as (host=nfsserver,uid=1000)?

-serge

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

Subject: Re: [PATCH] Send quota messages via netlink
Posted by Jan Kara on Mon, 03 Sep 2007 14:21:23 GMT

View Forum Message <> Reply to Message

On Thu 30-08-07 17:14:47, Serge E. Hallyn wrote:
> Quoting Jan Kara (jack@suse.cz):
> > Maybe before proceeding further with the discussion I'd like to
> > understand following: What are these user namespaces supposed to be good
> > for?
>
> (Please skip to the message end first, as I think you may not care about
> the next bit of my blathering)
>

> Right now they are only good for providing some separate accounting for
> uid 1000 in one user namespace versus uid 1000 in another namespace.
> All security enforcement must be done by actually providing separate
> filesystems and separate pid namespaces and, hopefully, with a selinux
> policy.
>
> Eventually the idea will be that uid 1000 in one user namespace and uid
> 1000 in another namespace will be completely separate entities. A
> mounted filesystem will be tied to a particuler user namespace, and
> the kernel will provide any cross-userns access perhaps the way I
> described, with uid equivalence implemented through the keyring.
  I see. Thanks for explanation.

> But note that this isn't really relevant when we get to NFS. Two user
> namespaces on one machine should have different network namespaces and
> network addresses as well, and so should look to the NFS server like two
> separate machines.
>
> So the user namespaces are only really relevant when talking about local
> filesystems.
>
> >   I imagine it so that you have a machine and on it several virtual
> > machines which are sharing a filesystem (or it could be a cluster). Now you
> > want UIDs to be independent between these virtual machines. That's it,
> > right?
> >   Now to continue the example: Alice has UID 100 on machineA, Bob has
> > UID 100 on machineB. These translate to UIDs 1000 and 1001 on the common
> > filesystem. Process of Alice writes to a file and Bob becomes to be over
> > quota. In this situation, there would be probably two processes (from
> > machineA and machineB) listening on the netlink socket. We want to send a
> > message so that on Alice's desktop we can show a message: "You caused
> > Bob to exceed his quotas" and of Bob's desktop: "Alice has caused that you
> > are over quota.".
>
> Since this is over NFS, you handle it the way you would any other time
> that user Alice on some other machine managed to do this.
  I meant this would actually happen over a local filesystem (imagine
something like "hostfs" from UML).

> >   Because there may be is not a notion of Bob on machineA or of Alice on
> > machineB, we are in trouble, right? What I like the most is to use the
> > filesystem identities (as you suggested in some other email). I. e. because
> > both Alice and Bob share a filesystem, identities of both have to make sense
> > to it (for example for purposes of permission checking). So we can probably
>
> Right, so long as we're talking about local filesystems that's the way
> to go. If a file write was allowed which brought bob over quota,
> clearly the person responsible had some uid valid on the filesystem to

> allow him to do so.
  Fine. So I'll keep UID in the quota netlink protocol with the meaning
"the identity of the user for filesystem operations".

> > send via netlink these (in our example ids 1000 and 1001) and hope that
> > inside machineA and machineB there will be a way to translate these
> > identities to names "Alice" and "Bob". So that user can understand what
> > is happenning. Does this sound plausible?
> >   If we go this route, then we only need a kernel function, that will
> > for a pair ($filesystem, $task) return indentity of that $task used
> > for operations on $filesystem...
>
> Ok, now I see. This is again unrelated to user namespaces, it's an
> issue regardless.
>
> Is there no way to just report Alice as the guilty party to Bob on his
> machine as (host=nfsserver,uid=1000)?
  You know, in fact this contains all the information but it is quite useless
for an ordinary user. The message should be understandable to average desktop
user so it should contain some name rather than UID - but resolving the
"filesystem" UID to some meaningful name is completely different issue
and I'd probably leave that for the moment when the kernel infrastructure
and use cases would be clearer...

Honza
--
Jan Kara <jack@suse.cz>
SuSE CR Labs

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

Subject: Re: [PATCH] Send quota messages via netlink
Posted by serue on Tue, 04 Sep 2007 21:32:10 GMT

View Forum Message <> Reply to Message

Quoting Jan Kara (jack@suse.cz):
> On Thu 30-08-07 17:14:47, Serge E. Hallyn wrote:
> > Quoting Jan Kara (jack@suse.cz):
> > >   Maybe before proceeding further with the discussion I'd like to
> > > understand following: What are these user namespaces supposed to be good
> > > for?
> >
> > (Please skip to the message end first, as I think you may not care about
> > the next bit of my blathering)
> >

> > Right now they are only good for providing some separate accounting for
> > uid 1000 in one user namespace versus uid 1000 in another namespace.
> > All security enforcement must be done by actually providing separate
> > filesystems and separate pid namespaces and, hopefully, with a selinux
> > policy.
> >
> > Eventually the idea will be that uid 1000 in one user namespace and uid
> > 1000 in another namespace will be completely separate entities. A
> > mounted filesystem will be tied to a particuler user namespace, and
> > the kernel will provide any cross-userns access perhaps the way I
> > described, with uid equivalence implemented through the keyring.
>   I see. Thanks for explanation.
>
> > But note that this isn't really relevant when we get to NFS. Two user
> > namespaces on one machine should have different network namespaces and
> > network addresses as well, and so should look to the NFS server like two
> > separate machines.
> >
> > So the user namespaces are only really relevant when talking about local
> > filesystems.
> >
> > > I imagine it so that you have a machine and on it several virtual
> > > machines which are sharing a filesystem (or it could be a cluster). Now you
> > > want UIDs to be independent between these virtual machines. That's it,
> > > right?
> > > Now to continue the example: Alice has UID 100 on machineA, Bob has
> > > UID 100 on machineB. These translate to UIDs 1000 and 1001 on the common
> > > filesystem. Process of Alice writes to a file and Bob becomes to be over
> > > quota. In this situation, there would be probably two processes (from
> > > machineA and machineB) listening on the netlink socket. We want to send a
> > > message so that on Alice's desktop we can show a message: "You caused
> > > Bob to exceed his quotas" and of Bob's desktop: "Alice has caused that you
> > > are over quota.".
> >
> > Since this is over NFS, you handle it the way you would any other time
> > that user Alice on some other machine managed to do this.
>   I meant this would actually happen over a local filesystem (imagine
> something like "hostfs" from UML).

Ok, then that is where I was previously suggesting that we use an api to
report a uid meaningful in bob's context, where we currently (in the
absense of meaningful mount uids and uid equivalence) tell Bob that root
was the one who brought him over quota. From a user pov 'nobody' would
make more sense, but I don't think we want the kernel to know about user
nobody, right?

So if the msg weren't broadcast, or netlink sockets were tied to one
user namespace, we could call a

int uid_in_user_ns(struct user *, struct user_ns *)
sending in Alice's user struct and Bob's userns, and use the result in
the netlink message. Otherwise I'm not sure what is the right answer.
We just might need the equivalent of 'struct pid' to struct user, or
persistant global user namespace ids (persistant after user namespace
destruction, not across reboot) so we can safely send the user_ns * in a
netlink msg.

> > >   Because there may be is not a notion of Bob on machineA or of Alice on
> > > machineB, we are in trouble, right? What I like the most is to use the
> > > filesystem identities (as you suggested in some other email). I. e. because
> > > both Alice and Bob share a filesystem, identities of both have to make sense
> > > to it (for example for purposes of permission checking). So we can probably
> >
> > Right, so long as we're talking about local filesystems that's the way
> > to go. If a file write was allowed which brought bob over quota,
> > clearly the person responsible had some uid valid on the filesystem to
> > allow him to do so.
>   Fine. So I'll keep UID in the quota netlink protocol with the meaning
> "the identity of the user for filesystem operations".

I think that's ok.

Hopefully when that changes to accomodate user namespaces, we can use
netlink field versioning to make that transition pretty seamless?

If not, then we probably should in fact make some decision now so as not
to change the api.

> > > send via netlink these (in our example ids 1000 and 1001) and hope that
> > > inside machineA and machineB there will be a way to translate these
> > > identities to names "Alice" and "Bob". So that user can understand what
> > > is happenning. Does this sound plausible?
> > >   If we go this route, then we only need a kernel function, that will
> > > for a pair ($filesystem, $task) return indentity of that $task used
> > > for operations on $filesystem...
> >
> > Ok, now I see. This is again unrelated to user namespaces, it's an
> > issue regardless.
> >
> > Is there no way to just report Alice as the guilty party to Bob on his
> > machine as (host=nfsserver,uid=1000)?
>   You know, in fact this contains all the information but it is quite useless
> for an ordinary user. The message should be understandable to average desktop

What is the ordinary user going to do about it? If the user didn't set
up the nfsserver and/or the second client, the only thing he can do is
report the guilty user to an admin. In which case the tuple

(host=nfsserver,uid=1000) is exactly the data he needs to report.

> user so it should contain some name rather than UID - but resolving the
> "filesystem" UID to some meaningful name is completely different issue
> and I'd probably leave that for the moment when the kernel infrastructure
> and use cases would be clearer...
>
> Honza
> --
> Jan Kara <jack@suse.cz>
> SuSE CR Labs

thanks,
-serge

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

Subject: Re: [PATCH] Send quota messages via netlink
Posted by Jan Kara on Tue, 04 Sep 2007 22:49:21 GMT

View Forum Message <> Reply to Message

On Tue 04-09-07 16:32:10, Serge E. Hallyn wrote:
> Quoting Jan Kara (jack@suse.cz):
> > On Thu 30-08-07 17:14:47, Serge E. Hallyn wrote:
> > > Quoting Jan Kara (jack@suse.cz):
> > > > I imagine it so that you have a machine and on it several virtual
> > > > machines which are sharing a filesystem (or it could be a cluster). Now you
> > > > want UIDs to be independent between these virtual machines. That's it,
> > > > right?
> > > > Now to continue the example: Alice has UID 100 on machineA, Bob has
> > > > UID 100 on machineB. These translate to UIDs 1000 and 1001 on the common
> > > > filesystem. Process of Alice writes to a file and Bob becomes to be over
> > > > quota. In this situation, there would be probably two processes (from
> > > > machineA and machineB) listening on the netlink socket. We want to send a
> > > > message so that on Alice's desktop we can show a message: "You caused
> > > > Bob to exceed his quotas" and of Bob's desktop: "Alice has caused that you
> > > > are over quota.".
> > >
> > > Since this is over NFS, you handle it the way you would any other time
> > > that user Alice on some other machine managed to do this.
> > I meant this would actually happen over a local filesystem (imagine
> > something like "hostfs" from UML).
>
> Ok, then that is where I was previously suggesting that we use an api to
> report a uid meaningful in bob's context, where we currently (in the

> absense of meaningful mount uids and uid equivalence) tell Bob that root
> was the one who brought him over quota. From a user pov 'nobody' would
> make more sense, but I don't think we want the kernel to know about user
> nobody, right?
  But what is the problem with using the filesystem ids? All virtual
machines in my example should have a notion of those...

> So if the msg weren't broadcast, or netlink sockets were tied to one
> user namespace, we could call a
> int uid_in_user_ns(struct user *, struct user_ns *)
> sending in Alice's user struct and Bob's userns, and use the result in
> the netlink message. Otherwise I'm not sure what is the right answer.
> We just might need the equivalent of 'struct pid' to struct user, or
> persistant global user namespace ids (persistant after user namespace
> destruction, not across reboot) so we can safely send the user_ns * in a
> netlink msg.
  Yes, that could also be a solution.

> > > >   Because there may be is not a notion of Bob on machineA or of Alice on
> > > > machineB, we are in trouble, right? What I like the most is to use the
> > > > filesystem identities (as you suggested in some other email). I. e. because
> > > > both Alice and Bob share a filesystem, identities of both have to make sense
> > > > to it (for example for purposes of permission checking). So we can probably
> > >
> > > Right, so long as we're talking about local filesystems that's the way
> > > to go. If a file write was allowed which brought bob over quota,
> > > clearly the person responsible had some uid valid on the filesystem to
> > > allow him to do so.
> >   Fine. So I'll keep UID in the quota netlink protocol with the meaning
> > "the identity of the user for filesystem operations".
>
> I think that's ok.
>
> Hopefully when that changes to accomodate user namespaces, we can use
> netlink field versioning to make that transition pretty seamless?
  Yes, we'd just assign the attribute a different number and teach
userspace about the new attribute format...

> If not, then we probably should in fact make some decision now so as not
> to change the api.
>
> > > > send via netlink these (in our example ids 1000 and 1001) and hope that
> > > > inside machineA and machineB there will be a way to translate these
> > > > identities to names "Alice" and "Bob". So that user can understand what
> > > > is happenning. Does this sound plausible?
> > > >   If we go this route, then we only need a kernel function, that will
> > > > for a pair ($filesystem, $task) return indentity of that $task used
> > > > for operations on $filesystem...

> > >
> > > Ok, now I see. This is again unrelated to user namespaces, it's an
> > > issue regardless.
> > >
> > > Is there no way to just report Alice as the guilty party to Bob on his
> > > machine as (host=nfsserver,uid=1000)?
> >   You know, in fact this contains all the information but it is quite useless
> > for an ordinary user. The message should be understandable to average desktop
>
> What is the ordinary user going to do about it? If the user didn't set
> up the nfsserver and/or the second client, the only thing he can do is
> report the guilty user to an admin. In which case the tuple
> (host=nfsserver,uid=1000) is exactly the data he needs to report.

Maybe write him an email or go and bang him with a baseball bat ;)
Seriously, if someone (like admin) is able to find a physical identity of the
guilty user, then we should be able to do this in a software too, shouldn't
we?

Honza
--
Jan Kara <jack@suse.cz>
SuSE CR Labs

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