Subject: namespaces compatibility list
Posted by Pavel Emelianov on Tue, 06 Nov 2007 10:51:21 GMT
View Forum Message <> Reply to Message

Hi guys!

As you might have seen, recently there was some spontaneous
discussion about the namespaces-working-together problems.

Ted T'so proposed to create some document that describes what
problems user may have when he/she creates some new namespace,
but keeps others shared. I like this idea, so here's the draft
with the problems I currently have in mind and can describe
somewhat audibly - the "namespaces compatibility list".

The Documentation/namespaces/ dir is about to contain more
docs about the namespaces stuff (e.g. I'm going to prepare
a doc about the pid namespaces, maybe Serge will want to
write something about the user namespaces development, Eric
may want to put some notes about the netns API and so on),
but currently there will be only one file.

What would you say about it?

Signed-off-by: Pavel Emelyanov <xemul@openvz.org>

```
diff --git a/Documentation/namespaces/compatibility-list.txt
b/Documentation/namespaces/compatibility-list.txt
new file mode 100644
index 0000000..4be4a3c
--- /dev/null
+++ b/Documentation/namespaces/compatibility-list.txt
@@ -0,0 +1,32 @@
+ Namespaces compatibility list
+
+This document contains the information about the problems user
+may have when creating tasks living in different namespaces.
+
+Here's the summary. This matrix shows the known problems, that
+occur when tasks share some namespace (the columns) while living
+in different other namespaces (the raws):
+
+ UTS IPC VFS PID User Net
+UTS
+IPC    1
+VFS
```

+PID 1 1
+User 2
+Net
+
+1. Both the IPC and the PID namespaces provide IDs to address
+ object inside the kernel. E.g. semaphore with ipcid or
+ process group with pid.
+
+ In both cases, tasks shouldn't try telling this id to some
+ other task living in different namespace vid shared filesystem
+ or IPC shmem/message. The fact is that this ID is only valid
+ within the namespace it was obtained in and may refer to some
+ other object in another namespace.
+
+2. Intentionnaly, two equal user ids in different user namespaces
+ should not be equal from the VFS point of view. In other
+ words, user 10 in one user namespace shouldn't have the same
+ access permissions to files, beloging to user 10 in another
+ namespace. But currently this is not so.

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

---

Subject: Re: namespaces compatibility list
Posted by dev on Tue, 06 Nov 2007 12:33:15 GMT
View Forum Message <> Reply to Message

imho very good idea and we'll have more and more docs there...
(the only issue I see - it would be nice to have cgroups docs in the same place,
though cgroups are not about namespaces directly.)

Acked-By: Kirill Korotaev <dev@sw.ru>

Pavel Emelyanov wrote:
> Hi guys!
>
> As you might have seen, recently there was some spontaneous
> discussion about the namespaces-working-together problems.
>
> Ted T'so proposed to create some document that describes what
> problems user may have when he/she creates some new namespace,
> but keeps others shared. I like this idea, so here's the draft
> with the problems I currently have in mind and can describe
> somewhat audibly - the "namespaces compatibility list".

>
> The Documentation/namespaces/ dir is about to contain more
> docs about the namespaces stuff (e.g. I'm going to prepare
> a doc about the pid namespaces, maybe Serge will want to
> write something about the user namespaces development, Eric
> may want to put some notes about the netns API and so on),
> but currently there will be only one file.
>
> What would you say about it?
>
> Signed-off-by: Pavel Emelyanov <xemul@openvz.org>
>
> ---
>
> diff --git a/Documentation/namespaces/compatibility-list.txt
b/Documentation/namespaces/compatibility-list.txt
> new file mode 100644
> index 0000000..4be4a3c
> --- /dev/null
> +++ b/Documentation/namespaces/compatibility-list.txt
> @@ -0,0 +1,32 @@
> + Namespaces compatibility list
> +
> +This document contains the information about the problems user
> +may have when creating tasks living in different namespaces.
> +
> +Here's the summary. This matrix shows the known problems, that
> +occur when tasks share some namespace (the columns) while living
> +in different other namespaces (the raws):
> +
> + UTS IPC VFS PID User Net
> +UTS
> +IPC   1
> +VFS
> +PID   1 1
> +User   2
> +Net
> +
> +1. Both the IPC and the PID namespaces provide IDs to address
> +   object inside the kernel. E.g. semaphore with ipcid or
> +   process group with pid.
> +
> +   In both cases, tasks shouldn't try telling this id to some
> +   other task living in different namespace vid shared filesystem
> +   or IPC shmem/message. The fact is that this ID is only valid
> +   within the namespace it was obtained in and may refer to some
> +   other object in another namespace.
> +

> +2. Intentionnaly, two equal user ids in different user namespaces
> + should not be equal from the VFS point of view. In other
> + words, user 10 in one user namespace shouldn't have the same
> + access permissions to files, beloging to user 10 in another
> + namespace. But currently this is not so.
>
> _____
> 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: namespaces compatibility list
Posted by Cedric Le Goater on Tue, 06 Nov 2007 12:54:37 GMT
View Forum Message <> Reply to Message

Pavel Emelyanov wrote:
> Hi guys!
>
> As you might have seen, recently there was some spontaneous
> discussion about the namespaces-working-together problems.
>
> Ted T'so proposed to create some document that describes what
> problems user may have when he/she creates some new namespace,
> but keeps others shared. I like this idea, so here's the draft
> with the problems I currently have in mind and can describe
> somewhat audibly - the "namespaces compatibility list".

that compatibility list could be encoded in the way we check
the clone flags in copy_process() and unshare(). It would
also be good to have it as a comment somewhere in kernel/fork.c

> The Documentation/namespaces/ dir is about to contain more
> docs about the namespaces stuff (e.g. I'm going to prepare
> a doc about the pid namespaces, maybe Serge will want to
> write something about the user namespaces development, Eric
> may want to put some notes about the netns API and so on),
> but currently there will be only one file.
>
> What would you say about it?

well, as this is user space issues, I'd say that we should

help building a good man page. What's in Documentation/
could help to do that but I don't trust documentation when
it's maintained in 2 places.

So a check_flags() routine for namespaces with all the required
comments would probably be more helpful for the manpage
maintainer.

I was thinking of merging some clone flags together also and
keep only 3, NS, PID and NET.

>
> Signed-off-by: Pavel Emelyanov <xemul@openvz.org>
>
> ---
>
> diff --git a/Documentation/namespaces/compatibility-list.txt
b/Documentation/namespaces/compatibility-list.txt
> new file mode 100644
> index 0000000..4be4a3c
> --- /dev/null
> +++ b/Documentation/namespaces/compatibility-list.txt
> @@ -0,0 +1,32 @@
> + Namespaces compatibility list
> +
> +This document contains the information about the problems user
> +may have when creating tasks living in different namespaces.
> +
> +Here's the summary. This matrix shows the known problems, that
> +occur when tasks share some namespace (the columns) while living
> +in different other namespaces (the raws):
> +
> + UTS IPC VFS PID User Net
> +UTS
> +IPC   1
> +VFS
> +PID   1 1
> +User   2
> +Net
> +

funny, I had just started doing :

depends on VFS PID IPC NET UTS MQ
VFS *
PID * *
IPC   * *
NET   * *

UTS *
MQ * * ? * *

I kept VFS out for the moment.

I would rather build a matrix giving the dependencies. nop ? which
is a way to enforce the clone flags.

C.

> +1. Both the IPC and the PID namespaces provide IDs to address
> + object inside the kernel. E.g. semaphore with ipcid or
> + process group with pid.
> +
> + In both cases, tasks shouldn't try telling this id to some
> + other task living in different namespace vid shared filesystem
> + or IPC shmem/message. The fact is that this ID is only valid
> + within the namespace it was obtained in and may refer to some
> + other object in another namespace.
> +
> +2. Intentionnaly, two equal user ids in different user namespaces
> + should not be equal from the VFS point of view. In other
> + words, user 10 in one user namespace shouldn't have the same
> + access permissions to files, beloging to user 10 in another
> + namespace. But currently this is not so.
>

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

---

Subject: Re: namespaces compatibility list
Posted by Pavel Emelianov on Tue, 06 Nov 2007 13:00:12 GMT
View Forum Message <> Reply to Message

Cedric Le Goater wrote:
> Pavel Emelyanov wrote:
>> Hi guys!
>>
>> As you might have seen, recently there was some spontaneous
>> discussion about the namespaces-working-together problems.
>>
>> Ted T'so proposed to create some document that describes what
>> problems user may have when he/she creates some new namespace,
>> but keeps others shared. I like this idea, so here's the draft

>> with the problems I currently have in mind and can describe
>> somewhat audibly - the "namespaces compatibility list".
>
> that compatibility list could be encoded in the way we check
> the clone flags in copy_process() and unshare(). It would
> also be good to have it as a comment somewhere in kernel/fork.c

How can we insure, that a new task will not share the files
with its parent to address the PID namespaces vs VFS namespaces
interaction? There's no way to do it. We can only keep them in
one IPC namespace...

>> The Documentation/namespaces/ dir is about to contain more
>> docs about the namespaces stuff (e.g. I'm going to prepare
>> a doc about the pid namespaces, maybe Serge will want to
>> write something about the user namespaces development, Eric
>> may want to put some notes about the netns API and so on),
>> but currently there will be only one file.
>>
>> What would you say about it?
>
> well, as this is user space issues, I'd say that we should
> help building a good man page. What's in Documentation/
> could help to do that but I don't trust documentation when
> it's maintained in 2 places.
>
> So a check_flags() routine for namespaces with all the required
> comments would probably be more helpful for the manpage
> maintainer.
>
> I was thinking of merging some clone flags together also and
> keep only 3, NS, PID and NET.
>
>> Signed-off-by: Pavel Emelyanov <xemul@openvz.org>
>>
>> ---
>>
>> diff --git a/Documentation/namespaces/compatibility-list.txt
b/Documentation/namespaces/compatibility-list.txt
>> new file mode 100644
>> index 0000000..4be4a3c
>> --- /dev/null
>> +++ b/Documentation/namespaces/compatibility-list.txt
>> @@ -0,0 +1,32 @@
>> + Namespaces compatibility list
>> +
>> +This document contains the information about the problems user
>> +may have when creating tasks living in different namespaces.

>> +
>> +Here's the summary. This matrix shows the known problems, that
>> +occur when tasks share some namespace (the columns) while living
>> +in different other namespaces (the raws):
>> +
>> + UTS IPC VFS PID User Net
>> +UTS
>> +IPC   1
>> +VFS
>> +PID   1 1
>> +User   2
>> +Net
>> +
>
> funny, I had just started doing :
>
> depends on VFS PID IPC NET UTS MQ
> VFS *
> PID * *
> IPC   * *
> NET   * *
> UTS   *
> MQ * * ? * *
>
> I kept VFS out for the moment.
>
> I would rather build a matrix giving the dependencies. nop ? which
> is a way to enforce the clone flags.
>
> C.
>
>
>> +1. Both the IPC and the PID namespaces provide IDs to address
>> +   object inside the kernel. E.g. semaphore with ipcid or
>> +   process group with pid.
>> +
>> +   In both cases, tasks shouldn't try telling this id to some
>> +   other task living in different namespace vid shared filesystem
>> +   or IPC shmem/message. The fact is that this ID is only valid
>> +   within the namespace it was obtained in and may refer to some
>> +   other object in another namespace.
>> +
>> +2. Intentionnaly, two equal user ids in different user namespaces
>> +   should not be equal from the VFS point of view. In other
>> +   words, user 10 in one user namespace shouldn't have the same
>> +   access permissions to files, beloging to user 10 in another
>> +   namespace. But currently this is not so.
>>

>
>

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

---

Subject: Re: namespaces compatibility list
Posted by Cedric Le Goater on Tue, 06 Nov 2007 16:01:31 GMT
View Forum Message <> Reply to Message

Pavel Emelyanov wrote:
> Cedric Le Goater wrote:
>> Pavel Emelyanov wrote:
>>> Hi guys!
>>>
>>> As you might have seen, recently there was some spontaneous
>>> discussion about the namespaces-working-together problems.
>>>
>>> Ted T'so proposed to create some document that describes what
>>> problems user may have when he/she creates some new namespace,
>>> but keeps others shared. I like this idea, so here's the draft
>>> with the problems I currently have in mind and can describe
>>> somewhat audibly - the "namespaces compatibility list".
>> that compatibility list could be encoded in the way we check
>> the clone flags in copy_process() and unshare(). It would
>> also be good to have it as a comment somewhere in kernel/fork.c
>
> How can we insure, that a new task will not share the files
> with its parent to address the PID namespaces vs VFS namespaces
> interaction? There's no way to do it. We can only keep them in
> one IPC namespace...

? I'm not sure I understand you.

C.

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

---

Subject: Re: namespaces compatibility list
Posted by Pavel Emelianov on Tue, 06 Nov 2007 16:10:33 GMT
View Forum Message <> Reply to Message

Cedric Le Goater wrote:
> Pavel Emelyanov wrote:
>> Cedric Le Goater wrote:
>>> Pavel Emelyanov wrote:
>>>> Hi guys!
>>>>
>>>> As you might have seen, recently there was some spontaneous
>>>> discussion about the namespaces-working-together problems.
>>>>
>>>> Ted T'so proposed to create some document that describes what
>>>> problems user may have when he/she creates some new namespace,
>>>> but keeps others shared. I like this idea, so here's the draft
>>>> with the problems I currently have in mind and can describe
>>>> somewhat audibly - the "namespaces compatibility list".
>>> that compatibility list could be encoded in the way we check
>>> the clone flags in copy_process() and unshare(). It would
>>> also be good to have it as a comment somewhere in kernel/fork.c
>> How can we insure, that a new task will not share the files
>> with its parent to address the PID namespaces vs VFS namespaces
>> interaction? There's no way to do it. We can only keep them in
>> one IPC namespace...
>
> ? I'm not sure I understand you.

As far as I understand, you propose the check for the clone flags
in the copy_process()/sys_unshare() and return -EINVAL for the cases
we consider to be unsafe. E.g. when a user wants to clone new pid
namespace, he must clone the ipc namespace as well.

But my point is that this check is not enough - user may kill himself
by cloning a pid namespace and sharing the pids via the filesystem
(like with the example with futexes) and there's no way to check for
this situation in the copy_process()/sys_unchare.

I mean that this list cannot be encoded. But we can warn user, that
some stuff will stop working if he violates some rules.

> C.
>

Subject: Re: namespaces compatibility list

Posted by ebiederm on Tue, 06 Nov 2007 16:36:26 GMT
View Forum Message <> Reply to Message

Pavel Emelyanov <xemul@openvz.org> writes:

> +2. Intentionnaly, two equal user ids in different user namespaces
> + should not be equal from the VFS point of view. In other
> + words, user 10 in one user namespace shouldn't have the same
> + access permissions to files, beloging to user 10 in another
> + namespace. But currently this is not so.

I don't know where this is going to land for a final call.
But if the pid namespace has a chance of landing under CONFIG_BROKEN
for the final stable release.

We seriously want to consider the user namespace for the same treatment.
We all seem to agree that it is incomplete.

Eric

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

Subject: Re: namespaces compatibility list
Posted by Cedric Le Goater on Tue, 06 Nov 2007 16:46:09 GMT
View Forum Message <> Reply to Message

Pavel Emelyanov wrote:
> Cedric Le Goater wrote:
>> Pavel Emelyanov wrote:
>>> Cedric Le Goater wrote:
>>>> Pavel Emelyanov wrote:
>>>>> Hi guys!
>>>>>
>>>>> As you might have seen, recently there was some spontaneous
>>>>> discussion about the namespaces-working-together problems.
>>>>>
>>>>> Ted T'so proposed to create some document that describes what
>>>>> problems user may have when he/she creates some new namespace,
>>>>> but keeps others shared. I like this idea, so here's the draft
>>>>> with the problems I currently have in mind and can describe
>>>>> somewhat audibly - the "namespaces compatibility list".
>>>> that compatibility list could be encoded in the way we check
>>>> the clone flags in copy_process() and unshare(). It would
>>>> also be good to have it as a comment somewhere in kernel/fork.c
>>> How can we insure, that a new task will not share the files

>>> with its parent to address the PID namespaces vs VFS namespaces
>>> interaction? There's no way to do it. We can only keep them in
>>> one IPC namespace...
>> ? I'm not sure I understand you.
>
> As far as I understand, you propose the check for the clone flags
> in the copy_process()/sys_unshare() and return -EINVAL for the cases
> we consider to be unsafe. E.g. when a user wants to clone new pid
> namespace, he must clone the ipc namespace as well.

yes.

> But my point is that this check is not enough - user may kill himself
> by cloning a pid namespace and sharing the pids via the filesystem
> (like with the example with futexes) and there's no way to check for
> this situation in the copy_process()/sys_unchare.

right. I think we can address Ulrich concerns first because we have
a solution for it (which looks like unsharing all namespaces at once,
here comes back the container object story :)

> I mean that this list cannot be encoded. But we can warn user, that
> some stuff will stop working if he violates some rules.

and then do that for the futexes, which are a real difficult case.

thanks,

C.

Subject: Re: namespaces compatibility list
Posted by Cedric Le Goater on Tue, 06 Nov 2007 16:48:32 GMT
View Forum Message <> Reply to Message

Eric W. Biederman wrote:
> Pavel Emelyanov <xemul@openvz.org> writes:
>
>> +2. Intentionnaly, two equal user ids in different user namespaces
>> + should not be equal from the VFS point of view. In other
>> + words, user 10 in one user namespace shouldn't have the same
>> + access permissions to files, beloging to user 10 in another
>> + namespace. But currently this is not so.

>
> I don't know where this is going to land for a final call.
> But if the pid namespace has a chance of landing under CONFIG_BROKEN
> for the final stable release.

I prefered immature but he, any config name would do, just to make sure it
doesn't get shipped by default in distros

> We seriously want to consider the user namespace for the same treatment.
> We all seem to agree that it is incomplete.

yes.

C.

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

---

Subject: Re: namespaces compatibility list
Posted by ebiederm on Tue, 06 Nov 2007 17:00:27 GMT
View Forum Message <> Reply to Message

Cedric Le Goater <clg@fr.ibm.com> writes:
> right. I think we can address Ulrich concerns first because we have
> a solution for it (which looks like unsharing all namespaces at once,
> here comes back the container object story :)

It doesn't work because we can't create a fresh mount namespace.

We need to create all new mounts (and deny access to the old ones)
if we want to prevent all possibility of user space goof ups.

While that is easy enough to build an application to do we can't
easily enforce that in the kernel. Currently this is all
CAP_SYS_ADMIN so only root can do this anyway. So we can easily
say don't do that then.

Clone flag consistency checking should only be used to enforce
cases where the kernel side cannot support correctly. Currently
the kernel has no problems with the current mix and match possibilities
short of implementation deficiencies. So I do not see us
addressing Ulrich's concerns with clone flags.

Eric

_____
Containers mailing list

Subject: Re: namespaces compatibility list
Posted by Pavel Emelianov on Tue, 06 Nov 2007 17:09:01 GMT
View Forum Message <> Reply to Message

Eric W. Biederman wrote:
> Cedric Le Goater <clg@fr.ibm.com> writes:
>> right. I think we can address Ulrich concerns first because we have
>> a solution for it (which looks like unsharing all namespaces at once,
>> here comes back the container object story :)
>
> It doesn't work because we can't create a fresh mount namespace.
>
> We need to create all new mounts (and deny access to the old ones)
> if we want to prevent all possibility of user space goof ups.
>
> While that is easy enough to build an application to do we can't
> easily enforce that in the kernel. Currently this is all
> CAP_SYS_ADMIN so only root can do this anyway. So we can easily
> say don't do that then.
>
> Clone flag consistency checking should only be used to enforce
> cases where the kernel side cannot support correctly. Currently
> the kernel has no problems with the current mix and match possibilities
> short of implementation deficiencies. So I do not see us
> addressing Ulrich's concerns with clone flags.

ACK :) Since this all is CAP_SYS_ADMIN-ed we can do with just a warning.

> Eric
>

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

Subject: Re: namespaces compatibility list
Posted by ebiederm on Tue, 06 Nov 2007 17:46:23 GMT
View Forum Message <> Reply to Message

Pavel Emelyanov <xemul@openvz.org> writes:

> Eric W. Biederman wrote:
>> Cedric Le Goater <clg@fr.ibm.com> writes:
>>> right. I think we can address Ulrich concerns first because we have
>>> a solution for it (which looks like unsharing all namespaces at once,
>>> here comes back the container object story :)
>>
>> It doesn't work because we can't create a fresh mount namespace.
>>
>> We need to create all new mounts (and deny access to the old ones)
>> if we want to prevent all possibility of user space goof ups.
>>
>> While that is easy enough to build an application to do we can't
>> easily enforce that in the kernel. Currently this is all
>> CAP_SYS_ADMIN so only root can do this anyway. So we can easily
>> say don't do that then.
>>
>> Clone flag consistency checking should only be used to enforce
>> cases where the kernel side cannot support correctly. Currently
>> the kernel has no problems with the current mix and match possibilities
>> short of implementation deficiencies. So I do not see us
>> addressing Ulrich's concerns with clone flags.
>
> ACK :) Since this all is CAP_SYS_ADMIN-ed we can do with just a warning.

So to restate.

clone flags consistency checks are for things the kernel can't do or
  for things that the kernel can't do securely.

If all we do is confuse user space if used improperly it's simply
a don't do that then.

CAP_SYS_ADMIN keeps us untrusted applications from confusing suid
executables, which is the only case where confusion counts as a
security hole.

Eric

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

---

Subject: Re: namespaces compatibility list
Posted by Cedric Le Goater on Wed, 07 Nov 2007 08:14:04 GMT
View Forum Message <> Reply to Message

Eric W. Biederman wrote:

> Cedric Le Goater <clg@fr.ibm.com> writes:
>> right. I think we can address Ulrich concerns first because we have
>> a solution for it (which looks like unsharing all namespaces at once,
>> here comes back the container object story :)
>
> It doesn't work because we can't create a fresh mount namespace.
>
> We need to create all new mounts (and deny access to the old ones)
> if we want to prevent all possibility of user space goof ups.

arg. yes, I keep on forgetting this one.

C.

> While that is easy enough to build an application to do we can't
> easily enforce that in the kernel. Currently this is all
> CAP_SYS_ADMIN so only root can do this anyway. So we can easily
> say don't do that then.
>
> Clone flag consistency checking should only be used to enforce
> cases where the kernel side cannot support correctly. Currently
> the kernel has no problems with the current mix and match possibilities
> short of implementation deficiencies. So I do not see us
> addressing Ulrich's concerns with clone flags.
>
> Eric

Subject: Re: namespaces compatibility list
Posted by Cedric Le Goater on Wed, 07 Nov 2007 08:20:58 GMT
View Forum Message <> Reply to Message

Pavel Emelyanov wrote:
> Eric W. Biederman wrote:
>> Cedric Le Goater <clg@fr.ibm.com> writes:
>>> right. I think we can address Ulrich concerns first because we have
>>> a solution for it (which looks like unsharing all namespaces at once,
>>> here comes back the container object story :)
>> It doesn't work because we can't create a fresh mount namespace.
>>
>> We need to create all new mounts (and deny access to the old ones)
>> if we want to prevent all possibility of user space goof ups.
>>

>> While that is easy enough to build an application to do we can't
>> easily enforce that in the kernel. Currently this is all
>> CAP_SYS_ADMIN so only root can do this anyway. So we can easily
>> say don't do that then.
>>
>> Clone flag consistency checking should only be used to enforce
>> cases where the kernel side cannot support correctly. Currently
>> the kernel has no problems with the current mix and match possibilities
>> short of implementation deficiencies. So I do not see us
>> addressing Ulrich's concerns with clone flags.
>
> ACK :) Since this all is CAP_SYS_ADMIN-ed we can do with just a warning.

Fine with me.

Let's come back to the document, then.

C.

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

---

Subject: Re: namespaces compatibility list
Posted by Pavel Emelianov on Wed, 07 Nov 2007 09:29:04 GMT
View Forum Message <> Reply to Message

Cedric Le Goater wrote:
> Pavel Emelyanov wrote:
>> Eric W. Biederman wrote:
>>> Cedric Le Goater <clg@fr.ibm.com> writes:
>>>> right. I think we can address Ulrich concerns first because we have
>>>> a solution for it (which looks like unsharing all namespaces at once,
>>>> here comes back the container object story :)
>>> It doesn't work because we can't create a fresh mount namespace.
>>>
>>> We need to create all new mounts (and deny access to the old ones)
>>> if we want to prevent all possibility of user space goof ups.
>>>
>>> While that is easy enough to build an application to do we can't
>>> easily enforce that in the kernel. Currently this is all
>>> CAP_SYS_ADMIN so only root can do this anyway. So we can easily
>>> say don't do that then.
>>>
>>> Clone flag consistency checking should only be used to enforce
>>> cases where the kernel side cannot support correctly. Currently
>>> the kernel has no problems with the current mix and match possibilities

>>> short of implementation deficiencies. So I do not see us
>>> addressing Ulrich's concerns with clone flags.
>> ACK :) Since this all is CAP_SYS_ADMIN-ed we can do with just a warning.
>
> Fine with me.
>
> Let's come back to the document, then.

:) Let's. Does anybody have any comments about the current text? :)

> C.
>

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

---

Subject: Re: namespaces compatibility list
Posted by Cedric Le Goater on Wed, 07 Nov 2007 13:49:03 GMT
View Forum Message <> Reply to Message

There's also a Documentation/unshare.txt. Should we update it also ?

> --- /dev/null
> +++ b/Documentation/namespaces/compatibility-list.txt
> @@ -0,0 +1,32 @@
> + Namespaces compatibility list
> +
> +This document contains the information about the problems user
> +may have when creating tasks living in different namespaces.
> +
> +Here's the summary. This matrix shows the known problems, that
> +occur when tasks share some namespace (the columns) while living
> +in different other namespaces (the raws):

s/raws/rows/

> +
> + UTS IPC VFS PID User Net
> +UTS
> +IPC   1
> +VFS
> +PID   1 1
> +User   2
> +Net

This is dense but I can't think of a better representation.

> +1. Both the IPC and the PID namespaces provide IDs to address
> + object inside the kernel. E.g. semaphore with ipcid or
> + process group with pid.
> +
> + In both cases, tasks shouldn't try telling this id to some

s/telling/exposing/ ?

> + other task living in different namespace vid shared filesystem

.. in a different namespace via a shared filesystem

> + or IPC shmem/message. The fact is that this ID is only valid
> + within the namespace it was obtained in and may refer to some
> + other object in another namespace.
> +
> +2. Intentionnaly, two equal user ids in different user namespaces
> + should not be equal from the VFS point of view. In other
> + words, user 10 in one user namespace shouldn't have the same
> + access permissions to files, beloging to user 10 in another
> + namespace. But currently this is not so.
>
>

Thanks Pavel,

C.