Subject: [PATCH] pidns: Place under CONFIG_EXPERIMENTAL
Posted by ebiederm on Fri, 26 Oct 2007 17:40:59 GMT
View Forum Message <> Reply to Message

I finally found a chance to review the pid namespace implementation in
detail and currently it is much to easy to find issues where the
kernel does the wrong thing outside of the initial pid namespace.
At the same time the pid namespace code we have does appear
to be a good base to build on.

Therefore until the dust settles and we are certain we have the
ABI and the implementation as correct as humanly possible let's
hide the availability of process ID namespaces behind
CONFIG_EXPERIMENTAL.

Allowing users to avoid bugs, and removing a guarantee of bug
compatibility. Allowing any issues that may be found to
be fixed properly.

If CONFIG_PID_NS=N this patch will cause copy_pid_ns to
unconditionally return -EINVAL removing the availability
of multiple pid namespaces.

Signed-off-by: Eric W. Biederman <ebiederm@xmission.com>

```
---
 init/Kconfig |   12 ++++++++++++
 kernel/pid.c |    4 ++++
 2 files changed, 16 insertions(+), 0 deletions(-)

diff --git a/init/Kconfig b/init/Kconfig
index 8b88d0b..72e37c0 100644
--- a/init/Kconfig
+++ b/init/Kconfig
@@ -215,6 +215,18 @@ config USER_NS
    vservers, to use user namespaces to provide different
    user info for different servers.  If unsure, say N.

+config PID_NS
+ bool "PID Namespaces (EXPERIMENTAL)"
+ default n
+ depends on EXPERIMENTAL
+ help
+   Suport process id namespaces.  This allows having multiple
+   process with the same pid as long as they are in different
+   pid namespaces.  This is a building block of containers.
+
+   Unless you want to work with an experimental feature
+   say N here.
```

```
+
 config AUDIT
  bool "Auditing support"
  depends on NET
diff --git a/kernel/pid.c b/kernel/pid.c
index d1db36b..8a5637b 100644
--- a/kernel/pid.c
+++ b/kernel/pid.c
@@ -599,6 +599,10 @@ struct pid_namespace *copy_pid_ns(unsigned long flags, struct pid_namespace *old
  if (flags & CLONE_THREAD)
   goto out_put;

+#ifndef CONFIG_PID_NS
+ goto out_put;
+#endif
+
  new_ns = create_pid_namespace(old_ns->level + 1);
  if (!IS_ERR(new_ns))
   new_ns->parent = get_pid_ns(old_ns);
--
1.5.3.rc6.17.g1911
```

Subject: Re: [PATCH] pidns: Place under CONFIG_EXPERIMENTAL
Posted by Linus Torvalds on Fri, 26 Oct 2007 17:55:24 GMT

View Forum Message <> Reply to Message

On Fri, 26 Oct 2007, Eric W. Biederman wrote:

```
>
> +#ifndef CONFIG_PID_NS
> + goto out_put;
> +#endif
> +
```

No. We don't do crap like this. That's just horrible.

If this is conditional, then we should have conditional versions of
"create/destroy_pid_namespace()" or something.

  Linus

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

---

Subject: [PATCH] pidns: Place under CONFIG_EXPERIMENTAL (take 2)
Posted by ebiederm on Fri, 26 Oct 2007 19:35:43 GMT
View Forum Message <> Reply to Message

This is my trivial patch to swat innumerable little bugs
with a single blow.

After some intensive review (my apologies for not having
gotten to this sooner) what we have looks like a good
base to build on with the current pid namespace code but
it is not complete, and it is still much to simple to find
issues where the kernel does the wrong thing outside of
the initial pid namespace.

Until the dust settles and we are certain we have the ABI and
the implementation is as correct as humanly possible let's keep
process ID namespaces behind CONFIG_EXPERIMENTAL.

Allowing us the option of fixing any ABI or other bugs
we find as long as they are minor.

Allowing users of the kernel to avoid those bugs simply
by ensuring their kernel does not have support for multiple
pid namespaces.

Signed-off-by: Eric W. Biederman <ebiederm@xmission.com>

```
---
 include/linux/pid_namespace.h |   22 ++++++++++++++++++++++
 init/Kconfig                  |   12 ++++++++++++
 kernel/pid.c                  |    2 ++
 3 files changed, 36 insertions(+), 0 deletions(-)

diff --git a/include/linux/pid_namespace.h b/include/linux/pid_namespace.h
index 0135c76..0227e68 100644
--- a/include/linux/pid_namespace.h
+++ b/include/linux/pid_namespace.h
@@ -29,6 +29,7 @@ struct pid_namespace {

 extern struct pid_namespace init_pid_ns;

+#ifdef CONFIG_PID_NS
 static inline struct pid_namespace *get_pid_ns(struct pid_namespace *ns)
 {
  if (ns != &init_pid_ns)
```

```
@@ -45,6 +46,27 @@ static inline void put_pid_ns(struct pid_namespace *ns)
   kref_put(&ns->kref, free_pid_ns);
 }

+#else /* !CONFIG_PID_NS */
+#include <linux/err.h>
+
+static inline struct pid_namespace *get_pid_ns(struct pid_namespace *ns)
+{
+ return ns;
+}
+
+static inline struct pid_namespace *copy_pid_ns(unsigned long flags, struct pid_namespace *ns)
+{
+ if (flags & CLONE_NEWPID)
+  ns = ERR_PTR(-EINVAL);
+ return ns;
+}
+
+static inline void put_pid_ns(struct pid_namespace *ns)
+{
+}
+
+#endif /* CONFIG_PID_NS */
+
 static inline struct pid_namespace *task_active_pid_ns(struct task_struct *tsk)
 {
  return tsk->nsproxy->pid_ns;
diff --git a/init/Kconfig b/init/Kconfig
index 8b88d0b..72e37c0 100644
--- a/init/Kconfig
+++ b/init/Kconfig
@@ -215,6 +215,18 @@ config USER_NS
    vservers, to use user namespaces to provide different
    user info for different servers.  If unsure, say N.

+config PID_NS
+ bool "PID Namespaces (EXPERIMENTAL)"
+ default n
+ depends on EXPERIMENTAL
+ help
+   Suport process id namespaces.  This allows having multiple
+   process with the same pid as long as they are in different
+   pid namespaces.  This is a building block of containers.
+
+   Unless you want to work with an experimental feature
+   say N here.
+
 config AUDIT
  bool "Auditing support"
  depends on NET
diff --git a/kernel/pid.c b/kernel/pid.c
index d1db36b..f815455 100644
--- a/kernel/pid.c
+++ b/kernel/pid.c
@@ -537,6 +537,7 @@ err_alloc:
  return NULL;
 }

+#ifdef CONFIG_PID_NS
 static struct pid_namespace *create_pid_namespace(int level)
 {
  struct pid_namespace *ns;
@@ -621,6 +622,7 @@ void free_pid_ns(struct kref *kref)
  if (parent != NULL)
   put_pid_ns(parent);
 }
+#endif /* CONFIG_PID_NS */

 void zap_pid_ns_processes(struct pid_namespace *pid_ns)
 {
--
1.5.3.rc6.17.g1911
```

Subject: Re: [PATCH] pidns: Place under CONFIG_EXPERIMENTAL (take 2)
Posted by Kir Kolyshkin on Fri, 26 Oct 2007 20:58:32 GMT

View Forum Message <> Reply to Message

Eric,

Could you please hold off the horses a bit and wait till Pavel Emelyanov returns? It means next Monday; he's currently at a conference whose organisers don't provide internet access.

I feel it makes great sense to review/discuss patches first on containers@ first before submitting directly to lkml/Linus.

Speaking of this particular patch -- I don't understand how you fix "innumerable little bugs" by providing stubs instead of real functions.
Sent from my BlackBerry; please reply to kir@openvz.org

-----Original Message-----
From: devel-bounces@openvz.org <devel-bounces@openvz.org>
To: Linus Torvalds <torvalds@linux-foundation.org>
CC: Linux Containers <containers@lists.osdl.org>; Andrew Morton <akpm@linux-foundation.org>; linux-kernel@vger.kernel.org <linux-kernel@vger.kernel.org>
Sent: Fri Oct 26 15:35:43 2007
Subject: [Devel] [PATCH] pidns: Place under CONFIG_EXPERIMENTAL (take 2)

This is my trivial patch to swat innumerable little bugs
with a single blow.

After some intensive review (my apologies for not having
gotten to this sooner) what we have looks like a good
base to build on with the current pid namespace code but
it is not complete, and it is still much to simple to find
issues where the kernel does the wrong thing outside of
the initial pid namespace.

Until the dust settles and we are certain we have the ABI and
the implementation is as correct as humanly possible let's keep
process ID namespaces behind CONFIG_EXPERIMENTAL.

Allowing us the option of fixing any ABI or other bugs
we find as long as they are minor.

Allowing users of the kernel to avoid those bugs simply
by ensuring their kernel does not have support for multiple
pid namespaces.

Signed-off-by: Eric W. Biederman <ebiederm@xmission.com>

```
---
 include/linux/pid_namespace.h |   22 ++++++++++++++++++++++
 init/Kconfig                  |   12 ++++++++++++
 kernel/pid.c                  |    2 ++
 3 files changed, 36 insertions(+), 0 deletions(-)

diff --git a/include/linux/pid_namespace.h b/include/linux/pid_namespace.h
index 0135c76..0227e68 100644
--- a/include/linux/pid_namespace.h
+++ b/include/linux/pid_namespace.h
@@ -29,6 +29,7 @@ struct pid_namespace {

 extern struct pid_namespace init_pid_ns;

+#ifdef CONFIG_PID_NS
 static inline struct pid_namespace *get_pid_ns(struct pid_namespace *ns)
 {
  if (ns != &init_pid_ns)
@@ -45,6 +46,27 @@ static inline void put_pid_ns(struct pid_namespace *ns)
   kref_put(&ns->kref, free_pid_ns);
 }

+#else /* !CONFIG_PID_NS */
+#include <linux/err.h>
+
+static inline struct pid_namespace *get_pid_ns(struct pid_namespace *ns)
+{
+ return ns;
+}
+
+static inline struct pid_namespace *copy_pid_ns(unsigned long flags, struct pid_namespace *ns)
+{
+ if (flags & CLONE_NEWPID)
+  ns = ERR_PTR(-EINVAL);
+ return ns;
+}
+
+static inline void put_pid_ns(struct pid_namespace *ns)
+{
+}
+
+#endif /* CONFIG_PID_NS */
+
 static inline struct pid_namespace *task_active_pid_ns(struct task_struct *tsk)
 {
  return tsk->nsproxy->pid_ns;
diff --git a/init/Kconfig b/init/Kconfig
index 8b88d0b..72e37c0 100644
--- a/init/Kconfig
+++ b/init/Kconfig
@@ -215,6 +215,18 @@ config USER_NS
    vservers, to use user namespaces to provide different
    user info for different servers.  If unsure, say N.

+config PID_NS
+ bool "PID Namespaces (EXPERIMENTAL)"
+ default n
+ depends on EXPERIMENTAL
+ help
+   Suport process id namespaces.  This allows having multiple
+   process with the same pid as long as they are in different
+   pid namespaces.  This is a building block of containers.
+
+   Unless you want to work with an experimental feature
+   say N here.
```

```
+
 config AUDIT
  bool "Auditing support"
  depends on NET
diff --git a/kernel/pid.c b/kernel/pid.c
index d1db36b..f815455 100644
--- a/kernel/pid.c
+++ b/kernel/pid.c
@@ -537,6 +537,7 @@ err_alloc:
  return NULL;
 }

+#ifdef CONFIG_PID_NS
 static struct pid_namespace *create_pid_namespace(int level)
 {
  struct pid_namespace *ns;
@@ -621,6 +622,7 @@ void free_pid_ns(struct kref *kref)
  if (parent != NULL)
   put_pid_ns(parent);
 }
+#endif /* CONFIG_PID_NS */

 void zap_pid_ns_processes(struct pid_namespace *pid_ns)
 {
--
1.5.3.rc6.17.g1911
```

Subject: Re: [PATCH] pidns: Place under CONFIG_EXPERIMENTAL (take 2)
Posted by ebiederm on Fri, 26 Oct 2007 21:59:29 GMT

View Forum Message <> Reply to Message

"Kir Kolyshkin" <kir@swsoft.com> writes:

> Eric,
>
> Could you please hold off the horses a bit and wait till Pavel Emelyanov
> returns? It means next Monday; he's currently at a conference whose organisers
> don't provide internet access.

When we decided to go top down (i.e. user interface first) instead of
bottom up with the pid namespace implementation it was my
understanding that we had agreed we would make the pid namespaces
depend on CONFIG_EXPERIMENTAL so that we wouldn't be stuck forever
supporting early ABI mistakes.

So to my knowledge the conversation has already happened. I believe
something in the confusion of trying to use these options to shrink
the kernel and the futility of that, caused whatever config options
we had before to be dropped.

Further I was happy to let Pavel and Suka work on this code because
the appeared to know what they were doing and it freed me to do other
things. I don't think there are any mysteries in what we are trying
to do that I need them to explain.

> I feel it makes great sense to review/discuss patches first on containers@
> first before submitting directly to lkml/Linus.

My feel before starting to review the pid namespace patches was that
the work was essentially done except a handful of minor details. Upon
closer examination, I found that not to be the case. My rough fix
queue had 25 or so patches as of last night to fix pid namespace
issues.

I have no confidence that we will fix all of the pid namespaces issues
before 2.6.24-final. I do think we can get most of them fixed.

Given that most of the remaining issues are integration issues
with the rest of the kernel having the code merged should make
it much easier to see what is going on and actually fix things.
So I am not in favor of reverting this code despite seeing numerous
problems.

> Speaking of this particular patch -- I don't understand how you fix
> "innumerable little bugs" by providing stubs instead of real functions.
> Sent from my BlackBerry; please reply to kir@openvz.org

It doesn't fix the bugs it avoids them because there is no way to
get to the them and trigger them. So far I have yet to find a bug
that is a problem with only a single pid namespace in the kernel.

Since everyone agrees that there are at least some deficiencies in
the current pid namespace I think this makes sense, to mark
the code as EXPERIMENTAL and have a way for people who care to
shut it off just so they don't have to worry about new issues.

As far as how the config option is implemented I don't much care
so long as I get the -EINVAL when I pass CLONE_NEWPID as root.

Essentially this patch is part of a defense in depth against pid
namespace problems hitting people. This patch is my first line
of defense. Actually fixing all of the rest of the known bugs
is the other line.

Eric

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

---

Subject: Re: [PATCH] pidns: Place under CONFIG_EXPERIMENTAL (take 2)
Posted by ebiederm on Fri, 26 Oct 2007 22:34:28 GMT

View Forum Message <> Reply to Message

"Kir Kolyshkin" <kir@swsoft.com> writes:

> Speaking of this particular patch -- I don't understand how you fix
> "innumerable little bugs" by providing stubs instead of real functions.

I think it would be a disaster to use pid namespaces as currently
implemented 2.6.24-rc1 in a production environment.

There are lots of little bugs and I am certain know one knows what
they are all right now.

Therefore not creating more then the initial pid namespace in a
production environment sounds like the responsible thing to do for
2.6.24.

This patch enables people to guarantee they don't run software
that will create additional pid namespaces and expose them to
the bugs we have not yet found, and it says look out. Don't
mess with this unless you know what you are doing.

That message of Look out be careful is what I really care
about sending to users of the kernel.

The best way I know to do that is to mark the feature

(EXPERIMENTAL) and have a config option for the feature
that depends on CONFIG_EXPERIMENTAL.

Eric

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

---

Subject: Re: [PATCH] pidns: Place under CONFIG_EXPERIMENTAL (take 2)
Posted by Adrian Bunk on Sat, 27 Oct 2007 00:24:49 GMT
View Forum Message <> Reply to Message

On Fri, Oct 26, 2007 at 03:59:29PM -0600, Eric W. Biederman wrote:
> "Kir Kolyshkin" <kir@swsoft.com> writes:
>
> > Eric,
> >
> > Could you please hold off the horses a bit and wait till Pavel Emelyanov
> > returns? It means next Monday; he's currently at a conference whose organisers
> > don't provide internet access.
>
> When we decided to go top down (i.e. user interface first) instead of
> bottom up with the pid namespace implementation it was my
> understanding that we had agreed we would make the pid namespaces
> depend on CONFIG_EXPERIMENTAL so that we wouldn't be stuck forever
> supporting early ABI mistakes.
>...

CONFIG_EXPERIMENTAL is a weak hint that some code might not (yet) be in
a perfect state, but it does not have any semantics regarding
userspace ABIs.

A dependency on BROKEN seems more appropriate.

> Eric

cu
Adrian

--

"Is there not promise of rain?" Ling Tan asked suddenly out
 of the darkness. There had been need of rain for many days.
"Only a promise," Lao Er said.
                                Pearl S. Buck - Dragon Seed

Subject: Re: [PATCH] pidns: Place under CONFIG_EXPERIMENTAL (take 2)
Posted by ebiederm on Sat, 27 Oct 2007 01:31:04 GMT

View Forum Message <> Reply to Message

Adrian Bunk <bunk@kernel.org> writes:

> CONFIG_EXPERIMENTAL is a weak hint that some code might not (yet) be in
> a perfect state, but it does not have any semantics regarding
> userspace ABIs.

Code that might not (yet) be in a perfect state sums it up pretty
well. There is not plan or expectation to change magic numbers or
things like that but the behavior of the code may change as bug and
such are fixed.

> A dependency on BROKEN seems more appropriate.

Since you can't select that it seems a little strong.

...

One of the things we talked about at the kernel summit is how
almost inevitably when new user space interfaces are introduced
there are problems. Someone over looks something, something
gets changed to get through the review something like that. There was
discussion but no consensus on how do introduce something like that
but still allow our selves the ability to fix it. Keeping the
code under CONFIG_EXPERIMENTAL is the best suggest I have seen
so far. Even if it is slightly expanding the definition of
CONFIG_EXPERIMENTAL.

Every place the kernel uses pids is a huge scope. It is very
easy to miss something with a scope that wide. So the engineer
in me says chances of us missing something are pretty huge.
Especially since I know we have bugs in -rc1.

If it turns out that making multiple pid namespaces work is
hopeless we can always change the dependency to BROKEN later.

As for ABI and behavioral characteristics currently that is
largely well defined. You are supposed to get the exact
same thing as you would on the system if you only had a

single pid namespace. The places where we have questionable
semantics is in the intersections between namespaces.

That is not an area I am willing to stand up and say we got
it perfect the first time, I'm going to support our behavior
quirks forever if I can find soon enough. Very few applications
will care, and the differences might really matter at some point.

So does any one have any better suggestions on how to deal
with features that are enough work you aren't going to get them
perfect the first time. You need the code merged or else you
can not complete the feature (too many dependencies through out the
code). You want early adopters to start playing with the feature
so you can get feed back and you can test to make certain everything
is going ok. You want to retain the ability to fix implementation
details even if those details are user visible, for a time until
things seem as good as they can reasonably get.

Roughly that sounds like CONFIG_EXPERIMENTAL to me. But I would
be happy to hear if someone has a better idea.

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

---

Subject: Re: [PATCH] pidns: Place under CONFIG_EXPERIMENTAL (take 2)
Posted by Adrian Bunk on Sat, 27 Oct 2007 02:04:08 GMT
View Forum Message <> Reply to Message

On Fri, Oct 26, 2007 at 07:31:04PM -0600, Eric W. Biederman wrote:
> Adrian Bunk <bunk@kernel.org> writes:
>
> > CONFIG_EXPERIMENTAL is a weak hint that some code might not (yet) be in
> > a perfect state, but it does not have any semantics regarding
> > userspace ABIs.
>
> Code that might not (yet) be in a perfect state sums it up pretty
> well. There is not plan or expectation to change magic numbers or
> things like that but the behavior of the code may change as bug and
> such are fixed.
>
> > A dependency on BROKEN seems more appropriate.
>
> Since you can't select that it seems a little strong.
>

> ...
>
> One of the things we talked about at the kernel summit is how
> almost inevitably when new user space interfaces are introduced
> there are problems. Someone over looks something, something
> gets changed to get through the review something like that. There was
> discussion but no consensus on how do introduce something like that
> but still allow our selves the ability to fix it. Keeping the
> code under CONFIG_EXPERIMENTAL is the best suggest I have seen
> so far. Even if it is slightly expanding the definition of
> CONFIG_EXPERIMENTAL.
>
> Every place the kernel uses pids is a huge scope. It is very
> easy to miss something with a scope that wide. So the engineer
> in me says chances of us missing something are pretty huge.
> Especially since I know we have bugs in -rc1.
>
> If it turns out that making multiple pid namespaces work is
> hopeless we can always change the dependency to BROKEN later.

No, we can't after 2.6.24 got released.

Let me make an example:
- looking at the timelines, e.g. Ubuntu 8.04 LTS is likely to
  ship with 2.6.24
- this experimental feature might be enabled there
- this Ubuntu release with this kernel will be supported and used
  for five years

> As for ABI and behavioral characteristics currently that is
> largely well defined. You are supposed to get the exact
> same thing as you would on the system if you only had a
> single pid namespace. The places where we have questionable
> semantics is in the intersections between namespaces.
>
> That is not an area I am willing to stand up and say we got
> it perfect the first time, I'm going to support our behavior
> quirks forever if I can find soon enough. Very few applications
> will care, and the differences might really matter at some point.
>
> So does any one have any better suggestions on how to deal
> with features that are enough work you aren't going to get them
> perfect the first time. You need the code merged or else you
> can not complete the feature (too many dependencies through out the
> code). You want early adopters to start playing with the feature
> so you can get feed back and you can test to make certain everything
> is going ok. You want to retain the ability to fix implementation
> details even if those details are user visible, for a time until

> things seem as good as they can reasonably get.
>
> Roughly that sounds like CONFIG_EXPERIMENTAL to me. But I would
> be happy to hear if someone has a better idea.

There is a difference between "complete the feature" and "early adopters
to start playing with the feature" on the one side, and making something
available in a released kernel on the other side.

For development and playing with it it can depend on BROKEN (perhaps
with the dependency removed through the first -rc kernels), but as soon
as it's available in a -final kernel the ABI is fixed.

> Eric

cu
Adrian

--

       "Is there not promise of rain?" Ling Tan asked suddenly out
        of the darkness. There had been need of rain for many days.
       "Only a promise," Lao Er said.
                                       Pearl S. Buck - Dragon Seed

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

---

Subject: Re: [PATCH] pidns: Place under CONFIG_EXPERIMENTAL (take 2)
Posted by akpm on Sat, 27 Oct 2007 02:18:45 GMT
View Forum Message <> Reply to Message

> On Sat, 27 Oct 2007 04:04:08 +0200 Adrian Bunk <bunk@kernel.org> wrote:
> > be happy to hear if someone has a better idea.
>
> There is a difference between "complete the feature" and "early adopters
> to start playing with the feature" on the one side, and making something
> available in a released kernel on the other side.
>
> For development and playing with it it can depend on BROKEN (perhaps
> with the dependency removed through the first -rc kernels), but as soon
> as it's available in a -final kernel the ABI is fixed.
>

Yes, if we're not 100% certain that the interfaces are correnct and unchanging and that the implementation is solid, we should disable the feature at Kconfig time.

The best option would be to fix things asap. But assuming that option isn't
reasonable and/or safe, we can slip a `depends on BROKEN' into -rc6 then
resume development for 2.6.25.

Subject: Re: [PATCH] pidns: Place under CONFIG_EXPERIMENTAL (take 2)
Posted by ebiederm on Sat, 27 Oct 2007 03:46:59 GMT

View Forum Message <> Reply to Message

Andrew Morton <akpm@linux-foundation.org> writes:

>> On Sat, 27 Oct 2007 04:04:08 +0200 Adrian Bunk <bunk@kernel.org> wrote:
>> > be happy to hear if someone has a better idea.
>>
>> There is a difference between "complete the feature" and "early adopters
>> to start playing with the feature" on the one side, and making something
>> available in a released kernel on the other side.
>>
>> For development and playing with it it can depend on BROKEN (perhaps
>> with the dependency removed through the first -rc kernels), but as soon
>> as it's available in a -final kernel the ABI is fixed.
>>
>
> Yes, if we're not 100% certain that the interfaces are correnct and unchanging
> and that the implementation is solid, we should disable the feature at Kconfig
> time.

Reasonable. So far things look good for a single pid namespace. Multiple
pid namespaces look iffy.

> The best option would be to fix things asap. But assuming that option isn't
> reasonable and/or safe, we can slip a `depends on BROKEN' into -rc6 then
> resume development for 2.6.25.

I think we can make a lot of progress but there is enough development
yet to do to reach the target of correct and unchanging interfaces,
with a solid interface. That unless we achieve a breakthrough I
don't see us achieving that target for 2.6.24.

The outstanding issues I can think of off the top of my head:
- signal handling for init on secondary pid namespaces.

- Properly setting si_pid on signals that cross namespaces.
- The kthread API conversion so we don't get kernel threads
  trapped in pid namespaces and make them unfreeable.
- At fork time I think we are doing a little bit too much work
  in setting the session and the pgrp, and removing the controlling
  tty.
- AF_unix domain credential passing.
- misc pid vs vpid sorting out (autofs autofs4, coda, arch specific
  syscalls, others?)
- Removal of task->pid, task->tgid, task->signal->__pgrp,
  tsk->signal->__session or some other way to ensure that we have
  touched and converted all of the kernel pid handling.
- flock pid handling.

It hurts me to even ponder what thinking makes it that
CONFIG_EXPERIMENTAL isn't enough to keep a stable distro
from shipping the code in their stable kernel, and locking us into
trouble.

With that said. I think I should just respin the patchset now and add
the "depends on BROKEN".

The user namespace appears to need that treatment as well.

The network namespace has so little there and it already depends
on !SYSFS so I don't think we are going to run into any trouble
with it. Happily I managed to parse that problem differently,
so I could slice of the parts of the networking stack that
had not been converted.

Eric

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

---

Subject: Re: [PATCH] pidns: Place under CONFIG_EXPERIMENTAL (take 2)
Posted by Adrian Bunk on Sat, 27 Oct 2007 04:03:46 GMT

View Forum Message <> Reply to Message

On Fri, Oct 26, 2007 at 09:46:59PM -0600, Eric W. Biederman wrote:
>...
> It hurts me to even ponder what thinking makes it that
> CONFIG_EXPERIMENTAL isn't enough to keep a stable distro
> from shipping the code in their stable kernel, and locking us into
> trouble.
>...

There isn't any hard semantics behind what is marked EXPERIMENTAL and
what not. In it's current state, we could even consider removing the
EXPERIMENTAL option and all dependencies on EXPERIMENTAL.

Currently CONFIG_EXPERIMENTAL=n would cost a distribution a three digit
number of device drivers plus several features like e.g. NFSv4.

> Eric

cu
Adrian

--

"Is there not promise of rain?" Ling Tan asked suddenly out
 of the darkness. There had been need of rain for many days.
"Only a promise," Lao Er said.
Pearl S. Buck - Dragon Seed

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

Subject: Re: [PATCH] pidns: Place under CONFIG_EXPERIMENTAL (take 2)
Posted by ebiederm on Sat, 27 Oct 2007 04:40:12 GMT
View Forum Message <> Reply to Message

Adrian Bunk <bunk@kernel.org> writes:

> There isn't any hard semantics behind what is marked EXPERIMENTAL and
> what not. In it's current state, we could even consider removing the
> EXPERIMENTAL option and all dependencies on EXPERIMENTAL.

Well I do know at least some of the things that depend on experimental
are legitimate.

I wonder if the problem is that we don't police experimental well
enough.

> Currently CONFIG_EXPERIMENTAL=n would cost a distribution a three digit
> number of device drivers plus several features like e.g. NFSv4.

I can see a distribution carefully cherry picking things, that the
have an intimate knowledge about out of experimental but it doesn't
sound right for taking things out of EXPERIMENTAL to be routine.

I know I'm a little slow about getting around to it but when ever I
have a feature that isn't EXPERIMENTAL anymore I remove the tag.

Eric

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

---

Subject: Re: [PATCH] pidns: Place under CONFIG_EXPERIMENTAL (take 2)
Posted by akpm on Sat, 27 Oct 2007 04:40:46 GMT

View Forum Message <> Reply to Message

On Fri, 26 Oct 2007 21:46:59 -0600 ebiederm@xmission.com (Eric W. Biederman) wrote:

> Andrew Morton <akpm@linux-foundation.org> writes:
>
> >> On Sat, 27 Oct 2007 04:04:08 +0200 Adrian Bunk <bunk@kernel.org> wrote:
> >> > be happy to hear if someone has a better idea.
> >>
> >> There is a difference between "complete the feature" and "early adopters
> >> to start playing with the feature" on the one side, and making something
> >> available in a released kernel on the other side.
> >>
> >> For development and playing with it it can depend on BROKEN (perhaps
> >> with the dependency removed through the first -rc kernels), but as soon
> >> as it's available in a -final kernel the ABI is fixed.
> >>
> >
> > Yes, if we're not 100% certain that the interfaces are corrénct and unchanging
> > and that the implementation is solid, we should disable the feature at Kconfig
> > time.
>
> Reasonable. So far things look good for a single pid namespace. Multiple
> pid namespaces look iffy.
>
> > The best option would be to fix things asap. But assuming that option isn't
> > reasonable and/or safe, we can slip a `depends on BROKEN' into -rc6 then
> > resume development for 2.6.25.
>
> I think we can make a lot of progress but there is enough development
> yet to do to reach the target of correct and unchanging interfaces,
> with a solid interface. That unless we achieve a breakthrough I
> don't see us achieving that target for 2.6.24.
>
> The outstanding issues I can think of off the top of my head:

> - signal handling for init on secondary pid namespaces.
> - Properly setting si_pid on signals that cross namespaces.
> - The kthread API conversion so we don't get kernel threads
>   trapped in pid namespaces and make them unfreeable.
> - At fork time I think we are doing a little bit too much work
>   in setting the session and the pgrp, and removing the controlling
>   tty.
> - AF_unix domain credential passing.
> - misc pid vs vpid sorting out (autofs autofs4, coda, arch specific
>   syscalls, others?)
> - Removal of task->pid, task->tgid, task->signal->__pgrp,
>   tsk->signal->__session or some other way to ensure that we have
>   touched and converted all of the kernel pid handling.
> - flock pid handling.

Given that a lot of this development will hopefully happen over the next
two months, ...

> It hurts me to even ponder what thinking makes it that
> CONFIG_EXPERIMENTAL isn't enough to keep a stable distro
> from shipping the code in their stable kernel, and locking us into
> trouble.
>
> With that said. I think I should just respin the patchset now and add
> the "depends on BROKEN".

it doesn't make sense to make it all dependent upon BROKEN now. Better
would be to make it dependant upon CONFIG_SOMETHING_ELSE now, which depends
upon EXPERIMENTAL and which will, around -rc6, be changed to depend upon
BROKEN.

If that makes sense.

It's all a bit unusual and complex, but this is an exceptional set of
features - let's hang in there.

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

---

Subject: Re: [PATCH] pidns: Place under CONFIG_EXPERIMENTAL (take 2)
Posted by Adrian Bunk on Sat, 27 Oct 2007 05:17:24 GMT
View Forum Message <> Reply to Message

On Fri, Oct 26, 2007 at 10:40:12PM -0600, Eric W. Biederman wrote:

> Adrian Bunk <bunk@kernel.org> writes:
>
> > There isn't any hard semantics behind what is marked EXPERIMENTAL and
> > what not. In it's current state, we could even consider removing the
> > EXPERIMENTAL option and all dependencies on EXPERIMENTAL.
>
> Well I do know at least some of the things that depend on experimental
> are legitimate.
>
> I wonder if the problem is that we don't police experimental well
> enough.
>
> > Currently CONFIG_EXPERIMENTAL=n would cost a distribution a three digit
> > number of device drivers plus several features like e.g. NFSv4.
>
> I can see a distribution carefully cherry picking things, that the
> have an intimate knowledge about out of experimental but it doesn't
> sound right for taking things out of EXPERIMENTAL to be routine.
>
> I know I'm a little slow about getting around to it but when ever I
> have a feature that isn't EXPERIMENTAL anymore I remove the tag.

Part of the picture might be that code that was included into the kernel
usually is in a state that it works at least most time for most of the
people.

And when you think about distributions, it's hard to imagine why a
distribution should not enable more or less all EXPERIMENTAL device
drivers - an EXPERIMENTAL driver is much better than no driver for this
hardware at all.

> Eric

cu
Adrian

--

"Is there not promise of rain?" Ling Tan asked suddenly out
 of the darkness. There had been need of rain for many days.
"Only a promise," Lao Er said.
Pearl S. Buck - Dragon Seed

____

Subject: Re: [PATCH] pidns: Place under CONFIG_EXPERIMENTAL (take 2)
Posted by ebiederm on Sat, 27 Oct 2007 07:41:43 GMT
View Forum Message <> Reply to Message

Andrew Morton <akpm@linux-foundation.org> writes:

> Given that a lot of this development will hopefully happen over the next
> two months, ...

A lot. Various pieces are a major effort in their own right.
Improving the kthread API so it can be used universally and
allow removal all of the kernel_thread users.
Reducing to an absolute minimum usage of pid_t.

I know several of the things with signal handling had Oleg
scratching his head.

There is enough development there I question if the code will even be
canidates for merging into 2.6.24.

I can imagine an -mm tree that has everything ready to go in
the next two months.

>> It hurts me to even ponder what thinking makes it that
>> CONFIG_EXPERIMENTAL isn't enough to keep a stable distro
>> from shipping the code in their stable kernel, and locking us into
>> trouble.
>>
>> With that said. I think I should just respin the patchset now and add
>> the "depends on BROKEN".
>
> it doesn't make sense to make it all dependent upon BROKEN now. Better
> would be to make it dependant upon CONFIG_SOMETHING_ELSE now, which depends
> upon EXPERIMENTAL and which will, around -rc6, be changed to depend upon
> BROKEN.

So we now have my patch which makes it depend on CONFIG_PID_NS.
Which is what started this thread.

> If that makes sense.

Yes.

> It's all a bit unusual and complex, but this is an exceptional set of
> features - let's hang in there.

Sure. One small step at a time.

- Step One add a config option.

Eric

Subject: Re: [PATCH] pidns: Place under CONFIG_EXPERIMENTAL (take 2)
Posted by Jeremy Fitzhardinge on Sun, 28 Oct 2007 16:12:34 GMT

View Forum Message <> Reply to Message

Eric W. Biederman wrote:
> Roughly that sounds like CONFIG_EXPERIMENTAL to me. But I would
> be happy to hear if someone has a better idea.
>

Rather than overload an existing config option, why not add one with the
specific semantics you want: CONFIG_UNSTABLE_UABI. The problem seems
like one which which may occur again, though one hopes not too often. I
guess the risk is that people will leave their subsystems depending on
it permanently (sysfs?), so it ends up being set all the time and
becomes as useless as EXPERIMENTAL...

    J

Subject: Re: [PATCH] pidns: Place under CONFIG_EXPERIMENTAL (take 2)
Posted by Adrian Bunk on Sun, 28 Oct 2007 17:00:08 GMT

View Forum Message <> Reply to Message

On Sun, Oct 28, 2007 at 09:12:34AM -0700, Jeremy Fitzhardinge wrote:
> Eric W. Biederman wrote:
> > Roughly that sounds like CONFIG_EXPERIMENTAL to me. But I would
> > be happy to hear if someone has a better idea.
>
> Rather than overload an existing config option, why not add one with the
> specific semantics you want: CONFIG_UNSTABLE_UABI. The problem seems
> like one which which may occur again, though one hopes not too often. I
> guess the risk is that people will leave their subsystems depending on
> it permanently (sysfs?), so it ends up being set all the time and
> becomes as useless as EXPERIMENTAL...

Then let SYSFS depend on UNSTABLE_UABI for the next 10 years and we have
an excuse for breaking the ABI with each new kernel...

Either the ABI is stable or it should not be exposed to users at all.

>   J

cu
Adrian

--

"Is there not promise of rain?" Ling Tan asked suddenly out
of the darkness. There had been need of rain for many days.
"Only a promise," Lao Er said.
Pearl S. Buck - Dragon Seed

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

Subject: Re: [PATCH] pidns: Place under CONFIG_EXPERIMENTAL (take 2)
Posted by ebiederm on Sun, 28 Oct 2007 18:31:30 GMT

View Forum Message <> Reply to Message

Adrian Bunk <bunk@kernel.org> writes:

> On Sun, Oct 28, 2007 at 09:12:34AM -0700, Jeremy Fitzhardinge wrote:
>> Eric W. Biederman wrote:
>> > Roughly that sounds like CONFIG_EXPERIMENTAL to me. But I would
>> > be happy to hear if someone has a better idea.
>>
>> Rather than overload an existing config option, why not add one with the
>> specific semantics you want: CONFIG_UNSTABLE_UABI. The problem seems
>> like one which which may occur again, though one hopes not too often. I
>> guess the risk is that people will leave their subsystems depending on
>> it permanently (sysfs?), so it ends up being set all the time and
>> becomes as useless as EXPERIMENTAL...
>
> Then let SYSFS depend on UNSTABLE_UABI for the next 10 years and we have
> an excuse for breaking the ABI with each new kernel...
>
> Either the ABI is stable or it should not be exposed to users at all.

If we need a new config for it. CONFIG_IMMATURE is the closest I can think of.

Horrible ABI damage like reassigning magic numbers won't happen.
Argument passing won't change (unless it is simply impossible).
However there are little details which might change. Things like
error codes or the behavior in unlikely circumstances. Bugs are
expected to exist and so we bug compatibility isn't expect to be
maintained.

The bottom line is that when things are new they are immature and
very rarely do they come out of the gate perfect. That is something
we need to recognize and deal with. We need to communicate this
to our users (distros and other developers) and we need to work
quickly to get the kinks out so that the code really is mature.

I'm not looking for excuses to mess up or do the wrong thing.
I just know that as an engineer getting it perfect out of the
gate isn't going to happen. So I'm doing what I can to ask
people to ease into using something new.

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

---

Subject: Re: [PATCH] pidns: Place under CONFIG_EXPERIMENTAL (take 2)
Posted by Kirill Korotaev on Mon, 29 Oct 2007 07:49:22 GMT

View Forum Message <> Reply to Message

Can you please send namespace related patches to containers@ ML first
before sending them to Linus/Andrew?

Thanks,
Kirill

Eric W. Biederman wrote:
> This is my trivial patch to swat innumerable little bugs
> with a single blow.
>
> After some intensive review (my apologies for not having
> gotten to this sooner) what we have looks like a good
> base to build on with the current pid namespace code but
> it is not complete, and it is still much to simple to find
> issues where the kernel does the wrong thing outside of
> the initial pid namespace.

```
>
> Until the dust settles and we are certain we have the ABI and
> the implementation is as correct as humanly possible let's keep
> process ID namespaces behind CONFIG_EXPERIMENTAL.
>
> Allowing us the option of fixing any ABI or other bugs
> we find as long as they are minor.
>
> Allowing users of the kernel to avoid those bugs simply
> by ensuring their kernel does not have support for multiple
> pid namespaces.
>
> Signed-off-by: Eric W. Biederman <ebiederm@xmission.com>
> ---
>  include/linux/pid_namespace.h |   22 ++++++++++++++++++++++
>  init/Kconfig                  |   12 ++++++++++++
>  kernel/pid.c                  |    2 ++
>  3 files changed, 36 insertions(+), 0 deletions(-)
>
> diff --git a/include/linux/pid_namespace.h b/include/linux/pid_namespace.h
> index 0135c76..0227e68 100644
> --- a/include/linux/pid_namespace.h
> +++ b/include/linux/pid_namespace.h
> @@ -29,6 +29,7 @@ struct pid_namespace {
>
>  extern struct pid_namespace init_pid_ns;
>
> +#ifdef CONFIG_PID_NS
>  static inline struct pid_namespace *get_pid_ns(struct pid_namespace *ns)
>  {
>   if (ns != &init_pid_ns)
> @@ -45,6 +46,27 @@ static inline void put_pid_ns(struct pid_namespace *ns)
>    kref_put(&ns->kref, free_pid_ns);
>  }
>
> +#else /* !CONFIG_PID_NS */
> +#include <linux/err.h>
> +
> +static inline struct pid_namespace *get_pid_ns(struct pid_namespace *ns)
> +{
> + return ns;
> +}
> +
> +static inline struct pid_namespace *copy_pid_ns(unsigned long flags, struct pid_namespace
*ns)
> +{
> + if (flags & CLONE_NEWPID)
> +  ns = ERR_PTR(-EINVAL);
> + return ns;
> +}
> +
> +static inline void put_pid_ns(struct pid_namespace *ns)
> +{
> +}
> +
> +#endif /* CONFIG_PID_NS */
> +
>  static inline struct pid_namespace *task_active_pid_ns(struct task_struct *tsk)
>  {
>   return tsk->nsproxy->pid_ns;
> diff --git a/init/Kconfig b/init/Kconfig
> index 8b88d0b..72e37c0 100644
> --- a/init/Kconfig
> +++ b/init/Kconfig
> @@ -215,6 +215,18 @@ config USER_NS
>     vservers, to use user namespaces to provide different
>     user info for different servers.  If unsure, say N.
>
> +config PID_NS
> + bool "PID Namespaces (EXPERIMENTAL)"
> + default n
> + depends on EXPERIMENTAL
> + help
> +   Suport process id namespaces.  This allows having multiple
> +   process with the same pid as long as they are in different
> +   pid namespaces.  This is a building block of containers.
> +
> +   Unless you want to work with an experimental feature
> +   say N here.
> +
>  config AUDIT
>   bool "Auditing support"
>   depends on NET
> diff --git a/kernel/pid.c b/kernel/pid.c
> index d1db36b..f815455 100644
> --- a/kernel/pid.c
> +++ b/kernel/pid.c
> @@ -537,6 +537,7 @@ err_alloc:
>   return NULL;
>  }
>
> +#ifdef CONFIG_PID_NS
>  static struct pid_namespace *create_pid_namespace(int level)
>  {
>   struct pid_namespace *ns;
> @@ -621,6 +622,7 @@ void free_pid_ns(struct kref *kref)
>   if (parent != NULL)
>    put_pid_ns(parent);
>  }
> +#endif /* CONFIG_PID_NS */
>
>  void zap_pid_ns_processes(struct pid_namespace *pid_ns)
>  {
```

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

---

Subject: Re: [PATCH] pidns: Place under CONFIG_EXPERIMENTAL (take 2)
Posted by Cedric Le Goater on Mon, 29 Oct 2007 10:13:42 GMT
View Forum Message <> Reply to Message

Eric W. Biederman wrote:
> Adrian Bunk <bunk@kernel.org> writes:
>
>> On Sun, Oct 28, 2007 at 09:12:34AM -0700, Jeremy Fitzhardinge wrote:
>>> Eric W. Biederman wrote:
>>>> Roughly that sounds like CONFIG_EXPERIMENTAL to me. But I would
>>>> be happy to hear if someone has a better idea.
>>> Rather than overload an existing config option, why not add one with the
>>> specific semantics you want: CONFIG_UNSTABLE_UABI. The problem seems
>>> like one which which may occur again, though one hopes not too often. I
>>> guess the risk is that people will leave their subsystems depending on
>>> it permanently (sysfs?), so it ends up being set all the time and
>>> becomes as useless as EXPERIMENTAL...
>> Then let SYSFS depend on UNSTABLE_UABI for the next 10 years and we have
>> an excuse for breaking the ABI with each new kernel...
>>
>> Either the ABI is stable or it should not be exposed to users at all.
>
> If we need a new config for it. CONFIG_IMMATURE is the closest I
> can think of.

Pavel also has a CONFIG_NAMESPACES patch that he should be resending to
andrew when 2.6.24-rc1-mm1 is released. pidns will go under this option,
like all the other namespaces, and should protect the distros from shipping
any immature namespace.

C.

____
Containers mailing list

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

Subject: Re: [PATCH] pidns: Place under CONFIG_EXPERIMENTAL (take 2)
Posted by Cedric Le Goater on Mon, 29 Oct 2007 18:05:12 GMT
View Forum Message <> Reply to Message

Eric W. Biederman wrote:
> Andrew Morton <akpm@linux-foundation.org> writes:
>
>>> On Sat, 27 Oct 2007 04:04:08 +0200 Adrian Bunk <bunk@kernel.org> wrote:
>>>> be happy to hear if someone has a better idea.
>>> There is a difference between "complete the feature" and "early adopters
>>> to start playing with the feature" on the one side, and making something
>>> available in a released kernel on the other side.
>>>
>>> For development and playing with it it can depend on BROKEN (perhaps
>>> with the dependency removed through the first -rc kernels), but as soon
>>> as it's available in a -final kernel the ABI is fixed.
>>>
>> Yes, if we're not 100% certain that the interfaces are corrent and unchanging
>> and that the implementation is solid, we should disable the feature at Kconfig
>> time.
>
> Reasonable. So far things look good for a single pid namespace. Multiple
> pid namespaces look iffy.
>
>> The best option would be to fix things asap. But assuming that option isn't
>> reasonable and/or safe, we can slip a `depends on BROKEN' into -rc6 then
>> resume development for 2.6.25.
>
> I think we can make a lot of progress but there is enough development
> yet to do to reach the target of correct and unchanging interfaces,
> with a solid interface. That unless we achieve a breakthrough I
> don't see us achieving that target for 2.6.24.
>
> The outstanding issues I can think of off the top of my head:
> - signal handling for init on secondary pid namespaces.
> - Properly setting si_pid on signals that cross namespaces.

these are being addressed by suka patches, and also you with the latest patch
you sent. right ?

> - The kthread API conversion so we don't get kernel threads
>   trapped in pid namespaces and make them unfreeable.

a lot of work has been done on that part. take a look at it. the clean up is really impressive !

NFS still uses the kernel_thread() API. the first thing to do on the kthread
topic is to improve the kthread API.

I think we can discard the remaining drivers for the moment.

> - At fork time I think we are doing a little bit too much work
>   in setting the session and the pgrp, and removing the controlling
>   tty.

yes probably. this needs to be sorted out. it makes a container init
process different from the system init process.

> - AF_unix domain credential passing.

see commit b488893a390edfe027bae7a46e9af8083e740668 which is covering
UNIX socket credentials and more. Are you thinking we should do more for
credentials and use a struct pid* ? This looks scary.

> - misc pid vs vpid sorting out (autofs autofs4, coda, arch specific
>   syscalls, others?)

autofs* is fixed. netlink ?

> - Removal of task->pid, task->tgid, task->signal->__pgrp,
>   tsk->signal->__session or some other way to ensure that we have
>   touched and converted all of the kernel pid handling.

well, __pgrp and __session are pretty well covered with the __deprecated
attribute. I don't see what else we could to do on these. we can't remove
the task_{session,pgrp}_* routines.

we could apply the same __deprecated technique to task->pid, task->tgid.
This is going to be a challenge :)

> - flock pid handling.

Pavel again.

> It hurts me to even ponder what thinking makes it that
> CONFIG_EXPERIMENTAL isn't enough to keep a stable distro
> from shipping the code in their stable kernel, and locking us into
> trouble.
>
> With that said. I think I should just respin the patchset now and add
> the "depends on BROKEN".
>

> The user namespace appears to need that treatment as well.

The kernel will be protected by a CONFIG_NAMESPACES option as soon as it
gets in. Unfortunately, it didn't make 2.6.24 so this will be 2.6.25
material.

Cheers,

C.

> The network namespace has so little there and it already depends
> on !SYSFS so I don't think we are going to run into any trouble
> with it. Happily I managed to parse that problem differently,
> so I could slice of the parts of the networking stack that
> had not been converted.
>
> Eric
> -
> To unsubscribe from this list: send the line "unsubscribe linux-kernel" in
> the body of a message to majordomo@vger.kernel.org
> More majordomo info at http://vger.kernel.org/majordomo-info.html
> Please read the FAQ at http://www.tux.org/lkml/
>

Subject: Re: [PATCH] pidns: Place under CONFIG_EXPERIMENTAL (take 2)
Posted by ebiederm on Mon, 29 Oct 2007 18:08:45 GMT

View Forum Message <> Reply to Message

Cedric Le Goater <clg@fr.ibm.com> writes:

> Pavel also has a CONFIG_NAMESPACES patch that he should be resending to
> andrew when 2.6.24-rc1-mm1 is released. pidns will go under this option,
> like all the other namespaces, and should protect the distros from shipping
> any immature namespace.

Thanks, As long as we get a config option on there I am happy.

Eric

Subject: Re: [PATCH] pidns: Place under CONFIG_EXPERIMENTAL (take 2)
Posted by ebiederm on Mon, 29 Oct 2007 18:54:46 GMT

View Forum Message <> Reply to Message

Kirill Korotaev <dev@openvz.org> writes:

> Can you please send namespace related patches to containers@ ML first
> before sending them to Linus/Andrew?

If you are so anxious to review my patches can you please review them?
I'd love to see an acked-by or an actual bug found.

I only did what I always do when I am focusing on fixing a bug
in the stable kernel. I fix the bug if I can with a simple obviously
correct patch and send that patch off to the maintainer. Copying
other people that I know of who have a chance of reviewing the
patch.

Eric

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

Subject: Re: [PATCH] pidns: Place under CONFIG_EXPERIMENTAL (take 2)
Posted by ebiederm on Mon, 29 Oct 2007 19:11:23 GMT

View Forum Message <> Reply to Message

Cedric Le Goater <clg@fr.ibm.com> writes:
>> The outstanding issues I can think of off the top of my head:
>> - signal handling for init on secondary pid namespaces.
>> - Properly setting si_pid on signals that cross namespaces.
>
> these are being addressed by suka patches, and also you with the latest patch
> you sent. right ?

I am just starting to review suka's patches it is a subtle area and tricky.
My signal related patches were aimed at just going through the global list
of processes.

>> - The kthread API conversion so we don't get kernel threads
>>   trapped in pid namespaces and make them unfreeable.
>
> a lot of work has been done on that part. take a look at it. the clean up
> is really impressive !

Agreed.

> NFS still uses the kernel_thread() API. the first thing to do on the kthread
> topic is to improve the kthread API.

yes, getting the kthread API up to snuff works.

> I think we can discard the remaining drivers for the moment.
>
>> - At fork time I think we are doing a little bit too much work
>>   in setting the session and the pgrp, and removing the controlling
>>   tty.
>
> yes probably. this needs to be sorted out. it makes a container init
> process different from the system init process.

Yes. I sent a patch for review that fixes just this one aspect and
I will see what happens.

>> - AF_unix domain credential passing.
>
> see commit b488893a390edfe027bae7a46e9af8083e740668 which is covering
> UNIX socket credentials and more. Are you thinking we should do more for
> credentials and use a struct pid* ? This looks scary.

Yes. We need to get the pid in the pid namespace that remove the
credentials. Not the pid in the pid namespace that places the credentials
on the socket.

As for scary and delicate I agree. We really need to pass the struct
pid, not a pid_t in the credentials.

>> - misc pid vs vpid sorting out (autofs autofs4, coda, arch specific
>>   syscalls, others?)
>
> autofs* is fixed. netlink ?

No. autofs compiles builds and works in a single pid namespace.
The issues with multiple pid namespaces have not been fully
addresses, and autofs4 is in worse shape. Unless there
are pending petches I'm not aware of.

Partly it may be Pavel's shift from my intent of having
pid_nr be the pid_t in current->nsproxy->pid_namespace.
To calling that function pid_vnr, and having pid_nr be
the pid_t value in init_pid_ns.

>> - Removal of task->pid, task->tgid, task->signal->__pgrp,
>>   tsk->signal->__session or some other way to ensure that we have

>>   touched and converted all of the kernel pid handling.
>
> well, __pgrp and __session are pretty well covered with the __deprecated
> attribute. I don't see what else we could to do on these. we can't remove
> the task_{session,pgrp}_* routines.
>
> we could apply the same __deprecated technique to task->pid, task->tgid.
> This is going to be a challenge :)

Well the point is not to use the pid_t for anything except passing to user
space. It isn't that I object to people using __session directly it
is that I object to people using task_session_nr because it hides
possible bugs. When we work with pids using struct pid pointers it is
clear we can't get it wrong. When we use pid_t values it is quite
easy to get things wrong. As the current autofs code demonstrates.

```
> diff --git a/fs/autofs/inode.c b/fs/autofs/inode.c
> index 45f5992..af82143 100644
> --- a/fs/autofs/inode.c
> +++ b/fs/autofs/inode.c
> @@ -80,7 +80,7 @@ static int parse_options(char *options, int *pipefd, uid_t *uid, gid_t *gid,
>
>        *uid = current->uid;
>        *gid = current->gid;
> -      *pgrp = task_pgrp_nr(current);
> +       *pgrp = task_pgrp_vnr(current);
>
>        *minproto = *maxproto = AUTOFS_PROTO_VERSION;
>
```

>> - flock pid handling.
>
> Pavel again.

Yes. I know he was looking at it.

> The kernel will be protected by a CONFIG_NAMESPACES option as soon as it
> gets in. Unfortunately, it didn't make 2.6.24 so this will be 2.6.25
> material.

Which sounds completely bass akwards. We want the option so we can
disable things now, when everything is immature, and not really
doing the right thing. I guess I really should look in Andrews queue
and see what didn't make it, and see if there are pieces that fixes
bugs that really should have.

I think the CONFIG_NAMESPACES work was focused primarily on size

reduction under CONFIG_EMBEDDED. Which means from a correctness
perspective I don't care.

Eric

Subject: Re: [PATCH] pidns: Place under CONFIG_EXPERIMENTAL
Posted by Pavel Emelianov on Wed, 31 Oct 2007 08:54:21 GMT

View Forum Message <> Reply to Message

Eric W. Biederman wrote:
> I finally found a chance to review the pid namespace implementation in
> detail and currently it is much to easy to find issues where the
> kernel does the wrong thing outside of the initial pid namespace.
> At the same time the pid namespace code we have does appear
> to be a good base to build on.
>
> Therefore until the dust settles and we are certain we have the
> ABI and the implementation as correct as humanly possible let's
> hide the availability of process ID namespaces behind
> CONFIG_EXPERIMENTAL.

Sorry for the late answer - I was out for a conference and the
organization committee didn't provide an internet access.

I currently have a set of patches that move all the namespaces
cloning code under the config option. This is done to help
embedded people have a small kernel.

I was planning to wait with this set untill 2.6.24-rc-mm1 kernel,
but since (as I see) this is required rather badly I will send this
set in a couple of days.

Thanks,
Pavel

> Allowing users to avoid bugs, and removing a guarantee of bug
> compatibility. Allowing any issues that may be found to
> be fixed properly.
>
> If CONFIG_PID_NS=N this patch will cause copy_pid_ns to
> unconditionally return -EINVAL removing the availability
> of multiple pid namespaces.
>

```
> Signed-off-by: Eric W. Biederman <ebiederm@xmission.com>
> ---
>  init/Kconfig |   12 ++++++++++++
>  kernel/pid.c |    4 ++++
>  2 files changed, 16 insertions(+), 0 deletions(-)
>
> diff --git a/init/Kconfig b/init/Kconfig
> index 8b88d0b..72e37c0 100644
> --- a/init/Kconfig
> +++ b/init/Kconfig
> @@ -215,6 +215,18 @@ config USER_NS
>     vservers, to use user namespaces to provide different
>     user info for different servers.  If unsure, say N.
>
> +config PID_NS
> + bool "PID Namespaces (EXPERIMENTAL)"
> + default n
> + depends on EXPERIMENTAL
> + help
> +   Suport process id namespaces.  This allows having multiple
> +   process with the same pid as long as they are in different
> +   pid namespaces.  This is a building block of containers.
> +
> +   Unless you want to work with an experimental feature
> +   say N here.
> +
>  config AUDIT
>   bool "Auditing support"
>   depends on NET
> diff --git a/kernel/pid.c b/kernel/pid.c
> index d1db36b..8a5637b 100644
> --- a/kernel/pid.c
> +++ b/kernel/pid.c
> @@ -599,6 +599,10 @@ struct pid_namespace *copy_pid_ns(unsigned long flags, struct
pid_namespace *old
>   if (flags & CLONE_THREAD)
>    goto out_put;
>
> +#ifndef CONFIG_PID_NS
> + goto out_put;
> +#endif
> +
>   new_ns = create_pid_namespace(old_ns->level + 1);
>   if (!IS_ERR(new_ns))
>    new_ns->parent = get_pid_ns(old_ns);
```

____
Containers mailing list

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