Subject: Re: [RFC][PATCH] Pid namespaces vs locks interaction
Posted by serue on Thu, 06 Dec 2007 14:53:40 GMT
View Forum Message <> Reply to Message

Quoting Vitaliy Gusev (vgusev@openvz.org):
> Hello!
>
> I am working on pid namespaces vs locks interaction and want to evaluate the
> idea.
> fcntl(F_GETLK,..) can return pid of process for not current pid namespace (if
> process is belonged to the several namespaces). It is true also for pids
> in /proc/locks. So correct behavior is saving pointer to the struct pid of
> the process lock owner.
> --
> Thank,
> Vitaliy Gusev

> diff --git a/fs/locks.c b/fs/locks.c
> index 8b8388e..d2d3d75 100644
> --- a/fs/locks.c
> +++ b/fs/locks.c
> @@ -125,6 +125,7 @@
> #include <linux/syscalls.h>
> #include <linux/time.h>
> #include <linux/rcupdate.h>
> +#include <linux/pid_namespace.h>
>
> #include <asm/semaphore.h>
> #include <asm/uaccess.h>
> @@ -185,6 +186,7 @@ void locks_init_lock(struct file_lock *fl)
>   fl->fl_fasync = NULL;
>   fl->fl_owner = NULL;
>   fl->fl_pid = 0;
> + fl->fl_nspid = NULL;

The idea seems right, but why are you keeping fl->fl_pid around?

Seems like the safer thing to do would be to have a separate
struct user_flock, with an integer pid, for communicating to userspace,
and a struct flock, with struct pid, for kernel use? Then fcntl_getlk()
and fcntl_setlk() do the appropriate conversions.

thanks,
-serge

>   fl->fl_file = NULL;
>   fl->fl_flags = 0;
>   fl->fl_type = 0;

```
> @@ -553,6 +555,8 @@ static void locks_insert_lock(struct file_lock **pos, struct file_lock *fl)
>  {
>   list_add(&fl->fl_link, &file_lock_list);
>
> + fl->fl_nspid = get_pid(task_tgid(current));
> +
>   /* insert into file's list */
>   fl->fl_next = *pos;
>   *pos = fl;
> @@ -584,6 +588,11 @@ static void locks_delete_lock(struct file_lock **thisfl_p)
>   if (fl->fl_ops && fl->fl_ops->fl_remove)
>    fl->fl_ops->fl_remove(fl);
>
> + if (fl->fl_nspid) {
> +  put_pid(fl->fl_nspid);
> +  fl->fl_nspid = NULL;
> + }
> +
>   locks_wake_up_blocks(fl);
>   locks_free_lock(fl);
>  }
> @@ -673,14 +682,16 @@ posix_test_lock(struct file *filp, struct file_lock *fl)
>    if (posix_locks_conflict(fl, cfl))
>     break;
>   }
> - if (cfl)
> + if (cfl) {
>    __locks_copy_lock(fl, cfl);
> - else
> +  if (cfl->fl_nspid)
> +   fl->fl_pid = pid_nr_ns(cfl->fl_nspid,
> +       task_active_pid_ns(current));
> + } else
>    fl->fl_type = F_UNLCK;
>   unlock_kernel();
>   return;
>  }
> -
>  EXPORT_SYMBOL(posix_test_lock);
>
>  /* This function tests for deadlock condition before putting a process to
> @@ -2084,6 +2095,12 @@ static void lock_get_status(struct seq_file *f, struct file_lock *fl,
>         int id, char *pfx)
>  {
>   struct inode *inode = NULL;
> + unsigned int fl_pid;
> +
> + if (fl->fl_nspid)
```

```
> +  fl_pid = pid_nr_ns(fl->fl_nspid, task_active_pid_ns(current));
> + else
> +  fl_pid = fl->fl_pid;
>
>   if (fl->fl_file != NULL)
>    inode = fl->fl_file->f_path.dentry->d_inode;
> @@ -2124,16 +2141,16 @@ static void lock_get_status(struct seq_file *f, struct file_lock *fl,
>   }
>   if (inode) {
>  #ifdef WE_CAN_BREAK_LSLK_NOW
> -  seq_printf(f, "%d %s:%ld ", fl->fl_pid,
> +  seq_printf(f, "%d %s:%ld ", fl_pid,
>      inode->i_sb->s_id, inode->i_ino);
>  #else
>    /* userspace relies on this representation of dev_t ;-( */
> -  seq_printf(f, "%d %02x:%02x:%ld ", fl->fl_pid,
> +  seq_printf(f, "%d %02x:%02x:%ld ", fl_pid,
>      MAJOR(inode->i_sb->s_dev),
>      MINOR(inode->i_sb->s_dev), inode->i_ino);
>  #endif
>   } else {
> -  seq_printf(f, "%d <none>:0 ", fl->fl_pid);
> +  seq_printf(f, "%d <none>:0 ", fl_pid);
>   }
>   if (IS_POSIX(fl)) {
>    if (fl->fl_end == OFFSET_MAX)
> diff --git a/include/linux/fs.h b/include/linux/fs.h
> index b3ec4a4..5876f68 100644
> --- a/include/linux/fs.h
> +++ b/include/linux/fs.h
> @@ -870,6 +870,7 @@ struct file_lock {
>   struct list_head fl_block; /* circular list of blocked processes */
>   fl_owner_t fl_owner;
>   unsigned int fl_pid;
> + struct pid *fl_nspid;
>   wait_queue_head_t fl_wait;
>   struct file *fl_file;
>   unsigned char fl_flags;
```

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

Subject: Re: [RFC][PATCH] Pid namespaces vs locks interaction
Posted by gblond on Thu, 06 Dec 2007 15:19:59 GMT

On 6 December 2007 17:53:40 Serge E. Hallyn wrote:
> Quoting Vitaliy Gusev (vgusev@openvz.org):
> > Hello!
> >
> > I am working on pid namespaces vs locks interaction and want to evaluate
> > the idea.
> > fcntl(F_GETLK,..) can return pid of process for not current pid namespace
> > (if process is belonged to the several namespaces). It is true also for
> > pids in /proc/locks. So correct behavior is saving pointer to the struct
> > pid of the process lock owner.
> > --
> > Thank,
> > Vitaliy Gusev
> >
> > diff --git a/fs/locks.c b/fs/locks.c
> > index 8b8388e..d2d3d75 100644
> > --- a/fs/locks.c
> > +++ b/fs/locks.c
> > @@ -125,6 +125,7 @@
> > #include <linux/syscalls.h>
> > #include <linux/time.h>
> > #include <linux/rcupdate.h>
> > +#include <linux/pid_namespace.h>
> >
> > #include <asm/semaphore.h>
> > #include <asm/uaccess.h>
> > @@ -185,6 +186,7 @@ void locks_init_lock(struct file_lock *fl)
> >   fl->fl_fasync = NULL;
> >   fl->fl_owner = NULL;
> >   fl->fl_pid = 0;
> > + fl->fl_nspid = NULL;
>
> The idea seems right, but why are you keeping fl->fl_pid around?
>
> Seems like the safer thing to do would be to have a separate
> struct user_flock, with an integer pid, for communicating to userspace,
> and a struct flock, with struct pid, for kernel use? Then fcntl_getlk()
> and fcntl_setlk() do the appropriate conversions.

fl_pid is used by nfs, fuse and gfs2. For instance nfs keeps in fl_pid some
unique id to identify locking process between hosts - it is not a process
pid.

>
> thanks,
> -serge
> -

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

--
Thank,
Vitaliy Gusev

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

---

Subject: Re: [RFC][PATCH] Pid namespaces vs locks interaction
Posted by serue on Thu, 06 Dec 2007 15:51:30 GMT
View Forum Message <> Reply to Message

Quoting Vitaliy Gusev (vgusev@openvz.org):
> On 6 December 2007 17:53:40 Serge E. Hallyn wrote:
> > Quoting Vitaliy Gusev (vgusev@openvz.org):
> > > Hello!
> > >
> > > I am working on pid namespaces vs locks interaction and want to evaluate
> > > the idea.
> > > fcntl(F_GETLK,..) can return pid of process for not current pid namespace
> > > (if process is belonged to the several namespaces). It is true also for
> > > pids in /proc/locks. So correct behavior is saving pointer to the struct
> > > pid of the process lock owner.
> > > --
> > > Thank,
> > > Vitaliy Gusev
> > >
> > > diff --git a/fs/locks.c b/fs/locks.c
> > > index 8b8388e..d2d3d75 100644
> > > --- a/fs/locks.c
> > > +++ b/fs/locks.c
> > > @@ -125,6 +125,7 @@
> > > #include <linux/syscalls.h>
> > > #include <linux/time.h>
> > > #include <linux/rcupdate.h>
> > > +#include <linux/pid_namespace.h>
> > >
> > > #include <asm/semaphore.h>
> > > #include <asm/uaccess.h>
> > > @@ -185,6 +186,7 @@ void locks_init_lock(struct file_lock *fl)
> > >   fl->fl_fasync = NULL;

> > > fl->fl_owner = NULL;
> > > fl->fl_pid = 0;
> > > + fl->fl_nspid = NULL;
> >
> > The idea seems right, but why are you keeping fl->fl_pid around?
> >
> > Seems like the safer thing to do would be to have a separate
> > struct user_flock, with an integer pid, for communicating to userspace,
> > and a struct flock, with struct pid, for kernel use? Then fcntl_getlk()
> > and fcntl_setlk() do the appropriate conversions.
>
> fl_pid is used by nfs, fuse and gfs2. For instance nfs keeps in fl_pid some
> unique id to identify locking process between hosts - it is not a process
> pid.

Ok, but so the struct user_flock->fl_pid is being set to the task's
virtual pid, while the struct kernel_flock->fl_pid is being set to
task->tgid for nfsd use.

Why can't nfs just generate a uniqueid from the struct pid when it
needs it?

Fuse just seems to copy the pid to report it to userspace, so it would
just copy pid_vnr(kernel_flock->pid) into user_flock->fl_pid.

Anyway I haven't looked at all the uses of struct fl_pid, but you
can always get the pidnr back from the struct pid if needed so there
should be no problem.

The split definately seems worthwhile to me, so that
user_flock->fl_pidnr can always be said to be the pid in the acting
process' namespace, and flock->fl_pid can always be a struct pid,
rather than having fl_pid sometimes be current->tgid, or sometimes
pid_vnr(flock->fl_nspid)...

-serge

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

---

Subject: Re: [RFC][PATCH] Pid namespaces vs locks interaction
Posted by Brad Boyer on Sat, 08 Dec 2007 22:21:19 GMT
View Forum Message <> Reply to Message

On Thu, Dec 06, 2007 at 09:51:30AM -0600, Serge E. Hallyn wrote:
> Quoting Vitaliy Gusev (vgusev@openvz.org):

> > fl_pid is used by nfs, fuse and gfs2. For instance nfs keeps in fl_pid some
> > unique id to identify locking process between hosts - it is not a process
> > pid.
>
> Ok, but so the struct user_flock->fl_pid is being set to the task's
> virtual pid, while the struct kernel_flock->fl_pid is being set to
> task->tgid for nfsd use.
>
> Why can't nfs just generate a uniqueid from the struct pid when it
> needs it?
>
> Fuse just seems to copy the pid to report it to userspace, so it would
> just copy pid_vnr(kernel_flock->pid) into user_flock->fl_pid.
>
> Anyway I haven't looked at all the uses of struct fl_pid, but you
> can always get the pidnr back from the struct pid if needed so there
> should be no problem.

Perhaps we could add a sysid field like some unix systems have. Here is the flock structure documentation from Sun:

The flock structure contains at least the following elements:

```
short   l_type;       /* lock operation type */
short   l_whence;     /* lock base indicator */
off_t   l_start;      /* starting offset from base */
off_t   l_len;        /* lock length; l_len == 0 means
                      until end of file */
int     l_sysid;      /* system ID running process holding lock */
pid_t   l_pid;        /* process ID of process holding lock */
```

Using the sysid could show that the pid field refers to a separate namespace, and might also be useful for NFS to show that the lock is really held by a process on a different system. This would also be something we could export to user space in a way that some programs are already written to expect and handle properly.

Brad Boyer
flar@allandria.com

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

Subject: Re: [RFC][PATCH] Pid namespaces vs locks interaction
Posted by gblond on Wed, 12 Dec 2007 16:07:25 GMT
View Forum Message <> Reply to Message

Hello

On 6 December 2007 18:51:30 Serge E. Hallyn wrote:
> > fl_pid is used by nfs, fuse and gfs2. For instance nfs keeps in fl_pid
> > some unique id to identify locking process between hosts - it is not a
> > process pid.
>
> Ok, but so the struct user_flock->fl_pid is being set to the task's
> virtual pid, while the struct kernel_flock->fl_pid is being set to
> task->tgid for nfsd use.
>
> Why can't nfs just generate a uniqueid from the struct pid when it
> needs it?

I think it is hard. lockd uses struct nlm_host to get process unique id (see __nlm_alloc_pid() function).

>
> Fuse just seems to copy the pid to report it to userspace, so it would
> just copy pid_vnr(kernel_flock->pid) into user_flock->fl_pid.
>
> Anyway I haven't looked at all the uses of struct fl_pid, but you
> can always get the pidnr back from the struct pid if needed so there
> should be no problem.
>
> The split definately seems worthwhile to me, so that
> user_flock->fl_pidnr can always be said to be the pid in the acting
> process' namespace, and flock->fl_pid can always be a struct pid,
> rather than having fl_pid sometimes be current->tgid, or sometimes
> pid_vnr(flock->fl_nspid)...
>
> -serge
> -
> To unsubscribe from this list: send the line "unsubscribe linux-fsdevel" in
> the body of a message to majordomo@vger.kernel.org
> More majordomo info at http://vger.kernel.org/majordomo-info.html

--
Thank,
Vitaliy Gusev

_____
Containers mailing list
Containers@lists.linux-foundation.org

Subject: Re: [RFC][PATCH] Pid namespaces vs locks interaction
Posted by serue on Wed, 12 Dec 2007 17:31:15 GMT
View Forum Message <> Reply to Message

Quoting Vitaliy Gusev (vgusev@openvz.org):
> Hello
>
> On 6 December 2007 18:51:30 Serge E. Hallyn wrote:
> > > fl_pid is used by nfs, fuse and gfs2. For instance nfs keeps in fl_pid
> > > some unique id to identify locking process between hosts - it is not a
> > > process pid.
> >
> > Ok, but so the struct user_flock->fl_pid is being set to the task's
> > virtual pid, while the struct kernel_flock->fl_pid is being set to
> > task->tgid for nfsd use.
> >
> > Why can't nfs just generate a uniqueid from the struct pid when it
> > needs it?
>
> I think it is hard. lockd uses struct nlm_host to get process unique id (see
> __nlm_alloc_pid() function).

Looks pretty simple though... That whole set of code could even stay
the same except for in __nlm_alloc_pid():

option 1: compare struct pid* instead of uint32_t pid
option 2: use the "global pid" out of the stored struct pid,
something like pid->numbers[0].nr.

> > Fuse just seems to copy the pid to report it to userspace, so it would
> > just copy pid_vnr(kernel_flock->pid) into user_flock->fl_pid.
> >
> > Anyway I haven't looked at all the uses of struct fl_pid, but you
> > can always get the pidnr back from the struct pid if needed so there
> > should be no problem.
> >
> > The split definately seems worthwhile to me, so that
> > user_flock->fl_pidnr can always be said to be the pid in the acting
> > process' namespace, and flock->fl_pid can always be a struct pid,
> > rather than having fl_pid sometimes be current->tgid, or sometimes
> > pid_vnr(flock->fl_nspid)...
> >
> > -serge
> > -
> > To unsubscribe from this list: send the line "unsubscribe linux-fsdevel" in

> > the body of a message to majordomo@vger.kernel.org
> > More majordomo info at http://vger.kernel.org/majordomo-info.html
>
>
>
> --
> Thank,
> Vitaliy Gusev

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

---

Subject: Re: [RFC][PATCH] Pid namespaces vs locks interaction
Posted by gblond on Wed, 12 Dec 2007 17:42:41 GMT
View Forum Message <> Reply to Message

On 12 December 2007 20:31:15 Serge E. Hallyn wrote:
> Quoting Vitaliy Gusev (vgusev@openvz.org):
> > Hello
> >
> > On 6 December 2007 18:51:30 Serge E. Hallyn wrote:
> > > > fl_pid is used by nfs, fuse and gfs2. For instance nfs keeps in
> > > > fl_pid some unique id to identify locking process between hosts - it
> > > > is not a process pid.
> > >
> > > Ok, but so the struct user_flock->fl_pid is being set to the task's
> > > virtual pid, while the struct kernel_flock->fl_pid is being set to
> > > task->tgid for nfsd use.
> > >
> > > Why can't nfs just generate a uniqueid from the struct pid when it
> > > needs it?
> >
> > I think it is hard. lockd uses struct nlm_host to get process unique id
> > (see __nlm_alloc_pid() function).
>
> Looks pretty simple though... That whole set of code could even stay
> the same except for in __nlm_alloc_pid():
>
> option 1: compare struct pid* instead of uint32_t pid
> option 2: use the "global pid" out of the stored struct pid,
>   something like pid->numbers[0].nr.

We can't use process pid. Process pid is circulated! NFS (lockd) needs
unique process id between hosts which can't repeat oneself.

>

> > > Fuse just seems to copy the pid to report it to userspace, so it would
> > > just copy pid_vnr(kernel_flock->pid) into user_flock->fl_pid.
> > >
> > > Anyway I haven't looked at all the uses of struct fl_pid, but you
> > > can always get the pidnr back from the struct pid if needed so there
> > > should be no problem.
> > >
> > > The split definately seems worthwhile to me, so that
> > > user_flock->fl_pidnr can always be said to be the pid in the acting
> > > process' namespace, and flock->fl_pid can always be a struct pid,
> > > rather than having fl_pid sometimes be current->tgid, or sometimes
> > > pid_vnr(flock->fl_nspid)...
> > >
> > > -serge
> > > -
> > > To unsubscribe from this list: send the line "unsubscribe
> > > linux-fsdevel" in the body of a message to majordomo@vger.kernel.org
> > > More majordomo info at http://vger.kernel.org/majordomo-info.html
> >
> > --
> > Thank,
> > Vitaliy Gusev
>
> -
> To unsubscribe from this list: send the line "unsubscribe linux-fsdevel" in
> the body of a message to majordomo@vger.kernel.org
> More majordomo info at http://vger.kernel.org/majordomo-info.html

--
Thank,
Vitaliy Gusev

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

---

Subject: Re: [RFC][PATCH] Pid namespaces vs locks interaction
Posted by serue on Wed, 12 Dec 2007 18:42:25 GMT
View Forum Message <> Reply to Message

Quoting Vitaliy Gusev (vgusev@openvz.org):
> On 12 December 2007 20:31:15 Serge E. Hallyn wrote:
> > Quoting Vitaliy Gusev (vgusev@openvz.org):
> > > Hello
> > >

> > > On 6 December 2007 18:51:30 Serge E. Hallyn wrote:
> > > > > fl_pid is used by nfs, fuse and gfs2. For instance nfs keeps in
> > > > > fl_pid some unique id to identify locking process between hosts - it
> > > > > is not a process pid.
> > > >
> > > > Ok, but so the struct user_flock->fl_pid is being set to the task's
> > > > virtual pid, while the struct kernel_flock->fl_pid is being set to
> > > > task->tgid for nfsd use.
> > > >
> > > > Why can't nfs just generate a uniqueid from the struct pid when it
> > > > needs it?
> > >
> > > I think it is hard. lockd uses struct nlm_host to get process unique id
> > > (see __nlm_alloc_pid() function).
> >
> > Looks pretty simple though... That whole set of code could even stay
> > the same except for in __nlm_alloc_pid():
> >
> > option 1: compare struct pid* instead of uint32_t pid
> > option 2: use the "global pid" out of the stored struct pid,
> > something like pid->numbers[0].nr.
>
> We can't use process pid. Process pid is circulated! NFS (lockd) needs
> unique process id between hosts which can't repeat oneself.

Ok sorry - by letting this thread sit a few days I lost track of where
we were.

I see now, so you're saying fl_pid for nfs is not in fact a task pid.
It's a magically derived unique id. (And you say it is unique across
all the nfs clients?)

So does the p in fl_pid stand for something, or could we rename it to
fl_id or fl_uniqueid?

Maybe that's too much bother, but so long as we're bothering with a pid
cleanup at all it seems worth it to me. On the other hand maybe
J. Bruce Fields was right and we should accept the fact that the
flock->fl_pid shouldn't be taken too seriously, and leave it be.

-serge

> > > > Fuse just seems to copy the pid to report it to userspace, so it would
> > > > just copy pid_vnr(kernel_flock->pid) into user_flock->fl_pid.
> > > >
> > > > Anyway I haven't looked at all the uses of struct fl_pid, but you
> > > > can always get the pidnr back from the struct pid if needed so there
> > > > should be no problem.

> > > >
> > > > The split definately seems worthwhile to me, so that
> > > > user_flock->fl_pidnr can always be said to be the pid in the acting
> > > > process' namespace, and flock->fl_pid can always be a struct pid,
> > > > rather than having fl_pid sometimes be current->tgid, or sometimes
> > > > pid_vnr(flock->fl_nspid)...
> > > >
> > > > -serge
> > > > -
> > > > To unsubscribe from this list: send the line "unsubscribe
> > > > linux-fsdevel" in the body of a message to majordomo@vger.kernel.org
> > > > More majordomo info at http://vger.kernel.org/majordomo-info.html
> > >
> > > --
> > > Thank,
> > > Vitaliy Gusev
> >
> > -
> > To unsubscribe from this list: send the line "unsubscribe linux-fsdevel" in
> > the body of a message to majordomo@vger.kernel.org
> > More majordomo info at http://vger.kernel.org/majordomo-info.html
>
>
>
> --
> Thank,
> Vitaliy Gusev

_____

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

---

Subject: Re: [RFC][PATCH] Pid namespaces vs locks interaction
Posted by gblond on Thu, 13 Dec 2007 14:13:56 GMT
View Forum Message <> Reply to Message

On 12 December 2007 21:42:25 Serge E. Hallyn wrote:
> Ok sorry - by letting this thread sit a few days I lost track of where
> we were.
>
> I see now, so you're saying fl_pid for nfs is not in fact a task pid.
> It's a magically derived unique id. (And you say it is unique across
> all the nfs clients?)

It is unique for pair client,server.

>

> So does the p in fl_pid stand for something, or could we rename it to
> fl_id or fl_uniqueid?

If fl_pid will be renamed with fl_uniqueid or something, it still need accessing from fs/locks.c: cat /proc/locks shows pids which also are NFS pids (unique id).

For example, let's look the /proc/locks in my system (NFS-server) when do flock on a NFS client:

1: POSIX ADVISORY WRITE 2 08:06:63116 0 EOF
2: POSIX ADVISORY WRITE 7047 08:09:1899694 0 EOF
3: FLOCK ADVISORY WRITE 3334 08:06:110497 0 EOF
4: FLOCK ADVISORY WRITE 3265 08:06:94786 0 EOF
5: POSIX ADVISORY WRITE 2582 08:06:110462 0 EOF

It indicates that process with pid 2 has a posix lock. Really it is a NFS unique id. Problem can be solved by using pid of lockd.

> Maybe that's too much bother, but so long as we're bothering with a pid
> cleanup at all it seems worth it to me. On the other hand maybe
> J. Bruce Fields was right and we should accept the fact that the
> flock->fl_pid shouldn't be taken too seriously, and leave it be.

Mix pids from some namespaces is not good. We can store process pid seen from init namespace to the flock->fl_pid (instead pid from the current namespace). Thus fcntl(F_GETLK,...) and "cat /proc/locks" will show global pids. But some LTP tests can fail.

>
> -serge
>

--
Thank,
Vitaliy Gusev

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

---

Subject: Re: [RFC][PATCH] Pid namespaces vs locks interaction
Posted by serue on Thu, 13 Dec 2007 16:40:55 GMT
View Forum Message <> Reply to Message

Quoting Vitaliy Gusev (vgusev@openvz.org):
> On 12 December 2007 21:42:25 Serge E. Hallyn wrote:

> > Ok sorry - by letting this thread sit a few days I lost track of where
> > we were.
> >
> > I see now, so you're saying fl_pid for nfs is not in fact a task pid.
> > It's a magically derived unique id. (And you say it is unique across
> > all the nfs clients?)
>
> It is unique for pair client,server.
>
> >
> > So does the p in fl_pid stand for something, or could we rename it to
> > fl_id or fl_uniqueid?
>
> If fl_pid will be renamed with fl_uniqueid or something, it still need
> accessing from fs/locks.c: cat /proc/locks shows pids which also are NFS
> pids (unique id).
>
> For example, let's look the /proc/locks in my system (NFS-server) when do
> flock on a NFS client:
>
> 1: POSIX ADVISORY WRITE 2 08:06:63116 0 EOF
> 2: POSIX ADVISORY WRITE 7047 08:09:1899694 0 EOF
> 3: FLOCK ADVISORY WRITE 3334 08:06:110497 0 EOF
> 4: FLOCK ADVISORY WRITE 3265 08:06:94786 0 EOF
> 5: POSIX ADVISORY WRITE 2582 08:06:110462 0 EOF
>
> It indicates that process with pid 2 has a posix lock. Really it is a NFS
> unique id. Problem can be solved by using pid of lockd.
>
> > Maybe that's too much bother, but so long as we're bothering with a pid
> > cleanup at all it seems worth it to me. On the other hand maybe
> > J. Bruce Fields was right and we should accept the fact that the
> > flock->fl_pid shouldn't be taken too seriously, and leave it be.
>
> Mix pids from some namespaces is not good. We can store process pid seen from

Agreed, and that was the basis for my earlier objection.

It sounds like it's clear to all people smarter than I that fl_pid is
not really a pid, so there is no reason for changing the name. And your
patch (contrary to my earlier read of it) only translates fl_nspid into
fl_pids in temporary flocks being passed to userspace, through fcntl and
/proc/locks.

So I completely withdraw my objection.

Except, for the sake of other cognitively challenged types like myself,
could you add a comment by fl_pid and fl_nspid in fs.h, to the effect of

unsigned int fl_pid; /* unique id and sometimes global pid */
struct pid *fl_nspid; /* to calculate owner pid_nr for userspace */

(or something more accurate if I'm off)?

So after all that,

Acked-by: Serge Hallyn <serue@us.ibm.com>

(sorry)

thanks,
-serge

> init namespace to the flock->fl_pid (instead pid from the current namespace).
> Thus fcntl(F_GETLK,...) and "cat /proc/locks" will show global pids.   But
> some LTP tests can fail.

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