

---

Subject: [PATCH] pid: Extend/Fix pid\_vnr  
Posted by [ebiederm](#) on Thu, 06 Dec 2007 04:51:43 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

pid\_vnr returns the user space pid with respect to the pid namespace the struct pid was allocated in. What we want before we return a pid to user space is the user space pid with respect to the pid namespace of current.

pid\_vnr is a very nice optimization but because it isn't quite what we want it is easy to use pid\_vnr at times when we aren't certain the struct pid was allocated in our pid namespace.

Currently this describes at least tiocgpggrp and tiocgsid in ttyio.c the parent process reported in the core dumps and the parent process in get\_signal\_to\_deliver.

So unless the performance impact is huge having an interface that does what we want instead of always what we want should be much more reliable and much less error prone.

Signed-off-by: Eric W. Biederman <[ebiederm@xmission.com](mailto:ebiederm@xmission.com)>

---

```
include/linux/pid.h | 14 +++-----
include/linux/sched.h | 5 ++---
kernel/pid.c | 6 ++++++
3 files changed, 11 insertions(+), 14 deletions(-)
```

```
diff --git a/include/linux/pid.h b/include/linux/pid.h
index 061abb6..b91f473 100644
--- a/include/linux/pid.h
+++ b/include/linux/pid.h
@@ -127,9 +127,8 @@ extern void FASTCALL(free_pid(struct pid *pid));
 * the helpers to get the pid's id seen from different namespaces
 *
 * pid_nr() : global id, i.e. the id seen from the init namespace;
- * pid_vnr() : virtual id, i.e. the id seen from the namespace this pid
- * belongs to. this only makes sense when called in the
- * context of the task that belongs to the same namespace;
+ * pid_vnr() : virtual id, i.e. the id seen from the pid namespace of
+ * current.
 * pid_nr_ns() : id seen from the ns specified.
 *
 * see also task_xid_nr() etc in include/linux/sched.h
@@ -144,14 +143,7 @@ static inline pid_t pid_nr(struct pid *pid)
}
```

```
pid_t pid_nr_ns(struct pid *pid, struct pid_namespace *ns);
```

```

-
-static inline pid_t pid_vnr(struct pid *pid)
-{
- pid_t nr = 0;
- if (pid)
- nr = pid->numbers[pid->level].nr;
- return nr;
-}
+pid_t pid_vnr(struct pid *pid);

#define do_each_pid_task(pid, type, task) \
do { \
diff --git a/include/linux/sched.h b/include/linux/sched.h
index 1b1e25b..9293114 100644
--- a/include/linux/sched.h
+++ b/include/linux/sched.h
@@ -1286,9 +1286,8 @@ struct pid_namespace;
 * from various namespaces
 *
 * task_xid_nr() : global id, i.e. the id seen from the init namespace;
- * task_xid_vnr() : virtual id, i.e. the id seen from the namespace the task
- * belongs to. this only makes sence when called in the
- * context of the task that belongs to the same namespace;
+ * task_xid_vnr() : virtual id, i.e. the id seen from the pid namespace of
+ * current.
 * task_xid_nr_ns() : id seen from the ns specified;
 *
 * set_task_vxid() : assigns a virtual id to a task;
diff --git a/kernel/pid.c b/kernel/pid.c
index 21f027c..c507ca7 100644
--- a/kernel/pid.c
+++ b/kernel/pid.c
@@ -442,6 +442,12 @@ pid_t pid_nr_ns(struct pid *pid, struct pid_namespace *ns)
return nr;
}

+pid_t pid_vnr(struct pid *pid)
+{
+ return pid_nr_ns(pid, current->nsproxy->pid_ns);
+}
+EXPORT_SYMBOL_GPL(pid_vnr);
+
+pid_t task_pid_nr_ns(struct task_struct *tsk, struct pid_namespace *ns)
+{
return pid_nr_ns(task_pid(tsk), ns);
--
1.5.3.rc6.17.g1911

```

---

Subject: Re: [PATCH] pid: Extend/Fix pid\_vnr  
Posted by [Pavel Emelianov](#) on Thu, 06 Dec 2007 10:02:03 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

Eric W. Biederman wrote:

> pid\_vnr returns the user space pid with respect to the pid namespace  
> the struct pid was allocated in. What we want before we return a pid  
> to user space is the user space pid with respect to the pid namespace  
> of current.  
>  
> pid\_vnr is a very nice optimization but because it isn't quite what we  
> want it is easy to use pid\_vnr at times when we aren't certain the  
> struct pid was allocated in our pid namespace.  
>  
> Currently this describes at least tiocgpggrp and tiocgsid in ttyio.c  
> the parent process reported in the core dumps and the parent  
> process in get\_signal\_to\_deliver.  
>  
> So unless the performance impact is huge having an interface that does  
> what we want instead of always what we want should be much more  
> reliable and much less error prone.  
>  
> Signed-off-by: Eric W. Biederman <[ebiederm@xmission.com](mailto:ebiederm@xmission.com)>

Yup. xxx\_vnr with the existing meaning turns out to be useless.

Thanks, Eric.

Acked-by: Pavel Emelyanov <[xemul@openvz.org](mailto:xemul@openvz.org)>

> ---  
> include/linux/pid.h | 14 +++-----  
> include/linux/sched.h | 5 ++---  
> kernel/pid.c | 6 ++++++  
> 3 files changed, 11 insertions(+), 14 deletions(-)  
>  
> diff --git a/include/linux/pid.h b/include/linux/pid.h  
> index 061abb6..b91f473 100644  
> --- a/include/linux/pid.h  
> +++ b/include/linux/pid.h  
> @@ -127,9 +127,8 @@ extern void FASTCALL(free\_pid(struct pid \*pid));  
> \* the helpers to get the pid's id seen from different namespaces

```

> *
> * pid_nr() : global id, i.e. the id seen from the init namespace;
> - * pid_vnr() : virtual id, i.e. the id seen from the namespace this pid
> - * belongs to. this only makes sense when called in the
> - * context of the task that belongs to the same namespace;
> + * pid_vnr() : virtual id, i.e. the id seen from the pid namespace of
> + * current.
> * pid_nr_ns() : id seen from the ns specified.
> *
> * see also task_xid_nr() etc in include/linux/sched.h
> @@ -144,14 +143,7 @@ static inline pid_t pid_nr(struct pid *pid)
> }
>
> pid_t pid_nr_ns(struct pid *pid, struct pid_namespace *ns);
> -
> -static inline pid_t pid_vnr(struct pid *pid)
> -{
> - pid_t nr = 0;
> - if (pid)
> - nr = pid->numbers[pid->level].nr;
> - return nr;
> -}
> +pid_t pid_vnr(struct pid *pid);
>
> #define do_each_pid_task(pid, type, task) \
> do { \
> diff --git a/include/linux/sched.h b/include/linux/sched.h
> index 1b1e25b..9293114 100644
> --- a/include/linux/sched.h
> +++ b/include/linux/sched.h
> @@ -1286,9 +1286,8 @@ struct pid_namespace;
> * from various namespaces
> *
> * task_xid_nr() : global id, i.e. the id seen from the init namespace;
> - * task_xid_vnr() : virtual id, i.e. the id seen from the namespace the task
> - * belongs to. this only makes sense when called in the
> - * context of the task that belongs to the same namespace;
> + * task_xid_vnr() : virtual id, i.e. the id seen from the pid namespace of
> + * current.
> * task_xid_nr_ns() : id seen from the ns specified;
> *
> * set_task_vxid() : assigns a virtual id to a task;
> diff --git a/kernel/pid.c b/kernel/pid.c
> index 21f027c..c507ca7 100644
> --- a/kernel/pid.c
> +++ b/kernel/pid.c
> @@ -442,6 +442,12 @@ pid_t pid_nr_ns(struct pid *pid, struct pid_namespace *ns)
> return nr;

```

```
> }
>
> +pid_t pid_vnr(struct pid *pid)
> +{
> + return pid_nr_ns(pid, current->nsproxy->pid_ns);
> +}
> +EXPORT_SYMBOL_GPL(pid_vnr);
> +
> pid_t task_pid_nr_ns(struct task_struct *tsk, struct pid_namespace *ns)
> {
> return pid_nr_ns(task_pid(tsk), ns);
```

---

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

---

---

Subject: Re: [PATCH] pid: Extend/Fix pid\_vnr  
Posted by [Oleg Nesterov](#) on Thu, 06 Dec 2007 17:15:13 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

On 12/05, Eric W. Biederman wrote:

```
>
> +pid_t pid_vnr(struct pid *pid)
> +{
> + return pid_nr_ns(pid, current->nsproxy->pid_ns);
> +}
```

Excellent!!!

This allows us to do many cleanups. I am sending the trivial patch just as example.

Oleg.

---

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

---

---

Subject: [PATCH] sys\_getsid: don't use ->nsproxy directly  
Posted by [Oleg Nesterov](#) on Thu, 06 Dec 2007 17:53:14 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

With the new semantics of find\_vpid() we don't need to play with ->nsproxy

explicitely, \_vxx() do the right things.

Also s/tasklist/rcu/.

Signed-off-by: Oleg Nesterov <oleg@tv-sign.ru>

```
--- PT/kernel/sys.c~ 2007-12-05 21:43:18.000000000 +0300
+++ PT/kernel/sys.c 2007-12-06 20:09:02.000000000 +0300
@@ -1025,19 +1025,16 @@ asmlinkage long sys_getsid(pid_t pid)
     else {
         int retval;
         struct task_struct *p;
-        struct pid_namespace *ns;

-        ns = current->nsproxy->pid_ns;
-
-        read_lock(&tasklist_lock);
-        p = find_task_by_pid_ns(pid, ns);
+        rcu_read_lock();
+        p = find_task_by_vpid(pid);
         retval = -ESRCH;
         if (p) {
             retval = security_task_getsid(p);
             if (!retval)
-                retval = task_session_nr_ns(p, ns);
+                retval = task_session_vnr(p);
         }
-        read_unlock(&tasklist_lock);
+        rcu_read_unlock();
         return retval;
     }
 }
```

---

Containers mailing list

Containers@lists.linux-foundation.org

<https://lists.linux-foundation.org/mailman/listinfo/containers>

---

---

Subject: Re: [PATCH] pid: Extend/Fix pid\_vnr

Posted by [Sukadev Bhattiprolu](#) on Fri, 07 Dec 2007 01:04:24 GMT

[View Forum Message](#) <> [Reply to Message](#)

---

Oleg Nesterov [oleg@tv-sign.ru] wrote:

| On 12/05, Eric W. Biederman wrote:

| >

| > +pid\_t pid\_vnr(struct pid \*pid)

| > +{

```
| > + return pid_nr_ns(pid, current->nsproxy->pid_ns);  
| > +}
```

Hmm. current->nsproxy be NULL during process exit ?  
So this safe as long as pid\_vnr() is not called after a  
process exits its namespaces. Probably no such callers atm.

```
|  
| Excellent!!!  
|  
| This allows us to do many cleanups. I am sending the trivial patch just  
| as example.  
|  
| Oleg.
```

---

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

---

---

Subject: Re: [PATCH] pid: Extend/Fix pid\_vnr  
Posted by [serue](#) on Fri, 07 Dec 2007 01:57:27 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

Quoting [sukadev@us.ibm.com](mailto:sukadev@us.ibm.com) ([sukadev@us.ibm.com](mailto:sukadev@us.ibm.com)):  
> Oleg Nesterov [[oleg@tv-sign.ru](mailto:oleg@tv-sign.ru)] wrote:  
> | On 12/05, Eric W. Biederman wrote:  
> | >  
> | > +pid\_t pid\_vnr(struct pid \*pid)  
> | > +{  
> | > + return pid\_nr\_ns(pid, current->nsproxy->pid\_ns);  
> | > +}  
>  
> Hmm. current->nsproxy be NULL during process exit ?  
> So this safe as long as pid\_vnr() is not called after a  
> process exits its namespaces. Probably no such callers atm.

Yes I did a little audit for those this morning bc I frankly didn't  
believe there weren't any. But I couldn't find any :)

-serge

```
> |  
> | Excellent!!!  
> |  
> | This allows us to do many cleanups. I am sending the trivial patch just  
> | as example.  
> |
```

> | Oleg.

>

> Containers mailing list

> Containers@lists.linux-foundation.org

> <https://lists.linux-foundation.org/mailman/listinfo/containers>

---

Containers mailing list

Containers@lists.linux-foundation.org

<https://lists.linux-foundation.org/mailman/listinfo/containers>

---

---

Subject: Re: [PATCH] pid: Extend/Fix pid\_vnr

Posted by [ebiederm](#) on Fri, 07 Dec 2007 18:10:09 GMT

[View Forum Message](#) <> [Reply to Message](#)

---

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

> Quoting sukadev@us.ibm.com (sukadev@us.ibm.com):

>> Oleg Nesterov [oleg@tv-sign.ru] wrote:

>> | On 12/05, Eric W. Biederman wrote:

>> | >

>> | > +pid\_t pid\_vnr(struct pid \*pid)

>> | > +{

>> | > + return pid\_nr\_ns(pid, current->nsproxy->pid\_ns);

>> | > +}

>>

>> Hmm. current->nsproxy be NULL during process exit ?

>> So this safe as long as pid\_vnr() is not called after a

>> process exits its namespaces. Probably no such callers atm.

>

> Yes I did a little audit for those this morning bc I frankly didn't

> believe there weren't any. But I couldn't find any :)

Cool. The only case pid\_vnr would make sense in that context is if we were talking to user space after we had exited our namespaces. Which is at least as stretch.

Eric

---

Containers mailing list

Containers@lists.linux-foundation.org

<https://lists.linux-foundation.org/mailman/listinfo/containers>

---