
Subject: process_group()

Posted by [Sukadev Bhattiprolu](#) on Sat, 20 Jan 2007 20:19:59 GMT

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

We currently have:

```
static inline pid_t process_group(struct task_struct *tsk)
{
    return tsk->signal->pgrp;
}
and
```

```
static inline struct pid *task_pgrp(struct task_struct *task)
{
    return task->group_leader->pids[PIDTYPE_PGID].pid;
}
```

and we are replacing process_group() with task_pgrp() and eventually plan to remove process_group().

But there are several places in the kernel where we interact with user space using a pid_t (obvious being sys_setpgid(), sys_getpgid() do_task_stat(), do_wait() etc).

In all these places, process_group(p) would simply be replaced by pid_nr(task_pgrp(p)). Rather than do that same replacement in many places, can we keep the interface and change the implementation to:

```
static inline pid_t process_group(struct task_struct *tsk)
{
    return pid_nr(task_pgrp(tsk));
}
```

i.e our ultimate goal is not really to remove process_group() but actually to remove the caching of pid_t in signal->pgrp right ?

The above discussion is also valid for process_session()/task_session().

Containers mailing list

Containers@lists.osdl.org

<https://lists.osdl.org/mailman/listinfo/containers>

Subject: Re: process_group()

Posted by [ebiederm](#) on Sun, 21 Jan 2007 02:59:46 GMT

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

Sukadev Bhattiprolu <sukadev@us.ibm.com> writes:

> We currently have:

```
>
>
> static inline pid_t process_group(struct task_struct *tsk)
```

```
> {
>     return tsk->signal->pgrp;
> }
```

> and

```
>
> static inline struct pid *task_pgrp(struct task_struct *task)
```

```
> {
>     return task->group_leader->pids[PIDTYPE_PGID].pid;
> }
```

> and we are replacing process_group() with task_pgrp() and eventually
> plan to remove process_group().

>

> But there are several places in the kernel where we interact with
> user space using a pid_t (obvious being sys_setpgid(), sys_getpgid()
> do_task_stat(), do_wait() etc).

>

> In all these places, process_group(p) would simply be replaced by
> pid_nr(task_pgrp(p)). Rather than do that same replacement in many
> places, can we keep the interface and change the implementation to:

>

```
> static inline pid_t process_group(struct task_struct *tsk)
```

```
> {
>     return pid_nr(task_pgrp(tsk));
> }
```

>

> i.e our ultimate goal is not really to remove process_group() but
> actually to remove the caching of pid_t in signal->pgrp right ?

>

> The above discussion is also valid for process_session()/task_session().

Close. Our ultimate goal is to make it so that when you talk within
the kernel you use a struct pid not a pid_t value. Attacking the
cached pid_t values is merely a way finding those places.

So fixing thing like the pid_t value passed as credentials in unix domain
sockets is a lot more important than fixing any use of process_session
that just goes to user space.

The reason it is important is because different processes may be in different
pid namespaces and raw pid_t values just won't make sense while struct pid
references are pid namespace independent.

Eric

Containers mailing list
Containers@lists.osdl.org
<https://lists.osdl.org/mailman/listinfo/containers>

Subject: Re: process_group()
Posted by [ebiederm](#) on Sun, 21 Jan 2007 06:23:43 GMT
[View Forum Message](#) <> [Reply to Message](#)

ebiederm@xmission.com (Eric W. Biederman) writes:

>
> Close. Our ultimate goal is to make it so that when you talk within
> the kernel you use a struct pid not a pid_t value. Attacking the
> cached pid_t values is merely a way finding those places.
>
> So fixing thing like the pid_t value passed as credentials in unix domain
> sockets is a lot more important than fixing any use of process_session
> that just goes to user space.
>
> The reason it is important is because different processes may be in different
> pid namespaces and raw pid_t values just won't make sense while struct pid
> references are pid namespace independent.

The other reason for preferring a struct pid form is that it avoids unnecessary hash table lookups, that we get processing pid in pid_t form.

Eric

Containers mailing list
Containers@lists.osdl.org
<https://lists.osdl.org/mailman/listinfo/containers>

Subject: Re: process_group()
Posted by [Cedric Le Goater](#) on Wed, 24 Jan 2007 17:31:23 GMT
[View Forum Message](#) <> [Reply to Message](#)

Eric W. Biederman wrote:

[...]

> Close. Our ultimate goal is to make it so that when you talk within
> the kernel you use a struct pid not a pid_t value. Attacking the
> cached pid_t values is merely a way finding those places.

>
> So fixing thing like the pid_t value passed as credentials in unix domain
> sockets is a lot more important than fixing any use of process_session
> that just goes to user space.
>
> The reason it is important is because different processes may be in different
> pid namespaces and raw pid_t values just won't make sense while struct pid
> references are pid namespace independent.

BTW, in rc4-mm1, we've nearly closed down the list from (needs an update) :

<http://wiki.openvz.org/Containers/Pidspace>

NFS is still pending.

kthread is doing fine also.

But, there are some pid_t values left over like in struct ucred you just mentioned. Any idea on how to track them down and prioritize them ? because we are real close to have all the prerequisites for the pid namespace.

thanks,

C.

Containers mailing list
Containers@lists.osdl.org
<https://lists.osdl.org/mailman/listinfo/containers>
