Subject: [PATCH 5/9] pid: Update pid_vnr to use task_active_pid_ns
Posted by ebiederm on Wed, 12 Dec 2007 12:48:18 GMT
View Forum Message <> Reply to Message

There is no real additional cost in using task_active_pid_ns and
by doing so it means we do not need to audit the callers of pid_vnr
closely, to ensure they are always called on living processes.

Signed-off-by: Eric W. Biederman <ebiederm@xmission.com>
---
 kernel/pid.c | 2 +-
 1 files changed, 1 insertions(+), 1 deletions(-)

diff --git a/kernel/pid.c b/kernel/pid.c
index 54d7634..a7ecfce 100644
--- a/kernel/pid.c
+++ b/kernel/pid.c
@@ -444,7 +444,7 @@ pid_t pid_nr_ns(struct pid *pid, struct pid_namespace *ns)

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

--
1.5.3.rc6.17.g1911

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

Subject: [PATCH 6/9] pid: Implement pid_in_pid_ns.
Posted by ebiederm on Wed, 12 Dec 2007 12:49:17 GMT
View Forum Message <> Reply to Message

Implement a simple check to see if a specified pid resides
in a specified pid namespace. Generally we are smart enough
just to operate on single pid namespace and avoid this test.
However in the case of sending a signal to init we must check
to see if our sender is a child of init and so this test is needed.

Signed-off-by: Eric W. Biederman <ebiederm@xmission.com>
---
 include/linux/pid.h | 3 +++

```
 kernel/pid.c        |    6 ++++++
 2 files changed, 9 insertions(+), 0 deletions(-)

diff --git a/include/linux/pid.h b/include/linux/pid.h
index c4b56c0..e409cc5 100644
--- a/include/linux/pid.h
+++ b/include/linux/pid.h
@@ -156,6 +156,9 @@ static inline pid_t pid_nr(struct pid *pid)
 pid_t pid_nr_ns(struct pid *pid, struct pid_namespace *ns);
 pid_t pid_vnr(struct pid *pid);

+/* Test to see if pid is present in pid namespace ns */
+extern int pid_in_pid_ns(struct pid *pid, struct pid_namespace *ns);
+
 #define do_each_pid_task(pid, type, task)    \
  do {        \
   struct hlist_node *pos___;    \
diff --git a/kernel/pid.c b/kernel/pid.c
index a7ecfce..873c00f 100644
--- a/kernel/pid.c
+++ b/kernel/pid.c
@@ -429,6 +429,12 @@ struct pid *find_get_pid(pid_t nr)
  return pid;
 }

+int pid_in_pid_ns(struct pid *pid, struct pid_namespace *ns)
+{
+ return pid && ns && (ns->level <= pid->level) &&
+  pid->numbers[ns->level].ns == ns;
+}
+
 pid_t pid_nr_ns(struct pid *pid, struct pid_namespace *ns)
 {
  struct upid *upid;
--
1.5.3.rc6.17.g1911
```

Subject: Re: [PATCH 5/9] pid: Update pid_vnr to use task_active_pid_ns
Posted by Pavel Emelianov on Wed, 12 Dec 2007 13:28:33 GMT
View Forum Message <> Reply to Message

Eric W. Biederman wrote:

> There is no real additional cost in using task_active_pid_ns and
> by doing so it means we do not need to audit the callers of pid_vnr
> closely, to ensure they are always called on living processes.
>
> Signed-off-by: Eric W. Biederman <ebiederm@xmission.com>
> ---
> kernel/pid.c | 2 +-
> 1 files changed, 1 insertions(+), 1 deletions(-)
>
> diff --git a/kernel/pid.c b/kernel/pid.c
> index 54d7634..a7ecfce 100644
> --- a/kernel/pid.c
> +++ b/kernel/pid.c
> @@ -444,7 +444,7 @@ pid_t pid_nr_ns(struct pid *pid, struct pid_namespace *ns)
>
> pid_t pid_vnr(struct pid *pid)
> {
> - return pid_nr_ns(pid, current->nsproxy->pid_ns);
> + return pid_nr_ns(pid, task_active_pid_ns(current));

NAK. Your version will make more dereferences and one
condition check (if (pid != NULL)). Besides, dead process
can never call this thing - it is even never scheduled
for execution.

> }
> EXPORT_SYMBOL_GPL(pid_vnr);
>

Subject: Re: [PATCH 6/9] pid: Implement pid_in_pid_ns.
Posted by Pavel Emelianov on Wed, 12 Dec 2007 13:33:29 GMT
View Forum Message <> Reply to Message

Eric W. Biederman wrote:
> Implement a simple check to see if a specified pid resides
> in a specified pid namespace. Generally we are smart enough
> just to operate on single pid namespace and avoid this test.
> However in the case of sending a signal to init we must check
> to see if our sender is a child of init and so this test is needed.
>
> Signed-off-by: Eric W. Biederman <ebiederm@xmission.com>
> ---

> include/linux/pid.h | 3 +++
> kernel/pid.c | 6 ++++++
> 2 files changed, 9 insertions(+), 0 deletions(-)
>
> diff --git a/include/linux/pid.h b/include/linux/pid.h
> index c4b56c0..e409cc5 100644
> --- a/include/linux/pid.h
> +++ b/include/linux/pid.h
> @@ -156,6 +156,9 @@ static inline pid_t pid_nr(struct pid *pid)
> pid_t pid_nr_ns(struct pid *pid, struct pid_namespace *ns);
> pid_t pid_vnr(struct pid *pid);
>
> +/* Test to see if pid is present in pid namespace ns */
> +extern int pid_in_pid_ns(struct pid *pid, struct pid_namespace *ns);
> +
> #define do_each_pid_task(pid, type, task)   \
>   do {   \
>   struct hlist_node *pos___;   \
> diff --git a/kernel/pid.c b/kernel/pid.c
> index a7ecfce..873c00f 100644
> --- a/kernel/pid.c
> +++ b/kernel/pid.c
> @@ -429,6 +429,12 @@ struct pid *find_get_pid(pid_t nr)
>   return pid;
> }
>
> +int pid_in_pid_ns(struct pid *pid, struct pid_namespace *ns)

Can we give it a better name? Like pid_in_ns() if we do want it
to be as short as possible, or pid_visible_in_ns() if we want
it ti reflect its nature.

> +{
> + return pid && ns && (ns->level <= pid->level) &&
> + pid->numbers[ns->level].ns == ns;
> +}
> +
> pid_t pid_nr_ns(struct pid *pid, struct pid_namespace *ns)
> {
>   struct upid *upid;

Thanks,
Pavel

Subject: Re: [PATCH 5/9] pid: Update pid_vnr to use task_active_pid_ns
Posted by ebiederm on Wed, 12 Dec 2007 14:20:19 GMT
View Forum Message <> Reply to Message

> NAK. Your version will make more dereferences and one
> condition check (if (pid != NULL)). Besides, dead process
> can never call this thing - it is even never scheduled
> for execution.

Andrew please disregard this patch, it should not affect
the rest of the patch series.

Well EXIT_ZOMBIE is enough to cause us problems, not EXIT_DEAD,
so we can be scheduled.

Still we don't have any users currently and I actually don't expect us
to develop any, so this patch was just overkill right now.

I was thinking that there was a path to send_signal when
we were zombie with SEND_SIG_NOINFO, but that doesn't seem
to exist, and if there isn't one now we aren't likely
to grow one, and my signal patches remove the possibility
of that path being a problem.

Hmm. As to the cost argument.
Both pid->level and pid->numbers[pid->level].ns should be
on the same cache line so it should not be any real
extra memory reads. The branch should be predicted not
to happen so again it should not have a cost except a small
bit of code size.

So while I don't expect any real world cost this is the
common case so we don't want to put to many hoops in our
way either.

Eric