Subject: Re: [RFC][PATCH 3/5] Use pid namespace from struct pid_nrs list
Posted by Sukadev Bhattiprolu on Mon, 12 Mar 2007 21:35:57 GMT
View Forum Message <> Reply to Message

Eric W. Biederman [ebiederm@xmission.com] wrote:
| "Serge E. Hallyn" <serue@us.ibm.com> writes:
|
| > Quoting Eric W. Biederman (ebiederm@xmission.com):
| >> sukadev@us.ibm.com writes:
| >>
| >> > From: Sukadev Bhattiprolu <sukadev@us.ibm.com>
| >> > Subject: [RFC][PATCH 3/5] Use pid namespace from struct pid_nrs list
| >> >
| >> > Stop using task->nsproxy->pid_ns. Use pid_namespace from pid->pid_nrs
| >> > list instead.
| >> >
| >> > To simplify error handling, this patch moves processing of CLONE_NEWPID
| >> > flag, currently in copy_namespaces()/copy_process(), to alloc_pid() which
| >> > is where the process association with a pid namespace is established.
| >> >
| >> > i.e when cloning a new pid namespace, alloc_pid() allocates a new pid_nr
| >> > for both the parent and child namespaces.
| >>
| >>
| >> This patch seems to do a bit much, it is hard to follow what changes you
| >> are making.
| >
| > Is this a design comment, or do you mean you'd like to see it broken
| > into two or more patches?
|
| The latter. There is no reason for the changes to remove the use
| of nsproxy->pid_ns need to be all in one patch.

I will try to break this up into smaller patches.

|
| >> It probably makes sense to modify things so alloc_pid can do everything
| >> it needs to.
| >>
| >> It looks like we can safely move alloc_pid into copy_process and
| >> just dig out the pid number and place it in nr if copy_process succeeds.
| >>
| >> Which should allow the special case for setting the child reaper to go
| >> away, because we can allocate the task_struct before allocating the struct
| >> pid.
| >
| > That would be nice. That little reaper setting helper bugs me.

Yes. I like that idea too.

|
| Which is why I suggested the reorganization....
|
|

```
| >> > Index: lx26-20-mm2b/kernel/pid.c
| >> > ===================================================================
| >> > --- lx26-20-mm2b.orig/kernel/pid.c 2007-03-09 19:00:42.000000000 -0800
| >> > +++ lx26-20-mm2b/kernel/pid.c 2007-03-09 19:01:09.000000000 -0800
| >> > @@ -221,8 +221,13 @@ fastcall void free_pid(struct pid *pid)
| >> >   hlist_del_rcu(&pid->pid_chain);
| >> >   spin_unlock_irqrestore(&pidmap_lock, flags);
| >> >
| >> > - hlist_for_each_entry(pid_nr, pos, &pid->pid_nrs, node)
| >> > + hlist_for_each_entry(pid_nr, pos, &pid->pid_nrs, node) {
| >> >    free_pidmap(pid_nr->pid_ns, pid_nr->nr);
| >> > +
| >> > +  /* put the reference we got in kref_init() in clone_pid_ns() */
| >> > +  if (pid_nr->nr == 1)
| >> > +   put_pid_ns(pid_nr->pid_ns);
```

| >>
| >> Ok. This seems to make sense, but why restrict this to only pid 1?
| >> I'm almost certain this will be the case, but... this seems a like
| >> a unwarranted special case at the moment.
| >>
| >> Basically why is it safe to restrict this to pid == 1. Is it possible
| >> that we can race here?
| >
| > I think he's dropping an extra reference due to the pid_ns count being
| > set to 1 then alloce'd for each pid. Rather than worry about a race
| > here i'd prefer the extra reference be gotten rid of.

Yes. I automatically get the first reference in the kref_init(). Subsequent
alloc_pid_nr()/free_pid_nr() calls would be properly paired to get/put the
kref count.

I needed a special-case to drop the reference I got in kref_init(). If I can
initialize ns->kref in clone_pid_ns() to zero, I would not need this special
case. For that I would have to call atomic_set(&ns->kref, 0) - and that looked
ugly/dangerous to me.

|
| This is the only put_pid_ns I could find, and free_pid is the only
| place I could find it.
|
| Regardless casual inspection of the code is not showing what is going
| on and why so this part of the patch needs to be addressed.

|
| >> > -struct pid *alloc_pid(void)
| >> > +struct pid *alloc_pid(int flags)
| >> > {
| >> >   struct pid *pid;
| >> >   enum pid_type type;
| >> > - int nr = -1;
| >> > - struct pid_nr *pid_nr;
| >> > + struct pid_nr *pid_nr[2] = { NULL, NULL};
| >> I would rather not see pid_nr special cased this way at all (a loop?)
| >> but if we are going to I think two separate variables makes more
| >> sense than this array.

Yes - we want to alloc pid_nrs for all ancestor namespaces (I should
have added the comments at the function header). When I do that I will
use a loop and a single variable.

| >
| > Yes, the plan is for it to become a loop, with another CLONE flag to
| > specify whether all parent pid_namespaces should get a pid entry for
| > these processes or not. I'd love to just make that always the case, but
| > I'm afraid the clone flag is necessary else kernel memory use is going
| > to skyrocket too quickly.
|
| I doubt kernel memory will sky rocket. The normal case is just two
| pids. We can have an arbitrary nesting limit to prevent the worst
| abuses.
|
| If you don't create all of the pids you get into weird semantic
| problems, and a lot more complex kernel/user space interface.
|
| Eric
____
Containers mailing list
Containers@lists.osdl.org
https://lists.osdl.org/mailman/listinfo/containers