
Subject: Re: [RFC][PATCH 4/4] PID: use the target ID specified in procs

Posted by [serue](#) on Tue, 11 Mar 2008 16:47:25 GMT

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

Quoting Pavel Emelyanov (xemul@openvz.org):

> Nadia Derby wrote:

> > Pavel Emelyanov wrote:

> > > Nadia.Derbey@bull.net wrote:

> > >

> > > @@ -122,14 +122,26 @@ static void free_pidmap(struct upid *upi

> > > atomic_inc(&map->nr_free);

> > > }

> > >

> > > -static int alloc_pidmap(struct pid_namespace *pid_ns)

> > > +static int alloc_pidmap(struct pid_namespace *pid_ns, struct pid_list *pid_l,

> > > + int level)

> > > {

> > > int i, offset, max_scan, pid, last = pid_ns->last_pid;

> > > struct pidmap *map;

> > >

> > > - pid = last + 1;

> > > - if (pid >= pid_max)

> > > - pid = RESERVED_PIDS;

> > > + if (!pid_l) {

> > > + pid = last + 1;

> > > + if (pid >= pid_max)

> > > + pid = RESERVED_PIDS;

> > > + } else {

> > > + /*

> > > + * There's a target pid, so use it instead

> > > + */

> > > + BUG_ON(level < 0);

> > > + pid = PID_AT(pid_l, level);

> > > + if (pid >= pid_max)

> > > + return -EINVAL;

> > > + }

> > > +

> > > offset = pid & BITS_PER_PAGE_MASK;

> > > map = &pid_ns->pidmap[pid/BITS_PER_PAGE];

> > > max_scan = (pid_max + BITS_PER_PAGE - 1)/BITS_PER_PAGE - !offset;

> > > @@ -153,9 +165,16 @@ static int alloc_pidmap(struct pid_names

> > > do {

> > > if (!test_and_set_bit(offset, map->page)) {

> > > atomic_dec(&map->nr_free);

> > > - pid_ns->last_pid = pid;

> > > + if (!pid_l)

> > > + pid_ns->last_pid = pid;

> > > + else

```

>>> +   pid_ns->last_pid = max(last,
>>> +   pid);
>>>   return pid;
>>>   }
>>> +   if (pid_l)
>>> +   /* Target pid is already in use */
>>> +   return -EBUSY;
>>>   offset = find_next_offset(map, offset);
>>>   pid = mk_pid(pid_ns, map, offset);
>>>   /*
>>> @@ -179,7 +198,7 @@ static int alloc_pidmap(struct pid_names
>>>   }
>>>   pid = mk_pid(pid_ns, map, offset);
>>>   }
>>> - return -1;
>>> + return -ENOMEM;
>>> }
>>>
>>> int next_pidmap(struct pid_namespace *pid_ns, int last)
>>>
>> As fas as this particular piece of code is concerned this all can
>> be shrunk down to
>>>
>> static int set_vpidmap(struct pid_namespace *ns, int pid)
>> {
>>     int offset;
>>     pidmap_t *map;
>>>
>>     offset = pid & BITS_PER_PAGE_MASK;
>>     map = ns->pidmap + vpid / BITS_PER_PAGE;
>>>
>>     if (unlikely(alloc_pidmap_page(map)))
>>         return -ENOMEM;
>>>
>>     if (test_and_set_bit(offset, map->page))
>>         return -EEXIST;
>>>
>>     atomic_dec(&map->nr_free);
>>     return pid;
>>> }
>>>
>> where the alloc_pidmap_page is a consolidated part of code from alloc_pidmap.
>>>
>> And I'm scared of what the alloc_pid is going to become.
>>>
>>>
>>
>> It's true that I made alloc_pid() become ugly, but this patchset was

```

```

> > more intended to continue a discussion.
> >
> > What we could do is the following (not compiled, not tested...):
> >
> > struct pid *alloc_pid(struct pid_namespace *ns)
> > {
> >     struct pid *pid;
> >     enum pid_type type;
> >     int i, nr;
> >     struct pid_namespace *tmp;
> >     struct upid *upid;
> >
> >     pid = kmem_cache_alloc(ns->pid_cachep, GFP_KERNEL);
> >     if (!pid) {
> >         pid = ERR_PTR(-ENOMEM);
> >         goto out;
> >     }
> >
> >     tmp = ns;
> >     i = ns->level;
> >
> >     if (current->next_id && (current->next_id->flag & SYS_ID_PID)) {
> >         tmp = set_predefined_pids(ns,
> >             current->next_id->pid_ids);
> >         if (IS_ERR(tmp)) {
> >             nr = PTR_ERR(tmp);
> >             goto out_free;
> >         }
> >     }
> >
> >     /*
> >     * Let the lower levels upid nrs be automatically allocated
> >     */
> >     for ( ; i >= 0; i--) {
> >         nr = alloc_pidmap(tmp, NULL, -1);
> >         if (nr < 0)
> >             goto out_free;
> >     ....
> >
> > which would only add a test and a function call to alloc_pid() ==> more
> > readable.
> > with set_predefined_pids defined as follows (still not compiled, not
> > tested, ...):
> >
> > struct pid_namespace *set_predefined_pids(struct pid_namespace *ns,
> >     struct pid_list *pid_l)
> > {
> >     int rel_level;

```

```

>>
>>     BUG_ON(!pid_l);
>>
>>     rel_level = pid_l->npids - 1;
>>     if (rel_level > ns->level)
>>         return ERR_PTR(-EINVAL);
>>
>>     /*
>>      * Use the predefined upid nrs for levels ns->level down to
>>      * ns->level - rel_level
>>      */
>>     for ( ; rel_level >= 0; i--, rel_level--) {
>>         nr = alloc_pidmap(tmp, pid_l, rel_level);
>>         if (nr < 0)
>>             return ERR_PTR(nr);
>>
>>         pid->numbers[i].nr = nr;
>>         pid->numbers[i].ns = tmp;
>>         tmp = tmp->parent;
>>     }
>>
>>     current->next_id->flag &= ~SYS_ID_PID;
>>     pids_free(pid_l);
>>     current->next_id->pid_ids = NULL;
>>
>>     return tmp;
>> }
>>
>>
>> Don't you think that mixing this with your 1st proposal (the
>> set_vpidmap() one), would make things look better?
>
> I'd prefer seeing
>
> --- a/kernel/pid.c
> +++ b/kernel/pid.c
> @@ -247,7 +247,7 @@ struct pid *alloc_pid(struct pid_namespace *ns)
> {
>     struct pid *pid;
>     enum pid_type type;
> -    int i, nr;
> +    int i, nr, req_nr;
>     struct pid_namespace *tmp;
>     struct upid *upid;
>
> @@ -257,7 +257,11 @@ struct pid *alloc_pid(struct pid_namespace *ns)
>
>     tmp = ns;

```

```

>     for (i = ns->level; i >= 0; i--) {
> -         nr = alloc_pidmap(tmp);
> +         req_nr = get_required_pidnr(ns, i);
> +         if (req_nr > 0)
> +             nr = set_pidmap(tmp, req_nr);

```

I assume you mean `set_vpidmap(tmp, req_nr);` here?

```

> +         else
> +             nr = alloc_pidmap(tmp);
>         if (nr < 0)
>             goto out_free;
>
>
> in alloc_pid() and not much than that.

```

So `get_required_pidnr(ns, i)` would do something like

```

int get_required_pidnr(struct pid_namespace *ns, int i)
{
    if (current->next_id && (current->next_id->flag & SYS_ID_PID)) {
        pid_l = current->next_id->pid_ids;
        if (!pid_l) return 0;
        rel_level = pid_l->npids - 1;
        if (rel_level <= i)
            return PID_AT(pid_l, i);
    }
    return 0;
}

```

?

```

>
> > Regards,
> > Nadia
> >

```

```

>
> _____
> 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

```
