

---

Subject: Re: Processes with multiple pid\_t values  
Posted by [Sukadev Bhattiprolu](#) on Tue, 09 Jan 2007 03:33:18 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

Hmm. I have this mail sitting in my sent folder since Dec 18 but do not see it in the Containers archives. If you already received, sorry to spam. Appreciate any clarifications of my doubt :-)

Suka

Date: Mon, 18 Dec 2006 18:37:01 -0800  
From: Sukadev Bhattiprolu <sukadev@us.ibm.com>  
To: "Eric W. Biederman" <ebiederm@xmission.com>  
Cc: Containers <containers@lists.osdl.org>  
Subject: Re: Processes with multiple pid\_t values

Eric W. Biederman [ebiederm@xmission.com] wrote:  
| Sukadev Bhattiprolu <sukadev@us.ibm.com> writes:

|  
| > A process that unshares its namespace gets a new pid\_t in the child  
| > namespace. Similarly its process group and session leaders get new pid\_ts  
| > in the child namespace right ?  
| >  
| > i.e do the following pid\_ts look reasonable when process 1234 unshares  
| > its pid namespace ?  
| >  
| >  
| > PID PPID PGID SID  
| >  
| > init pid ns 1234 1233 1230 1220  
| >  
| > child pid ns 3 2 1 0

| A slightly more complete answer.

| A pid that cannot be represented in the current pid namespace should be  
| 0.

For my example above, I guess you are saying we want the following ids.

PID PPID PGID SID

init pid ns 1234 1233 1230 1220

child pid ns 1 0 1 1

If so, my confusion is that for the process 1234, following are true

right ?

```
task_pgrp(current) != task_session(current) != task_pid(current)
```

If we associate a list of [namespace, id] tuples with each struct pid, then three different struct pids would have the same id in the child namespace (unless the process calling clone() is both a session and process group leader).

|  
|  
| pid 1 is very special and in the case of a clone should definitely  
| be the first pid in the namespace.

|  
| In the case of an unshare pid == 1 is probably the process that does  
| the unshare, and it's children all show up in the child namespace.

|  
| Any other pids should simply be allocated with the pid allocator if they  
| do not fall into the pid namespace.

|  
| > Assuming they are :-), we should expect find\_pid() call with nr == 0, 1,  
| > or 2 in the child pid namespace to also work right ?

|  
| >  
| > But processes 1220, 1230, 1233 are entered into the hash table based on  
| > their init pid ns values. And so the above find\_pid() calls would not  
| > find the process we want.

|  
| My expectation is that we will have a link entry that points to the  
| real struct pid. Or possibly a different data structure at that point.  
| The pid hash table does not scale well to very large numbers of pids.

|  
| > i.e some processes have two pid\_ts and we want to find them using either  
| > of the two values. The pid\_hash table can obviously hash on one value.

|  
| >  
| > We would need some serious changes to the pid\_hash table to do this ???  
| > (can we change the hash algorithm to generate a key based on all pid\_ts  
| > a process has ????)

|  
| The key would need to be pid\_namespace, pid\_t. We just need a few tweaks  
| to the lookup algorithm. It's 5-10 line patch most likely.

Yes that makes more sense and works for nested containers.

|  
| > Or should we just keep track of these special processes (4 per namespace  
| > including the child reaper) in the namespace object - just like we treat  
| > the child\_reaper special ?

|  
| It doesn't look hard to allow pids to appear in several namespaces,

| we need at least one pid to appear in multiple pid namespaces, so  
| it makes sense to handle that case. With struct pid and the helper  
| functions currently defined it is not hard to allow for all pids to  
| be in several namespaces.

| So it is my intention that all processes will have a pid in every parent  
| pid namespace as well as a pid in their current pid namespace.

| Eric

---

Containers mailing list

Containers@lists.osdl.org

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

---