Subject: Re: Pid namespaces approaches testing results
Posted by Cedric Le Goater on Fri, 01 Jun 2007 08:47:26 GMT
View Forum Message <> Reply to Message

Serge E. Hallyn wrote:
> Quoting Pavel Emelianov (xemul@openvz.org):
>> Dave Hansen wrote:
>>> On Tue, 2007-05-29 at 15:45 +0400, Pavel Emelianov wrote:
>>>> The detailed results are the following:
>>>> Test name: spawn execl shell ps (sys time)
>>>> 1(no ns) : 579.1 618.3 1623.2 3.052s
>>>> 2(suka's): 570.7 610.8 1600.2 3.107s
>>>> Slowdown : 1.5% 1.3% 1.4% 1.8%
>>>>
>>>> 3(no ns) : 580.6 616.0 1633.8 3.050s
>>>> 4(flat) : 580.8 615.1 1632.2 3.054s
>>>> Slowdown : 0% 0.1% <0.1% 0.1%
>>>> 5(multi) : 576.9 611.0 1618.8 3.065s
>>>> Slowdown : 0.6% 0.8% 0.9% 0.5%
>>> Wow, thanks so much for running those. You're a step ahead of us,
>>> there!
>> Thanks :) Maybe we shall cooperate then and make three series
>> of patches like
>>
>> 1. * The Kconfig options;
>>
>> * The API. I.e. calls like task_pid_nr(), task_session_nr_ns() etc;
>> This part is rather important as I found that some places in kernel
>> where I had to lookup the hash in multilevel model were just pid->vpid
>> dereference in flat model. This is a good optimization.
>>
>> * The changes in the generic code that intruduce a bunch of
>> #ifdef CONFIG_PID_NS
>> ...
>> #else
>> #ifdef CONFIG_PID_NS_FLAT
>> #endif
>> #ifdef CONFIG_PID_NS_MULTILEVEL
>> #endif
>> #endif
>> code in pid.c, sched.c, fork.c etc
>>
>> This patchset will have to make kernel prepared for namespaces injections
>> and (!) not to break normal kernel operation with CONFIG_PID_NS=n.
>
> In principle there's nothing at all wrong with that (imo). But the
> thing is, given the way Suka's patchset is set up, there really isn't
> any reason why it should be slower when using only one or two pid

> namespaces.
>
> Suka, right now are you allocating the struct upid separately from the
> struct pid? That alone might slow things down quite a bit. By
> allocating them as one large struct - saving both an alloc at clone, and
> a dereference when looking at pid.upid[0] to get the pid_ns for instance
> - you might get some of this perf back.
>
> (Hmm, taking a quick look, it seems you're allocating the memory as one
> chunk, but then even though the struct upid is just at the end of the
> struct pid, you use a pointer to find the struct upid. That could slow
> things down a bit)

what about being more agressive and defining :

```
 struct pid
 {
  atomic_t count;
  /* lists of tasks that use this pid */
  struct hlist_head tasks[PIDTYPE_MAX];
  int num_upids;
  struct upid upid_list[CONFIG_MAX_NESTED_PIDNS];
  struct rcu_head rcu;
 };
```

if CONFIG_MAX_NESTED_PIDNS is 1, then pid namespaces are not available.
at 2, the model is flat and at 3, we start nesting them.

it should improve performance as profiling gave higher memory usage
in the current 2.6.21-mm2-pidns3 patchset.

C.

C.