Subject: Re: [RFC][PATCH] Static init struct pid for swapper
Posted by ebiederm on Mon, 15 Jan 2007 15:16:22 GMT
View Forum Message <> Reply to Message

Sukadev Bhattiprolu <sukadev@us.ibm.com> writes:

> From: Sukadev Bhattiprolu <sukadev@us.ibm.com>
> Subject: Statically initialize struct pid for swapper
>
> Statically initialize a struct pid for the swapper process (pid_t == 0)
> and attach it to init_task. This is needed so task_pid(), task_pgrp()
> and task_session() interfaces work on the swapper process also.

This looks encouraging.

We still need to address the fact that we are placing pid == 1
in an invalid process group and session. We need to call something
like setsid() to address this before we start any other threads
or /sbin/init.

All of the other idle threads are created with fork_idle() from
pid == 1 in the smp startup code, so there are still some differences
but that is independent of your patch.

Just please pass the pid as a struct pid into copy_process and
then we can remove the if (likely(p->pid)) case and use attach_pid
for everything. And please call setsid() or the equivalent about
where we recognize we are the child_reaper in init(), before we run
any other threads.

Oh yes. Please fix your whitespace damage in the initializer.

> /*
> * INIT_TASK is used to set up the first task table, touch at
> * your own risk!. Base=0, limit=0x1fffff (=2MB)
> @@ -139,6 +152,26 @@ extern struct group_info init_groups;
> .cpu_timers = INIT_CPU_TIMERS(tsk.cpu_timers), \
> .fs_excl = ATOMIC_INIT(0), \
> .pi_lock = SPIN_LOCK_UNLOCKED, \
> + .pids = { \
> + { .node = { \
> + .next = NULL, \
> + .pprev = &init_struct_pid.tasks[PIDTYPE_PID].first,\
> + }, \
> + .pid = &init_struct_pid, \
> + }, \
> + { .node = { \
> + .next = NULL, \

```
> +     .pprev = &init_struct_pid.tasks[PIDTYPE_PGID].first,\
> +    },       \
> +    .pid = &init_struct_pid,    \
> +  },       \
> +  { .node = {      \
> +    .next = NULL,     \
> +     .pprev = &init_struct_pid.tasks[PIDTYPE_SID].first,\
> +    },       \
> +    .pid = &init_struct_pid,    \
> +  },       \
> + }       \
>   INIT_TRACE_IRQFLAGS      \
>   INIT_LOCKDEP       \
>  }
```

Say something like:

```
#define INIT_PID_LINK(TYPE) =      \
{       \
 .node = {      \
  .next = NULL,     \
  .pprev = &init_struct_pid.tasks[TYPE].first, \
 },       \
 .pid = &init_struct_pid,    \
}
```

And then:

```
 .pids = {
  [PIDTYPE_PID]  = INIT_PID_LINK(PIDTYPE_PID),
  [PIDTYPE_PGID] = INIT_PID_LINK(PIDTYPE_PGID),
              [PIDTYPE_SID]  = INIT_PID_LINK(PIDTYPE_SID),
 }
```

As for the question below. It would be very bad if pid 0 every gets into the pid hash table. We have a number of cases that explicitly

```
+#define INIT_STRUCT_PID {      \
+ .count   = ATOMIC_INIT(1),    \
+ .nr  = 0,       \
+ /* Do we need to put this struct pid in pid_hash ? */  \
+ .pid_chain = { .next = NULL, .pprev = NULL },  \
+ .tasks   =  {       \
+  { .first = &init_task.pids[PIDTYPE_PID].node },  \
+  { .first = &init_task.pids[PIDTYPE_PGID].node }, \
+  { .first = &init_task.pids[PIDTYPE_SID].node },  \
+ },        \
+ .rcu  = RCU_HEAD_INIT,     \
```

```
+}
+
```

And one final thing. Please place init_pid in pid.c. We can just put an "extern struct pid init_pid;" in pid.h

There is no reason to put a separate copy in every architecture and it will be easier to have just a single copy in the code.

Eric

_____
Containers mailing list
Containers@lists.osdl.org
https://lists.osdl.org/mailman/listinfo/containers