Subject: Re: [PATCH 14/15] Destroy pid namespace on init's death
Posted by Oleg Nesterov on Mon, 30 Jul 2007 15:44:57 GMT
View Forum Message <> Reply to Message

On 07/30, Pavel Emelyanov wrote:
>
> Oleg Nesterov wrote:
> >>+
> >>+ nfree = 0;
> >>+ for (i = 0; i < PIDMAP_ENTRIES; i++)
> >>+ nfree += atomic_read(&pid_ns->pidmap[i].nr_free);
> >>+
> >>+ /*
> >>+ * If pidmap has entries for processes other than 0 and 1, retry.
> >>+ */
> >>+ if (nfree < (BITS_PER_PAGE * PIDMAP_ENTRIES - 2))
> >>+ goto repeat;
> >
> >This doesn't look right.
> >
> >Suppose that some "struct pid" was pinned from the parent namespace.
> >In that case zap_pid_ns_processes() will burn CPU until put_pid(), bad.
>
> Nope. struct pid can be pinned, but the pidmap "fingerprint" cannot.

Heh. It was specially designed this way, but I managed to forget.

You are right, thanks for correcting me.

> So as soon as the release_task() is called the pidmap becomes free and
> we can proceed.

Well, it doesn't matter, but strictly speaking this is not true.
release_task()->detach_pid(PIDTYPE_PID) doesn't necessary free pidmap,
it could be "used" by other tasks as PGID/SID.

> However I agree with the "burn CPU" issue - wait must sleep if needed.
>
> >I think we can rely on forget_original_child() and do something like
> >this:
> >
> > zap_active_ns_processes(void)
> > {
> > // kill all tasks in our ns and below
> > kill(-1, SIGKILL);
>
> That would be too slow to walk through all the tasks in a node searching
> for a couple of them we need. fing_ge_pid() looks better to me.

OK. I personally dislike the "retry" logic (and it is not safe if we want wait() to actually sleep waiting for the child), but we can do the "kill" loop under tasklist_lock.

In any case, I don't think we should use next_pid() for wait(), or check pidmap[].nr_free,

> > do {
> > clear_thread_flag(TIF_SIGPENDING);
> > } while (wait(NULL) != -ECHLD);
> > }

just wait for any child until -ECHLD. After that we can return safely.

Oleg.
