Subject: Re: kernel thread accounted to a VE
Posted by dev on Wed, 12 Dec 2007 08:35:10 GMT
View Forum Message <> Reply to Message

Eric Keller wrote:
> Is it possible to start a kernel thread and then move it to a particular
> VE?
>
> I have the following code inside of a kernel thread:
> envid_t _veid = 200;
> // enter that VE
> unsigned flags = VE_ENTER;
> int err = real_env_create(_veid, flags, 0, 0, 0); // the last 3
> arguments are only used if flags is VE_CREATE
>
> I needed to modify ve_move_task() a bit. It has the following
> assignment: tsk->mm->vps_dumpable = 0; But for kernel_threads,
> tsk->mm is NULL, so I just check if it's null and don't do the
> assignment if it is null. Other than that, it appears to be
> successful. It returns successful and in the VE I moved the task to, I
> can see a new process running (using top).
>
> The problem is, I set a cpu limit for that VE to 10%, yet I can see this
> thread go well above that amount (~50%). User processes do get limited
> when I run them, so I know it's not a setting issue (unless there's
> something special I need to do for kernel threads). Note that I do not
> want to allow the VEs to install kernel modules, so I want the host
> system to do it on their behalf for a very specific circumstance.
>
> Any ideas of what I'm doing wrong or what it'll take to make this work?

you can fix the place about checking for tsk->mm != NULL.
But... plz keep in mind the following:
1. having a kernel thread inside VE will break checkpointing (live migration),
   since CPT doesn't know how to restore this thread.
   (it can be fixed by you if you know how to save/restore it's state).
2. your kernel thread should handle signals or have an ability
   to detect VE shutdown, otherwise it will block VE stop.

and maybe something else...

Thanks,
Kirill