Subject: Re: [ckrm-tech] [PATCH 4/13] BC: context handling
Posted by Pavel Emelianov on Thu, 23 Nov 2006 10:45:23 GMT
View Forum Message <> Reply to Message

Paul Menage wrote:
> On 11/23/06, Pavel Emelianov <xemul@openvz.org> wrote:
>> Paul Menage wrote:
>> > On 11/23/06, Pavel Emelianov <xemul@openvz.org> wrote:
>> >> You mean moving is like this:
>> >>
>> >> old_bc = task->real_bc;
>> >> task->real_bc = new_bc;
>> >> cmpxchg(&tsk->exec_bc, old_bc, new_bc);
>> >>
>> >> ? Then this won't work:
>> >>
>> >> Initialisation:
>> >> current->exec_bc = init_bc;
>> >> current->real_bc = init_bc;
>> >> ...
>> >> IRQ:
>> >> current->exec_bc = init_bc;
>> >> ...
>> >>   old_bc = tsk->real_bc; /* init_bc */
>> >>   tsk->real_bc = bc1;
>> >>   cx(tsk->exec_bc, init_bc, bc1); /* ok */
>> >> ...
>> >> Here at the middle of an interrupt
>> >> we have bc1 set as exec_bc on task
>> >> which IS wrong!
>> >
>> > You could get round that by having a separate "irq_bc" that's never
>> > valid for a task not in an interrupt.
>>
>> No no no. This is not what is needed. You see, we do have to
>> set exec_bc as temporary (and atomic) context. Having temporary
>> context is 1. flexible 2. needed by beancounters' network accountig.
>
> I don't see why having an irq_bc wouldn't solve this. At the start of
> the interrupt handler, set current->exec_bc to &irq_bc; at the end set
> it to current->real_bc; use the cmpxchg() that I suggested to ensure
> that you never update task->exec_bc from another task if it's not
> equal to task->real_bc; use RCU to ensure that a beancounter is never
> freed while someone might be accessing it.

Oh, I see. I just didn't get your idea. This will work, but
1. we separate interrupt accounting from all the others'
2. for interrupts only. In case we want to set init_bc as

temporary context all will be broken...

We need some generic solution independent from what
exactly is set as temporary exec_bc.

>>
>> Maybe we can make smth similar to wait_task_inactive and change
>> it's beancounter before unlocking the runqueue?
>
> That could work too.

Could work, but whether everyone will like such intrusion...
I agree that stop_machine isn't nicer. This is a temporary
solution that works for sure. Better one will follow...