Subject: Re: [RFC PATCH] posix timers: allocate timer id per task
Posted by Stanislav Kinsbursky on Tue, 16 Oct 2012 08:08:14 GMT
View Forum Message <> Reply to Message

> On Mon, 15 Oct 2012, Stanislav Kinsbursky wrote:
>
>> This patch is required CRIU project (www.criu.org).
>> To migrate processes with posix timers we have to make sure, that we can
>> restore posix timer with proper id.
>> Currently, this is not true, because timer ids are allocated globally.
>> So, this is precursor patch and it's purpose is make posix timer id to be
>> allocated per task.
>
> You can't allocate them per task. posix timers are process wide.
>

This is probably a misunderstanding.
I meant process process.

> What's the reason why you did not make the posix timer ids per name
> space instead of going down to the per process level ?
>

The reason is that CRIU have to support single processes regardless to namespaces.

>> Patch replaces global idr with global hash table for posix timers and
>> makes timer ids unique not globally, but per task. Next free timer id is type
>> of integer and stored on signal struct (posix_timer_id). If free timer id
>> reaches negative value on timer creation, it will be dropped to zero and
>> -EAGAIN will be returned to user.
>
> So you want to allow 2^31 posix timers created for a single process?
>

I don't really want it.
I just tried to preserve existent logic. I.e. currently one process can create
2^31 posix timers (if no other processes tried to do the same).
But the amount can  be decreased, of course.

>> +static struct k_itimer *__posix_timers_find(struct hlist_head *head, struct signal_struct *sig,
timer_t id)
>> +{
>> + struct hlist_node *node;
>> + struct k_itimer *timer;
>> +
>> + hlist_for_each_entry(timer, node, head, t_hash) {
>> +  if ((timer->it_signal == sig) && (timer->it_id == id))

```
>> +   return timer;
>> + }
>> + return NULL;
>> +}
>> +
>> +static struct k_itimer *posix_timer_find(timer_t id, unsigned long *flags)
>> +{
>> + struct k_itimer *timer;
>> + struct signal_struct *sig = current->signal;
>> + struct hlist_head *head = &posix_timers_hashtable[hash(sig, id)];
>> +
>> + spin_lock_irqsave(&hash_lock, *flags);
>
> This is not going to fly. You just reintroduced a massive scalability
> problem. See commit 8af08871
>
```

Yep, Eric already pointed to it.
I'll try to fix this problem, if the idea with hash table suits in general.
Thanks.

```
> Thanks,
>
>  tglx
>
```

--
Best regards,
Stanislav Kinsbursky