Subject: Re: [RFC] kernel/pid.c pid allocation wierdness
Posted by xemul on Fri, 16 Mar 2007 10:57:39 GMT
View Forum Message <> Reply to Message

Oleg Nesterov wrote:
> On 03/14, Eric W. Biederman wrote:
>> Pavel Emelianov <xemul@sw.ru> writes:
>>
>>> Hi.
>>>
>>> I'm looking at how alloc_pid() works and can't understand
>>> one (simple/stupid) thing.
>>>
>>> It first kmem_cache_alloc()-s a strct pid, then calls
>>> alloc_pidmap() and at the end it taks a global pidmap_lock()
>>> to add new pid to hash.
>
> We need some global lock. pidmap_lock is already here, and it is
> only used to protect pidmap->page allocation. Iow, it is almost
> unused. So it was very natural to re-use it while implementing
> pidrefs.
>
>>> The question is - why does alloc_pidmap() use at least
>>> two atomic ops and potentially loop to find a zero bit
>>> in pidmap? Why not call alloc_pidmap() under pidmap_lock
>>> and find zero pid in pidmap w/o any loops and atomics?
>
> Currently we search for zero bit lockless, why do you want
> to do it under spin_lock ?

Search isn't lockless. Look:

```
while (1) {
  if (!test_and_set_bit(...)) {
     atomic_dec(&nr_free);
     return pid;
  }
  ...
}
```

we use two atomic operations to find and set a bit in a map.

> Oleg.
>
>

_____
Containers mailing list