Subject: Re: [PATCH 5/15] Introduce struct upid
Posted by Oleg Nesterov on Sun, 29 Jul 2007 09:50:35 GMT
View Forum Message <> Reply to Message

On 07/26, Pavel Emelyanov wrote:
>
> --- linux-2.6.23-rc1-mm1.orig/include/linux/pid.h 2007-07-26 16:34:45.000000000 +0400
> +++ linux-2.6.23-rc1-mm1-7/include/linux/pid.h 2007-07-26 16:36:37.000000000 +0400
> @@ -40,15 +40,21 @@ enum pid_type
> * processes.
> */
>
> -struct pid
> -{
> - atomic_t count;
> +struct upid {
> /* Try to keep pid_chain in the same cacheline as nr for find_pid */
> int nr;
> + struct pid_namespace *ns;
> struct hlist_node pid_chain;
> +};
> +
> +struct pid
> +{
> + atomic_t count;
> /* lists of tasks that use this pid */
> struct hlist_head tasks[PIDTYPE_MAX];
> struct rcu_head rcu;
> + int level;
> + struct upid numbers[1];
> };

Well. Definitely, the kernel can't be compiled with this patch applied,
this seems to be against the rules...

So. The task has a single (PIDTYPE_MAX) pid no matter how many namespaces
can see it, and "struct pid" has an array of numbers for each namespace.

Still I can't understand why do we need upid->ns, can't we kill it?
Suppose we add "struct pid_namespace *parent_ns" to "struct pid_namespace",
init_pid_ns.parent_ns == NULL.

Now,

```
struct upid {
  int nr;
  struct hlist_node pid_chain;
};
```

```
struct pid
{
 atomic_t count;
 struct hlist_head tasks[PIDTYPE_MAX];
 struct rcu_head rcu;
 struct pid_namespace *active_ns;
 struct upid numbers[0];
};
```

We populate pid->numbers in "reverse" order, so that pid->numbers[0] lives in pid->active_ns.

Now, for example,

```
void free_pid(struct pid *pid)
{
 struct pid_namespace *ns;
 unsigned long flags;
 int i;

 spin_lock_irqsave(&pidmap_lock, flags);
 for (i = 0, ns = pid->active_ns; ns; i++, ns = ns->parent_ns)
  hlist_del_rcu(&pid->numbers[i].pid_chain);
 spin_unlock_irqrestore(&pidmap_lock, flags);

 for (i = 0, ns = pid->active_ns; ns; i++, ns = ns->parent_ns)
  free_pidmap(ns, pid->numbers[i].nr);

 call_rcu(&pid->rcu, delayed_put_pid);
}
```

Possible?

Oleg.