Subject: [RFC][PATCH] Cleanup the new thread's creation
Posted by Pavel Emelianov on Fri, 24 Aug 2007 12:46:30 GMT
View Forum Message <> Reply to Message

The major differences of creating a new thread from creating a
new process is that

1. newbie's tgid is set to leader's
2. newbie's leader is set to leader
3. newbie is added to leader's thread_list

So move the initialization of these in one place. This helps
in pid/tgid fields isolation.

Signed-off-by: Pavel Emelyanov <xemul@openvz.org>

---

```
diff --git a/kernel/fork.c b/kernel/fork.c
index 7517efe..0b282a8 100644
--- a/kernel/fork.c
+++ b/kernel/fork.c
@@ -950,6 +950,20 @@ static inline void rt_mutex_init_task(st
 #endif
 }

+static void setup_new_thread(struct task_struct *thr, struct task_struct *leader)
+{
+ thr->tgid = leader->tgid;
+ thr->group_leader = leader;
+ list_add_tail_rcu(&thr->thread_group, &leader->thread_group);
+}
+
+static void setup_new_leader(struct task_struct *tsk)
+{
+ tsk->tgid = task_pid_nr(tsk);
+ tsk->group_leader = tsk;
+ INIT_LIST_HEAD(&tsk->thread_group);
+}
+
 /*
  * This creates a new process as a copy of the old one,
  * but does not actually start it yet.
@@ -1147,9 +1161,6 @@ static struct task_struct *copy_process(
  }

  p->pid = pid_nr(pid);
- p->tgid = p->pid;
```

```
- if (clone_flags & CLONE_THREAD)
-  p->tgid = current->tgid;

  p->set_child_tid = (clone_flags & CLONE_CHILD_SETTID) ? child_tidptr : NULL;
  /*
@@ -1191,8 +1202,6 @@ static struct task_struct *copy_process(
   * Ok, make it visible to the rest of the system.
   * We dont wake it up yet.
   */
- p->group_leader = p;
- INIT_LIST_HEAD(&p->thread_group);
  INIT_LIST_HEAD(&p->ptrace_children);
  INIT_LIST_HEAD(&p->ptrace_list);

@@ -1251,8 +1260,7 @@ static struct task_struct *copy_process(
  }

  if (clone_flags & CLONE_THREAD) {
-  p->group_leader = current->group_leader;
-  list_add_tail_rcu(&p->thread_group, &p->group_leader->thread_group);
+  setup_new_thread(p, current->group_leader);

   if (!cputime_eq(current->signal->it_virt_expires,
     cputime_zero) ||
@@ -1268,7 +1276,8 @@ static struct task_struct *copy_process(
    p->it_prof_expires = jiffies_to_cputime(1);
   }
- }
+ } else
+  setup_new_leader(p);

  if (likely(p->pid)) {
   add_parent(p);
```



---

Subject: Re: [RFC][PATCH] Cleanup the new thread's creation
Posted by Oleg Nesterov on Sat, 25 Aug 2007 16:50:31 GMT
View Forum Message <> Reply to Message

On 08/24, Pavel Emelyanov wrote:
>
> The major differences of creating a new thread from creating a
> new process is that

>
> 1. newbie's tgid is set to leader's
> 2. newbie's leader is set to leader
> 3. newbie is added to leader's thread_list

(Surely, the are many other major differences, but from the pids virtualization
 POV - yes ;)

> +static void setup_new_thread(struct task_struct *thr, struct task_struct
> *leader)
> +{
> + thr->tgid = leader->tgid;
> + thr->group_leader = leader;
> + list_add_tail_rcu(&thr->thread_group, &leader->thread_group);
> +}

Imho, this name is a bit "too generic". Not that I can suggest something
better... copy_sub_thread/copy_group_leader ?

> @@ -1147,9 +1161,6 @@ static struct task_struct *copy_process(
>  }
>
>  p->pid = pid_nr(pid);
> - p->tgid = p->pid;
> - if (clone_flags & CLONE_THREAD)
> -  p->tgid = current->tgid;

I agree, it is absoulutely not clear why should we set ->tgid here, and it
would be nice to consolidate "if (CLONE_THREAD)" checks, but do we really
need the helpers above? There are very simple, and have the only one caller.
Sometimes it is good to see what's going on without pressing C-]

Not that I against this patch, just I'm not sure it really simplifies things.
Perhaps I missed something else you have in mind.

Oleg.



Subject: Re: [RFC][PATCH] Cleanup the new thread's creation
Posted by Pavel Emelianov on Mon, 27 Aug 2007 06:43:24 GMT
View Forum Message <> Reply to Message

Oleg Nesterov wrote:

> On 08/24, Pavel Emelyanov wrote:
>> The major differences of creating a new thread from creating a
>> new process is that
>>
>> 1. newbie's tgid is set to leader's
>> 2. newbie's leader is set to leader
>> 3. newbie is added to leader's thread_list
>
> (Surely, the are many other major differences, but from the pids virtualization
>  POV - yes ;)
>
>> +static void setup_new_thread(struct task_struct *thr, struct task_struct
>> *leader)
>> +{
>> + thr->tgid = leader->tgid;
>> + thr->group_leader = leader;
>> + list_add_tail_rcu(&thr->thread_group, &leader->thread_group);
>> +}
>
> Imho, this name is a bit "too generic". Not that I can suggest something
> better... copy_sub_thread/copy_group_leader ?
>
>> @@ -1147,9 +1161,6 @@ static struct task_struct *copy_process(
>>  }
>>
>>  p->pid = pid_nr(pid);
>> - p->tgid = p->pid;
>> - if (clone_flags & CLONE_THREAD)
>> -  p->tgid = current->tgid;
>
> I agree, it is absoulutely not clear why should we set ->tgid here, and it
> would be nice to consolidate "if (CLONE_THREAD)" checks, but do we really
> need the helpers above? There are very simple, and have the only one caller.
> Sometimes it is good to see what's going on without pressing C-]
>
> Not that I against this patch, just I'm not sure it really simplifies things.
> Perhaps I missed something else you have in mind.

Me too, but while cleaning up the pid_t usage over the kernel I found
this place to be one of the most difficult from "how to make it better"
point of view. We need to hide the pid/tgid explicit usage somehow, but
the problem is that pid and tgid are set in this place and de_thread()
only and making helpers like set_task_tgid() doesn't sound reasonable.

> Oleg.
>
>

Thanks,
Pavel

_____

Containers mailing list
Containers@lists.linux-foundation.org
https://lists.linux-foundation.org/mailman/listinfo/containers