Subject: [PATCH 5/20] Introduce struct upid
Posted by Pavel Emelianov on Fri, 10 Aug 2007 11:47:59 GMT
View Forum Message <> Reply to Message

From: Sukadev Bhattiprolu <sukadev@us.ibm.com>

Since task will be visible from different pid namespaces each of them have to
be addressed by multiple pids. struct upid is to store the information about
which id refers to which namespace.

The constuciton looks like this. Each struct pid carried the reference counter
and the list of tasks attached to this pid. At its end it has a variable length
array of struct upid-s. Each struct upid has a numerical id (pid itself),
pointer to the namespace, this ID is valid in and is hashed into a pid_hash
for searching the pids.

The nr and pid_chain fields are kept in struct pid for a while to make kernel
still work (no patch initialize the upids yet), but it will be removed at the
end of this series when we switch to upids completely.

Signed-off-by: Sukadev Bhattiprolu <sukadev@us.ibm.com>
Signed-off-by: Pavel Emelyanov <xemul@openvz.org>
Cc: Oleg Nesterov <oleg@tv-sign.ru>

---

 init_task.h | 6 ++++++
 pid.h | 16 ++++++++++++++++
 2 files changed, 22 insertions(+)

diff -upr linux-2.6.23-rc1-mm1.orig/include/linux/init_task.h
linux-2.6.23-rc1-mm1-7/include/linux/init_task.h
--- linux-2.6.23-rc1-mm1.orig/include/linux/init_task.h 2007-07-26 16:34:45.000000000 +0400
+++ linux-2.6.23-rc1-mm1-7/include/linux/init_task.h 2007-07-26 16:36:36.000000000 +0400
@@ -93,6 +93,12 @@ extern struct group_info init_groups;
   { .first = &init_task.pids[PIDTYPE_SID].node }, \
  },   \
  .rcu = RCU_HEAD_INIT,   \
+ .level = 0,   \
+ .numbers = { {   \
+ .nr = 0,   \
+ .ns = &init_pid_ns,   \
+ .pid_chain = { .next = NULL, .pprev = NULL }, \
+ }, }   \
 }

 #define INIT_PID_LINK(type)   \
diff -upr linux-2.6.23-rc1-mm1.orig/include/linux/pid.h linux-2.6.23-rc1-mm1-7/include/linux/pid.h

```
--- 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
@@ -40,6 +40,20 @@ enum pid_type
  * processes.
  */

+
+/*
+ * struct upid is used to get the id of the struct pid, as it is
+ * seen in particular namespace. Later the struct pid is found with
+ * find_pid_ns() using the int nr and struct pid_namespace *ns.
+ */
+
+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;
@@ -50,6 +50,8 @@ struct pid
  /* lists of tasks that use this pid */
  struct hlist_head tasks[PIDTYPE_MAX];
  struct rcu_head rcu;
+ int level;
+ struct upid numbers[1];
 };

 extern struct pid init_struct_pid;
```

Subject: Re: [PATCH 5/20] Introduce struct upid
Posted by Andrew Morton on Tue, 14 Aug 2007 20:05:58 GMT
View Forum Message <> Reply to Message

On Fri, 10 Aug 2007 15:47:59 +0400
xemul@openvz.org wrote:

```
>  struct pid
>  {
>   atomic_t count;
> @@ -50,6 +50,8 @@ struct pid
>   /* lists of tasks that use this pid */
>   struct hlist_head tasks[PIDTYPE_MAX];
>   struct rcu_head rcu;
```

> + int level;
> + struct upid numbers[1];

You can make this have size [0] now. It's a gcc extension and
is used elsewhere in the kernel.

Subject: Re: [PATCH 5/20] Introduce struct upid
Posted by Sukadev Bhattiprolu on Sat, 08 Sep 2007 02:16:56 GMT
View Forum Message <> Reply to Message

Andrew Morton [akpm@linux-foundation.org] wrote:
| On Fri, 10 Aug 2007 15:47:59 +0400
| xemul@openvz.org wrote:
|
| > struct pid
| > {
| >   atomic_t count;
| > @@ -50,6 +50,8 @@ struct pid
| >   /* lists of tasks that use this pid */
| >   struct hlist_head tasks[PIDTYPE_MAX];
| >   struct rcu_head rcu;
| > + int level;
| > + struct upid numbers[1];
|
| You can make this have size [0] now. It's a gcc extension and
| is used elsewhere in the kernel.

Sorry, we did not respond to this yet :-)

Well, every process has at least one 'struct upid'. The only "cost"
I see with size [1] is having to subtract 1 in create_pid_cachep().

Besides, we create/initialize the 'struct pid' for the idle process
by hand (see INIT_STRUCT_PID in init_task.h).

If we set this size to [0] now, we would need to dynamically allocate
a 'struct upid' during early boot and attach this upid to init_struct_pid.

Or is there a easy way to attach a 'upid' to init_struct_pid, statically ?

Suka
