Subject: Re: [PATCH 22/28] [MULTI 1/6] Changes in data structures for multilevel model
Posted by Sukadev Bhattiprolu on Tue, 19 Jun 2007 06:32:50 GMT
View Forum Message <> Reply to Message

Pavel Emelianov [xemul@openvz.org] wrote:
| This patch opens the multilevel model patches.
|
| The multilevel model idea is basically the same as for the flat one,
| but in this case task may have many virtual pids - one id for each
| sub-namespace this task is visible in. The struct pid carries the
| list of pid_number-s and two hash tables are used to find this number
| by numerical id and by struct pid.
|
|
|
| The struct pid doesn't need the numerical ids any longer. Instead it
| has a single linked list of struct pid_number-s which are hashed
| for quick search and have the numerical id.
|
| Signed-off-by: Pavel Emelianov <xemul@openvz.org>
|
| ---
|
| pid.h | 31 ++++++++++++++++++++++++++++++
| 1 files changed, 31 insertions(+)
|
| --- ./include/linux/pid.h.multdatast 2007-06-15 15:23:00.000000000 +0400
| +++ ./include/linux/pid.h 2007-06-15 15:32:15.000000000 +0400
| @@ -50,9 +50,33 @@ enum pid_type
| * id must be used.
| */
|
| +/*
| + * multilevel pid namespaces
| + * each task may belong to any number of namespaces and thus struct pid do
| + * not carry the number any longer. instead if this struct pid has a list of
| + * pid_number-s each belonging to one namespace. two hashes are used to find
| + * the number - by the numerical id and by the struct pid this nr belongs to.
| + * this allows for creating namespaces of infinite nesting, but has slight
| + * performance problems.
| + */
| +
| +struct pid_number
| +{
| + int nr;
| + struct pid_namespace *ns;
| + struct pid *pid;

```
| +
| + struct hlist_node pid_chain;
| + struct hlist_node nr_chain;
| + struct pid_number *next;
```

As you probably noticed, we had a similar linked list until recently.
But since we use only clone() to create a new pid namespace, we figured
we could use an array of 'struct pid_number' elements. That may perform
slightly better since all 'pid_number elements' are co-located.

We obviously need a list like this if we unshare (rather than clone())
pid namespace.

I have a few questions - not that I see any problems yet - just for my
understanding (they may be addressed in other patches, but am still
reviewing them).

- Can one process unshare() its namespace, create a few children,
  and unshare its namespace again ?

- If so, will that same process be the reaper for multiple pid
  namespaces ?

- Will we terminate all those namespaces if the reaper is terminated ?

```
| +};
| +
|  struct pid
|  {
|   atomic_t count;
| +#ifdef CONFIG_PID_NS_MULTILEVEL
| + struct pid_number *pid_nrs;
| +#else
|   /* Try to keep pid_chain in the same cacheline as nr for find_pid */
|   int nr;
|   struct hlist_node pid_chain;
| @@ -65,11 +89,18 @@ struct pid
|   struct pid_namespace *ns;
|   struct hlist_node vpid_chain;
|  #endif
| +#endif
|   /* lists of tasks that use this pid */
|   struct hlist_head tasks[PIDTYPE_MAX];
|   struct rcu_head rcu;
|  };
|
| +#ifdef CONFIG_PID_NS_MULTILEVEL
```

```
| +/* small helper to iterate over the pid's numbers */
| +#define for_each_pid_nr(nr, pid) \
| +  for (nr = pid->pid_nrs; nr != NULL; nr = nr->next)
| +#endif
| +
|  extern struct pid init_struct_pid;
|
|  struct pid_link
```

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