Subject: Re: [PATCH 08/17] Pid-NS(V3) Define/use pid->upid_list list.
Posted by Sukadev Bhattiprolu on Tue, 19 Jun 2007 06:48:51 GMT
View Forum Message <> Reply to Message

Pavel Emelianov [xemul@openvz.org] wrote:
| sukadev@us.ibm.com wrote:
| > Pavel Emelianov [xemul@openvz.org] wrote:
| > | sukadev@us.ibm.com wrote:
| > | > Subject: [PATCH 08/17] Pid-NS(V3) Define/use pid->upid_list list.
| > | >
| > | > From: Sukadev Bhattiprolu <sukadev@us.ibm.com>
| > | >
| > | >
| > | > With multiple pid namespaces, a process would be known by several pid_t
| > | > values, one in each pid namespace. To represent this, we introduce a
| > | > 'struct upid' which associates a single pid_t value with a single pid
| > | > namespace.
| > | >
| > | > We then replace the pid->nr field in 'struct pid' with a list of struct upid'
| > | > entries (referred to as 'pid->upid_list'). This list represents the multiple
| > | > pid_t values of the process, one in each namespace. The current patch adds
| > | > just one element to this list, corresponding to 'init_pid_ns'. Subsequent
| > | > patches implement multiple pid namespaces and add more elements to the list.
| > | >
| > | > The 'struct upid' also replaces 'struct pid' in the pid_hash table to enable us
| > | > to find processes given a pid_t from any namespace (i.e we find 'struct upid'
| > | > for a given pid_t and from the 'struct upid', we find the 'struct pid' of the
| > | > process)
| > | >
| > | > We finally reimplement find_pid() and pid_to_nr() to use pid->upid_list
| > | > and remove unused fields from 'struct pid'.
| > | >
| > | > Changelog:
| > | > 2.6.21-mm2-pidns3:
| > | >
| > | > - 'struct upid' used to be called 'struct pid_nr' and a list of these
| > | > were hanging off of 'struct pid'. So, we renamed 'struct pid_nr'
| > | > and now hold them in a statically sized array in 'struct pid' since
| > | > the number of 'struct upid's for a process is known at process-
| > | > creation time.
| > | >
| > | > 2.6.21-rc3-mm2:
| > | >
| > | > - [Eric Biederman] Combine all logical changes into one patch
| > | > - [Eric Biederman] Implement __pid_nr(pid_ns, pid) for use in procfs.
| > | > (now called pid_to_nr_in_ns()).
| > | > - [Serge Hallyn]: Remove (!pid_nr) check in free_pid_nr()
| > | >

| > | > Signed-off-by: Cedric Le Goater <clg@fr.ibm.com>
| > | > Signed-off-by: Sukadev Bhattiprolu <sukadev@us.ibm.com>
| > | > ---
| > | > fs/proc/array.c | 30 ++++++--
| > | > fs/proc/base.c | 9 ++
| > | > include/linux/init_task.h | 14 +++-
| > | > include/linux/pid.h | 62 +++++++++++++++---
| > | > include/linux/pid_namespace.h | 15 ++++
| > | > kernel/fork.c | 2
| > | > kernel/pid.c | 145 +++++++++++++++++++++++++++++++++++++---------
| > | > 7 files changed, 220 insertions(+), 57 deletions(-)
| > | >
| > | > Index: lx26-22-rc4-mm2/include/linux/pid.h
| > | > ===
| > | > --- lx26-22-rc4-mm2.orig/include/linux/pid.h 2007-06-15 18:44:50.000000000 -0700
| > | > +++ lx26-22-rc4-mm2/include/linux/pid.h 2007-06-15 19:47:58.000000000 -0700
| > | > @@ -16,6 +16,25 @@ enum pid_type
| > | >   PIDTYPE_MAX
| > | > };
| > | >
| > | > +struct pid_namespace;
| > | > +
| > | > +/*
| > | > + * A struct upid holds a process identifier (or pid->nr) for a given
| > | > + * pid namespace.
| > | > + *
| > | > + * A list of 'struct upid' entries is stored in the struct pid. This list
| > | > + * is used to get the process identifier associated with the pid
| > | > + * namespace it is being seen from.
| > | > + */
| > | > +struct upid
| > | > +{
| > | > + /* Try to keep pid_chain in the same cacheline as nr for find_pid */
| > | > + struct hlist_node pid_chain; /* link hash collisions on pid_hash */
| > | > + int nr;   /* user space pid number */
| > | > + struct pid_namespace *pid_ns; /* pid namespace in which nr is valid */
| > | > + struct pid *pid;   /* back to task's unique kernel pid */
| > | > +};
| > | > +
| > | > /*
| > | >   * What is struct pid?
| > | >   *
| > | > @@ -48,12 +67,11 @@ enum pid_type
| > | > struct pid
| > | > {
| > | >   atomic_t count;
| > | > - /* Try to keep pid_chain in the same cacheline as nr for find_pid */
| > | > - int nr;

| > | > - struct hlist_node pid_chain;
| > | >   /* lists of tasks that use this pid */
| > | >   struct hlist_head tasks[PIDTYPE_MAX];
| > | >   struct rcu_head rcu;
| > | > + int num_upids;
| > | > + struct upid upid_list[1];
| > |
| > | Further in your patches you define MAX_NESTED_PID_NS. What for, you
| > | use the linked list here!?
| >
| > Hmm. I don't understand. upid_list[] is an array (and not a linked
| > list). Are you saying the '_list' in 'upid_list' is misleading ?
|
| Oh, I see! You allocate all the upids in one chunk. I have missed
| that, sorry :)
|
| > Placing a limit like MAX_NESTED_PID_NS simplifies allocation of
| > 'struct pid'.
|
| How? If we have, say, 100-level namespace than we have to create
| the sizeof(struct pid) + 100 * sizeof(struct upid) bytes.

I should have been a little more clear.

I was comparing this with my previous version which did not have the
MAX_NESTED_PID_NS limit and allowed for arbitrary levels of nesting
(100 or even 1000 :-). Allocating that kind of 'struct pid' is more
complex and looks like an overkill at this time.

With a limit like MAX_NESTED_PID_NS, we could in theory create that
many pid caches, one for each level of nesting and use the appropriate
cache in clone().

|
| >
| > |
| > | > };
| > | >
| > | > extern struct pid init_struct_pid;
| > |
| > | [snip]
| > _____
| > Containers mailing list
| > Containers@lists.linux-foundation.org
| > https://lists.linux-foundation.org/mailman/listinfo/containers
| >
| > _____
| > Devel mailing list

| > Devel@openvz.org
| > https://openvz.org/mailman/listinfo/devel
| >

_____

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