Subject: [RFC][PATCH 10/14] Use pid ns from pid_nrs list
Posted by Sukadev Bhattiprolu on Wed, 21 Mar 2007 03:21:27 GMT
View Forum Message <> Reply to Message

From: Sukadev Bhattiprolu <sukadev@us.ibm.com>
Subject: [RFC][PATCH 10/14] Use pid ns from pid_nrs list

We need to decouple pid namespace from nsproxy to allow getting and releasing pid namespace independently from other namespaces. This is required since a process's reference to its pid namespace must exist even after its references to other namespaces are dropped during process exit.

As stated in earlier patches, to support containers, a process can have different pid_t values in different pid namespaces and struct pid_nr and the pid->pid_nrs list provide this association of pid_t value with pid namespace for a process.

To find the pid namespace of a process, we can now use the pid namespace stored in the struct pid_nr rather than the one stored in nsproxy.

This patch defines a function, pid_ns(), which returns the "current" or "primary" pid namespace of the given struct pid and reimplements the task_pid_ns() and child_reaper() wrapper functions using the pid_ns()

Changelog:

- Rewrite from an earlier verison which supported unshare() of pid namespace.

Signed-off-by: Sukadev Bhattiprolu <sukadev@us.ibm.com>
---
include/linux/pid_namespace.h | 5 +++--
kernel/pid.c | 37 ++
2 files changed, 40 insertions(+), 2 deletions(-)

Index: lx26-21-rc3-mm2/kernel/pid.c
==========
--- lx26-21-rc3-mm2.orig/kernel/pid.c 2007-03-20 15:55:52.000000000 -0700
+++ lx26-21-rc3-mm2/kernel/pid.c 2007-03-20 16:57:14.000000000 -0700
@@ -270,6 +270,43 @@ pid_t pid_nr(struct pid *pid)
}
EXPORT_SYMBOL_GPL(pid_nr);

+/*
+ * Return the current pid-namespace of this pid.
+ *
+ * Processes initially exist only in the init-pid-ns. For such processes,

```
+ * their "current pid namespace" is just init-pid-ns.
+ *
+ * When a process in init-pid-ns clones using the CLONE_NEWPID flag, it
+ * creates a new child-pid-ns and a child process. This child process then
+ * exists in two namespaces - i.e it has a pid_t value in init pid namespace
+ * and another, usually different pid_t value in the newly-created child-pid-ns.
+ *
+ * For this child process, we treat child-pid-ns as its "current pid namespace".
+ *
+ * The "current pid namespace" is used:
+ *  a. to determine which process will reap an orphaned process.
+ *  b. by find_pid() to select the appropriate pid -  note that since
+ *     there can be multiple processes with a given pid_t value (one
+ *     in each namespace) find_pid() returns the pid corresponding
+ *     to the current pid namespace.
+ *
+ * Processes inherit the current and all ancestor pid namespaces from their
+ * parent.
+ */
+struct pid_namespace *pid_ns(struct pid *pid)
+{
+ struct pid_nr* pid_nr;
+ struct hlist_node *head;
+
+ if (!pid)
+  return NULL;
+
+ head = pid->pid_nrs.first;
+ pid_nr = hlist_entry(head, struct pid_nr, node);
+
+ return pid_nr->pid_ns;
+}
+
 struct pid *alloc_pid(void)
 {
  struct pid *pid;
Index: lx26-21-rc3-mm2/include/linux/pid_namespace.h
===================================================================
--- lx26-21-rc3-mm2.orig/include/linux/pid_namespace.h 2007-03-20 15:55:52.000000000 -0700
+++ lx26-21-rc3-mm2/include/linux/pid_namespace.h 2007-03-20 15:55:52.000000000 -0700
@@ -31,6 +31,7 @@ static inline void get_pid_ns(struct pid

 extern struct pid_namespace *copy_pid_ns(int flags, struct pid_namespace *ns);
 extern void free_pid_ns(struct kref *kref);
+extern struct pid_namespace *pid_ns(struct pid * pid);

 static inline void put_pid_ns(struct pid_namespace *ns)
 {
```

```
@@ -39,12 +40,12 @@ static inline void put_pid_ns(struct pid

 static inline struct pid_namespace *task_pid_ns(struct task_struct *tsk)
 {
-       return tsk->nsproxy->pid_ns;
+ return pid_ns(task_pid(tsk));
 }

 static inline struct task_struct *child_reaper(struct task_struct *tsk)
 {
- return init_pid_ns.child_reaper;
+ return task_pid_ns(tsk)->child_reaper;
 }

 #endif /* _LINUX_PID_NS_H */
```

Subject: Re: [RFC][PATCH 10/14] Use pid ns from pid_nrs list
Posted by ebiederm on Wed, 21 Mar 2007 08:04:33 GMT
View Forum Message <> Reply to Message

sukadev@us.ibm.com writes:

> From: Sukadev Bhattiprolu <sukadev@us.ibm.com>
> Subject: [RFC][PATCH 10/14] Use pid ns from pid_nrs list
>
> We need to decouple pid namespace from nsproxy to allow getting and
> releasing pid namespace independently from other namespaces. This is
> required since a process's reference to its pid namespace must exist
> even after its references to other namespaces are dropped during
> process exit.
>
> As stated in earlier patches, to support containers, a process can
> have different pid_t values in different pid namespaces and struct
> pid_nr and the pid->pid_nrs list provide this association of pid_t
> value with pid namespace for a process.
>
> To find the pid namespace of a process, we can now use the pid namespace
> stored in the struct pid_nr rather than the one stored in nsproxy.
>
> This patch defines a function, pid_ns(), which returns the "current" or
> "primary" pid namespace of the given struct pid and reimplements the
> task_pid_ns() and child_reaper() wrapper functions using the pid_ns()
>

> Changelog:
>
> - Rewrite from an earlier verison which supported unshare()
>   of pid namespace.
>
> Signed-off-by: Sukadev Bhattiprolu <sukadev@us.ibm.com>
> ---
> include/linux/pid_namespace.h | 5 +++--
> kernel/pid.c | 37 ++
> 2 files changed, 40 insertions(+), 2 deletions(-)
>
> Index: lx26-21-rc3-mm2/kernel/pid.c
> ==
> --- lx26-21-rc3-mm2.orig/kernel/pid.c 2007-03-20 15:55:52.000000000 -0700
> +++ lx26-21-rc3-mm2/kernel/pid.c 2007-03-20 16:57:14.000000000 -0700
> @@ -270,6 +270,43 @@ pid_t pid_nr(struct pid *pid)
> }
> EXPORT_SYMBOL_GPL(pid_nr);
>
> +/*
> + * Return the current pid-namespace of this pid.
> + *
> + * Processes initially exist only in the init-pid-ns. For such processes,
> + * their "current pid namespace" is just init-pid-ns.
> + *
> + * When a process in init-pid-ns clones using the CLONE_NEWPID flag, it
> + * creates a new child-pid-ns and a child process. This child process then
> + * exists in two namespaces - i.e it has a pid_t value in init pid namespace
> + * and another, usually different pid_t value in the newly-created
> child-pid-ns.
> + *
> + * For this child process, we treat child-pid-ns as its "current pid
> namespace".
> + *
> + * The "current pid namespace" is used:
> + * a. to determine which process will reap an orphaned process.
> + * b. by find_pid() to select the appropriate pid - note that since
> + *   there can be multiple processes with a given pid_t value (one
> + *   in each namespace) find_pid() returns the pid corresponding
> + *   to the current pid namespace.
> + *
> + * Processes inherit the current and all ancestor pid namespaces from their
> + * parent.
> + */

Can we make this:
> +struct pid_namespace *pid_ns(struct pid *pid)
> +{

```
> + struct pid_nr* pid_nr;
> + struct hlist_node *head;
> +
> + if (!pid)
> +  return NULL;
> +
> + head = pid->pid_nrs.first;
> + pid_nr = hlist_entry(head, struct pid_nr, node);
> +
> + return pid_nr->pid_ns;
> +}
> +
```

```
struct pid_namespace *primary_pid_ns(struct pid *pid)
{
 struct pid_namespace *ns = &init_pid_ns;
 struct pid_nr* pid_nr;
 struct hlist_node *head;

 if (unlikely(!pid))
  goto out;

 head = pid->pid_nrs.first;
        if (unlikely(!head))
         goto out;

 pid_nr = hlist_entry(head, struct pid_nr, node);
 ns = pid_nr->pid_ns;
out:
 return ns;
}
```

I think changing the name to primary_pid_ns is a little clearer.
(It implies that there are others).

Further I don't know if the return NULL when we don't have a pid buys
us anything.

But returning &init_pid_ns instead allow us to drop the static
initializer for pid_nr for the idle processes allowing those unhashed
processes to have no hash table state at all.

Eric