
Subject: [RFC][PATCH 09/16] Use pid ns from pid->upid_list
Posted by [Sukadev Bhattiprolu](#) on Thu, 24 May 2007 01:12:36 GMT
[View Forum Message](#) <> [Reply to Message](#)

Subject: Use pid ns from pid->upid_list

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

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.

With multiple pid namespaces, a process can have different pid_t values in different pid namespaces and 'struct upid' and the pid->upid_list list provide this association of pid_t value with pid namespace for a process.

Use pid->upid_list list to find the active pid namespace of a process and remove nsproxy->pid_namespace.

(Review note: My description of active pid namespace is probably long-winded.
Appreciate any comments in explaining it better :-)

TODO:

- Include pid_namespace in pid_hash() so processes with same pid_t in different namespaces are on different hash lists.

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]

- Drop support for unshare() of pid namespace which simplifies cloning of pid namespace and reorganize several functions.

Signed-off-by: Sukadev Bhattiprolu <sukadev@us.ibm.com>

```
include/linux/init_task.h | 1 -
include/linux/nsproxy.h   | 2 --
include/linux/pid_namespace.h | 4 ++--
kernel/nsproxy.c          | 10 -----
kernel/pid.c              | 23 ++++++++-----
```

5 files changed, 18 insertions(+), 22 deletions(-)

Index: lx26-21-mm2/include/linux/init_task.h

```
=====
--- lx26-21-mm2.orig/include/linux/init_task.h 2007-05-22 16:59:49.000000000 -0700
+++ lx26-21-mm2/include/linux/init_task.h 2007-05-22 16:59:50.000000000 -0700
@@ -72,7 +72,6 @@
```

```
extern struct nsproxy init_nsproxy;
#define INIT_NS_PROXY(nsproxy) { \
- .pid_ns = &init_pid_ns, \
  .count = ATOMIC_INIT(1), \
  .nslock = __SPIN_LOCK_UNLOCKED(nsproxy.nslock), \
  .uts_ns = &init_uts_ns, \
```

Index: lx26-21-mm2/include/linux/nsproxy.h

```
=====
--- lx26-21-mm2.orig/include/linux/nsproxy.h 2007-05-22 16:58:37.000000000 -0700
+++ lx26-21-mm2/include/linux/nsproxy.h 2007-05-22 16:59:50.000000000 -0700
@@ -7,7 +7,6 @@
```

```
struct mnt_namespace;
struct uts_namespace;
struct ipc_namespace;
-struct pid_namespace;
```

```
/*
 * A structure to contain pointers to all per-process
```

```
@@ -27,7 +26,6 @@ struct nsproxy {
  struct uts_namespace *uts_ns;
  struct ipc_namespace *ipc_ns;
  struct mnt_namespace *mnt_ns;
- struct pid_namespace *pid_ns;
};
extern struct nsproxy init_nsproxy;
```

Index: lx26-21-mm2/include/linux/pid_namespace.h

```
=====
--- lx26-21-mm2.orig/include/linux/pid_namespace.h 2007-05-22 16:59:49.000000000 -0700
+++ lx26-21-mm2/include/linux/pid_namespace.h 2007-05-22 16:59:50.000000000 -0700
@@ -41,8 +41,8 @@ static inline void get_pid_ns(struct pid
  kref_get(&ns->kref);
}
```

```
-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_active_pid_ns(struct pid *pid);
```

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

@@ -51,7 +51,7 @@ static inline void put_pid_ns(struct pid

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

static inline struct task_struct *task_child_reaper(struct task_struct *tsk)

Index: lx26-21-mm2/kernel/nsproxy.c

=====

--- lx26-21-mm2.orig/kernel/nsproxy.c 2007-05-22 16:58:37.000000000 -0700

+++ lx26-21-mm2/kernel/nsproxy.c 2007-05-22 16:59:50.000000000 -0700

@@ -19,7 +19,6 @@

#include <linux/init_task.h>

#include <linux/mnt_namespace.h>

#include <linux/utsname.h>

-#include <linux/pid_namespace.h>

struct nsproxy init_nsproxy = INIT_NS_PROXY(init_nsproxy);

@@ -75,15 +74,8 @@ static struct nsproxy *create_new_namesp

if (IS_ERR(new_nsp->ipc_ns))

goto out_ipc;

- new_nsp->pid_ns = copy_pid_ns(flags, tsk->nsproxy->pid_ns);

- if (IS_ERR(new_nsp->pid_ns))

- goto out_pid;

-

return new_nsp;

-out_pid:

- if (new_nsp->ipc_ns)

- put_ipc_ns(new_nsp->ipc_ns);

out_ipc:

if (new_nsp->uts_ns)

put_uts_ns(new_nsp->uts_ns);

@@ -138,8 +130,6 @@ void free_nsproxy(struct nsproxy *ns)

put_uts_ns(ns->uts_ns);

if (ns->ipc_ns)

put_ipc_ns(ns->ipc_ns);

- if (ns->pid_ns)

- put_pid_ns(ns->pid_ns);

kfree(ns);

}

Index: lx26-21-mm2/kernel/pid.c

=====

```

--- lx26-21-mm2.orig/kernel/pid.c 2007-05-22 16:59:49.000000000 -0700
+++ lx26-21-mm2/kernel/pid.c 2007-05-22 16:59:50.000000000 -0700
@@ -242,6 +242,22 @@ static int init_upid(struct upid *upid,
    return 0;
}

+static struct upid *pid_active_upid(struct pid *pid)
+{
+ return &pid->upid_list[0];
+}
+
+/*
+ * Return the active pid namespace of the process @pid.
+ *
+ * Note: At present, there is only one pid namespace (init_pid_ns).
+ */
+struct pid_namespace *pid_active_pid_ns(struct pid *pid)
+{
+ return pid_active_upid(pid)->pid_ns;
+}
+EXPORT_SYMBOL_GPL(pid_active_pid_ns);
+
+/*
+ * Return the pid_t by which the process @pid is known in the pid
+ * namespace @ns.
+ */
@@ -515,13 +531,6 @@ struct pid *find_ge_pid(int nr)
}
EXPORT_SYMBOL_GPL(find_get_pid);

-struct pid_namespace *copy_pid_ns(int flags, struct pid_namespace *old_ns)
-{
- BUG_ON(!old_ns);
- get_pid_ns(old_ns);
- return old_ns;
-}
-
void free_pid_ns(struct kref *kref)
{
    struct pid_namespace *ns;

```

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