
Subject: [RFC][PATCH 5/6] Define helper functions to unshare pid namespace
Posted by [Sukadev Bhattiprolu](#) on Sat, 10 Mar 2007 03:59:54 GMT
[View Forum Message](#) <> [Reply to Message](#)

From: Sukadev Bhattiprolu <sukadev@us.ibm.com>
Subject: [RFC][PATCH 5/6] Define helper functions to unshare pid namespace

Define clone_pid_ns() and unshare_pid_ns() functions that will be used in the next patch to unshare pid namespace.

Changelog:

- Rewrite of original code in -lxc from Cedric Le Goater to enforce setsid() requirement on unshare().
- [Cedric Le Goater comment] Fix minor compile errors/warnings

Signed-off-by: Cedric Le Goater <clg@fr.ibm.com>

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

Cc: Dave Hansen <haveblue@us.ibm.com>

Cc: Serge Hallyn <serue@us.ibm.com>

Cc: containers@lists.osdl.org

```
include/linux/pid_namespace.h | 2 +
kernel/pid.c                  | 55 +++++
2 files changed, 57 insertions(+)
```

Index: lx26-20-mm2b/include/linux/pid_namespace.h

```
=====
--- lx26-20-mm2b.orig/include/linux/pid_namespace.h 2007-03-09 17:00:14.000000000 -0800
+++ lx26-20-mm2b/include/linux/pid_namespace.h 2007-03-09 17:14:25.000000000 -0800
@@ -29,6 +29,8 @@ static inline void get_pid_ns(struct pid
    kref_get(&ns->kref);
}
```

```
+extern int unshare_pid_ns(unsigned long unshare_ns_flags,
+ struct pid_nr **new_pid_nr);
extern int copy_pid_ns(int flags, struct task_struct *tsk);
extern void free_pid_ns(struct kref *kref);
```

Index: lx26-20-mm2b/kernel/pid.c

```
=====
--- lx26-20-mm2b.orig/kernel/pid.c 2007-03-09 17:03:53.000000000 -0800
+++ lx26-20-mm2b/kernel/pid.c 2007-03-09 17:14:57.000000000 -0800
@@ -298,6 +298,35 @@ pid_t pid_nr(struct pid *pid)
    return 0;
}
```

```
+static struct pid_namespace *clone_pid_ns(void)
+{
```

```

+ struct pid_namespace *ns;
+ int i;
+
+ ns = kmalloc(sizeof(struct pid_namespace), GFP_KERNEL);
+ if (!ns)
+ return ns;
+
+ kref_init(&ns->kref);
+
+ atomic_set(&ns->pidmap[0].nr_free, BITS_PER_PAGE - 1);
+ ns->pidmap[0].page = kzalloc(PAGE_SIZE, GFP_KERNEL);
+ if (!ns->pidmap[0].page) {
+ kfree(ns);
+ return NULL;
+ }
+
+ set_bit(0, ns->pidmap[0].page);
+
+ for (i = 1; i < PIDMAP_ENTRIES; i++) {
+ atomic_set(&ns->pidmap[i].nr_free, BITS_PER_PAGE);
+ ns->pidmap[i].page = NULL;
+ }
+ ns->last_pid = 0;
+ ns->child_reaper = current;
+ return ns;
+}
+
+ struct pid *alloc_pid(void)
+ {
+ struct pid *pid;
@@ -471,6 +500,32 @@ struct pid *find_ge_pid(int nr)
+ }
+ EXPORT_SYMBOL_GPL(find_get_pid);
+
+int unshare_pid_ns(unsigned long unshare_ns_flags, struct pid_nr **new_pid_nr)
+{
+ struct pid_namespace *pid_ns;
+
+ if (!(unshare_ns_flags & CLONE_NEWPID))
+ return 0;
+
+ if (!capable(CAP_SYS_ADMIN))
+ return -EPERM;
+
+ if (task_pid(current) != task_session(current))
+ return -EPERM;
+
+ pid_ns = clone_pid_ns();

```

```
+ if (!pid_ns)
+ return -ENOMEM;
+
+ *new_pid_nr = alloc_pidmap_pid_nr(pid_ns);
+ if (!*new_pid_nr) {
+ put_pid_ns(pid_ns);
+ return -ENOMEM;
+ }
+
+ return 0;
+}
+
int copy_pid_ns(int flags, struct task_struct *tsk)
{
    struct pid_namespace *old_ns = task_pid_ns(tsk);
```

Containers mailing list

Containers@lists.osdl.org

<https://lists.osdl.org/mailman/listinfo/containers>
