Subject: Re: [PATCH 0/16] Pid namespaces
Posted by Pavel Emelianov on Tue, 10 Jul 2007 13:08:42 GMT
View Forum Message <> Reply to Message

sukadev@us.ibm.com wrote:
> Pavel Emelianov [xemul@openvz.org] wrote:
> | This is "submition for inclusion" of hierarchical, not kconfig
> | configurable, zero overheaded ;) pid namespaces.
> |
> | The overall idea is the following:
> |
> | The namespace are organized as a tree - once a task is cloned
> | with CLONE_NEWPIDS (yes, I've also switched to it :) the new
>
> Can you really clone() a pid namespace all by itself ?
> copy_namespaces() has the following:
>
>
> if (!(flags & (CLONE_NEWNS | CLONE_NEWUTS | CLONE_NEWIPC |
CLONE_NEWUSER)))
> return 0;
>
> doesn't it mean you cannot create a pid namespace using clone() unless
> one of the above flags are also specified ?
>
> unshare_nsproxy_namespaces() has the following correct check:
>
> if (!(unshare_flags & (CLONE_NEWNS | CLONE_NEWUTS | CLONE_NEWIPC |
> CLONE_NEWUSER | CLONE_NEWPIDS)))
> return 0;

I have lost a couple of hunks when I splitted the patch :(
That's the correct version, cap_set fix and the renamed CLONE_ flag.

```
--- ./include/linux/sched.h.fix 2007-07-06 11:09:33.000000000 +0400
+++ ./include/linux/sched.h 2007-07-10 13:48:19.000000000 +0400
@@ -26,7 +26,7 @@
 #define CLONE_NEWUTS  0x04000000 /* New utsname group? */
 #define CLONE_NEWIPC  0x08000000 /* New ipcs */
 #define CLONE_NEWUSER  0x10000000 /* New user namespace */
-#define CLONE_NEWPIDS  0x20000000 /* New pids */
+#define CLONE_NEWPID  0x20000000 /* New pids */

 /*
  * Scheduling policies
--- ./kernel/capability.c.fix 2007-07-06 11:09:33.000000000 +0400
+++ ./kernel/capability.c 2007-07-10 13:50:16.000000000 +0400
@@ -103,7 +103,7 @@ static inline int cap_set_pg(int pgrp_nr
```

```
  int found = 0;
  struct pid *pgrp;

- pgrp = find_pid(pgrp_nr);
+ pgrp = find_pid_ns(pgrp_nr, current->nsproxy->pid_ns);
  do_each_pid_task(pgrp, PIDTYPE_PGID, g) {
   target = g;
   while_each_thread(g, target) {
--- ./kernel/fork.c.fix 2007-07-06 11:09:33.000000000 +0400
+++ ./kernel/fork.c 2007-07-10 13:48:13.000000000 +0400
@@ -1267,7 +1267,7 @@ static struct task_struct *copy_process(
    __ptrace_link(p, current->parent);

   if (thread_group_leader(p)) {
-   if (clone_flags & CLONE_NEWPIDS) {
+   if (clone_flags & CLONE_NEWPID) {
     p->nsproxy->pid_ns->child_reaper = p;
     p->signal->tty = NULL;
     p->signal->pgrp = p->pid;
@@ -1434,7 +1434,7 @@ long do_fork(unsigned long clone_flags,
   else
    p->state = TASK_STOPPED;

-  nr = (clone_flags & CLONE_NEWPIDS) ?
+  nr = (clone_flags & CLONE_NEWPID) ?
    pid_nr_ns(task_pid(p), current->nsproxy->pid_ns) :
    pid_vnr(task_pid(p));

--- ./kernel/nsproxy.c.fix 2007-07-06 11:09:33.000000000 +0400
+++ ./kernel/nsproxy.c 2007-07-10 13:48:13.000000000 +0400
@@ -132,7 +132,8 @@ int copy_namespaces(unsigned long flags,

  get_nsproxy(old_ns);

- if (!(flags & (CLONE_NEWNS | CLONE_NEWUTS | CLONE_NEWIPC | CLONE_NEWUSER)))
+ if (!(flags & (CLONE_NEWNS | CLONE_NEWUTS | CLONE_NEWIPC |
+     CLONE_NEWUSER | CLONE_NEWPID)))
   return 0;

  if (!capable(CAP_SYS_ADMIN)) {
@@ -184,7 +185,7 @@ int unshare_nsproxy_namespaces(unsigned
  int err = 0;

  if (!(unshare_flags & (CLONE_NEWNS | CLONE_NEWUTS | CLONE_NEWIPC |
-         CLONE_NEWUSER | CLONE_NEWPIDS)))
+         CLONE_NEWUSER)))
   return 0;
```

```
  if (!capable(CAP_SYS_ADMIN))
--- ./kernel/pid.c.fix 2007-07-06 11:09:33.000000000 +0400
+++ ./kernel/pid.c 2007-07-10 13:48:19.000000000 +0400
@@ -523,7 +523,7 @@ struct pid_namespace *copy_pid_ns(unsign
  BUG_ON(!old_ns);
  get_pid_ns(old_ns);
  new_ns = old_ns;
- if (!(flags & CLONE_NEWPIDS))
+ if (!(flags & CLONE_NEWPID))
   goto out;

  new_ns = ERR_PTR(-EINVAL);
```

_____

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