Subject: [PATCH 2/2] Uninline find_pid etc set of functions
Posted by Pavel Emelianov on Mon, 24 Sep 2007 15:24:39 GMT
View Forum Message <> Reply to Message

The find_pid/_vpid/_pid_ns functions are used to find the
struct pid by its id, depending on whic id - global or
virtual - is used.

The find_vpid() is a macro that pushes the
current->nsproxy->pid_ns on the stack to call another
function - find_pid_ns(). It turned out, that this
dereference together with the push itself cause the kernel
text size to grow too much.

Move all these out-of-line. Together with the previous patch
this saves a bit less that 400 bytes from .text section.

Signed-off-by: Pavel Emelyanov <xemul@openvz.org>

```
diff --git a/fs/locks.c b/fs/locks.c
diff --git a/include/linux/pid.h b/include/linux/pid.h
index 4817c66..f79ba6f 100644
--- a/include/linux/pid.h
+++ b/include/linux/pid.h
@@ -110,9 +110,8 @@ extern struct pid_namespace init_pid_ns;
  * see also find_task_by_pid() set in include/linux/sched.h
  */
 extern struct pid *FASTCALL(find_pid_ns(int nr, struct pid_namespace *ns));
-
-#define find_vpid(pid) find_pid_ns(pid, current->nsproxy->pid_ns)
-#define find_pid(pid) find_pid_ns(pid, &init_pid_ns)
+extern struct pid *find_vpid(int nr);
+extern struct pid *find_pid(int nr);

 /*
  * Lookup a PID in the hash table, and return with it's count elevated.
diff --git a/kernel/pid.c b/kernel/pid.c
index e2e060e..f029c17 100644
--- a/kernel/pid.c
+++ b/kernel/pid.c
@@ -302,6 +302,18 @@ struct pid * fastcall find_pid_ns(int nr
 }
 EXPORT_SYMBOL_GPL(find_pid_ns);

+struct pid *find_vpid(int nr)
```

```
+{
+ return find_pid_ns(nr, current->nsproxy->pid_ns);
+}
+EXPORT_SYMBOL_GPL(find_vpid);
+
+struct pid *find_pid(int nr)
+{
+ return find_pid_ns(nr, &init_pid_ns);
+}
+EXPORT_SYMBOL_GPL(find_pid);
+
 /*
  * attach_pid() must be called with the tasklist_lock write-held.
  */
diff --git a/kernel/capability.c b/kernel/capability.c
index 0440d6d..4a881b8 100644
--- a/kernel/capability.c
+++ b/kernel/capability.c
@@ -97,7 +96,7 @@ static inline int cap_set_pg(int pgrp_nr
  int found = 0;
  struct pid *pgrp;

- pgrp = find_pid_ns(pgrp_nr, current->nsproxy->pid_ns);
+ pgrp = find_vpid(pgrp_nr);
  do_each_pid_task(pgrp, PIDTYPE_PGID, g) {
   target = g;
   while_each_thread(g, target) {
```