Subject: [PATCH] vt: Make vt_pid a struct pid (making it pid wrap around safe).
Posted by ebiederm on Sun, 10 Sep 2006 12:41:52 GMT
View Forum Message <> Reply to Message

I took a good hard look at the locking and it appears the locking on
vt_pid is the console semaphore. Every modified path is called under
the console semaphore except reset_vc when it is called from fn_SAK or
do_SAK both of which appear to be in interrupt context. In addition I
need to be careful because in the presence of an oops the console_sem
may be arbitrarily dropped.

Which leads me to conclude the current locking is inadequate for my
needs.

Given the weird cases we could hit because of oops printing instead of
introducing an extra spin lock to protect the data and keep the pid to
signal and the signal to send in sync, I have opted to use xchg on
just the struct pid * pointer instead.

Due to console_sem we will stay in sync between vt_pid and vt_mode
except for a small window during a SAK, or oops handling. SAK
handling should kill any user space process that care, and oops
handling we are broken anyway. Besides the worst that can happen is
that I try to send the wrong signal.

Signed-off-by: Eric W. Biederman <ebiederm@xmission.com>
---
 drivers/char/vt.c | 1 +
 drivers/char/vt_ioctl.c | 8 ++++----
 include/linux/console_struct.h | 2 +-
 3 files changed, 6 insertions(+), 5 deletions(-)

```
diff --git a/drivers/char/vt.c b/drivers/char/vt.c
index 0037682..4228f08 100644
--- a/drivers/char/vt.c
+++ b/drivers/char/vt.c
@@ -896,6 +896,7 @@ void vc_deallocate(unsigned int currcons
  if (vc_cons_allocated(currcons)) {
   struct vc_data *vc = vc_cons[currcons].d;
   vc->vc_sw->con_deinit(vc);
+  put_pid(vc->vt_pid);
   module_put(vc->vc_sw->owner);
   if (vc->vc_kmalloced)
    kfree(vc->vc_screenbuf);
diff --git a/drivers/char/vt_ioctl.c b/drivers/char/vt_ioctl.c
index dc408af..ac5d60e 100644
--- a/drivers/char/vt_ioctl.c
```

```
+++ b/drivers/char/vt_ioctl.c
@@ -672,7 +672,7 @@ #endif
   vc->vt_mode = tmp;
   /* the frsig is ignored, so we set it to 0 */
   vc->vt_mode.frsig = 0;
-  vc->vt_pid = current->pid;
+  put_pid(xchg(&vc->vt_pid, get_pid(task_pid(current))));
   /* no switch is required -- saw@shade.msu.ru */
   vc->vt_newvt = -1;
   release_console_sem();
@@ -1063,7 +1063,7 @@ void reset_vc(struct vc_data *vc)
  vc->vt_mode.relsig = 0;
  vc->vt_mode.acqsig = 0;
  vc->vt_mode.frsig = 0;
- vc->vt_pid = -1;
+ put_pid(xchg(&vc->vt_pid, NULL));
  vc->vt_newvt = -1;
  if (!in_interrupt())    /* Via keyboard.c:SAK() - akpm */
   reset_palette(vc);
@@ -1114,7 +1114,7 @@ static void complete_change_console(stru
    * tell us if the process has gone or something else
    * is awry
    */
-  if (kill_proc(vc->vt_pid, vc->vt_mode.acqsig, 1) != 0) {
+  if (kill_pid(vc->vt_pid, vc->vt_mode.acqsig, 1) != 0) {
   /*
    * The controlling process has died, so we revert back to
    * normal operation. In this case, we'll also change back
@@ -1174,7 +1174,7 @@ void change_console(struct vc_data *new_
    * tell us if the process has gone or something else
    * is awry
    */
-  if (kill_proc(vc->vt_pid, vc->vt_mode.relsig, 1) == 0) {
+  if (kill_pid(vc->vt_pid, vc->vt_mode.relsig, 1) == 0) {
    /*
     * It worked. Mark the vt to switch to and
     * return. The process needs to send us a
diff --git a/include/linux/console_struct.h b/include/linux/console_struct.h
index 25423f7..ed6c0fe 100644
--- a/include/linux/console_struct.h
+++ b/include/linux/console_struct.h
@@ -54,7 +54,7 @@ struct vc_data {
  struct tty_struct *vc_tty;  /* TTY we are attached to */
  /* data for manual vt switching */
  struct vt_mode vt_mode;
- int  vt_pid;
+ struct pid  *vt_pid;
  int  vt_newvt;
```

```
  wait_queue_head_t paste_wait;
  /* mode flags */
--
1.4.2.rc3.g7e18e-dirty
```

_____

Containers mailing list
Containers@lists.osdl.org
https://lists.osdl.org/mailman/listinfo/containers