
Subject: [PATCH 2/7]: Factor out PTY index allocation
Posted by [Sukadev Bhattiprolu](#) on Tue, 25 Mar 2008 04:23:42 GMT
[View Forum Message](#) <> [Reply to Message](#)

From: Sukadev Bhattiprolu <sukadev@us.ibm.com>
Subject: [PATCH 2/7]: Factor out PTY index allocation

Factor out the code used to allocate/free a pts index into new interfaces, devpts_new_index() and devpts_kill_index(). This localizes the external data structures used in managing the pts indices.

Signed-off-by: Sukadev Bhattiprolu <sukadev@us.ibm.com>
Signed-off-by: Serge Hallyn <serue@us.ibm.com>
Signed-off-by: Matt Helsley <matthltc@us.ibm.com>

drivers/char/tty_io.c | 40 ++++++-----
fs/devpts/inode.c | 42 ++++++-----
include/linux/devpts_fs.h | 4 +---
3 files changed, 51 insertions(+), 35 deletions(-)

Index: 2.6.25-rc5-mm1/include/linux/devpts_fs.h

=====

--- 2.6.25-rc5-mm1.orig/include/linux/devpts_fs.h 2008-03-24 20:04:07.000000000 -0700
+++ 2.6.25-rc5-mm1/include/linux/devpts_fs.h 2008-03-24 20:04:26.000000000 -0700
@@ -17,6 +17,8 @@

#ifdef CONFIG_UNIX98_PTYS

+int devpts_new_index(void);
+void devpts_kill_index(int idx);
int devpts_pty_new(struct tty_struct *tty); /* mknod in devpts */
struct tty_struct *devpts_get_tty(int number); /* get tty structure */
void devpts_pty_kill(int number); /* unlink */
@@ -24,6 +26,8 @@ void devpts_pty_kill(int number); /* u
#else

/* Dummy stubs in the no-pty case */
+static inline int devpts_new_index(void) { return -EINVAL; }
+static inline void devpts_kill_index(int idx) { }
static inline int devpts_pty_new(struct tty_struct *tty) { return -EINVAL; }
static inline struct tty_struct *devpts_get_tty(int number) { return NULL; }
static inline void devpts_pty_kill(int number) { }

Index: 2.6.25-rc5-mm1/drivers/char/tty_io.c

=====

--- 2.6.25-rc5-mm1.orig/drivers/char/tty_io.c 2008-03-24 20:04:07.000000000 -0700
+++ 2.6.25-rc5-mm1/drivers/char/tty_io.c 2008-03-24 20:04:26.000000000 -0700
@@ -91,7 +91,6 @@

```

#include <linux/module.h>
#include <linux/smp_lock.h>
#include <linux/device.h>
#include <linux/idr.h>
#include <linux/wait.h>
#include <linux/bitops.h>
#include <linux/delay.h>
@@ -137,9 +136,6 @@ EXPORT_SYMBOL(tty_mutex);

#ifdef CONFIG_UNIX98_PTYS
extern struct tty_driver *ptm_driver; /* Unix98 pty masters; for /dev/ptmx */
extern int pty_limit; /* Config limit on Unix98 ptys */
static DEFINE_IDR(allocated_ptys);
static DEFINE_MUTEX(allocated_ptys_lock);
static int ptmx_open(struct inode *, struct file *);
#endif

@@ -2636,15 +2632,9 @@ static void release_dev(struct file *fil
    */
    release_tty(tty, idx);

#ifdef CONFIG_UNIX98_PTYS
    /* Make this pty number available for reallocation */
    if (devpts) {
        mutex_lock(&allocated_ptys_lock);
        idr_remove(&allocated_ptys, idx);
        mutex_unlock(&allocated_ptys_lock);
    }
#endif
-
+ if (devpts)
+ devpts_kill_index(idx);
    }

/**
@@ -2800,29 +2790,13 @@ static int ptmx_open(struct inode *inode
    struct tty_struct *tty;
    int retval;
    int index;
- int idr_ret;

    nonseekable_open(inode, filp);

    /* find a device that is not in use. */
- mutex_lock(&allocated_ptys_lock);
- if (!idr_pre_get(&allocated_ptys, GFP_KERNEL)) {
- mutex_unlock(&allocated_ptys_lock);
- return -ENOMEM;

```

```

- }
- idr_ret = idr_get_new(&allocated_ptys, NULL, &index);
- if (idr_ret < 0) {
-     mutex_unlock(&allocated_ptys_lock);
-     if (idr_ret == -EAGAIN)
-         return -ENOMEM;
-     return -EIO;
- }
- if (index >= pty_limit) {
-     idr_remove(&allocated_ptys, index);
-     mutex_unlock(&allocated_ptys_lock);
-     return -EIO;
- }
- mutex_unlock(&allocated_ptys_lock);
+ index = devpts_new_index();
+ if (index < 0)
+     return index;

```

```

    mutex_lock(&tty_mutex);
    retval = init_dev(ptm_driver, index, &tty);
@@ -2847,9 +2821,7 @@ out1:
    release_dev(filp);
    return retval;

```

```

out:
- mutex_lock(&allocated_ptys_lock);
- idr_remove(&allocated_ptys, index);
- mutex_unlock(&allocated_ptys_lock);
+ devpts_kill_index(index);
    return retval;
}

```

#endif

Index: 2.6.25-rc5-mm1/fs/devpts/inode.c

=====

--- 2.6.25-rc5-mm1.orig/fs/devpts/inode.c 2008-03-24 20:04:07.000000000 -0700

+++ 2.6.25-rc5-mm1/fs/devpts/inode.c 2008-03-24 20:04:26.000000000 -0700

@@ -17,6 +17,7 @@

#include <linux/namei.h>

#include <linux/mount.h>

#include <linux/tty.h>

+#include <linux/idr.h>

#include <linux/devpts_fs.h>

#include <linux/parser.h>

#include <linux/fsnotify.h>

@@ -26,6 +27,10 @@

#define DEVPTS_DEFAULT_MODE 0600

+extern int pty_limit; /* Config limit on Unix98 ptys */

```

+static DEFINE_IDR(allocated_ptys);
+static DECLARE_MUTEX(allocated_ptys_lock);
+
+static struct vfsmount *devpts_mnt;
+static struct dentry *devpts_root;

@@ -171,9 +176,44 @@ static struct dentry *get_node(int num)
+return lookup_one_len(s, root, sprintf(s, "%d", num));
+}

+int devpts_new_index(void)
+{
+int index;
+int idr_ret;
+
+retry:
+if (!idr_pre_get(&allocated_ptys, GFP_KERNEL)) {
+return -ENOMEM;
+}
+
+down(&allocated_ptys_lock);
+idr_ret = idr_get_new(&allocated_ptys, NULL, &index);
+if (idr_ret < 0) {
+up(&allocated_ptys_lock);
+if (idr_ret == -EAGAIN)
+goto retry;
+return -EIO;
+}
+
+if (index >= pty_limit) {
+idr_remove(&allocated_ptys, index);
+up(&allocated_ptys_lock);
+return -EIO;
+}
+up(&allocated_ptys_lock);
+return index;
+}
+
+void devpts_kill_index(int idx)
+{
+down(&allocated_ptys_lock);
+idr_remove(&allocated_ptys, idx);
+up(&allocated_ptys_lock);
+}
+
+int devpts_pty_new(struct tty_struct *tty)
+{
+int number = tty->index;

```

```
+ int number = tty->index; /* tty layer puts index from devpts_new_index() in here */  
  struct tty_driver *driver = tty->driver;  
  dev_t device = MKDEV(driver->major, driver->minor_start+number);  
  struct dentry *dentry;
```

Containers mailing list

Containers@lists.linux-foundation.org

<https://lists.linux-foundation.org/mailman/listinfo/containers>
