
Subject: [PATCH]: Factor out PTY index allocation
Posted by [Sukadev Bhattiprolu](#) on Wed, 16 Apr 2008 22:17:23 GMT
[View Forum Message](#) <> [Reply to Message](#)

We noticed this while working on pts namespaces and believe this might be an useful change even as we rework our pts/device namespace approach.

From: Sukadev Bhattiprolu <sukadev@us.ibm.com>
Subject: [PATCH]: 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-rc8-mm2/include/linux/devpts_fs.h

```
=====
```

--- 2.6.25-rc8-mm2.orig/include/linux/devpts_fs.h 2008-01-26 09:49:16.000000000 -0800
+++ 2.6.25-rc8-mm2/include/linux/devpts_fs.h 2008-04-16 09:51:15.000000000 -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-rc8-mm2/drivers/char/tty_io.c

```
=====
--- 2.6.25-rc8-mm2.orig/drivers/char/tty_io.c 2008-04-16 09:51:11.000000000 -0700
+++ 2.6.25-rc8-mm2/drivers/char/tty_io.c 2008-04-16 09:51:15.000000000 -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-rc8-mm2/fs/devpts/inode.c

```

=====
--- 2.6.25-rc8-mm2.orig/fs/devpts/inode.c 2008-02-27 15:17:59.000000000 -0800
+++ 2.6.25-rc8-mm2/fs/devpts/inode.c 2008-04-16 09:51:15.000000000 -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>

Subject: Re: [PATCH]: Factor out PTY index allocation
Posted by [serue](#) on Thu, 17 Apr 2008 15:42:41 GMT
[View Forum Message](#) <> [Reply to Message](#)

Quoting sukadev@us.ibm.com (sukadev@us.ibm.com):

- > We noticed this while working on pts namespaces and believe this might
- > be an useful change even as we rework our pts/device namespace approach.
- >
- > ---
- >
- > From: Sukadev Bhattiprolu <sukadev@us.ibm.com>
- > Subject: [PATCH]: 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<matthlhc@us.ibm.com>

No traces of devpts namespaces here, so I assume this should be non-offensive and fine for inclusion.

thanks,
-serge

- > ---
- > 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-rc8-mm2/include/linux/devpts_fs.h
> =====
> --- 2.6.25-rc8-mm2.orig/include/linux/devpts_fs.h 2008-01-26 09:49:16.000000000 -0800
> +++ 2.6.25-rc8-mm2/include/linux/devpts_fs.h 2008-04-16 09:51:15.000000000 -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-rc8-mm2/drivers/char/tty_io.c
> =====
> --- 2.6.25-rc8-mm2.orig/drivers/char/tty_io.c 2008-04-16 09:51:11.000000000 -0700
> +++ 2.6.25-rc8-mm2/drivers/char/tty_io.c 2008-04-16 09:51:15.000000000 -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);
>

```

```

> #ifndef 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-rc8-mm2/fs/devpts/inode.c
> =====
> --- 2.6.25-rc8-mm2.orig/fs/devpts/inode.c 2008-02-27 15:17:59.000000000 -0800
> +++ 2.6.25-rc8-mm2/fs/devpts/inode.c 2008-04-16 09:51:15.000000000 -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>

Subject: Re: [PATCH]: Factor out PTY index allocation
Posted by [hpa](#) on Thu, 17 Apr 2008 16:05:36 GMT
[View Forum Message](#) <> [Reply to Message](#)

Serge E. Hallyn wrote:

```

> Quoting sukadev@us.ibm.com (sukadev@us.ibm.com):
>> We noticed this while working on pts namespaces and believe this might
>> be an useful change even as we rework our pts/device namespace approach.
>>
>> ---
>>

```

>> From: Sukadev Bhattiprolu <sukadev@us.ibm.com>
>> Subject: [PATCH]: 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>
>
> No traces of devpts namespaces here, so I assume this should be
> non-offensive and fine for inclusion.
>

Acked-by: H. Peter Anvin <hpa@zytor.com>

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

Subject: Re: [PATCH]: Factor out PTY index allocation
Posted by [Jiri Slaby](#) on Thu, 17 Apr 2008 21:08:17 GMT
[View Forum Message](#) <> [Reply to Message](#)

On 04/17/2008 12:17 AM, sukadev@us.ibm.com wrote:

> We noticed this while working on pts namespaces and believe this might
> be an useful change even as we rework our pts/device namespace approach.
>
> ---
>
> From: Sukadev Bhattiprolu <sukadev@us.ibm.com>
> Subject: [PATCH]: 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-rc8-mm2/drivers/char/tty_io.c
> =====
> --- 2.6.25-rc8-mm2.orig/drivers/char/tty_io.c 2008-04-16 09:51:11.000000000 -0700
> +++ 2.6.25-rc8-mm2/drivers/char/tty_io.c 2008-04-16 09:51:15.000000000 -0700
> @@ -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);
[...]
```

```
> Index: 2.6.25-rc8-mm2/fs/devpts/inode.c
> =====
> --- 2.6.25-rc8-mm2.orig/fs/devpts/inode.c 2008-02-27 15:17:59.000000000 -0800
> +++ 2.6.25-rc8-mm2/fs/devpts/inode.c 2008-04-16 09:51:15.000000000 -0700
> @@ -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);
```

Ugh, why mutex->semaphore conversion?

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

Subject: Re: [PATCH]: Factor out PTY index allocation
Posted by [akpm](#) on Thu, 17 Apr 2008 21:17:25 GMT
[View Forum Message](#) <> [Reply to Message](#)

On Wed, 16 Apr 2008 15:17:23 -0700
sukadev@us.ibm.com wrote:

```
> 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.
```

err...

```
> - mutex_lock(&allocated_ptys_lock);
> + down(&allocated_ptys_lock);
```

Why the mutex-to-semaphore conversion?

--- a/fs/devpts/inode.c~devpts-factor-out-pty-index-allocation-fix

+++ a/fs/devpts/inode.c

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

#include <linux/namei.h>

#include <linux/mount.h>

#include <linux/tty.h>

+#include <linux/mutex.h>

#include <linux/idr.h>

#include <linux/devpts_fs.h>

#include <linux/parser.h>

@@ -29,7 +30,7 @@

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

static DEFINE_IDR(allocated_ptys);

-static DECLARE_MUTEX(allocated_ptys_lock);

+static DEFINE_MUTEX(allocated_ptys_lock);

static struct vfsmount *devpts_mnt;

static struct dentry *devpts_root;

@@ -186,10 +187,10 @@ retry:

return -ENOMEM;

}

- down(&allocated_ptys_lock);

+ mutex_lock(&allocated_ptys_lock);

idr_ret = idr_get_new(&allocated_ptys, NULL, &index);

if (idr_ret < 0) {

- up(&allocated_ptys_lock);

+ mutex_unlock(&allocated_ptys_lock);

if (idr_ret == -EAGAIN)

goto retry;

return -EIO;

@@ -197,18 +198,18 @@ retry:

if (index >= pty_limit) {

idr_remove(&allocated_ptys, index);

- up(&allocated_ptys_lock);

+ mutex_unlock(&allocated_ptys_lock);

return -EIO;

}

- up(&allocated_ptys_lock);

+ mutex_unlock(&allocated_ptys_lock);

return index;

}

```
void devpts_kill_index(int idx)
{
- down(&allocated_ptys_lock);
+ mutex_lock(&allocated_ptys_lock);
  idr_remove(&allocated_ptys, idx);
- up(&allocated_ptys_lock);
+ mutex_unlock(&allocated_ptys_lock);
}

int devpts_pty_new(struct tty_struct *tty)
-
```

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

Subject: Re: [PATCH]: Factor out PTY index allocation
Posted by [Matt Helsley](#) on Thu, 17 Apr 2008 23:53:49 GMT
[View Forum Message](#) <> [Reply to Message](#)

On Thu, 2008-04-17 at 14:17 -0700, Andrew Morton wrote:
> On Wed, 16 Apr 2008 15:17:23 -0700
> sukadev@us.ibm.com wrote:
>
> > 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.
>
> err...
>
> > - mutex_lock(&allocated_ptys_lock);
> > + down(&allocated_ptys_lock);
>
> Why the mutex-to-semaphore conversion?

The patch series was originally developed for 2.6.22 -- prior to the semaphore -> mutex migration. This appears to be a mistake in forward-porting the series.

Acked-by: Matt Helsley <matthltc@us.ibm.com>

```
>
> --- a/fs/devpts/inode.c~devpts-factor-out-pty-index-allocation-fix
> +++ a/fs/devpts/inode.c
> @@ -17,6 +17,7 @@
> #include <linux/namei.h>
```

```

> #include <linux/mount.h>
> #include <linux/tty.h>
> +#include <linux/mutex.h>
> #include <linux/idr.h>
> #include <linux/devpts_fs.h>
> #include <linux/parser.h>
> @@ -29,7 +30,7 @@
>
> extern int pty_limit; /* Config limit on Unix98 ptys */
> static DEFINE_IDR(allocated_ptys);
> -static DECLARE_MUTEX(allocated_ptys_lock);
> +static DEFINE_MUTEX(allocated_ptys_lock);
>
> static struct vfsmount *devpts_mnt;
> static struct dentry *devpts_root;
> @@ -186,10 +187,10 @@ retry:
>     return -ENOMEM;
> }
>
> - down(&allocated_ptys_lock);
> + mutex_lock(&allocated_ptys_lock);
>     idr_ret = idr_get_new(&allocated_ptys, NULL, &index);
>     if (idr_ret < 0) {
> - up(&allocated_ptys_lock);
> + mutex_unlock(&allocated_ptys_lock);
>         if (idr_ret == -EAGAIN)
>             goto retry;
>         return -EIO;
>     @@ -197,18 +198,18 @@ retry:
>
>     if (index >= pty_limit) {
>         idr_remove(&allocated_ptys, index);
> - up(&allocated_ptys_lock);
> + mutex_unlock(&allocated_ptys_lock);
>         return -EIO;
>     }
> - up(&allocated_ptys_lock);
> + mutex_unlock(&allocated_ptys_lock);
>     return index;
> }
>
> void devpts_kill_index(int idx)
> {
> - down(&allocated_ptys_lock);
> + mutex_lock(&allocated_ptys_lock);
>     idr_remove(&allocated_ptys, idx);
> - up(&allocated_ptys_lock);
> + mutex_unlock(&allocated_ptys_lock);

```

> }
>
> int devpts_pty_new(struct tty_struct *tty)
> _
>
> --
> To unsubscribe from this list: send the line "unsubscribe linux-kernel" in
> the body of a message to majordomo@vger.kernel.org
> More majordomo info at <http://vger.kernel.org/majordomo-info.html>
> Please read the FAQ at <http://www.tux.org/lkml/>
>

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