Subject: [PATCH 3/4]: Move devpts globals into init_pts_ns
Posted by Sukadev Bhattiprolu on Sat, 12 Apr 2008 17:33:27 GMT
View Forum Message <> Reply to Message

Matt, Serge, please sign-off on this version.

From: Sukadev Bhattiprolu <sukadev@us.ibm.com>
Subject: [PATCH 3/4]: Move devpts globals into init_pts_ns

Move devpts global variables 'allocated_ptys' and 'devpts_mnt' into a new
'pts_namespace' and remove the 'devpts_root'.

Changelog:
 - Split these relatively simpler changes off from the patch that
   supports remounting devpts.

Signed-off-by: Sukadev Bhattiprolu <sukadev@us.ibm.com>
---
 fs/devpts/inode.c | 84 ++++++++++++++++++++++++++++++++++++----------
 include/linux/devpts_fs.h | 10 +++++
 2 files changed, 70 insertions(+), 24 deletions(-)

Index: 2.6.25-rc8-mm1/fs/devpts/inode.c
===
--- 2.6.25-rc8-mm1.orig/fs/devpts/inode.c 2008-04-11 10:12:09.000000000 -0700
+++ 2.6.25-rc8-mm1/fs/devpts/inode.c 2008-04-12 10:10:33.000000000 -0700
@@ -28,12 +28,8 @@
 #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;
-
 static struct {
  int setuid;
  int setgid;
@@ -54,6 +50,14 @@ static match_table_t tokens = {
  {Opt_err, NULL}
 };

+struct pts_namespace init_pts_ns = {
+ .kref = {
+ .refcount = ATOMIC_INIT(2),
+ },
+ .allocated_ptys = IDR_INIT(init_pts_ns.allocated_ptys),

```
+ .mnt = NULL,
+};
+
 static int devpts_remount(struct super_block *sb, int *flags, char *data)
 {
  char *p;
@@ -140,7 +144,7 @@ devpts_fill_super(struct super_block *s,
  inode->i_fop = &simple_dir_operations;
  inode->i_nlink = 2;

- devpts_root = s->s_root = d_alloc_root(inode);
+ s->s_root = d_alloc_root(inode);
  if (s->s_root)
   return 0;

@@ -168,10 +172,9 @@ static struct file_system_type devpts_fs
  * to the System V naming convention
  */

-static struct dentry *get_node(int num)
+static struct dentry *get_node(struct dentry *root, int num)
 {
  char s[12];
- struct dentry *root = devpts_root;
  mutex_lock(&root->d_inode->i_mutex);
  return lookup_one_len(s, root, sprintf(s, "%d", num));
 }
@@ -180,14 +183,17 @@ int devpts_new_index(void)
 {
  int index;
  int idr_ret;
+ struct pts_namespace *pts_ns;
+
+ pts_ns = &init_pts_ns;

 retry:
- if (!idr_pre_get(&allocated_ptys, GFP_KERNEL)) {
+ if (!idr_pre_get(&pts_ns->allocated_ptys, GFP_KERNEL)) {
   return -ENOMEM;
  }

  down(&allocated_ptys_lock);
- idr_ret = idr_get_new(&allocated_ptys, NULL, &index);
+ idr_ret = idr_get_new(&pts_ns->allocated_ptys, NULL, &index);
  if (idr_ret < 0) {
   up(&allocated_ptys_lock);
   if (idr_ret == -EAGAIN)
@@ -196,7 +202,7 @@ retry:
```

```
  }

  if (index >= pty_limit) {
-  idr_remove(&allocated_ptys, index);
+  idr_remove(&pts_ns->allocated_ptys, index);
   up(&allocated_ptys_lock);
   return -EIO;
  }
@@ -206,8 +212,10 @@ retry:

 void devpts_kill_index(int idx)
 {
+ struct pts_namespace *pts_ns = &init_pts_ns;
+
  down(&allocated_ptys_lock);
- idr_remove(&allocated_ptys, idx);
+ idr_remove(&pts_ns->allocated_ptys, idx);
  up(&allocated_ptys_lock);
 }

@@ -217,12 +225,26 @@ int devpts_pty_new(struct tty_struct *tt
  struct tty_driver *driver = tty->driver;
  dev_t device = MKDEV(driver->major, driver->minor_start+number);
  struct dentry *dentry;
- struct inode *inode = new_inode(devpts_mnt->mnt_sb);
+ struct dentry *root;
+ struct inode *inode;
+ struct pts_namespace *pts_ns;

  /* We're supposed to be given the slave end of a pty */
  BUG_ON(driver->type != TTY_DRIVER_TYPE_PTY);
  BUG_ON(driver->subtype != PTY_TYPE_SLAVE);

+ pts_ns = &init_pts_ns;
+ root = pts_ns->mnt->mnt_root;
+
+ mutex_lock(&root->d_inode->i_mutex);
+ inode = idr_find(&pts_ns->allocated_ptys, number);
+ mutex_unlock(&root->d_inode->i_mutex);
+
+ if (inode && !IS_ERR(inode))
+  return -EEXIST;
+
+ inode = new_inode(pts_ns->mnt->mnt_sb);
+
  if (!inode)
   return -ENOMEM;
```

```
@@ -232,23 +254,28 @@ int devpts_pty_new(struct tty_struct *tt
  inode->i_mtime = inode->i_atime = inode->i_ctime = CURRENT_TIME;
  init_special_inode(inode, S_IFCHR|config.mode, device);
  inode->i_private = tty;
+ idr_replace(&pts_ns->allocated_ptys, inode, number);

- dentry = get_node(number);
+ dentry = get_node(root, number);
  if (!IS_ERR(dentry) && !dentry->d_inode) {
   d_instantiate(dentry, inode);
-  fsnotify_create(devpts_root->d_inode, dentry);
+  fsnotify_create(root->d_inode, dentry);
  }

- mutex_unlock(&devpts_root->d_inode->i_mutex);
+ mutex_unlock(&root->d_inode->i_mutex);

  return 0;
 }

 struct tty_struct *devpts_get_tty(int number)
 {
- struct dentry *dentry = get_node(number);
+ struct dentry *root;
+ struct dentry *dentry;
  struct tty_struct *tty;

+ root = init_pts_ns.mnt->mnt_root;
+ dentry = get_node(root, number);
+
  tty = NULL;
  if (!IS_ERR(dentry)) {
   if (dentry->d_inode)
@@ -256,14 +283,18 @@ struct tty_struct *devpts_get_tty(int nu
   dput(dentry);
  }

- mutex_unlock(&devpts_root->d_inode->i_mutex);
+ mutex_unlock(&root->d_inode->i_mutex);

  return tty;
 }

 void devpts_pty_kill(int number)
 {
- struct dentry *dentry = get_node(number);
+ struct dentry *root;
+ struct dentry *dentry;
```

```
+
+ root = init_pts_ns.mnt->mnt_root;
+ dentry = get_node(root, number);

  if (!IS_ERR(dentry)) {
   struct inode *inode = dentry->d_inode;
@@ -274,16 +305,21 @@ void devpts_pty_kill(int number)
   }
   dput(dentry);
  }
- mutex_unlock(&devpts_root->d_inode->i_mutex);
+ mutex_unlock(&root->d_inode->i_mutex);
 }

 static int __init init_devpts_fs(void)
 {
- int err = register_filesystem(&devpts_fs_type);
+ struct vfsmount *mnt;
+ int err;
+
+ err = register_filesystem(&devpts_fs_type);
  if (!err) {
-  devpts_mnt = kern_mount(&devpts_fs_type);
-  if (IS_ERR(devpts_mnt))
-   err = PTR_ERR(devpts_mnt);
+  mnt = kern_mount(&devpts_fs_type);
+  if (IS_ERR(mnt))
+   err = PTR_ERR(mnt);
+  else
+   init_pts_ns.mnt = mnt;
  }
  return err;
 }
@@ -291,7 +327,7 @@ static int __init init_devpts_fs(void)
 static void __exit exit_devpts_fs(void)
 {
  unregister_filesystem(&devpts_fs_type);
- mntput(devpts_mnt);
+ mntput(init_pts_ns.mnt);
 }

 module_init(init_devpts_fs)
Index: 2.6.25-rc8-mm1/include/linux/devpts_fs.h
===================================================================
--- 2.6.25-rc8-mm1.orig/include/linux/devpts_fs.h 2008-04-11 10:34:16.000000000 -0700
+++ 2.6.25-rc8-mm1/include/linux/devpts_fs.h 2008-04-12 08:52:57.000000000 -0700
@@ -14,6 +14,16 @@
 #define _LINUX_DEVPTS_FS_H
```

```
 #include <linux/errno.h>
+#include <linux/kref.h>
+#include <linux/idr.h>
+
+struct pts_namespace {
+ struct kref kref;
+ struct idr allocated_ptys;
+ struct vfsmount *mnt;
+};
+
+extern struct pts_namespace init_pts_ns;

 #ifdef CONFIG_UNIX98_PTYS
```