Subject: [PATCH 4/7][v2]: Allow mknod of ptmx and tty in devpts
Posted by Sukadev Bhattiprolu on Thu, 03 Apr 2008 07:09:41 GMT
View Forum Message <> Reply to Message

From: Sukadev Bhattiprolu <sukadev@us.ibm.com>
Subject: [PATCH 4/7][v2]: Allow mknod of ptmx and tty in devpts

We want to allow administrators to access PTYs in a descendant pts-namespaces,
for instance "echo foo > /vserver/vserver1/dev/pts/0". To enable such access
we must hold a reference to the pts-ns in which the device (ptmx or the
slave pty) exists.

Note that we cannot use the pts-ns of the 'current' process since that pts-ns
could be different from the pts-ns in which the PTY device was created. So
we find the pts-ns from the inode of the PTY (inode->i_sb->s_fs_info).
While this would work for the slave PTY devices like /dev/pts/0, it would
not work for either the master PTY device (/dev/ptmx) or controlling terminal
(/dev/tty).

To uniformly handle the master, slave and controlling ttys, we allow creation
of 'ptmx' and 'tty' devices in /dev/pts. When creating containers, the
administrator can then:

```
 $ umount /dev/pts
 $ mount -t devpts lxcpts /dev/pts
 $ mknod /dev/pts/ptmx c 5 2
 $ mknod /dev/pts/tty c 5 0
 $ rm /dev/ptmx /dev/tty
 $ ln -s /dev/pts/ptmx /dev/ptmx
 $ ln -s /dev/pts/tty /dev/tty
```

With this, even if the 'ptmx' is accessed from parent pts-ns we still find
and hold the pts-ns in which 'ptmx' actually belongs.

This patch merely allows creation of /dev/pts/ptmx and /dev/pts/tty. We
hold a reference to the dentries for these nodes to pin them in memory and
use 'kill_litter_super()' while unmounting to ensure we drop these dentries.

TODO: Ability to unlink the /dev/pts/ptmx and /dev/pts/tty nodes.

Note: if /dev/ptmx is a symlink to /vserver/vserver1/dev/pts/ptmx an open
 of /dev/ptmx in init-pts-ns will create a PTY in 'vserver1' !

Signed-off-by: Sukadev Bhattiprolu <sukadev@us.ibm.com>

```
 fs/devpts/inode.c |   55 ++++++++++++++++++++++++++++++++++++++++++++++++++----
 1 file changed, 51 insertions(+), 4 deletions(-)
```

```
Index: 2.6.25-rc5-mm1/fs/devpts/inode.c
===================================================================
--- 2.6.25-rc5-mm1.orig/fs/devpts/inode.c 2008-04-02 10:18:42.000000000 -0700
+++ 2.6.25-rc5-mm1/fs/devpts/inode.c 2008-04-02 22:51:02.000000000 -0700
@@ -58,7 +58,6 @@ struct pts_namespace init_pts_ns = {
  .mnt = NULL,
 };

-
 static int devpts_remount(struct super_block *sb, int *flags, char *data)
 {
  char *p;
@@ -122,6 +121,54 @@ static const struct super_operations dev
  .show_options = devpts_show_options,
 };

+
+static int devpts_mknod(struct inode *dir, struct dentry *dentry,
+   int mode, dev_t rdev)
+{
+ int inum;
+ struct inode *inode;
+ struct super_block *sb = dir->i_sb;
+
+ if (dentry->d_inode)
+  return -EEXIST;
+
+ if (!S_ISCHR(mode))
+  return -EPERM;
+
+ if (rdev == MKDEV(TTYAUX_MAJOR, 0))
+  inum = 2;
+ else if (rdev == MKDEV(TTYAUX_MAJOR, 2))
+  inum = 3;
+ else
+  return -EPERM;
+
+ inode = new_inode(sb);
+ if (!inode)
+  return -ENOMEM;
+
+ inode->i_ino = inum;
+ inode->i_uid = inode->i_gid = 0;
+ inode->i_blocks = 0;
+ inode->i_mtime = inode->i_atime = inode->i_ctime = CURRENT_TIME;
+
+ init_special_inode(inode, mode, rdev);
+
```

```
+ d_instantiate(dentry, inode);
+ /*
+  * Get a reference to the dentry so the device-nodes persist
+  * even when there are no active references to them. We use
+  * kill_litter_super() to remove this entry when unmounting
+  * devpts.
+  */
+ dget(dentry);
+ return 0;
+}
+
+const struct inode_operations devpts_dir_inode_operations = {
+        .lookup         = simple_lookup,
+ .mknod  = devpts_mknod,
+};
+
 static int
 devpts_fill_super(struct super_block *s, void *data, int silent)
 {
@@ -141,7 +188,7 @@ devpts_fill_super(struct super_block *s,
  inode->i_blocks = 0;
  inode->i_uid = inode->i_gid = 0;
  inode->i_mode = S_IFDIR | S_IRUGO | S_IXUGO | S_IWUSR;
- inode->i_op = &simple_dir_inode_operations;
+ inode->i_op = &devpts_dir_inode_operations;
  inode->i_fop = &simple_dir_operations;
  inode->i_nlink = 2;

@@ -214,7 +261,7 @@ static int devpts_get_sb(struct file_sys
 static void devpts_kill_sb(struct super_block *sb)
 {
  sb->s_fs_info = NULL;
- kill_anon_super(sb);
+ kill_litter_super(sb);
 }

 static struct file_system_type devpts_fs_type = {
@@ -303,7 +350,7 @@ int devpts_pty_new(struct tty_struct *tt
  if (!inode)
   return -ENOMEM;

- inode->i_ino = number+2;
+ inode->i_ino = number+4;
  inode->i_uid = config.setuid ? config.uid : current->fsuid;
  inode->i_gid = config.setgid ? config.gid : current->fsgid;
  inode->i_mtime = inode->i_atime = inode->i_ctime = CURRENT_TIME;
```

_____

Containers mailing list