Subject: [PATCH 0/4] Helper patches for PTY namespaces
Posted by Sukadev Bhattiprolu on Sat, 12 Apr 2008 17:29:33 GMT
View Forum Message <> Reply to Message

Some simple helper patches to enable implementation of multiple PTY
(or device) namespaces.

[PATCH 1/4]: Propagate error code from devpts_pty_new
[PATCH 2/4]: Factor out PTY index allocation
[PATCH 3/4]: Move devpts globals into init_pts_ns
[PATCH 3/4]: Enable multiple mounts of /dev/pts

This patchset is based on earlier versions developed by Serge Hallyn
and Matt Helsley.

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

Subject: [PATCH 1/4]: Propagate error code from devpts_pty_new
Posted by Sukadev Bhattiprolu on Sat, 12 Apr 2008 17:32:02 GMT
View Forum Message <> Reply to Message

From: Sukadev Bhattiprolu <sukadev@us.ibm.com>
Subject: [PATCH 1/4]: Propagate error code from devpts_pty_new

Have ptmx_open() propagate any error code returned by devpts_pty_new()
(which returns either 0 or -ENOMEM anyway).

Signed-off-by: Sukadev Bhattiprolu <sukadev@us.ibm.com>
---
drivers/char/tty_io.c | 4 ++--
1 file changed, 2 insertions(+), 2 deletions(-)

Index: 2.6.25-rc8-mm1/drivers/char/tty_io.c
===
--- 2.6.25-rc8-mm1.orig/drivers/char/tty_io.c 2008-04-07 14:49:56.000000000 -0700
+++ 2.6.25-rc8-mm1/drivers/char/tty_io.c 2008-04-09 13:54:00.000000000 -0700
@@ -2835,8 +2835,8 @@ static int ptmx_open(struct inode *inode
filp->private_data = tty;
file_move(filp, &tty->tty_files);

- retval = -ENOMEM;
- if (devpts_pty_new(tty->link))
+ retval = devpts_pty_new(tty->link);
+ if (retval)
goto out1;

check_tty_count(tty, "tty_open");

_____

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

---

Subject: [PATCH 2/4]: Factor out PTY index allocation
Posted by Sukadev Bhattiprolu on Sat, 12 Apr 2008 17:32:38 GMT
View Forum Message <> Reply to Message

From: Sukadev Bhattiprolu <sukadev@us.ibm.com>
Subject: [PATCH 2/4]: 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

---

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
```

Subject: [PATCH 4/4]: Enable multiple mounts of /dev/pts
Posted by Sukadev Bhattiprolu on Sat, 12 Apr 2008 17:34:28 GMT
View Forum Message <> Reply to Message

From: Sukadev Bhattiprolu <sukadev@us.ibm.com>
Subject:[PATCH 4/4]: Enable multiple mounts of /dev/pts

To support multiple PTY namespaces, allow multiple mounts of /dev/pts, once
within each PTY namespace.

This patch removes the get_sb_single() in devpts_get_sb() and uses test and
set sb interfaces to allow remounting /dev/pts.

Changelog [v4]:
 - Split-off the simpler changes of moving global=variables into
   'pts_namespace' to previous patch.

Changelog [v3]:
 - Removed some unnecessary comments from devpts_set_sb()

Changelog [v2]:

 - (Pavel Emelianov/Serge Hallyn) Remove reference to pts_ns from
   sb->s_fs_info to fix the circular reference (/dev/pts is not
   unmounted unless the pts_ns is destroyed, so we don't need a
   reference to the pts_ns).

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>
---
 fs/devpts/inode.c | 62 ++---
 1 file changed, 59 insertions(+), 3 deletions(-)

Index: 2.6.25-rc8-mm1/fs/devpts/inode.c
===
--- 2.6.25-rc8-mm1.orig/fs/devpts/inode.c 2008-04-12 10:10:33.000000000 -0700
+++ 2.6.25-rc8-mm1/fs/devpts/inode.c 2008-04-12 10:10:38.000000000 -0700
@@ -154,17 +154,73 @@ fail:
  return -ENOMEM;
 }

+/*
+ * We use test and set super-block operations to help determine whether we
+ * need a new super-block for this namespace. get_sb() walks the list of
+ * existing devpts supers, comparing them with the @data ptr. Since we
+ * passed 'current's namespace as the @data pointer we can compare the
+ * namespace pointer in the super-block's 's_fs_info'. If the test is
+ * TRUE then get_sb() returns a new active reference to the super block.
+ * Otherwise, it helps us build an active reference to a new one.
+ */
+
+static int devpts_test_sb(struct super_block *sb, void *data)
+{
+ return sb->s_fs_info == data;
+}
+

```
+static int devpts_set_sb(struct super_block *sb, void *data)
+{
+ sb->s_fs_info = data;
+ return set_anon_super(sb, NULL);
+}
+
 static int devpts_get_sb(struct file_system_type *fs_type,
  int flags, const char *dev_name, void *data, struct vfsmount *mnt)
 {
- return get_sb_single(fs_type, flags, data, devpts_fill_super, mnt);
+ struct super_block *sb;
+ struct pts_namespace *ns;
+ int err;
+
+ /* hereafter we're very similar to proc_get_sb */
+ if (flags & MS_KERNMOUNT)
+  ns = data;
+ else
+  ns = &init_pts_ns;
+
+ /* hereafter we're very simlar to get_sb_nodev */
+ sb = sget(fs_type, devpts_test_sb, devpts_set_sb, ns);
+ if (IS_ERR(sb))
+  return PTR_ERR(sb);
+
+ if (sb->s_root)
+  return simple_set_mnt(mnt, sb);
+
+ sb->s_flags = flags;
+ err = devpts_fill_super(sb, data, flags & MS_SILENT ? 1 : 0);
+ if (err) {
+  up_write(&sb->s_umount);
+  deactivate_super(sb);
+  return err;
+ }
+
+ sb->s_flags |= MS_ACTIVE;
+ ns->mnt = mnt;
+
+ return simple_set_mnt(mnt, sb);
+}
+
+static void devpts_kill_sb(struct super_block *sb)
+{
+ sb->s_fs_info = NULL;
+ kill_anon_super(sb);
 }
```

```
 static struct file_system_type devpts_fs_type = {
  .owner  = THIS_MODULE,
  .name  = "devpts",
  .get_sb  = devpts_get_sb,
- .kill_sb = kill_anon_super,
+ .kill_sb = devpts_kill_sb,
 };

 /*
@@ -315,7 +371,7 @@ static int __init init_devpts_fs(void)

  err = register_filesystem(&devpts_fs_type);
  if (!err) {
-  mnt = kern_mount(&devpts_fs_type);
+  mnt = kern_mount_data(&devpts_fs_type, &init_pts_ns);
   if (IS_ERR(mnt))
    err = PTR_ERR(mnt);
   else
```

_____

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

---

Subject: Re: [PATCH 0/4] Helper patches for PTY namespaces
Posted by Subrata Modak on Sat, 12 Apr 2008 17:39:50 GMT
View Forum Message <> Reply to Message

Sukadev,

Any corresponding test cases for LTP. We just have UTS, PID & SYSVIPC
Namespace till now.

Regards--
Subrata

On Sat, Apr 12, 2008 at 10:59 PM, <sukadev@us.ibm.com> wrote:

> Some simple helper patches to enable implementation of multiple PTY
> (or device) namespaces.
>
> [PATCH 1/4]: Propagate error code from devpts_pty_new
> [PATCH 2/4]: Factor out PTY index allocation
> [PATCH 3/4]: Move devpts globals into init_pts_ns
> [PATCH 3/4]: Enable multiple mounts of /dev/pts
>
> This patchset is based on earlier versions developed by Serge Hallyn

> and Matt Helsley.
> --
> 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/
>

--
Regards & Thanks--
Subrata

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

---

Subject: Re: [PATCH 0/4] Helper patches for PTY namespaces
Posted by Sukadev Bhattiprolu on Sat, 12 Apr 2008 18:05:34 GMT
View Forum Message <> Reply to Message

Subrata Modak [tosubrata@gmail.com] wrote:
| Sukadev,
|
| Any corresponding test cases for LTP. We just have UTS, PID & SYSVIPC
| Namespace till now.

I had some unit-tests that I used with the clone-pts-ns patchset.
But the patches in this set are just helpers and should not change
existing functionality.

I can send the tests I used (they are not in LTP format) when I tested
the clone-pts-ns patchset.

Thanks,

Sukadev

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

---

Subject: Re: [PATCH 0/4] Helper patches for PTY namespaces
Posted by hpa on Sat, 12 Apr 2008 18:09:31 GMT

sukadev@us.ibm.com wrote:
> Some simple helper patches to enable implementation of multiple PTY
> (or device) namespaces.
>
> [PATCH 1/4]: Propagate error code from devpts_pty_new
> [PATCH 2/4]: Factor out PTY index allocation
> [PATCH 3/4]: Move devpts globals into init_pts_ns
> [PATCH 3/4]: Enable multiple mounts of /dev/pts
>
> This patchset is based on earlier versions developed by Serge Hallyn
> and Matt Helsley.

Any measurable performance impact when not using these kinds of namespaces?

 -hpa

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

---

Subject: Re: [PATCH 0/4] Helper patches for PTY namespaces
Posted by Al Viro on Sat, 12 Apr 2008 18:35:14 GMT

On Sat, Apr 12, 2008 at 10:29:33AM -0700, sukadev@us.ibm.com wrote:
>
> Some simple helper patches to enable implementation of multiple PTY
> (or device) namespaces.
>
> [PATCH 1/4]: Propagate error code from devpts_pty_new
> [PATCH 2/4]: Factor out PTY index allocation
> [PATCH 3/4]: Move devpts globals into init_pts_ns
> [PATCH 3/4]: Enable multiple mounts of /dev/pts
>
> This patchset is based on earlier versions developed by Serge Hallyn
> and Matt Helsley.

boggle

Care to explain how that "namespace" is different from devpts instance?
IOW, why the devil do you guys ignore Occam's Razor?

Frankly, this nonsense has gone far enough; I can buy the need to compensate
for shitty APIs (sockets, non-fs-based-IPC, etc.), but devpts *is* *a*
fucking *filesystem*. Already. And as such it's already present in

normal, real, we-really-shouldn't-have-any-other-if-not-for-ancient-stupidity
namespace.

Why not simply allow independent instances of devpts and be done with that?

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

---

Subject: Multiple instances of devpts
Posted by hpa on Sat, 12 Apr 2008 18:54:59 GMT
View Forum Message <> Reply to Message

Al Viro wrote:
>
> *boggle*
>
> Care to explain how that "namespace" is different from devpts instance?
> IOW, why the devil do you guys ignore Occam's Razor?
>
> Frankly, this nonsense has gone far enough; I can buy the need to compensate
> for shitty APIs (sockets, non-fs-based-IPC, etc.), but devpts *is* *a*
> *fucking* *filesystem*. Already. And as such it's already present in
> normal, real, we-really-shouldn't-have-any-other-if-not-for-ancient-stupidity
> namespace.
>
> Why not simply allow independent instances of devpts and be done with that?

In particular:

/dev/ptmx can be a symlink ptmx -> pts/ptmx, and we add a ptmx instance
inside the devpts filesystem. Each devpts filesystem is responsible for
its own pool of ptys, with own numbering, etc.

This does mean that entries in /dev/pts are more than just plain device
nodes, which they are now (you can cp -a a device node from /dev/pts
into another filesystem and it will still "just work"), but I doubt this
actually matters to anyone. If anyone cares, now I guess would be a
good time to speak up.

 -hpa

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

---

Subject: Re: [PATCH 0/4] Helper patches for PTY namespaces
Posted by ebiederm on Sat, 12 Apr 2008 19:06:55 GMT
View Forum Message <> Reply to Message

On Sat, 2008-04-12 at 10:29 -0700, sukadev@us.ibm.com wrote:
> Some simple helper patches to enable implementation of multiple PTY
> (or device) namespaces.
>
> [PATCH 1/4]: Propagate error code from devpts_pty_new
> [PATCH 2/4]: Factor out PTY index allocation
> [PATCH 3/4]: Move devpts globals into init_pts_ns
> [PATCH 3/4]: Enable multiple mounts of /dev/pts
>
> This patchset is based on earlier versions developed by Serge Hallyn
> and Matt Helsley.

Suka Stop.

The first two patches appear to just be cleanups and as such should be able to stand on their own. Mentioning that you found these opportunities while working on your pts namespace is fine. Justifying the cleanups this way is not.

When you mentioned you intended to just resend the cleanups this is not at all what I thought you were about. So please just get the first two patches in a form that stands by themselves.

The pts namespace as designed is not acceptable.

The problem you are trying to solve with the pts namespace is real.

So what we need is a device namespace and possibly and incremental path to get there. A device namespace would abstract the device number to device mapping for all devices. A safe incremental path would disable all device number to device mappings for process in that namespace. So all functionality would be gone, then it would enable certain mappings and certain pieces of the functionality if the code in the kernel is not clean enough that we can do it all in one go.

We need to see that path. Only then can we take patches that add namespace specific goo. The pattern I am proposing has worked quite well for the network namespace. Meanwhile the uid namespace which follows the pattern you seem to be following now seems does not look to be completed any time soon.

So send your clean up patches and then let's architect this thing so we are really talking about a namespace. Then we can update devpts to capture the device namespace on mount and do it's work in a namespace specific manner.

Eric

_____

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

---

Subject: Re: Multiple instances of devpts
Posted by ebiederm on Sat, 12 Apr 2008 19:15:57 GMT
View Forum Message <> Reply to Message

On Sat, 2008-04-12 at 11:54 -0700, H. Peter Anvin wrote:
> Al Viro wrote:
> >
> > Why not simply allow independent instances of devpts and be done with that?
>
> In particular:
>
> /dev/ptmx can be a symlink ptmx -> pts/ptmx, and we add a ptmx instance
> inside the devpts filesystem. Each devpts filesystem is responsible for
> its own pool of ptys, with own numbering, etc.
>
> This does mean that entries in /dev/pts are more than just plain device
> nodes, which they are now (you can cp -a a device node from /dev/pts
> into another filesystem and it will still "just work"), but I doubt this
> actually matters to anyone. If anyone cares, now I guess would be a
> good time to speak up.

Agreed. That is another legitimate path. And if all you care about is isolation and not dealing with the general class of problems with the global device number to device mapping that is sane. I know we have several other virtual devices that we tend to care about but ptys are the real world pain point.

Further I don't see any conflict with the generalizing devpts in this manner (so you only see a subset of the ptys) and then later adding a namespace that deals with the whole device number to device mapping.

Eric

_____

Containers mailing list

Subject: Re: Multiple instances of devpts
Posted by hpa on Sat, 12 Apr 2008 19:24:32 GMT
View Forum Message <> Reply to Message

Eric W. Biederman wrote:
>>
>> /dev/ptmx can be a symlink ptmx -> pts/ptmx, and we add a ptmx instance
>> inside the devpts filesystem. Each devpts filesystem is responsible for
>> its own pool of ptys, with own numbering, etc.
>>
>> This does mean that entries in /dev/pts are more than just plain device
>> nodes, which they are now (you can cp -a a device node from /dev/pts
>> into another filesystem and it will still "just work"), but I doubt this
>> actually matters to anyone. If anyone cares, now I guess would be a
>> good time to speak up.
>
> Agreed. That is another legitimate path. And if all you care about is
> isolation and not dealing with the general class of problems with the
> global device number to device mapping that is sane. I know we have
> several other virtual devices that we tend to care about but ptys are
> the real world pain point.
>

Thinking about it further, allowing this restriction would also allow a
whole lot of cleanups inside the pty setup, since it would eliminate the
need to do a separate lookup to find the corresponding devpts entry in
pty_open(). The benefit here comes from the closer coupling between the
pty and the devpts filesystem and isn't at all related to namespaces,
but it's a very nice side benefit.

-hpa

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

Subject: Re: Multiple instances of devpts
Posted by hpa on Sat, 12 Apr 2008 19:30:05 GMT
View Forum Message <> Reply to Message

H. Peter Anvin wrote:
>

> Thinking about it further, allowing this restriction would also allow a
> whole lot of cleanups inside the pty setup, since it would eliminate the
> need to do a separate lookup to find the corresponding devpts entry in
> pty_open(). The benefit here comes from the closer coupling between the
> pty and the devpts filesystem and isn't at all related to namespaces,
> but it's a very nice side benefit.
>

Minor correction: the lookup is actually in init_dev() in tty_io.c; I'm
specifically referring to devpts_get_tty().

-hpa

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

---

Subject: Re: [PATCH 0/4] Helper patches for PTY namespaces
Posted by serue on Sun, 13 Apr 2008 00:59:36 GMT
View Forum Message <> Reply to Message

Quoting Eric W. Biederman (ebiederm@xmission.com):
> On Sat, 2008-04-12 at 10:29 -0700, sukadev@us.ibm.com wrote:
> > Some simple helper patches to enable implementation of multiple PTY
> > (or device) namespaces.
> >
> > [PATCH 1/4]: Propagate error code from devpts_pty_new
> > [PATCH 2/4]: Factor out PTY index allocation
> > [PATCH 3/4]: Move devpts globals into init_pts_ns
> > [PATCH 3/4]: Enable multiple mounts of /dev/pts
> >
> > This patchset is based on earlier versions developed by Serge Hallyn
> > and Matt Helsley.
>
> Suka Stop.
>
> The first two patches appear to just be cleanups and as such should be
> able to stand on their own. Mentioning that you found these
> opportunities while working on your pts namespace is fine. Justifying
> the cleanups this way is not.
>
> When you mentioned you intended to just resend the cleanups this is not
> at all what I thought you were about. So please just get the first two
> patches in a form that stands by themselves.

It took me a minute to figure out what you were offended by, but I see,
the patches still introduce a "pts ns", which shouldn't exist at all.

> The pts namespace as designed is not acceptable.
>
> The problem you are trying to solve with the pts namespace is real.
>
> So what we need is a device namespace and possibly and incremental path
> to get there. A device namespace would abstract the device number to
> device mapping for all devices. A safe incremental path would disable
> all device number to device mappings for process in that namespace. So
> all functionality would be gone, then it would enable certain mappings
> and certain pieces of the functionality if the code in the kernel is not
> clean enough that we can do it all in one go.
>
> We need to see that path. Only then can we take patches that add
> namespace specific goo. The pattern I am proposing has worked quite
> well for the network namespace. Meanwhile the uid namespace which
> follows the pattern you seem to be following now seems does not look to
> be completed any time soon.

(you know perfectly well that we're holding off on any further real
uidns work until netns is complete and you have time to partake in the
design - it wouldn't be any fun without you :)

> So send your clean up patches and then let's architect this thing so we
> are really talking about a namespace. Then we can update devpts to
> capture the device namespace on mount and do it's work in a namespace
> specific manner.

Sounds reasonable to me.

thanks,
-serge

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

---

Subject: Re: [LTP] [PATCH 0/4] Helper patches for PTY namespaces
Posted by Subrata Modak on Mon, 14 Apr 2008 11:20:06 GMT
View Forum Message <> Reply to Message

On Sat, 2008-04-12 at 11:05 -0700, sukadev@us.ibm.com wrote:
> Subrata Modak [tosubrata@gmail.com] wrote:
> | Sukadev,
> |
> | Any corresponding test cases for LTP. We just have UTS, PID & SYSVIPC
> | Namespace till now.

>
> I had some unit-tests that I used with the clone-pts-ns patchset.
> But the patches in this set are just helpers and should not change
> existing functionality.
>
> I can send the tests I used (they are not in LTP format) when I tested
> the clone-pts-ns patchset.

Looping in Veerandra. He may be able to help us here.

Regards--
Subrata

>
> Thanks,
>
> Sukadev
>
> ---
> This SF.net email is sponsored by the 2008 JavaOne(SM) Conference
> Don't miss this year's exciting event. There's still time to save $100.
> Use priority code J8TL2D2.
> http://ad.doubleclick.net/clk;198757673;13503038;p?http://java.sun.com/javaone
> _____
> Ltp-list mailing list
> Ltp-list@lists.sourceforge.net
> https://lists.sourceforge.net/lists/listinfo/ltp-list

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

---

Subject: Re: [LTP] [PATCH 0/4] Helper patches for PTY namespaces
Posted by Veerendra Chandrappa on Mon, 14 Apr 2008 12:31:11 GMT
View Forum Message <> Reply to Message

Hi Sukadev,

That sounds nice, I will integrate them with the LTP.
Of course it depends on the order of priority, and as I am learning the
tricks of container/ltp .

Regards, Veerendra C

_____
Linux Technology Center, India Software Labs, Bangalore, Ph: 080-4177 6428

Subrata Modak <subrata@linux.vnet.ibm.com>	To	sukadev@us.ibm.com, Veerendra Chandrappa/India/IBM@IBMIN
04/14/2008 04:50 PM	cc	Subrata Modak <tosubrata@gmail.com>, Andrew Morton <akpm@osdl.org>, ltp-list <ltp-list@lists.sourceforge.net>, "Eric W. Biederman" <ebiederm@xmission.com>, Rishikesh K Rajak/India/IBM@IBMIN, matthltc@us.ibm.com, hpa@zytor.com, Containers <containers@lists.osdl.org>, Pavel Emelyanov <xemul@openvz.org>
Please respond to subrata@linux.vnet.ibm.com	Subject	Re: [LTP] [PATCH 0/4] Helper patches for PTY namespaces

On Sat, 2008-04-12 at 11:05 -0700, sukadev@us.ibm.com wrote:
> Subrata Modak [tosubrata@gmail.com] wrote:
> | Sukadev,
> |
> | Any corresponding test cases for LTP. We just have UTS, PID & SYSVIPC
> | Namespace till now.
>
> I had some unit-tests that I used with the clone-pts-ns patchset.
> But the patches in this set are just helpers and should not change
> existing functionality.
>
> I can send the tests I used (they are not in LTP format) when I tested
> the clone-pts-ns patchset.

Looping in Veerandra. He may be able to help us here.

Regards--
Subrata

>
> Thanks,
>
> Sukadev
>
> ---
> This SF.net email is sponsored by the 2008 JavaOne(SM) Conference
> Don't miss this year's exciting event. There's still time to save $100.
> Use priority code J8TL2D2.
>
http://ad.doubleclick.net/clk;198757673;13503038;p?http://java.sun.com/javaone

> _____
> Ltp-list mailing list
> Ltp-list@lists.sourceforge.net
> https://lists.sourceforge.net/lists/listinfo/ltp-list

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