Subject: [PATCH 3/20] Introduce MS_KERNMOUNT flag
Posted by Pavel Emelianov on Fri, 10 Aug 2007 11:47:55 GMT
View Forum Message <> Reply to Message

This flag tells the .get_sb callback that this is a kern_mount() call
so that it can trust *data pointer to be valid in-kernel one. If this
flag is passed from the user process, it is cleared since the *data
pointer is not a valid kernel object.

Running a few steps forward - this will be needed for proc to create the
superblock and store a valid pid namespace on it during the namespace
creation. The reason, why the namespace cannot live without proc mount
is described in the appropriate patch.

Signed-off-by: Pavel Emelyanov <xemul@openvz.org>
Cc: Oleg Nesterov <oleg@tv-sign.ru>

---

 fs/namespace.c | 3 ++-
 fs/super.c | 6 +++---
 include/linux/fs.h | 4 +++-
 3 files changed, 8 insertions(+), 5 deletions(-)

```
diff -upr linux-2.6.23-rc1-mm1.orig/fs/namespace.c linux-2.6.23-rc1-mm1-7/fs/namespace.c
--- linux-2.6.23-rc1-mm1.orig/fs/namespace.c 2007-07-26 16:34:45.000000000 +0400
+++ linux-2.6.23-rc1-mm1-7/fs/namespace.c 2007-07-26 16:36:36.000000000 +0400
@@ -1579,7 +1579,8 @@ long do_mount(char *dev_name, char *dir_
   mnt_flags |= MNT_NOMNT;

  flags &= ~(MS_NOSUID | MS_NOEXEC | MS_NODEV | MS_ACTIVE |
-     MS_NOATIME | MS_NODIRATIME | MS_RELATIME | MS_NOMNT);
+     MS_NOATIME | MS_NODIRATIME | MS_RELATIME |
+     MS_NOMNT | MS_KERNMOUNT);

  /* ... and get the mountpoint */
  retval = path_lookup(dir_name, LOOKUP_FOLLOW, &nd);
diff -upr linux-2.6.23-rc1-mm1.orig/fs/super.c linux-2.6.23-rc1-mm1-7/fs/super.c
--- linux-2.6.23-rc1-mm1.orig/fs/super.c 2007-07-26 16:34:45.000000000 +0400
+++ linux-2.6.23-rc1-mm1-7/fs/super.c 2007-07-26 16:36:36.000000000 +0400
@@ -944,9 +944,9 @@ do_kern_mount(const char *fstype, int fl
  return mnt;
 }

-struct vfsmount *kern_mount(struct file_system_type *type)
+struct vfsmount *kern_mount_data(struct file_system_type *type, void *data)
 {
- return vfs_kern_mount(type, 0, type->name, NULL);
```

```
+ return vfs_kern_mount(type, MS_KERNMOUNT, type->name, data);
 }

-EXPORT_SYMBOL(kern_mount);
+EXPORT_SYMBOL_GPL(kern_mount_data);
diff -upr linux-2.6.23-rc1-mm1.orig/include/linux/fs.h linux-2.6.23-rc1-mm1-7/include/linux/fs.h
--- linux-2.6.23-rc1-mm1.orig/include/linux/fs.h 2007-07-26 16:34:45.000000000 +0400
+++ linux-2.6.23-rc1-mm1-7/include/linux/fs.h 2007-07-26 16:36:36.000000000 +0400
@@ -129,6 +129,7 @@ extern int dir_notify_enable;
 #define MS_RELATIME (1<<21) /* Update atime relative to mtime/ctime. */
 #define MS_SETUSER (1<<23) /* set mnt_uid to current user */
 #define MS_NOMNT (1<<24) /* don't allow unprivileged submounts */
+#define MS_KERNMOUNT (1<<25) /* this is a kern_mount call */
 #define MS_ACTIVE (1<<30)
 #define MS_NOUSER (1<<31)

@@ -1459,7 +1460,8 @@ void unnamed_dev_init(void);

 extern int register_filesystem(struct file_system_type *);
 extern int unregister_filesystem(struct file_system_type *);
-extern struct vfsmount *kern_mount(struct file_system_type *);
+extern struct vfsmount *kern_mount_data(struct file_system_type *, void *data);
+#define kern_mount(type) kern_mount_data(type, NULL)
 extern int may_umount_tree(struct vfsmount *);
 extern int may_umount(struct vfsmount *);
 extern void umount_tree(struct vfsmount *, int, struct list_head *);
```