Subject: [patch -mm 3/4] mqueue namespace : enable the mqueue namespace
Posted by Cedric Le Goater on Wed, 28 Nov 2007 16:37:31 GMT
View Forum Message <> Reply to Message

From: Cedric Le Goater <clg@fr.ibm.com>

Move forward and start using the mqueue namespace.

The single super block mount of the file system is modified to allow
one mount per namespace. This is achieved by storing the namespace
in the super_block s_fs_info attribute.

Signed-off-by: Cedric Le Goater <clg@fr.ibm.com>
---
 include/linux/mq_namespace.h | 2 ++
 ipc/mq_namespace.c | 8 +++++++-
 ipc/mqueue.c | 42 ++++++++++++++++++++++++++++++++++++------
 3 files changed, 44 insertions(+), 8 deletions(-)

Index: 2.6.24-rc3-mm2/ipc/mqueue.c
===
--- 2.6.24-rc3-mm2.orig/ipc/mqueue.c
+++ 2.6.24-rc3-mm2/ipc/mqueue.c
@@ -32,6 +32,7 @@
 #include <linux/nsproxy.h>
 #include <linux/pid.h>
 #include <linux/mq_namespace.h>
+#include <linux/parser.h>

 #include <net/sock.h>
 #include "util.h"
@@ -200,11 +201,38 @@ static int mqueue_fill_super(struct supe
  return 0;
 }

+static int compare_data(struct super_block *sb, void *data)
+{
+ return sb->s_fs_info == data;
+}
+
 static int mqueue_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, mqueue_fill_super, mnt);
+ struct super_block *s;
+ int error;
+ struct mq_namespace *mq_ns = current->nsproxy->mq_ns;

```
+
+ if (flags & MS_KERNMOUNT)
+  mq_ns = (struct mq_namespace *) data;
+
+ s = sget(fs_type, compare_data, set_anon_super, mq_ns);
+ if (IS_ERR(s))
+  return PTR_ERR(s);
+ if (!s->s_root) {
+  s->s_flags = flags;
+  s->s_fs_info = mq_ns;
+  error = mqueue_fill_super(s, data, flags & MS_SILENT ? 1 : 0);
+  if (error) {
+   up_write(&s->s_umount);
+   deactivate_super(s);
+   return error;
+  }
+  s->s_flags |= MS_ACTIVE;
+ }
+ do_remount_sb(s, flags, data, 0);
+ return simple_set_mnt(mnt, s);
 }

 static void init_once(struct kmem_cache *cachep, void *foo)
@@ -235,7 +263,7 @@ static void mqueue_delete_inode(struct i
  struct user_struct *user;
  unsigned long mq_bytes;
  int i;
- struct mq_namespace *mq_ns = &init_mq_ns;
+ struct mq_namespace *mq_ns = inode->i_sb->s_fs_info;

  if (S_ISDIR(inode->i_mode)) {
   clear_inode(inode);
@@ -268,7 +296,7 @@ static int mqueue_create(struct inode *d
  struct inode *inode;
  struct mq_attr *attr = dentry->d_fsdata;
  int error;
- struct mq_namespace *mq_ns = &init_mq_ns;
+ struct mq_namespace *mq_ns = dir->i_sb->s_fs_info;

  spin_lock(&mq_lock);
  if (mq_ns->queues_count >= mq_ns->queues_max &&
@@ -659,7 +687,7 @@ asmlinkage long sys_mq_open(const char _
  struct file *filp;
  char *name;
  int fd, error;
- struct mq_namespace *mq_ns = &init_mq_ns;
+ struct mq_namespace *mq_ns = current->nsproxy->mq_ns;
```

```
  error = audit_mq_open(oflag, mode, u_attr);
  if (error != 0)
@@ -731,7 +759,7 @@ asmlinkage long sys_mq_unlink(const char
  char *name;
  struct dentry *dentry;
  struct inode *inode = NULL;
- struct mq_namespace *mq_ns = &init_mq_ns;
+ struct mq_namespace *mq_ns = current->nsproxy->mq_ns;

  name = getname(u_name);
  if (IS_ERR(name))
@@ -1196,7 +1224,7 @@ static struct super_operations mqueue_su
  .drop_inode = generic_delete_inode,
 };

-static struct file_system_type mqueue_fs_type = {
+struct file_system_type mqueue_fs_type = {
  .name = "mqueue",
  .get_sb = mqueue_get_sb,
  .kill_sb = kill_litter_super,
@@ -1273,7 +1301,7 @@ static int __init init_mqueue_fs(void)
  if (error)
   goto out_sysctl;

- init_mq_ns.mnt = kern_mount(&mqueue_fs_type);
+ init_mq_ns.mnt = kern_mount_data(&mqueue_fs_type, &init_mq_ns);
  if (IS_ERR(init_mq_ns.mnt)) {
   error = PTR_ERR(init_mq_ns.mnt);
   goto out_filesystem;
Index: 2.6.24-rc3-mm2/include/linux/mq_namespace.h
===================================================================
--- 2.6.24-rc3-mm2.orig/include/linux/mq_namespace.h
+++ 2.6.24-rc3-mm2/include/linux/mq_namespace.h
@@ -3,6 +3,7 @@

 #include <linux/kref.h>
 #include <linux/err.h>
+#include <linux/fs.h>

 struct vfsmount;

@@ -17,6 +18,7 @@ struct mq_namespace {
 };

 extern struct mq_namespace init_mq_ns;
+extern struct file_system_type mqueue_fs_type;

 /* default values */
```

```
 #define DFLT_QUEUESMAX 256 /* max number of message queues */
Index: 2.6.24-rc3-mm2/ipc/mq_namespace.c
===================================================================
--- 2.6.24-rc3-mm2.orig/ipc/mq_namespace.c
+++ 2.6.24-rc3-mm2/ipc/mq_namespace.c
@@ -27,7 +27,12 @@ static struct mq_namespace *clone_mq_ns(
  mq_ns->queues_max = DFLT_QUEUESMAX;
  mq_ns->msg_max  = DFLT_MSGMAX;
  mq_ns->msgsize_max = DFLT_MSGSIZEMAX;
- mq_ns->mnt  = NULL;
+ mq_ns->mnt  = kern_mount_data(&mqueue_fs_type, mq_ns);
+ if (IS_ERR(mq_ns->mnt)) {
+  void *error = mq_ns->mnt;
+  kfree(mq_ns);
+  return error;
+ }
  return mq_ns;
 }

@@ -53,5 +58,6 @@ void free_mq_ns(struct kref *kref)
  struct mq_namespace *ns;

  ns = container_of(kref, struct mq_namespace, kref);
+ mntput(ns->mnt);
  kfree(ns);
 }

--
```