Subject: [PATCH v3, RESEND 09/16] sunrpc: introduce rpc_pipefs_add_destroy_cb()
Posted by Kirill A. Shutemov on Tue, 08 Feb 2011 18:42:00 GMT
View Forum Message <> Reply to Message

Add facility to do some action on destroying of rpc_pipefs superblock.

Signed-off-by: Kirill A. Shutemov <kas@openvz.org>
Reviewed-by: Rob Landley <rlandley@parallels.com>
---
include/linux/sunrpc/rpc_pipe_fs.h | 3 ++
net/sunrpc/rpc_pipe.c | 51 ++-
2 files changed, 52 insertions(+), 2 deletions(-)

```
diff --git a/include/linux/sunrpc/rpc_pipe_fs.h b/include/linux/sunrpc/rpc_pipe_fs.h
index b09bfa5..f5216f1 100644
--- a/include/linux/sunrpc/rpc_pipe_fs.h
+++ b/include/linux/sunrpc/rpc_pipe_fs.h
@@ -46,6 +46,9 @@ RPC_I(struct inode *inode)

 extern struct vfsmount *init_rpc_pipefs;

+extern int rpc_pipefs_add_destroy_cb(struct super_block *sb,
+  void (*destroy_cb)(void *data), void *data);
+
 extern int rpc_queue_upcall(struct inode *, struct rpc_pipe_msg *);

 struct rpc_clnt;
diff --git a/net/sunrpc/rpc_pipe.c b/net/sunrpc/rpc_pipe.c
index 484c9a3..58312fa 100644
--- a/net/sunrpc/rpc_pipe.c
+++ b/net/sunrpc/rpc_pipe.c
@@ -939,6 +939,31 @@ static const struct super_operations s_ops = {

 #define RPCAUTH_GSSMAGIC 0x67596969

+struct destroy_cb {
+ struct list_head list;
+ void (*callback)(void *data);
+ void *data;
+};
+
+int rpc_pipefs_add_destroy_cb(struct super_block *sb,
+  void (*destroy_cb)(void *data), void *data)
+{
+ struct destroy_cb *dcb;
+ struct list_head *destroy_cb_list = sb->s_fs_info;
+
```

```
+ dcb = kmalloc(sizeof(*dcb), GFP_KERNEL);
+ if (!dcb)
+  return -ENOMEM;
+
+ dcb->callback = destroy_cb;
+ dcb->data = data;
+ INIT_LIST_HEAD(&dcb->list);
+ list_add(&dcb->list, destroy_cb_list);
+
+ return 0;
+}
+EXPORT_SYMBOL_GPL(rpc_pipefs_add_destroy_cb);
+
 /*
  * We have a single directory with 1 node in it.
  */
@@ -1004,8 +1029,16 @@ rpc_fill_super(struct super_block *sb, void *data, int silent)
   iput(inode);
   return -ENOMEM;
  }
- if (rpc_populate(root, files, RPCAUTH_lockd, RPCAUTH_RootEOF, NULL))
+ /* List of destroy callbacks */
+ sb->s_fs_info = kmalloc(sizeof(struct list_head), GFP_KERNEL);
+ if (!sb->s_fs_info)
+  return -ENOMEM;
+ INIT_LIST_HEAD((struct list_head*) sb->s_fs_info);
+ if (rpc_populate(root, files, RPCAUTH_lockd, RPCAUTH_RootEOF, NULL)) {
+  kfree(sb->s_fs_info);
   return -ENOMEM;
+ }
+
  return 0;
 }

@@ -1016,11 +1049,25 @@ rpc_mount(struct file_system_type *fs_type,
  return mount_single(fs_type, flags, data, rpc_fill_super);
 }

+static void rpc_kill_sb(struct super_block *sb)
+{
+ struct list_head *destroy_cb_list = sb->s_fs_info;
+ struct destroy_cb *dcb, *tmp;
+
+ list_for_each_entry_safe(dcb, tmp, destroy_cb_list, list) {
+  dcb->callback(dcb->data);
+  list_del(&dcb->list);
+  kfree(dcb);
+ }
```

```
+ kfree(destroy_cb_list);
+ kill_litter_super(sb);
+}
+
 static struct file_system_type rpc_pipe_fs_type = {
  .owner  = THIS_MODULE,
  .name  = "rpc_pipefs",
  .mount  = rpc_mount,
- .kill_sb = kill_litter_super,
+ .kill_sb = rpc_kill_sb,
 };

 static void
--
1.7.4
```

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