Subject: [RFC PATCH 2/5] SUNRPC: send notification events on pipefs sb creation and destruction
Posted by Stanislav Kinsbursky on Mon, 17 Oct 2011 12:12:33 GMT
View Forum Message <> Reply to Message

It will be used to notify subscribers about pipefs superblock creation and
destruction.
Cubcriders have to create their dentries on passed superblock on mount event
and destroy otherwise.

Signed-off-by: Stanislav Kinsbursky <skinsbursky@parallels.com>

---
 include/linux/sunrpc/rpc_pipe_fs.h | 8 ++++++++
 net/sunrpc/rpc_pipe.c | 27 +++++++++++++++++++++++++++
 2 files changed, 35 insertions(+), 0 deletions(-)

```
diff --git a/include/linux/sunrpc/rpc_pipe_fs.h b/include/linux/sunrpc/rpc_pipe_fs.h
index cf14db9..c1cdb2f 100644
--- a/include/linux/sunrpc/rpc_pipe_fs.h
+++ b/include/linux/sunrpc/rpc_pipe_fs.h
@@ -44,6 +44,14 @@ RPC_I(struct inode *inode)
  return container_of(inode, struct rpc_inode, vfs_inode);
 }

+extern int rpc_pipefs_notifier_register(struct notifier_block *);
+extern void rpc_pipefs_notifier_unregister(struct notifier_block *);
+
+enum {
+ RPC_PIPEFS_MOUNT,
+ RPC_PIPEFS_UMOUNT,
+};
+
 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 8322080..c90a7e0 100644
--- a/net/sunrpc/rpc_pipe.c
+++ b/net/sunrpc/rpc_pipe.c
@@ -28,6 +28,7 @@
 #include <linux/sunrpc/rpc_pipe_fs.h>
 #include <linux/sunrpc/cache.h>
 #include <linux/nsproxy.h>
+#include <linux/notifier.h>

 #include "netns.h"
```

```
@@ -41,6 +42,18 @@ static struct kmem_cache *rpc_inode_cachep __read_mostly;

 #define RPC_UPCALL_TIMEOUT (30*HZ)

+static BLOCKING_NOTIFIER_HEAD(rpc_pipefs_notifier_list);
+
+int rpc_pipefs_notifier_register(struct notifier_block *nb)
+{
+ return  blocking_notifier_chain_cond_register(&rpc_pipefs_notifi er_list, nb);
+}
+
+void rpc_pipefs_notifier_unregister(struct notifier_block *nb)
+{
+  blocking_notifier_chain_unregister(&rpc_pipefs_notifier_ list, nb);
+}
+
 static void rpc_purge_list(struct rpc_inode *rpci, struct list_head *head,
   void (*destroy_msg)(struct rpc_pipe_msg *), int err)
 {
@@ -1010,6 +1023,7 @@ rpc_fill_super(struct super_block *sb, void *data, int silent)
  struct inode *inode;
  struct dentry *root;
  struct net *net = current->nsproxy->net_ns;
+ int err;

  sb->s_blocksize = PAGE_CACHE_SIZE;
  sb->s_blocksize_bits = PAGE_CACHE_SHIFT;
@@ -1027,8 +1041,18 @@ rpc_fill_super(struct super_block *sb, void *data, int silent)
  }
  if (rpc_populate(root, files, RPCAUTH_lockd, RPCAUTH_RootEOF, NULL))
   return -ENOMEM;
+ err = blocking_notifier_call_chain(&rpc_pipefs_notifier_list,
+       RPC_PIPEFS_MOUNT,
+       sb);
+ if (err)
+  goto err_depopulate;
  sb->s_fs_info = get_net(net);
  return 0;
+
+err_depopulate:
+ /* TODO: check if we need to send RPC_PIPEFS_UMOUNT notification. */
+ __rpc_depopulate(root, files, RPCAUTH_lockd, RPCAUTH_RootEOF);
+ return err;
 }

 static struct dentry *
@@ -1043,6 +1067,9 @@ void rpc_kill_sb(struct super_block *sb)
  struct net *net = sb->s_fs_info;
```

```
  put_net(net);
+ blocking_notifier_call_chain(&rpc_pipefs_notifier_list,
+       RPC_PIPEFS_UMOUNT,
+       sb);
  kill_litter_super(sb);
 }
```