
Subject: [PATCH 4/6] SUNRPC: cleanup RPC PipeFS pipes upcall interface
Posted by [Stanislav Kinsbursky](#) on Tue, 22 Nov 2011 14:45:37 GMT
[View Forum Message](#) <> [Reply to Message](#)

RPC pipe upcall doesn't requires only private pipe data. Thus RPC inode references in this code can be removed.

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

```
fs/nfs/blocklayout/blocklayoutdev.c | 2 +-  
fs/nfs/blocklayout/blocklayoutdm.c | 2 +-  
fs/nfs/idmap.c                       | 4 ++--  
include/linux/sunrpc/rpc_pipe_fs.h | 2 +-  
net/sunrpc/auth_gss/auth_gss.c      | 3 ++-  
net/sunrpc/rpc_pipe.c               | 3 ++-  
6 files changed, 7 insertions(+), 9 deletions(-)
```

diff --git a/fs/nfs/blocklayout/blocklayoutdev.c b/fs/nfs/blocklayout/blocklayoutdev.c

index a83b393..44dc348 100644

--- a/fs/nfs/blocklayout/blocklayoutdev.c

+++ b/fs/nfs/blocklayout/blocklayoutdev.c

@@ -168,7 +168,7 @@ nfs4_blk_decode_device(struct nfs_server *server,

```
    dprintk("%s CALLING USERSPACE DAEMON\n", __func__);  
    add_wait_queue(&bl_wq, &wq);  
- if (rpc_queue_upcall(bl_device_pipe->d_inode, &msg) < 0) {  
+ if (rpc_queue_upcall(RPC_I(bl_device_pipe->d_inode)->pipe, &msg) < 0) {  
    remove_wait_queue(&bl_wq, &wq);  
    goto out;  
}
```

diff --git a/fs/nfs/blocklayout/blocklayoutdm.c b/fs/nfs/blocklayout/blocklayoutdm.c

index d055c75..3c38244 100644

--- a/fs/nfs/blocklayout/blocklayoutdm.c

+++ b/fs/nfs/blocklayout/blocklayoutdm.c

@@ -66,7 +66,7 @@ static void dev_remove(dev_t dev)

msg.len = sizeof(bl_msg) + bl_msg.totallen;

```
    add_wait_queue(&bl_wq, &wq);  
- if (rpc_queue_upcall(bl_device_pipe->d_inode, &msg) < 0) {  
+ if (rpc_queue_upcall(RPC_I(bl_device_pipe->d_inode)->pipe, &msg) < 0) {  
    remove_wait_queue(&bl_wq, &wq);  
    goto out;  
}
```

diff --git a/fs/nfs/idmap.c b/fs/nfs/idmap.c

index f20801a..7e3d8dd 100644

--- a/fs/nfs/idmap.c

+++ b/fs/nfs/idmap.c

```

@@ -508,7 +508,7 @@ nfs_idmap_id(struct idmap *idmap, struct idmap_hashtable *h,
    msg.len = sizeof(*im);

    add_wait_queue(&idmap->idmap_wq, &wq);
- if (rpc_queue_upcall(idmap->idmap_dentry->d_inode, &msg) < 0) {
+ if (rpc_queue_upcall(RPC_I(idmap->idmap_dentry->d_inode)->pipe, &msg) < 0) {
    remove_wait_queue(&idmap->idmap_wq, &wq);
    goto out;
}
@@ -569,7 +569,7 @@ nfs_idmap_name(struct idmap *idmap, struct idmap_hashtable *h,

    add_wait_queue(&idmap->idmap_wq, &wq);

- if (rpc_queue_upcall(idmap->idmap_dentry->d_inode, &msg) < 0) {
+ if (rpc_queue_upcall(RPC_I(idmap->idmap_dentry->d_inode)->pipe, &msg) < 0) {
    remove_wait_queue(&idmap->idmap_wq, &wq);
    goto out;
}
diff --git a/include/linux/sunrpc/rpc_pipe_fs.h b/include/linux/sunrpc/rpc_pipe_fs.h
index c2fa330..ad78bea 100644
--- a/include/linux/sunrpc/rpc_pipe_fs.h
+++ b/include/linux/sunrpc/rpc_pipe_fs.h
@@ -64,7 +64,7 @@ extern void rpc_put_sb_net(const struct net *net);

extern ssize_t rpc_pipe_generic_upcall(struct file *, struct rpc_pipe_msg *,
    char __user *, size_t);
-extern int rpc_queue_upcall(struct inode *, struct rpc_pipe_msg *);
+extern int rpc_queue_upcall(struct rpc_pipe *, struct rpc_pipe_msg *);

struct rpc_clnt;
extern struct dentry *rpc_create_client_dir(struct dentry *, struct qstr *, struct rpc_clnt *);
diff --git a/net/sunrpc/auth_gss/auth_gss.c b/net/sunrpc/auth_gss/auth_gss.c
index 70a7953..40227ef 100644
--- a/net/sunrpc/auth_gss/auth_gss.c
+++ b/net/sunrpc/auth_gss/auth_gss.c
@@ -475,8 +475,7 @@ gss_setup_upcall(struct rpc_clnt *clnt, struct gss_auth *gss_auth, struct
rpc_cr
    return gss_new;
    gss_msg = gss_add_msg(gss_new);
    if (gss_msg == gss_new) {
- struct inode *inode = &gss_new->inode->vfs_inode;
- int res = rpc_queue_upcall(inode, &gss_new->msg);
+ int res = rpc_queue_upcall(gss_new->inode->pipe, &gss_new->msg);
    if (res) {
        gss_unhash_msg(gss_new);
        gss_msg = ERR_PTR(res);
diff --git a/net/sunrpc/rpc_pipe.c b/net/sunrpc/rpc_pipe.c
index edf140a..0eed975 100644

```

```
--- a/net/sunrpc/rpc_pipe.c
+++ b/net/sunrpc/rpc_pipe.c
@@ -110,9 +110,8 @@ rpc_timeout_upcall_queue(struct work_struct *work)
 * initialize the fields of @msg (other than @msg->list) appropriately.
 */
int
-rpc_queue_upcall(struct inode *inode, struct rpc_pipe_msg *msg)
+rpc_queue_upcall(struct rpc_pipe *pipe, struct rpc_pipe_msg *msg)
{
- struct rpc_pipe *pipe = RPC_I(inode)->pipe;
  int res = -EPIPE;

  spin_lock(&pipe->lock);
```
