
Subject: [RFC PATCH 5/5] SUNRPC: pipefs per-net operations helper introduced
Posted by [Stanislav Kinsbursky](#) on Mon, 17 Oct 2011 12:12:30 GMT
[View Forum Message](#) <> [Reply to Message](#)

During per-net operation pipes creation and destruction we have to make sure, that pipefs sb exists for the whole creation/destruction procedure. This is done by using special mutex which controls pipefs sb reference on network namespace context.

Helper consist of two parts: first of them (rpc_get_dentry_net) searches for dentry with specified name and returns with mutex taken on success. When pipe creation is completed, caller should release this mutex by rpc_put_dentry_net call.

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

```
include/linux/sunrpc/rpc_pipe_fs.h | 5 +++++
net/sunrpc/netns.h                  | 1 +
net/sunrpc/rpc_pipe.c               | 41 ++++++++++++++++++++++++++++++++++++++
net/sunrpc/sunrpc_syms.c            | 1 +
4 files changed, 48 insertions(+), 0 deletions(-)
```

```
diff --git a/include/linux/sunrpc/rpc_pipe_fs.h b/include/linux/sunrpc/rpc_pipe_fs.h
index 4a327ad..d48ff0b 100644
```

```
--- a/include/linux/sunrpc/rpc_pipe_fs.h
+++ b/include/linux/sunrpc/rpc_pipe_fs.h
@@ -55,6 +55,11 @@ enum {
extern struct dentry *rpc_d_lookup_sb(const struct super_block *sb,
    const unsigned char *dir_name);
```

```
+extern void rpc_pipefs_init_net(struct net *net);
+extern struct dentry *rpc_get_dentry_net(const struct net *net,
+    const unsigned char *dir_name);
+extern void rpc_put_dentry_net(const struct net *net, struct dentry *dentry);
+
extern int rpc_queue_upcall(struct inode *, struct rpc_pipe_msg *);
```

```
struct rpc_clnt;
diff --git a/net/sunrpc/netns.h b/net/sunrpc/netns.h
index b384252..11d2f48 100644
```

```
--- a/net/sunrpc/netns.h
+++ b/net/sunrpc/netns.h
@@ -11,6 +11,7 @@ struct sunrpc_net {
    struct cache_detail *ip_map_cache;
```

```
    struct super_block *pipefs_sb;
+ struct mutex pipefs_sb_lock;
};
```

```

extern int sunrpc_net_id;
diff --git a/net/sunrpc/rpc_pipe.c b/net/sunrpc/rpc_pipe.c
index a873f03..331be28 100644
--- a/net/sunrpc/rpc_pipe.c
+++ b/net/sunrpc/rpc_pipe.c
@@ -1033,6 +1033,45 @@ struct dentry *rpc_d_lookup_sb(const struct super_block *sb,
 }
EXPORT_SYMBOL_GPL(rpc_d_lookup_sb);

+void rpc_pipefs_init_net(struct net *net)
+{
+ struct sunrpc_net *sn = net_generic(net, sunrpc_net_id);
+
+ mutex_init(&sn->pipefs_sb_lock);
+}
+
+/*
+ * This call will be used for per network namespace operations calls.
+ * Note: Function will be returned with pipefs_sb_lock taken if dentry was
+ * found. This lock have to be released by rpc_put_dentry_net() when all
+ * operations will be completed.
+ */
+struct dentry *rpc_get_dentry_net(const struct net *net,
+    const unsigned char *dir_name)
+{
+ struct sunrpc_net *sn = net_generic(net, sunrpc_net_id);
+ struct dentry *dentry = ERR_PTR(-ENOENT);
+
+ mutex_lock(&sn->pipefs_sb_lock);
+ if (sn->pipefs_sb) {
+ dentry = rpc_d_lookup_sb(sn->pipefs_sb, dir_name);
+ if (dentry)
+ return dentry;
+ }
+ mutex_unlock(&sn->pipefs_sb_lock);
+ return ERR_PTR(-ENOENT);
+}
+EXPORT_SYMBOL_GPL(rpc_get_dentry_net);
+
+void rpc_put_dentry_net(const struct net *net, struct dentry *dentry)
+{
+ struct sunrpc_net *sn = net_generic(net, sunrpc_net_id);
+
+ dput(dentry);
+ mutex_unlock(&sn->pipefs_sb_lock);
+}
+EXPORT_SYMBOL_GPL(rpc_put_dentry_net);

```

```

+
static int
rpc_fill_super(struct super_block *sb, void *data, int silent)
{
@@ -1085,7 +1124,9 @@ void rpc_kill_sb(struct super_block *sb)
    struct net *net = sb->s_fs_info;
    struct sunrpc_net *sn = net_generic(net, sunrpc_net_id);

+ mutex_lock(&sn->pipefs_sb_lock);
    sn->pipefs_sb = NULL;
+ mutex_unlock(&sn->pipefs_sb_lock);
    put_net(net);
    blocking_notifier_call_chain(&rpc_pipefs_notifier_list,
        RPC_PIPEFS_UMOUNT,
diff --git a/net/sunrpc/sunrpc_syms.c b/net/sunrpc/sunrpc_syms.c
index 9d08091..880de8b 100644
--- a/net/sunrpc/sunrpc_syms.c
+++ b/net/sunrpc/sunrpc_syms.c
@@ -38,6 +38,7 @@ static __net_init int sunrpc_init_net(struct net *net)
    if (err)
        goto err_ipmap;

+ rpc_pipefs_init_net(net);
    return 0;

err_ipmap:

```
