Subject: [PATCH v2 2/6] SUNRPC: handle GSS AUTH pipes by network namespace aware routines
Posted by Stanislav Kinsbursky on Wed, 11 Jan 2012 15:18:09 GMT
View Forum Message <> Reply to Message

This patch makes RPC GSS PipeFs pipes allocated in it's RPC client owner
network namespace context.
Pipes creation and destruction now done in separated functions, which takes
care about PipeFS superblock locking.

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

---
 net/sunrpc/auth_gss/auth_gss.c | 95 ++++++++++++++++++++++++++++++---------
 1 files changed, 73 insertions(+), 22 deletions(-)

diff --git a/net/sunrpc/auth_gss/auth_gss.c b/net/sunrpc/auth_gss/auth_gss.c
index 7544305..164193e 100644
--- a/net/sunrpc/auth_gss/auth_gss.c
+++ b/net/sunrpc/auth_gss/auth_gss.c
@@ -759,6 +759,73 @@ gss_pipe_destroy_msg(struct rpc_pipe_msg *msg)
  }
 }

+static void gss_pipes_dentries_destroy(struct rpc_auth *auth)
+{
+ struct gss_auth *gss_auth;
+
+ gss_auth = container_of(auth, struct gss_auth, rpc_auth);
+ rpc_unlink(gss_auth->pipe[0]->dentry);
+ rpc_unlink(gss_auth->pipe[1]->dentry);
+}
+
+static int gss_pipes_dentries_create(struct rpc_auth *auth)
+{
+ int err;
+ struct gss_auth *gss_auth;
+ struct rpc_clnt *clnt;
+
+ gss_auth = container_of(auth, struct gss_auth, rpc_auth);
+ clnt = gss_auth->client;
+
+ gss_auth->pipe[1]->dentry = rpc_mkpipe_dentry(clnt->cl_path.dentry,
+   "gssd",
+   clnt, gss_auth->pipe[1]);
+ if (IS_ERR(gss_auth->pipe[1]->dentry))
+ return PTR_ERR(gss_auth->pipe[1]->dentry);
+ gss_auth->pipe[0]->dentry = rpc_mkpipe_dentry(clnt->cl_path.dentry,

```
+           gss_auth->mech->gm_name,
+           clnt, gss_auth->pipe[0]);
+ if (IS_ERR(gss_auth->pipe[0]->dentry)) {
+  err = PTR_ERR(gss_auth->pipe[0]->dentry);
+  goto err_unlink_pipe_1;
+ }
+ return 0;
+
+err_unlink_pipe_1:
+ rpc_unlink(gss_auth->pipe[1]->dentry);
+ return err;
+}
+
+static void gss_pipes_dentries_destroy_net(struct rpc_clnt *clnt,
+        struct rpc_auth *auth)
+{
+ struct net *net = clnt->cl_xprt->xprt_net;
+ struct super_block *sb;
+
+ sb = rpc_get_sb_net(net);
+ if (sb) {
+  if (clnt->cl_path.dentry)
+   gss_pipes_dentries_destroy(auth);
+  rpc_put_sb_net(net);
+ }
+}
+
+static int gss_pipes_dentries_create_net(struct rpc_clnt *clnt,
+       struct rpc_auth *auth)
+{
+ struct net *net = clnt->cl_xprt->xprt_net;
+ struct super_block *sb;
+ int err = 0;
+
+ sb = rpc_get_sb_net(net);
+ if (sb) {
+  if (clnt->cl_path.dentry)
+   err = gss_pipes_dentries_create(auth);
+  rpc_put_sb_net(net);
+ }
+ return err;
+}
+
 /*
  * NOTE: we have the opportunity to use different
  * parameters based on the input flavor (which must be a pseudoflavor)
@@ -814,31 +881,16 @@ gss_create(struct rpc_clnt *clnt, rpc_authflavor_t flavor)
   err = PTR_ERR(gss_auth->pipe[0]);
```

```
   goto err_destroy_pipe_1;
  }
-
- gss_auth->pipe[1]->dentry = rpc_mkpipe_dentry(clnt->cl_path.dentry,
-           "gssd",
-           clnt, gss_auth->pipe[1]);
- if (IS_ERR(gss_auth->pipe[1]->dentry)) {
-  err = PTR_ERR(gss_auth->pipe[1]->dentry);
+ err = gss_pipes_dentries_create_net(clnt, auth);
+ if (err)
   goto err_destroy_pipe_0;
- }
-
- gss_auth->pipe[0]->dentry = rpc_mkpipe_dentry(clnt->cl_path.dentry,
-           gss_auth->mech->gm_name,
-           clnt, gss_auth->pipe[0]);
- if (IS_ERR(gss_auth->pipe[0]->dentry)) {
-  err = PTR_ERR(gss_auth->pipe[0]->dentry);
-  goto err_unlink_pipe_1;
- }
  err = rpcauth_init_credcache(auth);
  if (err)
-  goto err_unlink_pipe_0;
+  goto err_unlink_pipes;

  return auth;
-err_unlink_pipe_0:
- rpc_unlink(gss_auth->pipe[0]->dentry);
-err_unlink_pipe_1:
- rpc_unlink(gss_auth->pipe[1]->dentry);
+err_unlink_pipes:
+ gss_pipes_dentries_destroy_net(clnt, auth);
 err_destroy_pipe_0:
  rpc_destroy_pipe_data(gss_auth->pipe[0]);
 err_destroy_pipe_1:
@@ -855,8 +907,7 @@ out_dec:
 static void
 gss_free(struct gss_auth *gss_auth)
 {
- rpc_unlink(gss_auth->pipe[0]->dentry);
- rpc_unlink(gss_auth->pipe[1]->dentry);
+ gss_pipes_dentries_destroy_net(gss_auth->client, &gss_auth->rpc_auth);
  rpc_destroy_pipe_data(gss_auth->pipe[0]);
  rpc_destroy_pipe_data(gss_auth->pipe[1]);
  gss_mech_put(gss_auth->mech);
```