
Subject: [PATCH v2 3/6] SUNRPC: subscribe RPC clients to pipefs notifications
Posted by [Stanislav Kinsbursky](#) on Wed, 11 Jan 2012 15:18:17 GMT
[View Forum Message](#) <> [Reply to Message](#)

This patch subscribes RPC clients to RPC pipefs notifications. RPC clients notifier block is registering with pipefs initialization during SUNRPC module init.

This notifier callback is responsible for RPC client PipeFS directory and GSS pipes creation. For pipes creation and destruction two additional callbacks were added to struct rpc_authops.

Note that no locking required in notifier callback because PipeFS superblock pointer is passed as an argument from it's creation or destruction routine and thus we can be sure about it's validity.

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

```
---
include/linux/sunrpc/auth.h | 2 +
net/sunrpc/auth_gss/auth_gss.c | 10 ++++--
net/sunrpc/clnt.c | 69 ++++++-----
net/sunrpc/rpc_pipe.c | 19 ++++++---
net/sunrpc/sunrpc.h | 2 +
5 files changed, 92 insertions(+), 10 deletions(-)

diff --git a/include/linux/sunrpc/auth.h b/include/linux/sunrpc/auth.h
index febc4db..83f493f 100644
--- a/include/linux/sunrpc/auth.h
+++ b/include/linux/sunrpc/auth.h
@@ -98,6 +98,8 @@ struct rpc_authops {

    struct rpc_cred * (*lookup_cred)(struct rpc_auth *, struct auth_cred *, int);
    struct rpc_cred * (*create)(struct rpc_auth *, struct auth_cred *, int);
+ int (*pipes_create)(struct rpc_auth *);
+ void (*pipes_destroy)(struct rpc_auth *);
};

struct rpc_credops {
diff --git a/net/sunrpc/auth_gss/auth_gss.c b/net/sunrpc/auth_gss/auth_gss.c
index 164193e..d7a5719 100644
--- a/net/sunrpc/auth_gss/auth_gss.c
+++ b/net/sunrpc/auth_gss/auth_gss.c
@@ -764,8 +764,10 @@ 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);
+ if (gss_auth->pipe[0]->dentry)
```

```

+ rpc_unlink(gss_auth->pipe[0]->dentry);
+ if (gss_auth->pipe[1]->dentry)
+ rpc_unlink(gss_auth->pipe[1]->dentry);
}

static int gss_pipes_dentries_create(struct rpc_auth *auth)
@@ -1608,7 +1610,9 @@ static const struct rpc_authops authgss_ops = {
    .create = gss_create,
    .destroy = gss_destroy,
    .lookup_cred = gss_lookup_cred,
- .crcreate = gss_create_cred
+ .crcreate = gss_create_cred,
+ .pipes_create = gss_pipes_dentries_create,
+ .pipes_destroy = gss_pipes_dentries_destroy,
};

static const struct rpc_credops gss_credops = {
diff --git a/net/sunrpc/clnt.c b/net/sunrpc/clnt.c
index 90e82c5..4170745 100644
--- a/net/sunrpc/clnt.c
+++ b/net/sunrpc/clnt.c
@@ -98,8 +98,11 @@ static void rpc_unregister_client(struct rpc_clnt *clnt)

static void __rpc_clnt_remove_pipedir(struct rpc_clnt *clnt)
{
- if (clnt->cl_path.dentry)
+ if (clnt->cl_path.dentry) {
+ if (clnt->cl_auth && clnt->cl_auth->au_ops->pipes_destroy)
+ clnt->cl_auth->au_ops->pipes_destroy(clnt->cl_auth);
    rpc_remove_client_dir(clnt->cl_path.dentry);
+ }
    clnt->cl_path.dentry = NULL;
}

@@ -181,6 +184,70 @@ rpc_setup_pipedir(struct rpc_clnt *clnt, char *dir_name)
    return 0;
}

+static int __rpc_pipefs_event(struct rpc_clnt *clnt, unsigned long event,
+ struct super_block *sb)
+{
+ struct dentry *dentry;
+ int err = 0;
+
+ switch (event) {
+ case RPC_PIPEFS_MOUNT:
+ if (clnt->cl_program->pipe_dir_name == NULL)
+ break;

```

```

+ dentry = rpc_setup_pipedir_sb(sb, clnt,
+   clnt->cl_program->pipe_dir_name);
+ BUG_ON(dentry == NULL);
+ if (IS_ERR(dentry))
+   return PTR_ERR(dentry);
+ clnt->cl_path.dentry = dentry;
+ if (clnt->cl_auth->au_ops->pipes_create) {
+   err = clnt->cl_auth->au_ops->pipes_create(clnt->cl_auth);
+   if (err)
+     __rpc_clnt_remove_pipedir(clnt);
+ }
+ break;
+ case RPC_PIPEFS_UMOUNT:
+   __rpc_clnt_remove_pipedir(clnt);
+   break;
+ default:
+   printk(KERN_ERR "%s: unknown event: %ld\n", __func__, event);
+   return -ENOTSUPP;
+ }
+ return err;
+}
+
+static int rpc_pipefs_event(struct notifier_block *nb, unsigned long event,
+   void *ptr)
+{
+   struct super_block *sb = ptr;
+   struct rpc_clnt *clnt;
+   int error = 0;
+   struct sunrpc_net *sn = net_generic(sb->s_fs_info, sunrpc_net_id);
+
+   spin_lock(&sn->rpc_client_lock);
+   list_for_each_entry(clnt, &sn->all_clients, cl_clients) {
+     error = __rpc_pipefs_event(clnt, event, sb);
+     if (error)
+       break;
+   }
+   spin_unlock(&sn->rpc_client_lock);
+   return error;
+}
+
+static struct notifier_block rpc_clients_block = {
+   .notifier_call = rpc_pipefs_event,
+};
+
+int rpc_clients_notifier_register(void)
+{
+   return rpc_pipefs_notifier_register(&rpc_clients_block);
+}

```

```

+
+void rpc_clients_notifier_unregister(void)
+{
+ return rpc_pipefs_notifier_unregister(&rpc_clients_block);
+}
+
static struct rpc_clnt * rpc_new_client(const struct rpc_create_args *args, struct rpc_xprt *xprt)
{
    struct rpc_program *program = args->program;
diff --git a/net/sunrpc/rpc_pipe.c b/net/sunrpc/rpc_pipe.c
index d3a0d26..8d74575 100644
--- a/net/sunrpc/rpc_pipe.c
+++ b/net/sunrpc/rpc_pipe.c
@@ -938,7 +938,7 @@ struct dentry *rpc_create_client_dir(struct dentry *dentry,

/**
 * rpc_remove_client_dir - Remove a directory created with rpc_create_client_dir()
- * @dentry: directory to remove
+ * @clnt: rpc client
 */
int rpc_remove_client_dir(struct dentry *dentry)
{
@@ -1189,17 +1189,24 @@ int register_rpc_pipefs(void)
    init_once);
    if (!rpc_inode_cachep)
        return -ENOMEM;
+ err = rpc_clients_notifier_register();
+ if (err)
+ goto err_notifier;
    err = register_filesystem(&rpc_pipe_fs_type);
- if (err) {
- kmem_cache_destroy(rpc_inode_cachep);
- return err;
- }
-
+ if (err)
+ goto err_register;
    return 0;
+
+err_register:
+ rpc_clients_notifier_unregister();
+err_notifier:
+ kmem_cache_destroy(rpc_inode_cachep);
+ return err;
}

void unregister_rpc_pipefs(void)
{

```

```
+ rpc_clients_notifier_unregister();
  kmem_cache_destroy(rpc_inode_cachep);
  unregister_filesystem(&rpc_pipe_fs_type);
}
diff --git a/net/sunrpc/sunrpc.h b/net/sunrpc/sunrpc.h
index 90c292e..14c9f6d 100644
--- a/net/sunrpc/sunrpc.h
+++ b/net/sunrpc/sunrpc.h
@@ -47,5 +47,7 @@ int svc_send_common(struct socket *sock, struct xdr_buf *xdr,
    struct page *headpage, unsigned long headoffset,
    struct page *tailpage, unsigned long tailoffset);

+int rpc_clients_notifier_register(void);
+void rpc_clients_notifier_unregister(void);
#endif /* _NET_SUNRPC_SUNRPC_H */
```
