
Subject: [PATCH 2/5] SUNRPC: create unix gid cache per network namespace

Posted by [Stanislav Kinsbursky](#) on Thu, 19 Jan 2012 14:49:14 GMT

[View Forum Message](#) <> [Reply to Message](#)

This patch makes unix_gid_cache cache detail allocated and registered per network namespace context.

Thus with this patch unix_gid_cache contents for network namespace "X" are controlled from proc file system mount for the same network namespace "X".

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

```
net/sunrpc/netns.h      | 1 +
net/sunrpc/sunrpc_syms.c | 14 ++++++-----
net/sunrpc/svcauth_unix.c | 36 ++++++-----
3 files changed, 42 insertions(+), 9 deletions(-)
```

diff --git a/net/sunrpc/netns.h b/net/sunrpc/netns.h

index 1fdeb1b..309f88d 100644

--- a/net/sunrpc/netns.h

+++ b/net/sunrpc/netns.h

@@ -9,6 +9,7 @@ struct cache_detail;

struct sunrpc_net {

struct proc_dir_entry *proc_net_rpc;

struct cache_detail *ip_map_cache;

+ struct cache_detail *unix_gid_cache;

struct super_block *pipefs_sb;

struct mutex pipefs_sb_lock;

diff --git a/net/sunrpc/sunrpc_syms.c b/net/sunrpc/sunrpc_syms.c

index b4217dc..38a72a1 100644

--- a/net/sunrpc/sunrpc_syms.c

+++ b/net/sunrpc/sunrpc_syms.c

@@ -26,6 +26,9 @@

int sunrpc_net_id;

+extern int unix_gid_cache_create(struct net *net);

+extern int unix_gid_cache_destroy(struct net *net);

+

static __net_init int sunrpc_init_net(struct net *net)

{

int err;

@@ -39,11 +42,17 @@ static __net_init int sunrpc_init_net(struct net *net)

if (err)

goto err_ipmap;

+ err = unix_gid_cache_create(net);

```

+ if (err)
+ goto err_unixgid;
+
+   rpc_pipefs_init_net(net);
+   INIT_LIST_HEAD(&sn->all_clients);
+   spin_lock_init(&sn->rpc_client_lock);
+   return 0;

+err_unixgid:
+ ip_map_cache_destroy(net);
err_ipmap:
+   rpc_proc_exit(net);
err_proc:
@@ -52,6 +61,7 @@ err_proc:

static __net_exit void sunrpc_exit_net(struct net *net)
{
+ unix_gid_cache_destroy(net);
+ ip_map_cache_destroy(net);
+   rpc_proc_exit(net);
+ }
@@ -63,8 +73,6 @@ static struct pernet_operations sunrpc_net_ops = {
+   .size = sizeof(struct sunrpc_net),
+ };

-extern struct cache_detail unix_gid_cache;
-
static int __init
init_sunrpc(void)
{
@@ -86,7 +94,6 @@ init_sunrpc(void)
#ifdef RPC_DEBUG
+   rpc_register_sysctl();
#endif
- cache_register(&unix_gid_cache);
+   svc_init_xprt_sock(); /* svc sock transport */
+   init_socket_xprt(); /* clnt sock transport */
+   return 0;
@@ -109,7 +116,6 @@ cleanup_sunrpc(void)
+   svc_cleanup_xprt_sock();
+   unregister_rpc_pipefs();
+   rpc_destroy_mempool();
- cache_unregister(&unix_gid_cache);
+   unregister_pernet_subsys(&sunrpc_net_ops);
#ifdef RPC_DEBUG
+   rpc_unregister_sysctl();
diff --git a/net/sunrpc/svcauth_unix.c b/net/sunrpc/svcauth_unix.c
index 2f8c426..c753ae4 100644

```

```

--- a/net/sunrpc/svcauth_unix.c
+++ b/net/sunrpc/svcauth_unix.c
@@ -436,7 +436,6 @@ struct unix_gid {
    uid_t uid;
    struct group_info *gi;
};
-static struct cache_head *gid_table[GID_HASHMAX];

static void unix_gid_put(struct kref *kref)
{
@@ -495,7 +494,6 @@ static int unix_gid_upcall(struct cache_detail *cd, struct cache_head *h)
}

static struct unix_gid *unix_gid_lookup(uid_t uid);
-extern struct cache_detail unix_gid_cache;

static int unix_gid_parse(struct cache_detail *cd,
    char *mesg, int mlen)
@@ -544,14 +542,14 @@ static int unix_gid_parse(struct cache_detail *cd,
    struct cache_head *ch;
    ug.h.flags = 0;
    ug.h.expiry_time = expiry;
- ch = sunrpc_cache_update(&unix_gid_cache,
+ ch = sunrpc_cache_update(cd,
    &ug.h, &ugp->h,
    hash_long(uid, GID_HASHBITS));
    if (!ch)
        err = -ENOMEM;
    else {
        err = 0;
- cache_put(ch, &unix_gid_cache);
+ cache_put(ch, cd);
    }
} else
    err = -ENOMEM;
@@ -590,7 +588,6 @@ static int unix_gid_show(struct seq_file *m,
struct cache_detail unix_gid_cache = {
    .owner = THIS_MODULE,
    .hash_size = GID_HASHMAX,
- .hash_table = gid_table,
    .name = "auth.unix.gid",
    .cache_put = unix_gid_put,
    .cache_upcall = unix_gid_upcall,
@@ -602,6 +599,35 @@ struct cache_detail unix_gid_cache = {
    .alloc = unix_gid_alloc,
};

+int unix_gid_cache_create(struct net *net)

```

```

+{
+ struct sunrpc_net *sn = net_generic(net, sunrpc_net_id);
+ struct cache_detail *cd;
+ int err;
+
+ cd = cache_create_net(&unix_gid_cache, net);
+ if (IS_ERR(cd))
+ return PTR_ERR(cd);
+ err = cache_register_net(cd, net);
+ if (err) {
+ cache_destroy_net(&unix_gid_cache, net);
+ return err;
+ }
+ sn->unix_gid_cache = cd;
+ return 0;
+}
+
+void unix_gid_cache_destroy(struct net *net)
+{
+ struct sunrpc_net *sn = net_generic(net, sunrpc_net_id);
+ struct cache_detail *cd = sn->unix_gid_cache;
+
+ sn->unix_gid_cache = NULL;
+ cache_purge(cd);
+ cache_unregister_net(cd, net);
+ cache_destroy_net(cd, net);
+}
+
+static struct unix_gid *unix_gid_lookup(uid_t uid)
+{
+ struct unix_gid ug;

```
