Subject: [PATCH 0/4] NFS: make internal list network namespace aware
Posted by Stanislav Kinsbursky on Mon, 23 Jan 2012 17:25:56 GMT
View Forum Message <> Reply to Message

This patch set moves static internal lists of NFS clients and NFS servers, as well as static IDR's structure and lock, protecting all that stuff, to network namespace context.

The following series consists of:

Stanislav Kinsbursky (4):
 NFS: make nfs_client_list per net ns
 NFS: make nfs_volume_list per net ns
 NFS: make cb_ident_idr per net ns
 NFS: make nfs_client_lock per net ns

```
 fs/nfs/callback_xdr.c |    2 -
 fs/nfs/client.c       |  137 ++++++++++++++++++++++++++++++++++------------------
 fs/nfs/idmap.c        |    9 +--
 fs/nfs/inode.c        |    3 +
 fs/nfs/internal.h     |    9 +--
 fs/nfs/netns.h        |    6 ++
 6 files changed, 104 insertions(+), 62 deletions(-)
```

Subject: [PATCH 2/4] NFS: make nfs_volume_list per net ns
Posted by Stanislav Kinsbursky on Mon, 23 Jan 2012 17:26:14 GMT
View Forum Message <> Reply to Message

This patch splits global list of NFS servers into per-net-ns array of lists.
This looks more strict and clearer.
BTW, this patch also makes "/proc/fs/nfsfs/volumes" content depends on /proc mount owner pid namespace. See below for details.

NOTE: few words about how was /proc/fs/nfsfs/ entries content show per network namespace done. This is a little bit tricky and not the best is could be. But it's cheap (proper fix for /proc conteinerization is a hard nut to crack).
The idea is simple: take proper network namespace from pid namespace child reaper nsproxy of /proc/ mount creator.
This actually means, that if there are 2 containers with different net namespace sharing pid namespace, then read of /proc/fs/nfsfs/ entries will always return content, taken from net namespace of pid namespace creator task (and thus second namespace set wil be unvisible).

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

```
---
 fs/nfs/client.c |   20 ++++++++++++++------
 fs/nfs/netns.h  |    1 +
 2 files changed, 15 insertions(+), 6 deletions(-)

diff --git a/fs/nfs/client.c b/fs/nfs/client.c
index 058eb9b..d58e838 100644
--- a/fs/nfs/client.c
+++ b/fs/nfs/client.c
@@ -56,7 +56,6 @@
 #define NFSDBG_FACILITY  NFSDBG_CLIENT

 DEFINE_SPINLOCK(nfs_client_lock);
-static LIST_HEAD(nfs_volume_list);
 static DECLARE_WAIT_QUEUE_HEAD(nfs_client_active_wq);
 #ifdef CONFIG_NFS_V4
 static DEFINE_IDR(cb_ident_idr); /* Protected by nfs_client_lock */
@@ -1036,10 +1035,11 @@ static void nfs_server_copy_userdata(struct nfs_server *target,
struct nfs_serve
 static void nfs_server_insert_lists(struct nfs_server *server)
 {
  struct nfs_client *clp = server->nfs_client;
+ struct nfs_net *nn = net_generic(clp->net, nfs_net_id);

  spin_lock(&nfs_client_lock);
  list_add_tail_rcu(&server->client_link, &clp->cl_superblocks);
- list_add_tail(&server->master_link, &nfs_volume_list);
+ list_add_tail(&server->master_link, &nn->nfs_volume_list);
  clear_bit(NFS_CS_STOP_RENEW, &clp->cl_res_state);
  spin_unlock(&nfs_client_lock);

@@ -1764,6 +1764,7 @@ void nfs_clients_init(struct net *net)
  struct nfs_net *nn = net_generic(net, nfs_net_id);

  INIT_LIST_HEAD(&nn->nfs_client_list);
+ INIT_LIST_HEAD(&nn->nfs_volume_list);
 }

 #ifdef CONFIG_PROC_FS
@@ -1900,13 +1901,15 @@ static int nfs_volume_list_open(struct inode *inode, struct file *file)
 {
  struct seq_file *m;
  int ret;
+ struct pid_namespace *pid_ns = file->f_dentry->d_sb->s_fs_info;
+ struct net *net = pid_ns->child_reaper->nsproxy->net_ns;

  ret = seq_open(file, &nfs_volume_list_ops);
```

```
  if (ret < 0)
   return ret;

  m = file->private_data;
- m->private = PDE(inode)->data;
+ m->private = net;

  return 0;
 }
@@ -1916,9 +1919,11 @@ static int nfs_volume_list_open(struct inode *inode, struct file *file)
  */
 static void *nfs_volume_list_start(struct seq_file *m, loff_t *_pos)
 {
+ struct nfs_net *nn = net_generic(m->private, nfs_net_id);
+
  /* lock the list against modification */
  spin_lock(&nfs_client_lock);
- return seq_list_start_head(&nfs_volume_list, *_pos);
+ return seq_list_start_head(&nn->nfs_volume_list, *_pos);
 }

 /*
@@ -1926,7 +1931,9 @@ static void *nfs_volume_list_start(struct seq_file *m, loff_t *_pos)
  */
 static void *nfs_volume_list_next(struct seq_file *p, void *v, loff_t *pos)
 {
- return seq_list_next(v, &nfs_volume_list, pos);
+ struct nfs_net *nn = net_generic(p->private, nfs_net_id);
+
+ return seq_list_next(v, &nn->nfs_volume_list, pos);
 }

 /*
@@ -1945,9 +1952,10 @@ static int nfs_volume_list_show(struct seq_file *m, void *v)
  struct nfs_server *server;
  struct nfs_client *clp;
  char dev[8], fsid[17];
+ struct nfs_net *nn = net_generic(m->private, nfs_net_id);

  /* display header on line 1 */
- if (v == &nfs_volume_list) {
+ if (v == &nn->nfs_volume_list) {
   seq_puts(m, "NV SERVER   PORT DEV     FSID              FSC\n");
   return 0;
  }
diff --git a/fs/nfs/netns.h b/fs/nfs/netns.h
index feb33c3..0fbd4e0 100644
--- a/fs/nfs/netns.h
```

```
+++ b/fs/nfs/netns.h
@@ -8,6 +8,7 @@ struct nfs_net {
  struct cache_detail *nfs_dns_resolve;
  struct rpc_pipe *bl_device_pipe;
  struct list_head nfs_client_list;
+ struct list_head nfs_volume_list;
 };

 extern int nfs_net_id;
```