Subject: [PATCH v3 00/16] make rpc_pipefs be mountable multiple time
Posted by Kirill A. Shutemov on Fri, 14 Jan 2011 13:48:58 GMT
View Forum Message <> Reply to Message

Prepare nfs/sunrpc stack to use multiple instances of rpc_pipefs.
Only for client for now.

It's step forwart to get nfs work from container.

Changelog:

v3:
 - rebase to the current Linus' tree (52cfd503ad)
 - rework get_rpc_pipefs() once again;
 - solve problem with rmmod sunrpc moudle;
 - free dns cache on killing rpc_pipefs superblock.

v2:
 - one of rpc_create() calls was missed initially, fixed;
 - change logic for get_rpc_pipefs(NULL);
 - export get_rpc_pipefs() to be able to use from modules (tnx J. Bruce Field);
 - change "From:" and "Signed-off-by:" addresses.

v1:
 - initial revision of the patchset.

Kirill A. Shutemov (16):
  sunrpc: mount rpc_pipefs on initialization
  sunrpc: introduce init_rpc_pipefs
  sunrpc: push init_rpc_pipefs up to rpc_create() callers
  sunrpc: tag svc_serv with rpc_pipefs mount point
  sunrpc: get rpc_pipefs mount point for svc_serv from callers
  lockd: get rpc_pipefs mount point from callers
  sunrpc: get rpc_pipefs mount point for rpcb_create[_local] from
    callers
  sunrpc: tag pipefs field of cache_detail with rpc_pipefs mount point
  sunrpc: introduce rpc_pipefs_add_destroy_cb()
  nfs: per-rpc_pipefs dns cache
  Export iterate_mounts symbol to be able to use from sunrpc module.
  sunrpc: introduce get_rpc_pipefs()
  nfs: introduce mount option 'rpcmount'
  sunrpc: make rpc_pipefs be mountable multiple times
  sunrpc: remove global init_rpc_pipefs
  Rework get_rpc_pipefs and introduce put_rpc_pipefs()

 fs/lockd/clntlock.c | 8 +-
 fs/lockd/host.c | 15 ++-
 fs/lockd/mon.c | 13 ++-

```
 fs/lockd/svc.c                  |    4 +-
 fs/namespace.c                  |    1 +
 fs/nfs/cache_lib.c              |   18 +--
 fs/nfs/cache_lib.h              |    3 +-
 fs/nfs/callback.c               |    7 +-
 fs/nfs/callback.h               |    3 +-
 fs/nfs/client.c                 |   45 ++++++-
 fs/nfs/dns_resolve.c            |  137 +++++++++++++++-----
 fs/nfs/dns_resolve.h            |   15 +--
 fs/nfs/inode.c                  |    9 +-
 fs/nfs/internal.h               |   10 ++-
 fs/nfs/mount_clnt.c             |    1 +
 fs/nfs/namespace.c              |    3 +-
 fs/nfs/nfs4namespace.c          |   20 ++--
 fs/nfs/super.c                  |   20 +++
 fs/nfsd/nfs4callback.c          |    5 +
 fs/nfsd/nfssvc.c                |   20 +++-
 include/linux/lockd/bind.h      |    3 +-
 include/linux/lockd/lockd.h     |    4 +-
 include/linux/nfs_fs_sb.h       |    1 +
 include/linux/sunrpc/cache.h    |    9 +-
 include/linux/sunrpc/clnt.h     |    5 +-
 include/linux/sunrpc/rpc_pipe_fs.h |    7 +-
 include/linux/sunrpc/svc.h      |    9 +-
 net/sunrpc/cache.c              |   16 ++-
 net/sunrpc/clnt.c               |   19 ++--
 net/sunrpc/rpc_pipe.c           |  235 ++++++++++++++++++++++++++++++++++----
 net/sunrpc/rpcb_clnt.c          |   19 ++-
 net/sunrpc/svc.c                |   52 +++++---
 32 files changed, 549 insertions(+), 187 deletions(-)

--
1.7.3.4
```


Subject: [PATCH v3 01/16] sunrpc: mount rpc_pipefs on initialization
Posted by Kirill A. Shutemov on Fri, 14 Jan 2011 13:48:59 GMT
View Forum Message <> Reply to Message

Mount rpc_pipefs on register_rpc_pipefs() and replace
rpc_get_mount()/rpc_put_mount() implementation with mntget()/mntput().

This commit introduces temprorary regression: there is no way to

remove the module. It will be fixed at the end of the patchset.

Signed-off-by: Kirill A. Shutemov <kas@openvz.org>
---
 net/sunrpc/rpc_pipe.c | 26 ++++++++++++++++++----------
 1 files changed, 15 insertions(+), 11 deletions(-)

diff --git a/net/sunrpc/rpc_pipe.c b/net/sunrpc/rpc_pipe.c
index 72bc536..9ab9355 100644
--- a/net/sunrpc/rpc_pipe.c
+++ b/net/sunrpc/rpc_pipe.c
@@ -29,7 +29,6 @@
 #include <linux/sunrpc/cache.h>

 static struct vfsmount *rpc_mnt __read_mostly;
-static int rpc_mount_count;

 static struct file_system_type rpc_pipe_fs_type;

@@ -423,18 +422,13 @@ struct rpc_filelist {

 struct vfsmount *rpc_get_mount(void)
 {
- int err;
-
- err = simple_pin_fs(&rpc_pipe_fs_type, &rpc_mnt, &rpc_mount_count);
- if (err != 0)
- return ERR_PTR(err);
- return rpc_mnt;
+ return mntget(rpc_mnt);
 }
 EXPORT_SYMBOL_GPL(rpc_get_mount);

 void rpc_put_mount(void)
 {
- simple_release_fs(&rpc_mnt, &rpc_mount_count);
+ mntput(rpc_mnt);
 }
 EXPORT_SYMBOL_GPL(rpc_put_mount);

@@ -1071,12 +1065,22 @@ int register_rpc_pipefs(void)
   if (!rpc_inode_cachep)
    return -ENOMEM;
   err = register_filesystem(&rpc_pipe_fs_type);
- if (err) {
- kmem_cache_destroy(rpc_inode_cachep);
- return err;
+ if (err)

```
+  goto destroy_cache;
+
+ rpc_mnt = kern_mount(&rpc_pipe_fs_type);
+ if (IS_ERR(rpc_mnt)) {
+  err = PTR_ERR(rpc_mnt);
+  goto unregister_fs;
  }

  return 0;
+
+unregister_fs:
+ unregister_filesystem(&rpc_pipe_fs_type);
+destroy_cache:
+ kmem_cache_destroy(rpc_inode_cachep);
+ return err;
 }

 void unregister_rpc_pipefs(void)
--
1.7.3.4
```

Subject: [PATCH v3 02/16] sunrpc: introduce init_rpc_pipefs
Posted by Kirill A. Shutemov on Fri, 14 Jan 2011 13:49:00 GMT
View Forum Message <> Reply to Message

Introduce global variable init_rpc_pipefs and use it instead of
rpc_get_mount()/rpc_put_mount().

Signed-off-by: Kirill A. Shutemov <kas@openvz.org>

```
 fs/nfs/cache_lib.c                |    6 +++---
 include/linux/sunrpc/rpc_pipe_fs.h |    4 ++--
 net/sunrpc/clnt.c                 |   10 ++++++------
 net/sunrpc/rpc_pipe.c             |   21 +++++++++----------------
 4 files changed, 14 insertions(+), 27 deletions(-)

diff --git a/fs/nfs/cache_lib.c b/fs/nfs/cache_lib.c
index 8469031..dd7ca5f 100644
--- a/fs/nfs/cache_lib.c
+++ b/fs/nfs/cache_lib.c
@@ -117,7 +117,7 @@ int nfs_cache_register(struct cache_detail *cd)
  struct vfsmount *mnt;
```

```
  int ret;

- mnt = rpc_get_mount();
+ mnt = mntget(init_rpc_pipefs);
  if (IS_ERR(mnt))
   return PTR_ERR(mnt);
  ret = vfs_path_lookup(mnt->mnt_root, mnt, "/cache", 0, &nd);
@@ -129,13 +129,13 @@ int nfs_cache_register(struct cache_detail *cd)
  if (!ret)
   return ret;
 err:
- rpc_put_mount();
+ mntput(mnt);
  return ret;
 }

 void nfs_cache_unregister(struct cache_detail *cd)
 {
  sunrpc_cache_unregister_pipefs(cd);
- rpc_put_mount();
+ mntput(init_rpc_pipefs);
 }

diff --git a/include/linux/sunrpc/rpc_pipe_fs.h b/include/linux/sunrpc/rpc_pipe_fs.h
index cf14db9..b09bfa5 100644
--- a/include/linux/sunrpc/rpc_pipe_fs.h
+++ b/include/linux/sunrpc/rpc_pipe_fs.h
@@ -44,6 +44,8 @@ RPC_I(struct inode *inode)
  return container_of(inode, struct rpc_inode, vfs_inode);
 }

+extern struct vfsmount *init_rpc_pipefs;
+
 extern int rpc_queue_upcall(struct inode *, struct rpc_pipe_msg *);

 struct rpc_clnt;
@@ -60,8 +62,6 @@ extern void rpc_remove_cache_dir(struct dentry *);
 extern struct dentry *rpc_mkpipe(struct dentry *, const char *, void *,
      const struct rpc_pipe_ops *, int flags);
 extern int rpc_unlink(struct dentry *);
-extern struct vfsmount *rpc_get_mount(void);
-extern void rpc_put_mount(void);
 extern int register_rpc_pipefs(void);
 extern void unregister_rpc_pipefs(void);

diff --git a/net/sunrpc/clnt.c b/net/sunrpc/clnt.c
index 57d344c..f3812d0 100644
--- a/net/sunrpc/clnt.c
```

```
+++ b/net/sunrpc/clnt.c
@@ -112,9 +112,7 @@ rpc_setup_pipedir(struct rpc_clnt *clnt, char *dir_name)
  if (dir_name == NULL)
   return 0;

- path.mnt = rpc_get_mount();
- if (IS_ERR(path.mnt))
-  return PTR_ERR(path.mnt);
+ path.mnt = mntget(init_rpc_pipefs);
  error = vfs_path_lookup(path.mnt->mnt_root, path.mnt, dir_name, 0, &nd);
  if (error)
   goto err;
@@ -140,7 +138,7 @@ rpc_setup_pipedir(struct rpc_clnt *clnt, char *dir_name)
 err_path_put:
  path_put(&nd.path);
 err:
- rpc_put_mount();
+ mntput(path.mnt);
  return error;
 }

@@ -251,7 +249,7 @@ static struct rpc_clnt * rpc_new_client(const struct rpc_create_args *args,
stru
 out_no_auth:
  if (!IS_ERR(clnt->cl_path.dentry)) {
   rpc_remove_client_dir(clnt->cl_path.dentry);
-  rpc_put_mount();
+  mntput(clnt->cl_path.mnt);
  }
 out_no_path:
  kfree(clnt->cl_principal);
@@ -472,7 +470,7 @@ rpc_free_client(struct rpc_clnt *clnt)
    clnt->cl_protname, clnt->cl_server);
  if (!IS_ERR(clnt->cl_path.dentry)) {
   rpc_remove_client_dir(clnt->cl_path.dentry);
-  rpc_put_mount();
+  mntput(clnt->cl_path.mnt);
  }
  if (clnt->cl_parent != clnt) {
   rpc_release_client(clnt->cl_parent);
diff --git a/net/sunrpc/rpc_pipe.c b/net/sunrpc/rpc_pipe.c
index 9ab9355..484c9a3 100644
--- a/net/sunrpc/rpc_pipe.c
+++ b/net/sunrpc/rpc_pipe.c
@@ -28,7 +28,8 @@
 #include <linux/sunrpc/rpc_pipe_fs.h>
 #include <linux/sunrpc/cache.h>
```

```
-static struct vfsmount *rpc_mnt __read_mostly;
+struct vfsmount *init_rpc_pipefs __read_mostly;
+EXPORT_SYMBOL_GPL(init_rpc_pipefs);

 static struct file_system_type rpc_pipe_fs_type;

@@ -420,18 +421,6 @@ struct rpc_filelist {
  umode_t mode;
 };

-struct vfsmount *rpc_get_mount(void)
-{
- return mntget(rpc_mnt);
-}
-EXPORT_SYMBOL_GPL(rpc_get_mount);
-
-void rpc_put_mount(void)
-{
- mntput(rpc_mnt);
-}
-EXPORT_SYMBOL_GPL(rpc_put_mount);
-
 static int rpc_delete_dentry(const struct dentry *dentry)
 {
  return 1;
@@ -1068,9 +1057,9 @@ int register_rpc_pipefs(void)
  if (err)
   goto destroy_cache;

- rpc_mnt = kern_mount(&rpc_pipe_fs_type);
- if (IS_ERR(rpc_mnt)) {
-  err = PTR_ERR(rpc_mnt);
+ init_rpc_pipefs = kern_mount(&rpc_pipe_fs_type);
+ if (IS_ERR(init_rpc_pipefs)) {
+  err = PTR_ERR(init_rpc_pipefs);
   goto unregister_fs;
  }

--
1.7.3.4
```

Subject: [PATCH v3 03/16] sunrpc: push init_rpc_pipefs up to rpc_create() callers
Posted by Kirill A. Shutemov on Fri, 14 Jan 2011 13:49:01 GMT
View Forum Message <> Reply to Message

Signed-off-by: Kirill A. Shutemov <kas@openvz.org>
---
 fs/lockd/host.c | 2 ++
 fs/lockd/mon.c | 2 ++
 fs/nfs/client.c | 2 ++
 fs/nfs/mount_clnt.c | 2 ++
 fs/nfsd/nfs4callback.c | 2 ++
 include/linux/sunrpc/clnt.h | 1 +
 net/sunrpc/clnt.c | 11 +++++++----
 net/sunrpc/rpcb_clnt.c | 3 +++
 8 files changed, 21 insertions(+), 4 deletions(-)

```
diff --git a/fs/lockd/host.c b/fs/lockd/host.c
index 5f1bcb2..7ed06d2 100644
--- a/fs/lockd/host.c
+++ b/fs/lockd/host.c
@@ -14,6 +14,7 @@
 #include <linux/in6.h>
 #include <linux/sunrpc/clnt.h>
 #include <linux/sunrpc/svc.h>
+#include <linux/sunrpc/rpc_pipe_fs.h>
 #include <linux/lockd/lockd.h>
 #include <linux/mutex.h>

@@ -463,6 +464,7 @@ nlm_bind_host(struct nlm_host *host)
    .authflavor = RPC_AUTH_UNIX,
    .flags  = (RPC_CLNT_CREATE_NOPING |
        RPC_CLNT_CREATE_AUTOBIND),
+   .rpcmount = init_rpc_pipefs,
   };

   /*
diff --git a/fs/lockd/mon.c b/fs/lockd/mon.c
index 23d7451..6219026 100644
--- a/fs/lockd/mon.c
+++ b/fs/lockd/mon.c
@@ -15,6 +15,7 @@
 #include <linux/sunrpc/clnt.h>
 #include <linux/sunrpc/xprtsock.h>
 #include <linux/sunrpc/svc.h>
+#include <linux/sunrpc/rpc_pipe_fs.h>
 #include <linux/lockd/lockd.h>

 #include <asm/unaligned.h>
@@ -78,6 +79,7 @@ static struct rpc_clnt *nsm_create(void)
```

```
   .version  = NSM_VERSION,
   .authflavor  = RPC_AUTH_NULL,
   .flags   = RPC_CLNT_CREATE_NOPING,
+  .rpcmount  = init_rpc_pipefs,
  };

  return rpc_create(&args);
diff --git a/fs/nfs/client.c b/fs/nfs/client.c
index 192f2f8..36847f8 100644
--- a/fs/nfs/client.c
+++ b/fs/nfs/client.c
@@ -25,6 +25,7 @@
 #include <linux/sunrpc/metrics.h>
 #include <linux/sunrpc/xprtsock.h>
 #include <linux/sunrpc/xprtrdma.h>
+#include <linux/sunrpc/rpc_pipe_fs.h>
 #include <linux/nfs_fs.h>
 #include <linux/nfs_mount.h>
 #include <linux/nfs4_mount.h>
@@ -628,6 +629,7 @@ static int nfs_create_rpc_client(struct nfs_client *clp,
   .program = &nfs_program,
   .version = clp->rpc_ops->version,
   .authflavor = flavor,
+  .rpcmount = init_rpc_pipefs,
  };

  if (discrtry)
diff --git a/fs/nfs/mount_clnt.c b/fs/nfs/mount_clnt.c
index d4c2d6b..6227875 100644
--- a/fs/nfs/mount_clnt.c
+++ b/fs/nfs/mount_clnt.c
@@ -13,6 +13,7 @@
 #include <linux/in.h>
 #include <linux/sunrpc/clnt.h>
 #include <linux/sunrpc/sched.h>
+#include <linux/sunrpc/rpc_pipe_fs.h>
 #include <linux/nfs_fs.h>
 #include "internal.h"

@@ -161,6 +162,7 @@ int nfs_mount(struct nfs_mount_request *info)
   .program = &mnt_program,
   .version = info->version,
   .authflavor = RPC_AUTH_UNIX,
+  .rpcmount = init_rpc_pipefs,
  };
  struct rpc_clnt  *mnt_clnt;
  int   status;
diff --git a/fs/nfsd/nfs4callback.c b/fs/nfsd/nfs4callback.c
```

```
index 21a63da..9bec643 100644
--- a/fs/nfsd/nfs4callback.c
+++ b/fs/nfsd/nfs4callback.c
@@ -33,6 +33,7 @@

 #include <linux/sunrpc/clnt.h>
 #include <linux/sunrpc/svc_xprt.h>
+#include <linux/sunrpc/rpc_pipe_fs.h>
 #include <linux/slab.h>
 #include "nfsd.h"
 #include "state.h"
@@ -646,6 +647,7 @@ int setup_callback_client(struct nfs4_client *clp, struct nfs4_cb_conn
*conn)
   .version = 0,
   .authflavor = clp->cl_flavor,
   .flags  = (RPC_CLNT_CREATE_NOPING | RPC_CLNT_CREATE_QUIET),
+  .rpcmount = init_rpc_pipefs,
  };
  struct rpc_clnt *client;

diff --git a/include/linux/sunrpc/clnt.h b/include/linux/sunrpc/clnt.h
index ef9476a..dffaaaa 100644
--- a/include/linux/sunrpc/clnt.h
+++ b/include/linux/sunrpc/clnt.h
@@ -116,6 +116,7 @@ struct rpc_create_args {
  unsigned long  flags;
  char   *client_name;
  struct svc_xprt  *bc_xprt; /* NFSv4.1 backchannel */
+ struct vfsmount  *rpcmount;
 };

 /* Values for "flags" field */
diff --git a/net/sunrpc/clnt.c b/net/sunrpc/clnt.c
index f3812d0..6e1d923 100644
--- a/net/sunrpc/clnt.c
+++ b/net/sunrpc/clnt.c
@@ -96,7 +96,8 @@ static void rpc_unregister_client(struct rpc_clnt *clnt)
 }

 static int
-rpc_setup_pipedir(struct rpc_clnt *clnt, char *dir_name)
+rpc_setup_pipedir(struct rpc_clnt *clnt, struct vfsmount *rpcmount,
+  char *dir_name)
 {
  static uint32_t clntid;
  struct nameidata nd;
@@ -112,7 +113,7 @@ rpc_setup_pipedir(struct rpc_clnt *clnt, char *dir_name)
  if (dir_name == NULL)
```

```
   return 0;

- path.mnt = mntget(init_rpc_pipefs);
+ path.mnt = mntget(rpcmount);
  error = vfs_path_lookup(path.mnt->mnt_root, path.mnt, dir_name, 0, &nd);
  if (error)
   goto err;
@@ -226,7 +227,8 @@ static struct rpc_clnt * rpc_new_client(const struct rpc_create_args *args,
stru

  atomic_set(&clnt->cl_count, 1);

- err = rpc_setup_pipedir(clnt, program->pipe_dir_name);
+ BUG_ON(!args->rpcmount);
+ err = rpc_setup_pipedir(clnt, args->rpcmount, program->pipe_dir_name);
  if (err < 0)
   goto out_no_path;

@@ -390,7 +392,8 @@ rpc_clone_client(struct rpc_clnt *clnt)
    goto out_no_principal;
  }
  atomic_set(&new->cl_count, 1);
- err = rpc_setup_pipedir(new, clnt->cl_program->pipe_dir_name);
+ err = rpc_setup_pipedir(new, clnt->cl_path.mnt,
+    clnt->cl_program->pipe_dir_name);
  if (err != 0)
   goto out_no_path;
  if (new->cl_auth)
diff --git a/net/sunrpc/rpcb_clnt.c b/net/sunrpc/rpcb_clnt.c
index c652e4c..b059cbe 100644
--- a/net/sunrpc/rpcb_clnt.c
+++ b/net/sunrpc/rpcb_clnt.c
@@ -27,6 +27,7 @@
 #include <linux/sunrpc/clnt.h>
 #include <linux/sunrpc/sched.h>
 #include <linux/sunrpc/xprtsock.h>
+#include <linux/sunrpc/rpc_pipe_fs.h>

 #ifdef RPC_DEBUG
 # define RPCDBG_FACILITY RPCDBG_BIND
@@ -182,6 +183,7 @@ static int rpcb_create_local(void)
   .version = RPCBVERS_2,
   .authflavor = RPC_AUTH_UNIX,
   .flags  = RPC_CLNT_CREATE_NOPING,
+  .rpcmount = init_rpc_pipefs,
  };
  struct rpc_clnt *clnt, *clnt4;
  int result = 0;
```

```
@@ -236,6 +238,7 @@ static struct rpc_clnt *rpcb_create(char *hostname, struct sockaddr
*srvaddr,
   .authflavor = RPC_AUTH_UNIX,
   .flags  = (RPC_CLNT_CREATE_NOPING |
     RPC_CLNT_CREATE_NONPRIVPORT),
+  .rpcmount = init_rpc_pipefs,
  };

  switch (srvaddr->sa_family) {
--
1.7.3.4
```

_____
Containers mailing list
Containers@lists.linux-foundation.org
https://lists.linux-foundation.org/mailman/listinfo/containe rs

---

Subject: [PATCH v3 04/16] sunrpc: tag svc_serv with rpc_pipefs mount point
Posted by Kirill A. Shutemov on Fri, 14 Jan 2011 13:49:02 GMT
View Forum Message <> Reply to Message

Signed-off-by: Kirill A. Shutemov <kas@openvz.org>
---
 include/linux/sunrpc/svc.h | 1 +
 net/sunrpc/svc.c | 4 ++++
 2 files changed, 5 insertions(+), 0 deletions(-)

```
diff --git a/include/linux/sunrpc/svc.h b/include/linux/sunrpc/svc.h
index c81d4d8..534ea8e 100644
--- a/include/linux/sunrpc/svc.h
+++ b/include/linux/sunrpc/svc.h
@@ -64,6 +64,7 @@ struct svc_pool {
  */
 struct svc_serv {
  struct svc_program * sv_program; /* RPC program */
+ struct vfsmount * sv_rpcmount; /* rpc_pipefs mount point*/
  struct svc_stat * sv_stats; /* RPC statistics */
  spinlock_t  sv_lock;
  unsigned int  sv_nrthreads; /* # of server threads */
diff --git a/net/sunrpc/svc.c b/net/sunrpc/svc.c
index 0e659c6..8cc6e79 100644
--- a/net/sunrpc/svc.c
+++ b/net/sunrpc/svc.c
@@ -20,6 +20,7 @@
 #include <linux/module.h>
 #include <linux/kthread.h>
 #include <linux/slab.h>
```

```
+#include <linux/mount.h>

 #include <linux/sunrpc/types.h>
 #include <linux/sunrpc/xdr.h>
@@ -27,6 +28,7 @@
 #include <linux/sunrpc/svcsock.h>
 #include <linux/sunrpc/clnt.h>
 #include <linux/sunrpc/bc_xprt.h>
+#include <linux/sunrpc/rpc_pipe_fs.h>

 #define RPCDBG_FACILITY RPCDBG_SVCDSP

@@ -371,6 +373,7 @@ __svc_create(struct svc_program *prog, unsigned int bufsize, int npools,
   return NULL;
  serv->sv_name      = prog->pg_name;
  serv->sv_program   = prog;
+ serv->sv_rpcmount  = mntget(init_rpc_pipefs);
  serv->sv_nrthreads = 1;
  serv->sv_stats     = prog->pg_stats;
  if (bufsize > RPCSVC_MAXPAYLOAD)
@@ -488,6 +491,7 @@ svc_destroy(struct svc_serv *serv)
  if (svc_serv_is_pooled(serv))
   svc_pool_map_put();

+ mntput(serv->sv_rpcmount);
  svc_unregister(serv);
  kfree(serv->sv_pools);
  kfree(serv);
--
1.7.3.4


_____
Containers mailing list
Containers@lists.linux-foundation.org
 https://lists.linux-foundation.org/mailman/listinfo/containe rs
```

Subject: [PATCH v3 05/16] sunrpc: get rpc_pipefs mount point for svc_serv from callers

Posted by Kirill A. Shutemov on Fri, 14 Jan 2011 13:49:03 GMT

View Forum Message <> Reply to Message

```
Signed-off-by: Kirill A. Shutemov <kas@openvz.org>
---
 fs/lockd/svc.c            |    4 +++-
 fs/nfs/callback.c         |    4 +++-
 fs/nfsd/nfssvc.c          |    6 ++++--
 include/linux/sunrpc/svc.h |    8 ++++----
```

```
 net/sunrpc/svc.c          |   18 +++++++++---------
 5 files changed, 23 insertions(+), 17 deletions(-)

diff --git a/fs/lockd/svc.c b/fs/lockd/svc.c
index abfff9d..32310b1 100644
--- a/fs/lockd/svc.c
+++ b/fs/lockd/svc.c
@@ -31,6 +31,7 @@
 #include <linux/sunrpc/clnt.h>
 #include <linux/sunrpc/svc.h>
 #include <linux/sunrpc/svcsock.h>
+#include <linux/sunrpc/rpc_pipe_fs.h>
 #include <net/ip.h>
 #include <linux/lockd/lockd.h>
 #include <linux/nfs.h>
@@ -269,7 +270,8 @@ int lockd_up(void)
     "lockd_up: no pid, %d users??\n", nlmsvc_users);

  error = -ENOMEM;
- serv = svc_create(&nlmsvc_program, LOCKD_BUFSIZE, NULL);
+ serv = svc_create(&nlmsvc_program, init_rpc_pipefs, LOCKD_BUFSIZE,
+    NULL);
  if (!serv) {
   printk(KERN_WARNING "lockd_up: create service failed\n");
   goto out;
diff --git a/fs/nfs/callback.c b/fs/nfs/callback.c
index 1990165..7a535c8 100644
--- a/fs/nfs/callback.c
+++ b/fs/nfs/callback.c
@@ -16,6 +16,7 @@
 #include <linux/freezer.h>
 #include <linux/kthread.h>
 #include <linux/sunrpc/svcauth_gss.h>
+#include <linux/sunrpc/rpc_pipe_fs.h>
 #include <linux/sunrpc/bc_xprt.h>

 #include <net/inet_sock.h>
@@ -290,7 +291,8 @@ int nfs_callback_up(u32 minorversion, struct rpc_xprt *xprt)
   nfs_callback_bc_serv(minorversion, xprt, cb_info);
   goto out;
  }
- serv = svc_create(&nfs4_callback_program, NFS4_CALLBACK_BUFSIZE, NULL);
+ serv = svc_create(&nfs4_callback_program, init_rpc_pipefs,
+    NFS4_CALLBACK_BUFSIZE, NULL);
  if (!serv) {
   ret = -ENOMEM;
   goto out_err;
diff --git a/fs/nfsd/nfssvc.c b/fs/nfsd/nfssvc.c
```

```
index 2bae1d8..d96c32b 100644
--- a/fs/nfsd/nfssvc.c
+++ b/fs/nfsd/nfssvc.c
@@ -13,6 +13,7 @@

 #include <linux/sunrpc/stats.h>
 #include <linux/sunrpc/svcsock.h>
+#include <linux/sunrpc/rpc_pipe_fs.h>
 #include <linux/lockd/bind.h>
 #include <linux/nfsacl.h>
 #include <linux/seq_file.h>
@@ -331,8 +332,9 @@ int nfsd_create_serv(void)
  }
  nfsd_reset_versions();

- nfsd_serv = svc_create_pooled(&nfsd_program, nfsd_max_blksize,
-         nfsd_last_thread, nfsd, THIS_MODULE);
+ nfsd_serv = svc_create_pooled(&nfsd_program, init_rpc_pipefs,
+          nfsd_max_blksize, nfsd_last_thread, nfsd,
+          THIS_MODULE);
  if (nfsd_serv == NULL)
   return -ENOMEM;

diff --git a/include/linux/sunrpc/svc.h b/include/linux/sunrpc/svc.h
index 534ea8e..ad30e5d 100644
--- a/include/linux/sunrpc/svc.h
+++ b/include/linux/sunrpc/svc.h
@@ -400,13 +400,13 @@ struct svc_procedure {
 /*
  * Function prototypes.
  */
-struct svc_serv *svc_create(struct svc_program *, unsigned int,
-       void (*shutdown)(struct svc_serv *));
+struct svc_serv *svc_create(struct svc_program *, struct vfsmount *,
+       unsigned int, void (*shutdown)(struct svc_serv *));
 struct svc_rqst *svc_prepare_thread(struct svc_serv *serv,
      struct svc_pool *pool);
 void     svc_exit_thread(struct svc_rqst *);
-struct svc_serv *  svc_create_pooled(struct svc_program *, unsigned int,
-   void (*shutdown)(struct svc_serv *),
+struct svc_serv *  svc_create_pooled(struct svc_program *, struct vfsmount *,
+   unsigned int, void (*shutdown)(struct svc_serv *),
    svc_thread_fn, struct module *);
 int     svc_set_num_threads(struct svc_serv *, struct svc_pool *, int);
 int     svc_pool_stats_open(struct svc_serv *serv, struct file *file);
diff --git a/net/sunrpc/svc.c b/net/sunrpc/svc.c
index 8cc6e79..8472798 100644
--- a/net/sunrpc/svc.c
```

```
+++ b/net/sunrpc/svc.c
@@ -28,7 +28,6 @@
 #include <linux/sunrpc/svcsock.h>
 #include <linux/sunrpc/clnt.h>
 #include <linux/sunrpc/bc_xprt.h>
-#include <linux/sunrpc/rpc_pipe_fs.h>

 #define RPCDBG_FACILITY RPCDBG_SVCDSP

@@ -361,7 +360,8 @@ svc_pool_for_cpu(struct svc_serv *serv, int cpu)
  * Create an RPC service
  */
 static struct svc_serv *
-__svc_create(struct svc_program *prog, unsigned int bufsize, int npools,
+__svc_create(struct svc_program *prog, struct vfsmount *rpcmount,
+      unsigned int bufsize, int npools,
       void (*shutdown)(struct svc_serv *serv))
 {
  struct svc_serv *serv;
@@ -373,7 +373,7 @@ __svc_create(struct svc_program *prog, unsigned int bufsize, int npools,
   return NULL;
  serv->sv_name      = prog->pg_name;
  serv->sv_program   = prog;
- serv->sv_rpcmount  = mntget(init_rpc_pipefs);
+ serv->sv_rpcmount  = mntget(rpcmount);
  serv->sv_nrthreads = 1;
  serv->sv_stats     = prog->pg_stats;
  if (bufsize > RPCSVC_MAXPAYLOAD)
@@ -429,22 +429,22 @@ __svc_create(struct svc_program *prog, unsigned int bufsize, int
npools,
 }

 struct svc_serv *
-svc_create(struct svc_program *prog, unsigned int bufsize,
-    void (*shutdown)(struct svc_serv *serv))
+svc_create(struct svc_program *prog, struct vfsmount *rpcmount,
+    unsigned int bufsize, void (*shutdown)(struct svc_serv *serv))
 {
- return __svc_create(prog, bufsize, /*npools*/1, shutdown);
+ return __svc_create(prog, rpcmount, bufsize, /*npools*/1, shutdown);
 }
 EXPORT_SYMBOL_GPL(svc_create);

 struct svc_serv *
-svc_create_pooled(struct svc_program *prog, unsigned int bufsize,
-    void (*shutdown)(struct svc_serv *serv),
+svc_create_pooled(struct svc_program *prog, struct vfsmount *rpcmount,
+    unsigned int bufsize, void (*shutdown)(struct svc_serv *serv),
```

```
     svc_thread_fn func, struct module *mod)
 {
  struct svc_serv *serv;
  unsigned int npools = svc_pool_map_get();

- serv = __svc_create(prog, bufsize, npools, shutdown);
+ serv = __svc_create(prog, rpcmount, bufsize, npools, shutdown);

  if (serv != NULL) {
   serv->sv_function = func;
--
1.7.3.4
```

_____
Containers mailing list
Containers@lists.linux-foundation.org
https://lists.linux-foundation.org/mailman/listinfo/containe rs

---

Subject: [PATCH v3 06/16] lockd: get rpc_pipefs mount point from callers
Posted by Kirill A. Shutemov on Fri, 14 Jan 2011 13:49:04 GMT
View Forum Message <> Reply to Message

Signed-off-by: Kirill A. Shutemov <kas@openvz.org>

```
 fs/lockd/clntlock.c         |    8 +++++---
 fs/lockd/host.c             |   17 +++++++++++++---
 fs/lockd/mon.c              |   15 ++++++++-------
 fs/lockd/svc.c              |    6 ++----
 fs/nfs/client.c             |    1 +
 fs/nfsd/nfssvc.c            |    2 +-
 include/linux/lockd/bind.h  |    3 ++-
 include/linux/lockd/lockd.h |    4 +++-
 8 files changed, 36 insertions(+), 20 deletions(-)

diff --git a/fs/lockd/clntlock.c b/fs/lockd/clntlock.c
index 8d4ea83..4664c56 100644
--- a/fs/lockd/clntlock.c
+++ b/fs/lockd/clntlock.c
@@ -56,13 +56,14 @@ struct nlm_host *nlmclnt_init(const struct nlmclnt_initdata *nlm_init)
  u32 nlm_version = (nlm_init->nfs_version == 2) ? 1 : 4;
  int status;

- status = lockd_up();
+ status = lockd_up(nlm_init->rpcmount);
  if (status < 0)
   return ERR_PTR(status);
```

```
  host = nlmclnt_lookup_host(nlm_init->address, nlm_init->addrlen,
        nlm_init->protocol, nlm_version,
-       nlm_init->hostname, nlm_init->noresvport);
+       nlm_init->hostname, nlm_init->noresvport,
+       nlm_init->rpcmount);
  if (host == NULL) {
   lockd_down();
   return ERR_PTR(-ENOLCK);
@@ -223,7 +224,8 @@ reclaimer(void *ptr)
  allow_signal(SIGKILL);

  down_write(&host->h_rwsem);
- lockd_up(); /* note: this cannot fail as lockd is already running */
+ /* note: this cannot fail as lockd is already running */
+ lockd_up(host->h_rpcmount);

  dprintk("lockd: reclaiming locks for host %s\n", host->h_name);

diff --git a/fs/lockd/host.c b/fs/lockd/host.c
index 7ed06d2..d5ca8f9 100644
--- a/fs/lockd/host.c
+++ b/fs/lockd/host.c
@@ -14,9 +14,10 @@
 #include <linux/in6.h>
 #include <linux/sunrpc/clnt.h>
 #include <linux/sunrpc/svc.h>
-#include <linux/sunrpc/rpc_pipe_fs.h>
 #include <linux/lockd/lockd.h>
 #include <linux/mutex.h>
+#include <linux/mount.h>
+

 #include <net/ipv6.h>

@@ -55,6 +56,7 @@ struct nlm_lookup_host_info {
  const char  *hostname; /* remote's hostname */
  const size_t  hostname_len; /* it's length */
  const int  noresvport; /* use non-priv port */
+ struct vfsmount  *rpcmount; /* rpc_pipefs mount point */
 };

 /*
@@ -134,6 +136,7 @@ static struct nlm_host *nlm_alloc_host(struct nlm_lookup_host_info *ni,
  host->h_srcaddrlen = 0;

  host->h_rpcclnt    = NULL;
+ host->h_rpcmount   = mntget(ni->rpcmount);
  host->h_name    = nsm->sm_name;
```

```
  host->h_version    = ni->version;
  host->h_proto      = ni->protocol;
@@ -179,6 +182,7 @@ static void nlm_destroy_host_locked(struct nlm_host *host)

  nsm_unmonitor(host);
  nsm_release(host->h_nsmhandle);
+ mntput(host->h_rpcmount);

  clnt = host->h_rpcclnt;
  if (clnt != NULL)
@@ -207,7 +211,8 @@ struct nlm_host *nlmclnt_lookup_host(const struct sockaddr *sap,
          const unsigned short protocol,
          const u32 version,
          const char *hostname,
-         int noresvport)
+         int noresvport,
+         struct vfsmount *rpcmount)
 {
  struct nlm_lookup_host_info ni = {
   .server  = 0,
@@ -218,6 +223,7 @@ struct nlm_host *nlmclnt_lookup_host(const struct sockaddr *sap,
   .hostname = hostname,
   .hostname_len = strlen(hostname),
   .noresvport = noresvport,
+  .rpcmount = rpcmount,
  };
  struct hlist_head *chain;
  struct hlist_node *pos;
@@ -243,6 +249,8 @@ struct nlm_host *nlmclnt_lookup_host(const struct sockaddr *sap,
    continue;
   if (host->h_version != version)
    continue;
+  if (host->h_rpcmount->mnt_sb != ni.rpcmount->mnt_sb)
+   continue;

   nlm_get_host(host);
   dprintk("lockd: %s found host %s (%s)\n", __func__,
@@ -333,6 +341,7 @@ struct nlm_host *nlmsvc_lookup_host(const struct svc_rqst *rqstp,
   .version = rqstp->rq_vers,
   .hostname = hostname,
   .hostname_len = hostname_len,
+  .rpcmount = rqstp->rq_server->sv_rpcmount,
  };

  dprintk("lockd: %s(host='%*s', vers=%u, proto=%s)\n", __func__,
@@ -374,6 +383,8 @@ struct nlm_host *nlmsvc_lookup_host(const struct svc_rqst *rqstp,
    continue;
   if (!rpc_cmp_addr(nlm_srcaddr(host), src_sap))
```

```
    continue;
+  if (host->h_rpcmount->mnt_sb != ni.rpcmount->mnt_sb)
+   continue;

   /* Move to head of hash chain. */
   hlist_del(&host->h_hash);
@@ -464,7 +475,7 @@ nlm_bind_host(struct nlm_host *host)
    .authflavor = RPC_AUTH_UNIX,
    .flags  = (RPC_CLNT_CREATE_NOPING |
        RPC_CLNT_CREATE_AUTOBIND),
-   .rpcmount = init_rpc_pipefs,
+   .rpcmount = host->h_rpcmount,
   };

   /*
diff --git a/fs/lockd/mon.c b/fs/lockd/mon.c
index 6219026..a121f5e 100644
--- a/fs/lockd/mon.c
+++ b/fs/lockd/mon.c
@@ -15,7 +15,6 @@
 #include <linux/sunrpc/clnt.h>
 #include <linux/sunrpc/xprtsock.h>
 #include <linux/sunrpc/svc.h>
-#include <linux/sunrpc/rpc_pipe_fs.h>
 #include <linux/lockd/lockd.h>

 #include <asm/unaligned.h>
@@ -63,7 +62,7 @@ static inline struct sockaddr *nsm_addr(const struct nsm_handle *nsm)
  return (struct sockaddr *)&nsm->sm_addr;
 }

-static struct rpc_clnt *nsm_create(void)
+static struct rpc_clnt *nsm_create(struct vfsmount *rpcmount)
 {
  struct sockaddr_in sin = {
   .sin_family  = AF_INET,
@@ -79,13 +78,14 @@ static struct rpc_clnt *nsm_create(void)
   .version  = NSM_VERSION,
   .authflavor  = RPC_AUTH_NULL,
   .flags   = RPC_CLNT_CREATE_NOPING,
-  .rpcmount  = init_rpc_pipefs,
+  .rpcmount  = rpcmount,
  };

  return rpc_create(&args);
 }

-static int nsm_mon_unmon(struct nsm_handle *nsm, u32 proc, struct nsm_res *res)
```

```
+static int nsm_mon_unmon(struct nsm_handle *nsm, u32 proc, struct nsm_res *res,
+  struct vfsmount *rpcmount)
 {
  struct rpc_clnt *clnt;
  int  status;
@@ -101,7 +101,7 @@ static int nsm_mon_unmon(struct nsm_handle *nsm, u32 proc, struct
nsm_res *res)
   .rpc_resp = res,
  };

- clnt = nsm_create();
+ clnt = nsm_create(rpcmount);
  if (IS_ERR(clnt)) {
   status = PTR_ERR(clnt);
   dprintk("lockd: failed to create NSM upcall transport, "
@@ -151,7 +151,7 @@ int nsm_monitor(const struct nlm_host *host)
   */
  nsm->sm_mon_name = nsm_use_hostnames ? nsm->sm_name : nsm->sm_addrbuf;

- status = nsm_mon_unmon(nsm, NSMPROC_MON, &res);
+ status = nsm_mon_unmon(nsm, NSMPROC_MON, &res, host->h_rpcmount);
  if (unlikely(res.status != 0))
   status = -EIO;
  if (unlikely(status < 0)) {
@@ -185,7 +185,8 @@ void nsm_unmonitor(const struct nlm_host *host)
   && nsm->sm_monitored && !nsm->sm_sticky) {
   dprintk("lockd: nsm_unmonitor(%s)\n", nsm->sm_name);

-  status = nsm_mon_unmon(nsm, NSMPROC_UNMON, &res);
+  status = nsm_mon_unmon(nsm, NSMPROC_UNMON, &res,
+    host->h_rpcmount);
   if (res.status != 0)
    status = -EIO;
   if (status < 0)
diff --git a/fs/lockd/svc.c b/fs/lockd/svc.c
index 32310b1..7387b04 100644
--- a/fs/lockd/svc.c
+++ b/fs/lockd/svc.c
@@ -31,7 +31,6 @@
 #include <linux/sunrpc/clnt.h>
 #include <linux/sunrpc/svc.h>
 #include <linux/sunrpc/svcsock.h>
-#include <linux/sunrpc/rpc_pipe_fs.h>
 #include <net/ip.h>
 #include <linux/lockd/lockd.h>
 #include <linux/nfs.h>
@@ -249,7 +248,7 @@ out_err:
 /*
```

```
  * Bring up the lockd process if it's not already up.
  */
-int lockd_up(void)
+int lockd_up(struct vfsmount *rpcmount)
 {
  struct svc_serv *serv;
  int  error = 0;
@@ -270,8 +269,7 @@ int lockd_up(void)
    "lockd_up: no pid, %d users??\n", nlmsvc_users);

  error = -ENOMEM;
- serv = svc_create(&nlmsvc_program, init_rpc_pipefs, LOCKD_BUFSIZE,
-   NULL);
+ serv = svc_create(&nlmsvc_program, rpcmount, LOCKD_BUFSIZE, NULL);
  if (!serv) {
   printk(KERN_WARNING "lockd_up: create service failed\n");
   goto out;
diff --git a/fs/nfs/client.c b/fs/nfs/client.c
index 36847f8..ad3b5e8 100644
--- a/fs/nfs/client.c
+++ b/fs/nfs/client.c
@@ -675,6 +675,7 @@ static int nfs_start_lockd(struct nfs_server *server)
   .nfs_version = clp->rpc_ops->version,
   .noresvport = server->flags & NFS_MOUNT_NORESVPORT ?
     1 : 0,
+  .rpcmount = init_rpc_pipefs,
  };

  if (nlm_init.nfs_version > 3)
diff --git a/fs/nfsd/nfssvc.c b/fs/nfsd/nfssvc.c
index d96c32b..17d78d3 100644
--- a/fs/nfsd/nfssvc.c
+++ b/fs/nfsd/nfssvc.c
@@ -220,7 +220,7 @@ static int nfsd_startup(unsigned short port, int nrservs)
  ret = nfsd_init_socks(port);
  if (ret)
   goto out_racache;
- ret = lockd_up();
+ ret = lockd_up(init_rpc_pipefs);
  if (ret)
   goto out_racache;
  ret = nfs4_state_start();
diff --git a/include/linux/lockd/bind.h b/include/linux/lockd/bind.h
index fbc48f8..97cd4bf 100644
--- a/include/linux/lockd/bind.h
+++ b/include/linux/lockd/bind.h
@@ -42,6 +42,7 @@ struct nlmclnt_initdata {
  unsigned short  protocol;
```

```
  u32   nfs_version;
  int   noresvport;
+ struct vfsmount  *rpcmount;
 };

 /*
@@ -53,7 +54,7 @@ extern void nlmclnt_done(struct nlm_host *host);

 extern int nlmclnt_proc(struct nlm_host *host, int cmd,
       struct file_lock *fl);
-extern int lockd_up(void);
+extern int lockd_up(struct vfsmount *rpcmount);
 extern void lockd_down(void);

 #endif /* LINUX_LOCKD_BIND_H */
diff --git a/include/linux/lockd/lockd.h b/include/linux/lockd/lockd.h
index ff9abff..32dbb7f 100644
--- a/include/linux/lockd/lockd.h
+++ b/include/linux/lockd/lockd.h
@@ -44,6 +44,7 @@ struct nlm_host {
  size_t   h_addrlen;
  struct sockaddr_storage h_srcaddr; /* our address (optional) */
  size_t   h_srcaddrlen;
+ struct vfsmount  *h_rpcmount; /* rpc_pipefs mount point */
  struct rpc_clnt  *h_rpcclnt; /* RPC client to talk to peer */
  char   *h_name;  /* remote hostname */
  u32   h_version; /* interface version */
@@ -222,7 +223,8 @@ struct nlm_host  *nlmclnt_lookup_host(const struct sockaddr *sap,
       const unsigned short protocol,
       const u32 version,
       const char *hostname,
-      int noresvport);
+      int noresvport,
+      struct vfsmount *rpcmount);
 void    nlmclnt_release_host(struct nlm_host *);
 struct nlm_host  *nlmsvc_lookup_host(const struct svc_rqst *rqstp,
       const char *hostname,
--
1.7.3.4
```

_____
Containers mailing list
Containers@lists.linux-foundation.org
https://lists.linux-foundation.org/mailman/listinfo/containe rs

---

Subject: [PATCH v3 07/16] sunrpc: get rpc_pipefs mount point for

rpcb_create[_local] from callers
Posted by Kirill A. Shutemov on Fri, 14 Jan 2011 13:49:05 GMT
View Forum Message <> Reply to Message

Signed-off-by: Kirill A. Shutemov <kas@openvz.org>
---
 include/linux/sunrpc/clnt.h | 4 ++--
 net/sunrpc/rpcb_clnt.c | 22 ++++++++++++----------
 net/sunrpc/svc.c | 34 ++++++++++++++++++++++------------
 3 files changed, 35 insertions(+), 25 deletions(-)

```
diff --git a/include/linux/sunrpc/clnt.h b/include/linux/sunrpc/clnt.h
index dffaaaa..52f6142 100644
--- a/include/linux/sunrpc/clnt.h
+++ b/include/linux/sunrpc/clnt.h
@@ -135,10 +135,10 @@ void  rpc_shutdown_client(struct rpc_clnt *);
 void  rpc_release_client(struct rpc_clnt *);
 void  rpc_task_release_client(struct rpc_task *);

-int  rpcb_register(u32, u32, int, unsigned short);
+int  rpcb_register(u32, u32, int, unsigned short, struct vfsmount *);
 int  rpcb_v4_register(const u32 program, const u32 version,
      const struct sockaddr *address,
-     const char *netid);
+     const char *netid, struct vfsmount *rpcmount);
 void  rpcb_getport_async(struct rpc_task *);

 void  rpc_call_start(struct rpc_task *);
diff --git a/net/sunrpc/rpcb_clnt.c b/net/sunrpc/rpcb_clnt.c
index b059cbe..7fddafa 100644
--- a/net/sunrpc/rpcb_clnt.c
+++ b/net/sunrpc/rpcb_clnt.c
@@ -27,7 +27,6 @@
 #include <linux/sunrpc/clnt.h>
 #include <linux/sunrpc/sched.h>
 #include <linux/sunrpc/xprtsock.h>
-#include <linux/sunrpc/rpc_pipe_fs.h>

 #ifdef RPC_DEBUG
 # define RPCDBG_FACILITY RPCDBG_BIND
@@ -171,7 +170,7 @@ static DEFINE_MUTEX(rpcb_create_local_mutex);
  * Returns zero on success, otherwise a negative errno value
  * is returned.
  */
-static int rpcb_create_local(void)
+static int rpcb_create_local(struct vfsmount *rpcmount)
 {
  struct rpc_create_args args = {
   .net  = &init_net,
```

```
@@ -183,7 +182,7 @@ static int rpcb_create_local(void)
   .version = RPCBVERS_2,
   .authflavor = RPC_AUTH_UNIX,
   .flags  = RPC_CLNT_CREATE_NOPING,
-  .rpcmount = init_rpc_pipefs,
+  .rpcmount = rpcmount,
  };
  struct rpc_clnt *clnt, *clnt4;
  int result = 0;
@@ -225,7 +224,8 @@ out:
 }

 static struct rpc_clnt *rpcb_create(char *hostname, struct sockaddr *srvaddr,
-        size_t salen, int proto, u32 version)
+        size_t salen, int proto, u32 version,
+        struct vfsmount *rpcmount)
 {
  struct rpc_create_args args = {
   .net  = &init_net,
@@ -238,7 +238,7 @@ static struct rpc_clnt *rpcb_create(char *hostname, struct sockaddr
*srvaddr,
   .authflavor = RPC_AUTH_UNIX,
   .flags  = (RPC_CLNT_CREATE_NOPING |
     RPC_CLNT_CREATE_NONPRIVPORT),
-  .rpcmount = init_rpc_pipefs,
+  .rpcmount = rpcmount,
  };

  switch (srvaddr->sa_family) {
@@ -305,7 +305,8 @@ static int rpcb_register_call(struct rpc_clnt *clnt, struct rpc_message
*msg)
  * IN6ADDR_ANY (ie available for all AF_INET and AF_INET6
  * addresses).
  */
-int rpcb_register(u32 prog, u32 vers, int prot, unsigned short port)
+int rpcb_register(u32 prog, u32 vers, int prot, unsigned short port,
+  struct vfsmount *rpcmount)
 {
  struct rpcbind_args map = {
   .r_prog  = prog,
@@ -318,7 +319,7 @@ int rpcb_register(u32 prog, u32 vers, int prot, unsigned short port)
  };
  int error;

- error = rpcb_create_local();
+ error = rpcb_create_local(rpcmount);
  if (error)
   return error;
```

```
@@ -445,7 +446,8 @@ static int rpcb_unregister_all_protofamilies(struct rpc_message *msg)
  * advertises the service on all IPv4 and IPv6 addresses.
  */
 int rpcb_v4_register(const u32 program, const u32 version,
-       const struct sockaddr *address, const char *netid)
+       const struct sockaddr *address, const char *netid,
+       struct vfsmount *rpcmount)
 {
  struct rpcbind_args map = {
   .r_prog  = program,
@@ -458,7 +460,7 @@ int rpcb_v4_register(const u32 program, const u32 version,
  };
  int error;

- error = rpcb_create_local();
+ error = rpcb_create_local(rpcmount);
  if (error)
   return error;
  if (rpcb_local_clnt4 == NULL)
@@ -594,7 +596,7 @@ void rpcb_getport_async(struct rpc_task *task)
   task->tk_pid, __func__, bind_version);

  rpcb_clnt = rpcb_create(clnt->cl_server, sap, salen, xprt->prot,
-    bind_version);
+    bind_version, clnt->cl_path.mnt);
  if (IS_ERR(rpcb_clnt)) {
   status = PTR_ERR(rpcb_clnt);
   dprintk("RPC: %5u %s: rpcb_create failed, error %ld\n",
diff --git a/net/sunrpc/svc.c b/net/sunrpc/svc.c
index 8472798..4d0b45a 100644
--- a/net/sunrpc/svc.c
+++ b/net/sunrpc/svc.c
@@ -739,7 +739,8 @@ EXPORT_SYMBOL_GPL(svc_exit_thread);
  */
 static int __svc_rpcb_register4(const u32 program, const u32 version,
     const unsigned short protocol,
-    const unsigned short port)
+    const unsigned short port,
+    struct vfsmount *rpcmount)
 {
  const struct sockaddr_in sin = {
   .sin_family  = AF_INET,
@@ -761,14 +762,16 @@ static int __svc_rpcb_register4(const u32 program, const u32 version,
  }

  error = rpcb_v4_register(program, version,
-     (const struct sockaddr *)&sin, netid);
```

```
+     (const struct sockaddr *)&sin, netid,
+     rpcmount);

  /*
   * User space didn't support rpcbind v4, so retry this
   * registration request with the legacy rpcbind v2 protocol.
   */
  if (error == -EPROTONOSUPPORT)
-  error = rpcb_register(program, version, protocol, port);
+  error = rpcb_register(program, version, protocol, port,
+    rpcmount);

  return error;
 }
@@ -786,7 +789,8 @@ static int __svc_rpcb_register4(const u32 program, const u32 version,
  */
 static int __svc_rpcb_register6(const u32 program, const u32 version,
     const unsigned short protocol,
-    const unsigned short port)
+    const unsigned short port,
+    struct vfsmount *rpcmount)
 {
  const struct sockaddr_in6 sin6 = {
   .sin6_family  = AF_INET6,
@@ -808,7 +812,8 @@ static int __svc_rpcb_register6(const u32 program, const u32 version,
  }

  error = rpcb_v4_register(program, version,
-     (const struct sockaddr *)&sin6, netid);
+     (const struct sockaddr *)&sin6, netid,
+     rpcmount);

  /*
   * User space didn't support rpcbind version 4, so we won't
@@ -831,19 +836,20 @@ static int __svc_register(const char *progname,
      const u32 program, const u32 version,
      const int family,
      const unsigned short protocol,
-     const unsigned short port)
+     const unsigned short port,
+     struct vfsmount *rpcmount)
 {
  int error = -EAFNOSUPPORT;

  switch (family) {
  case PF_INET:
   error = __svc_rpcb_register4(program, version,
-      protocol, port);
```

```
+       protocol, port, rpcmount);
    break;
 #if defined(CONFIG_IPV6) || defined(CONFIG_IPV6_MODULE)
  case PF_INET6:
   error = __svc_rpcb_register6(program, version,
-       protocol, port);
+       protocol, port, rpcmount);
 #endif /* defined(CONFIG_IPV6) || defined(CONFIG_IPV6_MODULE) */
  }

@@ -889,7 +895,8 @@ int svc_register(const struct svc_serv *serv, const int family,
     continue;

    error = __svc_register(progp->pg_name, progp->pg_prog,
-       i, family, proto, port);
+       i, family, proto, port,
+       serv->sv_rpcmount);
    if (error < 0)
     break;
   }
@@ -906,18 +913,18 @@ int svc_register(const struct svc_serv *serv, const int family,
  * in this case to clear all existing entries for [program, version].
  */
 static void __svc_unregister(const u32 program, const u32 version,
-        const char *progname)
+        const char *progname, struct vfsmount *rpcmount)
 {
  int error;

- error = rpcb_v4_register(program, version, NULL, "");
+ error = rpcb_v4_register(program, version, NULL, "", rpcmount);

  /*
   * User space didn't support rpcbind v4, so retry this
   * request with the legacy rpcbind v2 protocol.
   */
  if (error == -EPROTONOSUPPORT)
-  error = rpcb_register(program, version, 0, 0);
+  error = rpcb_register(program, version, 0, 0, rpcmount);

  dprintk("svc: %s(%sv%u), error %d\n",
    __func__, progname, version, error);
@@ -946,7 +953,8 @@ static void svc_unregister(const struct svc_serv *serv)
    if (progp->pg_vers[i]->vs_hidden)
     continue;

-   __svc_unregister(progp->pg_prog, i, progp->pg_name);
+   __svc_unregister(progp->pg_prog, i, progp->pg_name,
```

```
+     serv->sv_rpcmount);
   }
  }

--
1.7.3.4
```

_____
Containers mailing list
Containers@lists.linux-foundation.org
https://lists.linux-foundation.org/mailman/listinfo/containe rs

---

Subject: [PATCH v3 08/16] sunrpc: tag pipefs field of cache_detail with rpc_pipefs mount point
Posted by Kirill A. Shutemov on Fri, 14 Jan 2011 13:49:06 GMT
View Forum Message <> Reply to Message

Signed-off-by: Kirill A. Shutemov <kas@openvz.org>
---
fs/nfs/cache_lib.c | 3 +--
include/linux/sunrpc/cache.h | 9 +++------
net/sunrpc/cache.c | 16 ++++++++++------
3 files changed, 14 insertions(+), 14 deletions(-)

```
diff --git a/fs/nfs/cache_lib.c b/fs/nfs/cache_lib.c
index dd7ca5f..0944d4e 100644
--- a/fs/nfs/cache_lib.c
+++ b/fs/nfs/cache_lib.c
@@ -123,7 +123,7 @@ int nfs_cache_register(struct cache_detail *cd)
  ret = vfs_path_lookup(mnt->mnt_root, mnt, "/cache", 0, &nd);
  if (ret)
   goto err;
- ret = sunrpc_cache_register_pipefs(nd.path.dentry,
+ ret = sunrpc_cache_register_pipefs(mnt, nd.path.dentry,
    cd->name, 0600, cd);
  path_put(&nd.path);
  if (!ret)
@@ -136,6 +136,5 @@ err:
 void nfs_cache_unregister(struct cache_detail *cd)
 {
  sunrpc_cache_unregister_pipefs(cd);
- mntput(init_rpc_pipefs);
 }

diff --git a/include/linux/sunrpc/cache.h b/include/linux/sunrpc/cache.h
index 78aa104..339bca3 100644
--- a/include/linux/sunrpc/cache.h
```

```
+++ b/include/linux/sunrpc/cache.h
@@ -65,10 +65,6 @@ struct cache_detail_procfs {
  struct proc_dir_entry   *flush_ent, *channel_ent, *content_ent;
 };

-struct cache_detail_pipefs {
- struct dentry *dir;
-};
-
 struct cache_detail {
  struct module *  owner;
  int   hash_size;
@@ -115,7 +111,7 @@ struct cache_detail {

  union {
   struct cache_detail_procfs procfs;
-  struct cache_detail_pipefs pipefs;
+  struct path pipefs;
  } u;
 };

@@ -202,7 +198,8 @@ extern int cache_register_net(struct cache_detail *cd, struct net *net);
 extern void cache_unregister(struct cache_detail *cd);
 extern void cache_unregister_net(struct cache_detail *cd, struct net *net);

-extern int sunrpc_cache_register_pipefs(struct dentry *parent, const char *,
+extern int sunrpc_cache_register_pipefs(struct vfsmount *rpcmount,
+     struct dentry *parent, const char *,
      mode_t, struct cache_detail *);
 extern void sunrpc_cache_unregister_pipefs(struct cache_detail *);

diff --git a/net/sunrpc/cache.c b/net/sunrpc/cache.c
index e433e75..ed50d49 100644
--- a/net/sunrpc/cache.c
+++ b/net/sunrpc/cache.c
@@ -28,6 +28,7 @@
 #include <linux/workqueue.h>
 #include <linux/mutex.h>
 #include <linux/pagemap.h>
+#include <linux/mount.h>
 #include <asm/ioctls.h>
 #include <linux/sunrpc/types.h>
 #include <linux/sunrpc/cache.h>
@@ -1753,7 +1754,8 @@ const struct file_operations cache_flush_operations_pipefs = {
  .llseek  = no_llseek,
 };

-int sunrpc_cache_register_pipefs(struct dentry *parent,
```

```
+int sunrpc_cache_register_pipefs(struct vfsmount *rpcmount,
+     struct dentry *parent,
      const char *name, mode_t umode,
      struct cache_detail *cd)
 {
@@ -1766,9 +1768,10 @@ int sunrpc_cache_register_pipefs(struct dentry *parent,
  q.len = strlen(name);
  q.hash = full_name_hash(q.name, q.len);
  dir = rpc_create_cache_dir(parent, &q, umode, cd);
- if (!IS_ERR(dir))
-  cd->u.pipefs.dir = dir;
- else {
+ if (!IS_ERR(dir)) {
+  cd->u.pipefs.mnt = mntget(rpcmount);
+  cd->u.pipefs.dentry = dir;
+ } else {
   sunrpc_destroy_cache_detail(cd);
   ret = PTR_ERR(dir);
  }
@@ -1778,8 +1781,9 @@ EXPORT_SYMBOL_GPL(sunrpc_cache_register_pipefs);

 void sunrpc_cache_unregister_pipefs(struct cache_detail *cd)
 {
- rpc_remove_cache_dir(cd->u.pipefs.dir);
- cd->u.pipefs.dir = NULL;
+ rpc_remove_cache_dir(cd->u.pipefs.dentry);
+ cd->u.pipefs.dentry = NULL;
+ mntput(cd->u.pipefs.mnt);
  sunrpc_destroy_cache_detail(cd);
 }
 EXPORT_SYMBOL_GPL(sunrpc_cache_unregister_pipefs);
--
1.7.3.4
```

Subject: [PATCH v3 09/16] sunrpc: introduce rpc_pipefs_add_destroy_cb()
Posted by Kirill A. Shutemov on Fri, 14 Jan 2011 13:49:07 GMT
View Forum Message <> Reply to Message

Add facility to do some action on destroying of rpc_pipefs superblock.

Signed-off-by: Kirill A. Shutemov <kas@openvz.org>

```
 include/linux/sunrpc/rpc_pipe_fs.h |    3 ++
 net/sunrpc/rpc_pipe.c              |   51 ++++++++++++++++++++++++++++++++++++++++++++++++-
 2 files changed, 52 insertions(+), 2 deletions(-)

diff --git a/include/linux/sunrpc/rpc_pipe_fs.h b/include/linux/sunrpc/rpc_pipe_fs.h
index b09bfa5..f5216f1 100644
--- a/include/linux/sunrpc/rpc_pipe_fs.h
+++ b/include/linux/sunrpc/rpc_pipe_fs.h
@@ -46,6 +46,9 @@ RPC_I(struct inode *inode)

 extern struct vfsmount *init_rpc_pipefs;

+extern int rpc_pipefs_add_destroy_cb(struct super_block *sb,
+  void (*destroy_cb)(void *data), void *data);
+
 extern int rpc_queue_upcall(struct inode *, struct rpc_pipe_msg *);

 struct rpc_clnt;
diff --git a/net/sunrpc/rpc_pipe.c b/net/sunrpc/rpc_pipe.c
index 484c9a3..39511e3 100644
--- a/net/sunrpc/rpc_pipe.c
+++ b/net/sunrpc/rpc_pipe.c
@@ -939,6 +939,31 @@ static const struct super_operations s_ops = {

 #define RPCAUTH_GSSMAGIC 0x67596969

+struct destroy_cb {
+ struct list_head list;
+ void (*callback)(void *data);
+ void *data;
+};
+
+int rpc_pipefs_add_destroy_cb(struct super_block *sb,
+  void (*destroy_cb)(void *data), void *data)
+{
+ struct destroy_cb *dcb;
+ struct list_head *destroy_cb_list = sb->s_fs_info;
+
+ dcb = kmalloc(sizeof(*dcb), GFP_KERNEL);
+ if (!dcb)
+  return -ENOMEM;
+
+ dcb->callback = destroy_cb;
+ dcb->data = data;
+ INIT_LIST_HEAD(&dcb->list);
+ list_add(&dcb->list, destroy_cb_list);
+
+ return 0;
```

```
+}
+EXPORT_SYMBOL_GPL(rpc_pipefs_add_destroy_cb);
+
 /*
  * We have a single directory with 1 node in it.
  */
@@ -1004,8 +1029,16 @@ rpc_fill_super(struct super_block *sb, void *data, int silent)
   iput(inode);
   return -ENOMEM;
  }
- if (rpc_populate(root, files, RPCAUTH_lockd, RPCAUTH_RootEOF, NULL))
+ /* List of destroy callbacks */
+ sb->s_fs_info = kmalloc(sizeof(struct list_head), GFP_KERNEL);
+ if (!sb->s_fs_info)
+  return -ENOMEM;
+ INIT_LIST_HEAD((struct list_head*) &sb->s_fs_info);
+ if (rpc_populate(root, files, RPCAUTH_lockd, RPCAUTH_RootEOF, NULL)) {
+  kfree(sb->s_fs_info);
   return -ENOMEM;
+ }
+
  return 0;
 }

@@ -1016,11 +1049,25 @@ rpc_mount(struct file_system_type *fs_type,
  return mount_single(fs_type, flags, data, rpc_fill_super);
 }

+static void rpc_kill_sb(struct super_block *sb)
+{
+ struct list_head *destroy_cb_list = sb->s_fs_info;
+ struct destroy_cb *dcb, *tmp;
+
+ list_for_each_entry_safe(dcb, tmp, destroy_cb_list, list) {
+  dcb->callback(dcb->data);
+  list_del(&dcb->list);
+  kfree(dcb);
+ }
+ kfree(destroy_cb_list);
+ kill_litter_super(sb);
+}
+
 static struct file_system_type rpc_pipe_fs_type = {
  .owner  = THIS_MODULE,
  .name  = "rpc_pipefs",
  .mount  = rpc_mount,
- .kill_sb = kill_litter_super,
+ .kill_sb = rpc_kill_sb,
```

};

 static void
--
1.7.3.4

____
Containers mailing list
Containers@lists.linux-foundation.org
https://lists.linux-foundation.org/mailman/listinfo/containe rs

Subject: [PATCH v3 10/16] nfs: per-rpc_pipefs dns cache
Posted by Kirill A. Shutemov on Fri, 14 Jan 2011 13:49:08 GMT
View Forum Message <> Reply to Message

Lazy initialization of dns cache: on first call nfs_dns_resolve_name().
Every rpc_pipefs has separate dns cache now.

Signed-off-by: Kirill A. Shutemov <kas@openvz.org>

```
---
 fs/nfs/cache_lib.c     |   17 ++-----
 fs/nfs/cache_lib.h     |    3 +-
 fs/nfs/dns_resolve.c   |  137 ++++++++++++++++++++++++++++++++++++++----------
 fs/nfs/dns_resolve.h   |   15 +-----
 fs/nfs/inode.c         |    9 +---
 fs/nfs/nfs4namespace.c |    4 +-
 6 files changed, 117 insertions(+), 68 deletions(-)

diff --git a/fs/nfs/cache_lib.c b/fs/nfs/cache_lib.c
index 0944d4e..9b99d9e 100644
--- a/fs/nfs/cache_lib.c
+++ b/fs/nfs/cache_lib.c
@@ -12,7 +12,6 @@
 #include <linux/namei.h>
 #include <linux/slab.h>
 #include <linux/sunrpc/cache.h>
-#include <linux/sunrpc/rpc_pipe_fs.h>

 #include "cache_lib.h"

@@ -111,25 +110,17 @@ int nfs_cache_wait_for_upcall(struct nfs_cache_defer_req *dreq)
  return 0;
 }

-int nfs_cache_register(struct cache_detail *cd)
+int nfs_cache_register(struct cache_detail *cd, struct vfsmount *rpcmount)
 {
```

```
  struct nameidata nd;
- struct vfsmount *mnt;
  int ret;

- mnt = mntget(init_rpc_pipefs);
- if (IS_ERR(mnt))
-  return PTR_ERR(mnt);
- ret = vfs_path_lookup(mnt->mnt_root, mnt, "/cache", 0, &nd);
+ ret = vfs_path_lookup(rpcmount->mnt_root, rpcmount, "/cache", 0, &nd);
  if (ret)
-  goto err;
- ret = sunrpc_cache_register_pipefs(mnt, nd.path.dentry,
+  return ret;
+ ret = sunrpc_cache_register_pipefs(rpcmount, nd.path.dentry,
     cd->name, 0600, cd);
  path_put(&nd.path);
- if (!ret)
-  return ret;
-err:
- mntput(mnt);
  return ret;
 }

diff --git a/fs/nfs/cache_lib.h b/fs/nfs/cache_lib.h
index 76f856e..1d4a0a5 100644
--- a/fs/nfs/cache_lib.h
+++ b/fs/nfs/cache_lib.h
@@ -23,5 +23,6 @@ extern struct nfs_cache_defer_req *nfs_cache_defer_req_alloc(void);
 extern void nfs_cache_defer_req_put(struct nfs_cache_defer_req *dreq);
 extern int nfs_cache_wait_for_upcall(struct nfs_cache_defer_req *dreq);

-extern int nfs_cache_register(struct cache_detail *cd);
+extern int nfs_cache_register(struct cache_detail *cd,
+  struct vfsmount *rpcmount);
 extern void nfs_cache_unregister(struct cache_detail *cd);
diff --git a/fs/nfs/dns_resolve.c b/fs/nfs/dns_resolve.c
index a6e711a..a832e64 100644
--- a/fs/nfs/dns_resolve.c
+++ b/fs/nfs/dns_resolve.c
@@ -12,7 +12,7 @@
 #include <linux/dns_resolver.h>

 ssize_t nfs_dns_resolve_name(char *name, size_t namelen,
-  struct sockaddr *sa, size_t salen)
+  struct sockaddr *sa, size_t salen, struct vfsmount *rpcmount)
 {
  ssize_t ret;
  char *ip_addr = NULL;
```

```
@@ -37,9 +37,11 @@ ssize_t nfs_dns_resolve_name(char *name, size_t namelen,
 #include <linux/socket.h>
 #include <linux/seq_file.h>
 #include <linux/inet.h>
+#include <linux/mount.h>
 #include <linux/sunrpc/clnt.h>
 #include <linux/sunrpc/cache.h>
 #include <linux/sunrpc/svcauth.h>
+#include <linux/sunrpc/rpc_pipe_fs.h>

 #include "dns_resolve.h"
 #include "cache_lib.h"
@@ -47,7 +49,13 @@ ssize_t nfs_dns_resolve_name(char *name, size_t namelen,
 #define NFS_DNS_HASHBITS 4
 #define NFS_DNS_HASHTBL_SIZE (1 << NFS_DNS_HASHBITS)

-static struct cache_head *nfs_dns_table[NFS_DNS_HASHTBL_SIZE];
+static DEFINE_SPINLOCK(nfs_dns_resolve_lock);
+static LIST_HEAD(nfs_dns_resolve_list);
+
+struct nfs_dns_resolve_list {
+ struct list_head list;
+ struct cache_detail *cd;
+};

 struct nfs_dns_ent {
  struct cache_head h;
@@ -259,21 +267,6 @@ out:
  return ret;
 }

-static struct cache_detail nfs_dns_resolve = {
- .owner = THIS_MODULE,
- .hash_size = NFS_DNS_HASHTBL_SIZE,
- .hash_table = nfs_dns_table,
- .name = "dns_resolve",
- .cache_put = nfs_dns_ent_put,
- .cache_upcall = nfs_dns_upcall,
- .cache_parse = nfs_dns_parse,
- .cache_show = nfs_dns_show,
- .match = nfs_dns_match,
- .init = nfs_dns_ent_init,
- .update = nfs_dns_ent_update,
- .alloc = nfs_dns_ent_alloc,
-};
-
 static int do_cache_lookup(struct cache_detail *cd,
   struct nfs_dns_ent *key,
```

```
   struct nfs_dns_ent **item,
@@ -336,37 +329,119 @@ out:
  return ret;
 }

+static struct cache_detail *nfs_alloc_dns_resolve(void)
+{
+ struct cache_detail *dns_resolve;
+ struct cache_head **hash_table;
+
+ dns_resolve = kmalloc(sizeof(*dns_resolve), GFP_KERNEL);
+ if (!dns_resolve)
+  return NULL;
+
+ hash_table = kmalloc(sizeof(*hash_table) * NFS_DNS_HASHTBL_SIZE,
+    GFP_KERNEL);
+ if (!hash_table) {
+  kfree(dns_resolve);
+  return NULL;
+ }
+
+ dns_resolve->owner = THIS_MODULE;
+ dns_resolve->hash_size = NFS_DNS_HASHTBL_SIZE;
+ dns_resolve->hash_table = hash_table;
+ dns_resolve->name = "dns_resolve";
+ dns_resolve->cache_put = nfs_dns_ent_put;
+ dns_resolve->cache_upcall = nfs_dns_upcall;
+ dns_resolve->cache_parse = nfs_dns_parse;
+ dns_resolve->cache_show = nfs_dns_show;
+ dns_resolve->match = nfs_dns_match;
+ dns_resolve->init = nfs_dns_ent_init;
+ dns_resolve->update = nfs_dns_ent_update;
+ dns_resolve->alloc = nfs_dns_ent_alloc;
+
+ return dns_resolve;
+}
+
+static void nfs_free_dns_resolve(struct cache_detail *dns_resolve)
+{
+ kfree(dns_resolve->hash_table);
+ kfree(dns_resolve);
+}
+
+static struct cache_detail *nfs_get_dns_resolve(struct vfsmount *rpcmount)
+{
+ struct nfs_dns_resolve_list *dns_resolve;
+ int error = 0;
+
```

```
+ spin_lock(&nfs_dns_resolve_lock);
+ list_for_each_entry(dns_resolve, &nfs_dns_resolve_list, list) {
+  if (dns_resolve->cd->u.pipefs.mnt->mnt_sb != rpcmount->mnt_sb)
+   continue;
+
+  spin_unlock(&nfs_dns_resolve_lock);
+  return dns_resolve->cd;
+ }
+
+ dns_resolve = kmalloc(sizeof(*dns_resolve), GFP_KERNEL);
+ if (dns_resolve)
+  dns_resolve->cd = nfs_alloc_dns_resolve();
+ if (!dns_resolve || !dns_resolve->cd) {
+  error = -ENOMEM;
+  goto err;
+ }
+
+ error = nfs_cache_register(dns_resolve->cd, rpcmount);
+ if (error)
+  goto err;
+
+ INIT_LIST_HEAD(&dns_resolve->list);
+ list_add(&dns_resolve->list, &nfs_dns_resolve_list);
+ spin_unlock(&nfs_dns_resolve_lock);
+
+ return dns_resolve->cd;
+err:
+ spin_unlock(&nfs_dns_resolve_lock);
+ if (dns_resolve)
+  kfree(dns_resolve->cd);
+ kfree(dns_resolve);
+ return dns_resolve->cd;
+}
+
+static void nfs_dns_resolver_destroy(void *data)
+{
+ struct nfs_dns_resolve_list *dns_resolve = data;
+
+ spin_lock(&nfs_dns_resolve_lock);
+ nfs_cache_unregister(dns_resolve->cd);
+ nfs_free_dns_resolve(dns_resolve->cd);
+ list_del(&dns_resolve->list);
+ kfree(dns_resolve);
+ spin_unlock(&nfs_dns_resolve_lock);
+}
+
 ssize_t nfs_dns_resolve_name(char *name, size_t namelen,
-  struct sockaddr *sa, size_t salen)
```

```
+  struct sockaddr *sa, size_t salen, struct vfsmount *rpcmount)
 {
  struct nfs_dns_ent key = {
   .hostname = name,
   .namelen = namelen,
  };
+ struct cache_detail *dns_resolve;
  struct nfs_dns_ent *item = NULL;
  ssize_t ret;

- ret = do_cache_lookup_wait(&nfs_dns_resolve, &key, &item);
+ dns_resolve = nfs_get_dns_resolve(rpcmount);
+ ret = do_cache_lookup_wait(dns_resolve, &key, &item);
  if (ret == 0) {
   if (salen >= item->addrlen) {
    memcpy(sa, &item->addr, item->addrlen);
    ret = item->addrlen;
   } else
    ret = -EOVERFLOW;
-  cache_put(&item->h, &nfs_dns_resolve);
+  cache_put(&item->h, dns_resolve);
+  rpc_pipefs_add_destroy_cb(rpcmount->mnt_sb,
+    nfs_dns_resolver_destroy, dns_resolve);
  } else if (ret == -ENOENT)
   ret = -ESRCH;
  return ret;
 }
-
-int nfs_dns_resolver_init(void)
-{
- return nfs_cache_register(&nfs_dns_resolve);
-}
-
-void nfs_dns_resolver_destroy(void)
-{
- nfs_cache_unregister(&nfs_dns_resolve);
-}
-
 #endif
diff --git a/fs/nfs/dns_resolve.h b/fs/nfs/dns_resolve.h
index 199bb55..74ade60 100644
--- a/fs/nfs/dns_resolve.h
+++ b/fs/nfs/dns_resolve.h
@@ -7,20 +7,7 @@
 #define NFS_DNS_HOSTNAME_MAXLEN (128)


-#ifdef CONFIG_NFS_USE_KERNEL_DNS
```

```
-static inline int nfs_dns_resolver_init(void)
-{
- return 0;
-}
-
-static inline void nfs_dns_resolver_destroy(void)
-{}
-#else
-extern int nfs_dns_resolver_init(void);
-extern void nfs_dns_resolver_destroy(void);
-#endif
-
 extern ssize_t nfs_dns_resolve_name(char *name, size_t namelen,
-  struct sockaddr *sa, size_t salen);
+  struct sockaddr *sa, size_t salen, struct vfsmount *rpcmount);

 #endif
diff --git a/fs/nfs/inode.c b/fs/nfs/inode.c
index ce00b70..414570b 100644
--- a/fs/nfs/inode.c
+++ b/fs/nfs/inode.c
@@ -1535,10 +1535,6 @@ static int __init init_nfs_fs(void)

  err = nfs_idmap_init();
  if (err < 0)
-  goto out9;
-
- err = nfs_dns_resolver_init();
- if (err < 0)
   goto out8;

  err = nfs_fscache_register();
@@ -1599,10 +1595,8 @@ out5:
 out6:
  nfs_fscache_unregister();
 out7:
- nfs_dns_resolver_destroy();
-out8:
  nfs_idmap_quit();
-out9:
+out8:
  return err;
 }

@@ -1614,7 +1608,6 @@ static void __exit exit_nfs_fs(void)
  nfs_destroy_inodecache();
  nfs_destroy_nfspagecache();
  nfs_fscache_unregister();
```

```
- nfs_dns_resolver_destroy();
  nfs_idmap_quit();
 #ifdef CONFIG_PROC_FS
  rpc_proc_unregister("nfs");
diff --git a/fs/nfs/nfs4namespace.c b/fs/nfs/nfs4namespace.c
index 3c2a172..7a61fdb 100644
--- a/fs/nfs/nfs4namespace.c
+++ b/fs/nfs/nfs4namespace.c
@@ -14,6 +14,7 @@
 #include <linux/slab.h>
 #include <linux/string.h>
 #include <linux/sunrpc/clnt.h>
+#include <linux/sunrpc/rpc_pipe_fs.h>
 #include <linux/vfs.h>
 #include <linux/inet.h>
 #include "internal.h"
@@ -104,7 +105,8 @@ static size_t nfs_parse_server_name(char *string, size_t len,

  ret = rpc_pton(string, len, sa, salen);
  if (ret == 0) {
-  ret = nfs_dns_resolve_name(string, len, sa, salen);
+  ret = nfs_dns_resolve_name(string, len, sa, salen,
+     init_rpc_pipefs);
   if (ret < 0)
    ret = 0;
  }
--
1.7.3.4
```

Subject: [PATCH v3 13/16] nfs: introduce mount option 'rpcmount'
Posted by Kirill A. Shutemov on Fri, 14 Jan 2011 13:49:11 GMT
View Forum Message <> Reply to Message

It specifies rpc_pipefs to use. /var/lib/nfs/rpc_pipefs, by default.

Signed-off-by: Kirill A. Shutemov <kas@openvz.org>

```
 fs/nfs/callback.c        |    5 ++-
 fs/nfs/callback.h        |    3 +-
 fs/nfs/client.c          |   46 ++++++++++++++++++++++++++++++++++++++--------
 fs/nfs/internal.h        |   10 +++++++---
 fs/nfs/mount_clnt.c      |    3 +-
```

```
 fs/nfs/namespace.c        |    3 +-
 fs/nfs/nfs4namespace.c    |   22 ++++++++++---------
 fs/nfs/super.c            |   20 ++++++++++++++++++
 include/linux/nfs_fs_sb.h |    1 +
 9 files changed, 86 insertions(+), 27 deletions(-)

diff --git a/fs/nfs/callback.c b/fs/nfs/callback.c
index 7a535c8..c9814fb 100644
--- a/fs/nfs/callback.c
+++ b/fs/nfs/callback.c
@@ -276,7 +276,8 @@ int nfs4_set_callback_sessionid(struct nfs_client *clp)
 /*
  * Bring up the callback thread if it is not already up.
  */
-int nfs_callback_up(u32 minorversion, struct rpc_xprt *xprt)
+int nfs_callback_up(u32 minorversion, struct rpc_xprt *xprt,
+  struct vfsmount *rpcmount)
 {
  struct svc_serv *serv = NULL;
  struct svc_rqst *rqstp;
@@ -291,7 +292,7 @@ int nfs_callback_up(u32 minorversion, struct rpc_xprt *xprt)
   nfs_callback_bc_serv(minorversion, xprt, cb_info);
   goto out;
  }
- serv = svc_create(&nfs4_callback_program, init_rpc_pipefs,
+ serv = svc_create(&nfs4_callback_program, rpcmount,
    NFS4_CALLBACK_BUFSIZE, NULL);
  if (!serv) {
   ret = -ENOMEM;
diff --git a/fs/nfs/callback.h b/fs/nfs/callback.h
index d3b44f9..9496e0f 100644
--- a/fs/nfs/callback.h
+++ b/fs/nfs/callback.h
@@ -175,7 +175,8 @@ extern __be32 nfs4_callback_getattr(struct cb_getattrargs *args,
 extern __be32 nfs4_callback_recall(struct cb_recallargs *args, void *dummy,
       struct cb_process_state *cps);
 #ifdef CONFIG_NFS_V4
-extern int nfs_callback_up(u32 minorversion, struct rpc_xprt *xprt);
+extern int nfs_callback_up(u32 minorversion, struct rpc_xprt *xprt,
+  struct vfsmount *rpcmount);
 extern void nfs_callback_down(int minorversion);
 extern int nfs4_validate_delegation_stateid(struct nfs_delegation *delegation,
         const nfs4_stateid *stateid);
diff --git a/fs/nfs/client.c b/fs/nfs/client.c
index ad3b5e8..8dd0e99 100644
--- a/fs/nfs/client.c
+++ b/fs/nfs/client.c
@@ -131,6 +131,7 @@ struct nfs_client_initdata {
```

```
  const struct nfs_rpc_ops *rpc_ops;
  int proto;
  u32 minorversion;
+ struct vfsmount *rpcmount;
 };

 /*
@@ -167,6 +168,7 @@ static struct nfs_client *nfs_alloc_client(const struct nfs_client_initdata
*cl_
  clp->cl_rpcclient = ERR_PTR(-EINVAL);

  clp->cl_proto = cl_init->proto;
+ clp->cl_rpcmount = mntget(cl_init->rpcmount);

 #ifdef CONFIG_NFS_V4
  err = nfs_get_cb_ident_idr(clp, cl_init->minorversion);
@@ -286,6 +288,7 @@ static void nfs_free_client(struct nfs_client *clp)
  if (clp->cl_machine_cred != NULL)
   put_rpccred(clp->cl_machine_cred);

+ mntput(clp->cl_rpcmount);
  kfree(clp->cl_hostname);
  kfree(clp);

@@ -471,6 +474,9 @@ static struct nfs_client *nfs_match_client(const struct nfs_client_initdata
*dat
   /* Match the full socket address */
   if (!nfs_sockaddr_cmp(sap, clap))
    continue;
+  /* Match rpc_pipefs mount point */
+  if (clp->cl_rpcmount->mnt_sb != data->rpcmount->mnt_sb)
+   continue;

   atomic_inc(&clp->cl_count);
   return clp;
@@ -629,7 +635,7 @@ static int nfs_create_rpc_client(struct nfs_client *clp,
   .program = &nfs_program,
   .version = clp->rpc_ops->version,
   .authflavor = flavor,
-  .rpcmount = init_rpc_pipefs,
+  .rpcmount = clp->cl_rpcmount,
  };

  if (discrtry)
@@ -664,7 +670,7 @@ static void nfs_destroy_server(struct nfs_server *server)
 /*
  * Version 2 or 3 lockd setup
  */
```

```
-static int nfs_start_lockd(struct nfs_server *server)
+static int nfs_start_lockd(struct nfs_server *server, struct vfsmount *rpcmount)
 {
  struct nlm_host *host;
  struct nfs_client *clp = server->nfs_client;
@@ -675,7 +681,7 @@ static int nfs_start_lockd(struct nfs_server *server)
    .nfs_version = clp->rpc_ops->version,
    .noresvport = server->flags & NFS_MOUNT_NORESVPORT ?
      1 : 0,
-   .rpcmount = init_rpc_pipefs,
+   .rpcmount = rpcmount,
  };

  if (nlm_init.nfs_version > 3)
@@ -823,8 +829,16 @@ static int nfs_init_server(struct nfs_server *server,
   cl_init.rpc_ops = &nfs_v3_clientops;
 #endif

+ cl_init.rpcmount = get_rpc_pipefs(data->rpcmount);
+ if (IS_ERR(cl_init.rpcmount)) {
+  dprintk("<-- nfs_init_server() = error %ld\n",
+    PTR_ERR(cl_init.rpcmount));
+  return PTR_ERR(cl_init.rpcmount);
+ }
+
  /* Allocate or find a client reference we can use */
  clp = nfs_get_client(&cl_init);
+ mntput(cl_init.rpcmount);
  if (IS_ERR(clp)) {
   dprintk("<-- nfs_init_server() = error %ld\n", PTR_ERR(clp));
   return PTR_ERR(clp);
@@ -856,7 +870,7 @@ static int nfs_init_server(struct nfs_server *server,
  server->acdirmax = data->acdirmax * HZ;

  /* Start lockd here, before we might error out */
- error = nfs_start_lockd(server);
+ error = nfs_start_lockd(server, clp->cl_rpcmount);
  if (error < 0)
   goto error;

@@ -1270,7 +1284,8 @@ static int nfs4_init_callback(struct nfs_client *clp)
   }

   error = nfs_callback_up(clp->cl_mvops->minor_version,
-     clp->cl_rpcclient->cl_xprt);
+     clp->cl_rpcclient->cl_xprt,
+     clp->cl_rpcmount);
   if (error < 0) {
```

```
    dprintk("%s: failed to start callback. Error = %d\n",
     __func__, error);
@@ -1370,7 +1385,8 @@ static int nfs4_set_client(struct nfs_server *server,
   const char *ip_addr,
   rpc_authflavor_t authflavour,
   int proto, const struct rpc_timeout *timeparms,
-  u32 minorversion)
+  u32 minorversion,
+  struct vfsmount *rpcmount)
 {
  struct nfs_client_initdata cl_init = {
   .hostname = hostname,
@@ -1379,6 +1395,7 @@ static int nfs4_set_client(struct nfs_server *server,
   .rpc_ops = &nfs_v4_clientops,
   .proto = proto,
   .minorversion = minorversion,
+  .rpcmount = rpcmount,
  };
  struct nfs_client *clp;
  int error;
@@ -1485,6 +1502,7 @@ static int nfs4_init_server(struct nfs_server *server,
   const struct nfs_parsed_mount_data *data)
 {
  struct rpc_timeout timeparms;
+ struct vfsmount *rpcmount;
  int error;

  dprintk("--> nfs4_init_server()\n");
@@ -1499,6 +1517,11 @@ static int nfs4_init_server(struct nfs_server *server,
    server->caps |= NFS_CAP_READDIRPLUS;
  server->options = data->options;

+ rpcmount = get_rpc_pipefs(data->rpcmount);
+ if (IS_ERR(rpcmount)) {
+  error = PTR_ERR(rpcmount);
+  goto error;
+ }
  /* Get a client record */
  error = nfs4_set_client(server,
    data->nfs_server.hostname,
@@ -1508,7 +1531,9 @@ static int nfs4_init_server(struct nfs_server *server,
    data->auth_flavors[0],
    data->nfs_server.protocol,
    &timeparms,
-   data->minorversion);
+   data->minorversion,
+   rpcmount);
+ mntput(rpcmount);
```

```
  if (error < 0)
   goto error;

@@ -1598,7 +1623,10 @@ struct nfs_server *nfs4_create_referral_server(struct
nfs_clone_mount *data,
      data->authflavor,
      parent_server->client->cl_xprt->prot,
      parent_server->client->cl_timeout,
-     parent_client->cl_mvops->minor_version);
+     parent_client->cl_mvops->minor_version,
+     parent_client->cl_rpcmount);
+
+
  if (error < 0)
   goto error;

@@ -1672,7 +1700,7 @@ struct nfs_server *nfs_clone_server(struct nfs_server *source,
   (unsigned long long) server->fsid.major,
   (unsigned long long) server->fsid.minor);

- error = nfs_start_lockd(server);
+ error = nfs_start_lockd(server, server->nfs_client->cl_rpcmount);
  if (error < 0)
   goto out_free_server;

diff --git a/fs/nfs/internal.h b/fs/nfs/internal.h
index bfa3a34..b9554d7 100644
--- a/fs/nfs/internal.h
+++ b/fs/nfs/internal.h
@@ -86,6 +86,7 @@ struct nfs_parsed_mount_data {
  unsigned int  version;
  unsigned int  minorversion;
  char   *fscache_uniq;
+ char   *rpcmount;

  struct {
   struct sockaddr_storage address;
@@ -120,6 +121,7 @@ struct nfs_mount_request {
  int   noresvport;
  unsigned int  *auth_flav_len;
  rpc_authflavor_t *auth_flavs;
+ struct vfsmount  *rpcmount;
 };

 extern int nfs_mount(struct nfs_mount_request *info);
@@ -164,10 +166,14 @@ static inline void nfs_fs_proc_exit(void)

 /* nfs4namespace.c */
```

```
 #ifdef CONFIG_NFS_V4
-extern struct vfsmount *nfs_do_refmount(const struct vfsmount *mnt_parent, struct dentry
*dentry);
+extern struct vfsmount *nfs_do_refmount(const struct vfsmount *mnt_parent,
+  struct dentry *dentry,
+  struct vfsmount *rpcmount);
 #else
 static inline
-struct vfsmount *nfs_do_refmount(const struct vfsmount *mnt_parent, struct dentry *dentry)
+struct vfsmount *nfs_do_refmount(const struct vfsmount *mnt_parent,
+  struct dentry *dentry,
+  struct vfsmount *rpcmount)
 {
  return ERR_PTR(-ENOENT);
 }
diff --git a/fs/nfs/mount_clnt.c b/fs/nfs/mount_clnt.c
index 6227875..205cc02 100644
--- a/fs/nfs/mount_clnt.c
+++ b/fs/nfs/mount_clnt.c
@@ -13,7 +13,6 @@
 #include <linux/in.h>
 #include <linux/sunrpc/clnt.h>
 #include <linux/sunrpc/sched.h>
-#include <linux/sunrpc/rpc_pipe_fs.h>
 #include <linux/nfs_fs.h>
 #include "internal.h"

@@ -162,7 +161,7 @@ int nfs_mount(struct nfs_mount_request *info)
   .program = &mnt_program,
   .version = info->version,
   .authflavor = RPC_AUTH_UNIX,
-  .rpcmount = init_rpc_pipefs,
+  .rpcmount = info->rpcmount,
  };
  struct rpc_clnt  *mnt_clnt;
  int   status;
diff --git a/fs/nfs/namespace.c b/fs/nfs/namespace.c
index 74aaf39..dad0129 100644
--- a/fs/nfs/namespace.c
+++ b/fs/nfs/namespace.c
@@ -144,7 +144,8 @@ static void * nfs_follow_mountpoint(struct dentry *dentry, struct
nameidata *nd)
   goto out_err;

  if (fattr->valid & NFS_ATTR_FATTR_V4_REFERRAL)
-  mnt = nfs_do_refmount(nd->path.mnt, nd->path.dentry);
+  mnt = nfs_do_refmount(nd->path.mnt, nd->path.dentry,
+    server->nfs_client->cl_rpcmount);
```

```
  else
   mnt = nfs_do_submount(nd->path.mnt, nd->path.dentry, fh,
         fattr);
diff --git a/fs/nfs/nfs4namespace.c b/fs/nfs/nfs4namespace.c
index 7a61fdb..92d5d63 100644
--- a/fs/nfs/nfs4namespace.c
+++ b/fs/nfs/nfs4namespace.c
@@ -14,7 +14,6 @@
 #include <linux/slab.h>
 #include <linux/string.h>
 #include <linux/sunrpc/clnt.h>
-#include <linux/sunrpc/rpc_pipe_fs.h>
 #include <linux/vfs.h>
 #include <linux/inet.h>
 #include "internal.h"
@@ -99,14 +98,13 @@ static int nfs4_validate_fspath(const struct vfsmount *mnt_parent,
 }

 static size_t nfs_parse_server_name(char *string, size_t len,
-  struct sockaddr *sa, size_t salen)
+  struct sockaddr *sa, size_t salen, struct vfsmount *rpcmount)
 {
  ssize_t ret;

  ret = rpc_pton(string, len, sa, salen);
  if (ret == 0) {
-  ret = nfs_dns_resolve_name(string, len, sa, salen,
-    init_rpc_pipefs);
+  ret = nfs_dns_resolve_name(string, len, sa, salen, rpcmount);
   if (ret < 0)
    ret = 0;
  }
@@ -115,7 +113,8 @@ static size_t nfs_parse_server_name(char *string, size_t len,

 static struct vfsmount *try_location(struct nfs_clone_mount *mountdata,
         char *page, char *page2,
-        const struct nfs4_fs_location *location)
+        const struct nfs4_fs_location *location,
+        struct vfsmount *rpcmount)
 {
  const size_t addr_bufsize = sizeof(struct sockaddr_storage);
  struct vfsmount *mnt = ERR_PTR(-ENOENT);
@@ -143,7 +142,7 @@ static struct vfsmount *try_location(struct nfs_clone_mount *mountdata,
    continue;

   mountdata->addrlen = nfs_parse_server_name(buf->data, buf->len,
-    mountdata->addr, addr_bufsize);
+    mountdata->addr, addr_bufsize, rpcmount);
```

```
   if (mountdata->addrlen == 0)
    continue;

@@ -174,7 +173,8 @@ static struct vfsmount *try_location(struct nfs_clone_mount *mountdata,
  */
 static struct vfsmount *nfs_follow_referral(const struct vfsmount *mnt_parent,
          const struct dentry *dentry,
-         const struct nfs4_fs_locations *locations)
+         const struct nfs4_fs_locations *locations,
+         struct vfsmount *rpcmount)
 {
  struct vfsmount *mnt = ERR_PTR(-ENOENT);
  struct nfs_clone_mount mountdata = {
@@ -213,7 +213,7 @@ static struct vfsmount *nfs_follow_referral(const struct vfsmount
*mnt_parent,
       location->rootpath.ncomponents == 0)
     continue;

-  mnt = try_location(&mountdata, page, page2, location);
+  mnt = try_location(&mountdata, page, page2, location, rpcmount);
    if (!IS_ERR(mnt))
     break;
  }
@@ -231,7 +231,9 @@ out:
  * @dentry - dentry of referral
  *
  */
-struct vfsmount *nfs_do_refmount(const struct vfsmount *mnt_parent, struct dentry *dentry)
+struct vfsmount *nfs_do_refmount(const struct vfsmount *mnt_parent,
+  struct dentry *dentry,
+  struct vfsmount *rpcmount)
 {
  struct vfsmount *mnt = ERR_PTR(-ENOMEM);
  struct dentry *parent;
@@ -264,7 +266,7 @@ struct vfsmount *nfs_do_refmount(const struct vfsmount *mnt_parent,
struct dentr
      fs_locations->fs_path.ncomponents <= 0)
   goto out_free;

- mnt = nfs_follow_referral(mnt_parent, dentry, fs_locations);
+ mnt = nfs_follow_referral(mnt_parent, dentry, fs_locations, rpcmount);
 out_free:
  __free_page(page);
  kfree(fs_locations);
diff --git a/fs/nfs/super.c b/fs/nfs/super.c
index b68c860..be4852b 100644
--- a/fs/nfs/super.c
+++ b/fs/nfs/super.c
```

```
@@ -35,6 +35,7 @@
 #include <linux/sunrpc/metrics.h>
 #include <linux/sunrpc/xprtsock.h>
 #include <linux/sunrpc/xprtrdma.h>
+#include <linux/sunrpc/rpc_pipe_fs.h>
 #include <linux/nfs_fs.h>
 #include <linux/nfs_mount.h>
 #include <linux/nfs4_mount.h>
@@ -106,6 +107,7 @@ enum {
  Opt_lookupcache,
  Opt_fscache_uniq,
  Opt_local_lock,
+ Opt_rpcmount,

  /* Special mount options */
  Opt_userspace, Opt_deprecated, Opt_sloppy,
@@ -178,6 +180,7 @@ static const match_table_t nfs_mount_option_tokens = {
  { Opt_lookupcache, "lookupcache=%s" },
  { Opt_fscache_uniq, "fsc=%s" },
  { Opt_local_lock, "local_lock=%s" },
+ { Opt_rpcmount, "rpcmount=%s" },

  { Opt_err, NULL }
 };
@@ -1486,6 +1489,13 @@ static int nfs_parse_mount_options(char *raw,
     return 0;
    };
    break;
+  case Opt_rpcmount:
+   string = match_strdup(args);
+   if (string == NULL)
+    goto out_nomem;
+   kfree(mnt->rpcmount);
+   mnt->rpcmount = string;
+   break;

   /*
    * Special options
@@ -1646,11 +1656,19 @@ static int nfs_try_mount(struct nfs_parsed_mount_data *args,
  request.salen = args->mount_server.addrlen;
  nfs_set_port(request.sap, &args->mount_server.port, 0);

+ request.rpcmount = get_rpc_pipefs(args->rpcmount);
+ if (IS_ERR(request.rpcmount)) {
+  dfprintk(MOUNT, "NFS: unable get rpc_pipefs mount point, "
+    "error %ld\n", PTR_ERR(request.rpcmount));
+  return PTR_ERR(request.rpcmount);
+ }
```

```
+
  /*
   * Now ask the mount server to map our export path
   * to a file handle.
   */
  status = nfs_mount(&request);
+ mntput(request.rpcmount);
  if (status != 0) {
   dfprintk(MOUNT, "NFS: unable to mount server %s, error %d\n",
     request.hostname, status);
@@ -2355,6 +2373,7 @@ out:
  kfree(data->nfs_server.hostname);
  kfree(data->mount_server.hostname);
  kfree(data->fscache_uniq);
+ kfree(data->rpcmount);
  security_free_mnt_opts(&data->lsm_opts);
 out_free_fh:
  nfs_free_fhandle(mntfh);
@@ -2962,6 +2981,7 @@ out:
  kfree(data->nfs_server.export_path);
  kfree(data->nfs_server.hostname);
  kfree(data->fscache_uniq);
+ kfree(data->rpcmount);
 out_free_data:
  kfree(data);
  dprintk("<-- nfs4_get_sb() = %d%s\n", error,
diff --git a/include/linux/nfs_fs_sb.h b/include/linux/nfs_fs_sb.h
index b197563..ad8d913 100644
--- a/include/linux/nfs_fs_sb.h
+++ b/include/linux/nfs_fs_sb.h
@@ -36,6 +36,7 @@ struct nfs_client {
  struct list_head cl_share_link; /* link in global client list */
  struct list_head cl_superblocks; /* List of nfs_server structs */

+ struct vfsmount  *cl_rpcmount; /* rpc_pipefs mount point */
  struct rpc_clnt * cl_rpcclient;
  const struct nfs_rpc_ops *rpc_ops; /* NFS protocol vector */
  int   cl_proto; /* Network transport protocol */
--
1.7.3.4
```

_____

Containers mailing list
Containers@lists.linux-foundation.org
https://lists.linux-foundation.org/mailman/listinfo/containe rs

Subject: [PATCH v3 15/16] sunrpc: remove global init_rpc_pipefs
Posted by Kirill A. Shutemov on Fri, 14 Jan 2011 13:49:13 GMT
View Forum Message <> Reply to Message

Replace remaining init_rpc_pipefs references with get_rpc_pipefs(NULL)
and make init_rpc_pipefs static.

Signed-off-by: Kirill A. Shutemov <kas@openvz.org>
---
 fs/nfsd/nfs4callback.c | 5 ++++-
 fs/nfsd/nfssvc.c | 16 ++++++++++++++--
 include/linux/sunrpc/rpc_pipe_fs.h | 2 --
 net/sunrpc/rpc_pipe.c | 1 -
 4 files changed, 18 insertions(+), 6 deletions(-)

```
diff --git a/fs/nfsd/nfs4callback.c b/fs/nfsd/nfs4callback.c
index 9bec643..61210b6 100644
--- a/fs/nfsd/nfs4callback.c
+++ b/fs/nfsd/nfs4callback.c
@@ -647,7 +647,6 @@ int setup_callback_client(struct nfs4_client *clp, struct nfs4_cb_conn
*conn)
   .version = 0,
   .authflavor = clp->cl_flavor,
   .flags  = (RPC_CLNT_CREATE_NOPING | RPC_CLNT_CREATE_QUIET),
-  .rpcmount = init_rpc_pipefs,
  };
  struct rpc_clnt *client;

@@ -663,8 +662,12 @@ int setup_callback_client(struct nfs4_client *clp, struct nfs4_cb_conn
*conn)
   args.prognumber = clp->cl_cb_session->se_cb_prog;
   args.protocol = XPRT_TRANSPORT_BC_TCP;
  }
+ args.rpcmount = get_rpc_pipefs(NULL);
+ if (IS_ERR(args.rpcmount))
+  return PTR_ERR(args.rpcmount);
  /* Create RPC client */
  client = rpc_create(&args);
+ mntput(args.rpcmount);
  if (IS_ERR(client)) {
   dprintk("NFSD: couldn't create callback client: %ld\n",
    PTR_ERR(client));
diff --git a/fs/nfsd/nfssvc.c b/fs/nfsd/nfssvc.c
index 17d78d3..7353420 100644
--- a/fs/nfsd/nfssvc.c
+++ b/fs/nfsd/nfssvc.c
@@ -206,6 +206,7 @@ static bool nfsd_up = false;
 static int nfsd_startup(unsigned short port, int nrservs)
 {
```

```
  int ret;
+ struct vfsmount *rpcmount;

  if (nfsd_up)
   return 0;
@@ -220,7 +221,13 @@ static int nfsd_startup(unsigned short port, int nrservs)
  ret = nfsd_init_socks(port);
  if (ret)
   goto out_racache;
- ret = lockd_up(init_rpc_pipefs);
+ rpcmount = get_rpc_pipefs(NULL);
+ if (IS_ERR(rpcmount)) {
+  ret = PTR_ERR(rpcmount);
+  goto out_racache;
+ }
+ ret = lockd_up(rpcmount);
+ mntput(rpcmount);
  if (ret)
   goto out_racache;
  ret = nfs4_state_start();
@@ -308,6 +315,7 @@ static void set_max_drc(void)
 int nfsd_create_serv(void)
 {
  int err = 0;
+ struct vfsmount *rpcmount;

  WARN_ON(!mutex_is_locked(&nfsd_mutex));
  if (nfsd_serv) {
@@ -332,9 +340,13 @@ int nfsd_create_serv(void)
  }
  nfsd_reset_versions();

- nfsd_serv = svc_create_pooled(&nfsd_program, init_rpc_pipefs,
+ rpcmount = get_rpc_pipefs(NULL);
+ if (IS_ERR(rpcmount))
+  return PTR_ERR(rpcmount);
+ nfsd_serv = svc_create_pooled(&nfsd_program, rpcmount,
          nfsd_max_blksize, nfsd_last_thread, nfsd,
          THIS_MODULE);
+ mntput(rpcmount);
  if (nfsd_serv == NULL)
   return -ENOMEM;

diff --git a/include/linux/sunrpc/rpc_pipe_fs.h b/include/linux/sunrpc/rpc_pipe_fs.h
index 4a8830a..a0b9c46 100644
--- a/include/linux/sunrpc/rpc_pipe_fs.h
+++ b/include/linux/sunrpc/rpc_pipe_fs.h
@@ -44,8 +44,6 @@ RPC_I(struct inode *inode)
```

```
 return container_of(inode, struct rpc_inode, vfs_inode);
}

-extern struct vfsmount *init_rpc_pipefs;
-
 extern struct vfsmount *get_rpc_pipefs(const char *path);
 extern int rpc_pipefs_add_destroy_cb(struct super_block *sb,
   void (*destroy_cb)(void *data), void *data);
diff --git a/net/sunrpc/rpc_pipe.c b/net/sunrpc/rpc_pipe.c
index 02416f1..96973c0 100644
--- a/net/sunrpc/rpc_pipe.c
+++ b/net/sunrpc/rpc_pipe.c
@@ -33,7 +33,6 @@
 #include <linux/sunrpc/cache.h>

 struct vfsmount *init_rpc_pipefs __read_mostly;
-EXPORT_SYMBOL_GPL(init_rpc_pipefs);

 static struct file_system_type rpc_pipe_fs_type;

--
1.7.3.4
```

Containers mailing list
Containers@lists.linux-foundation.org
https://lists.linux-foundation.org/mailman/listinfo/containe rs

---

Subject: [PATCH v3 16/16] Rework get_rpc_pipefs and introduce put_rpc_pipefs()

Posted by Kirill A. Shutemov on Fri, 14 Jan 2011 13:49:14 GMT

View Forum Message <> Reply to Message

Now sunrpc module can be removed normally.

Signed-off-by: Kirill A. Shutemov <kas@openvz.org>

```
---
 fs/nfs/client.c                  |    4 +-
 fs/nfs/super.c                   |    2 +-
 fs/nfsd/nfs4callback.c           |    2 +-
 fs/nfsd/nfssvc.c                 |    4 +-
 include/linux/sunrpc/rpc_pipe_fs.h |    1 +
 net/sunrpc/rpc_pipe.c            |   51 ++++++++++++++++++++++++++------------
 6 files changed, 40 insertions(+), 24 deletions(-)

diff --git a/fs/nfs/client.c b/fs/nfs/client.c
index 8dd0e99..0b8ac12 100644
--- a/fs/nfs/client.c
```

```
+++ b/fs/nfs/client.c
@@ -838,7 +838,7 @@ static int nfs_init_server(struct nfs_server *server,

  /* Allocate or find a client reference we can use */
  clp = nfs_get_client(&cl_init);
- mntput(cl_init.rpcmount);
+ put_rpc_pipefs(cl_init.rpcmount);
  if (IS_ERR(clp)) {
   dprintk("<-- nfs_init_server() = error %ld\n", PTR_ERR(clp));
   return PTR_ERR(clp);
@@ -1533,7 +1533,7 @@ static int nfs4_init_server(struct nfs_server *server,
    &timeparms,
    data->minorversion,
    rpcmount);
- mntput(rpcmount);
+ put_rpc_pipefs(rpcmount);
  if (error < 0)
   goto error;

diff --git a/fs/nfs/super.c b/fs/nfs/super.c
index be4852b..148843e 100644
--- a/fs/nfs/super.c
+++ b/fs/nfs/super.c
@@ -1668,7 +1668,7 @@ static int nfs_try_mount(struct nfs_parsed_mount_data *args,
   * to a file handle.
   */
  status = nfs_mount(&request);
- mntput(request.rpcmount);
+ put_rpc_pipefs(request.rpcmount);
  if (status != 0) {
   dfprintk(MOUNT, "NFS: unable to mount server %s, error %d\n",
     request.hostname, status);
diff --git a/fs/nfsd/nfs4callback.c b/fs/nfsd/nfs4callback.c
index 61210b6..4d8383b 100644
--- a/fs/nfsd/nfs4callback.c
+++ b/fs/nfsd/nfs4callback.c
@@ -667,7 +667,7 @@ int setup_callback_client(struct nfs4_client *clp, struct nfs4_cb_conn
*conn)
   return PTR_ERR(args.rpcmount);
  /* Create RPC client */
  client = rpc_create(&args);
- mntput(args.rpcmount);
+ put_rpc_pipefs(args.rpcmount);
  if (IS_ERR(client)) {
   dprintk("NFSD: couldn't create callback client: %ld\n",
    PTR_ERR(client));
diff --git a/fs/nfsd/nfssvc.c b/fs/nfsd/nfssvc.c
index 7353420..cff49fc 100644
```

```
--- a/fs/nfsd/nfssvc.c
+++ b/fs/nfsd/nfssvc.c
@@ -227,7 +227,7 @@ static int nfsd_startup(unsigned short port, int nrservs)
   goto out_racache;
  }
  ret = lockd_up(rpcmount);
- mntput(rpcmount);
+ put_rpc_pipefs(rpcmount);
  if (ret)
   goto out_racache;
  ret = nfs4_state_start();
@@ -346,7 +346,7 @@ int nfsd_create_serv(void)
  nfsd_serv = svc_create_pooled(&nfsd_program, rpcmount,
          nfsd_max_blksize, nfsd_last_thread, nfsd,
          THIS_MODULE);
- mntput(rpcmount);
+ put_rpc_pipefs(rpcmount);
  if (nfsd_serv == NULL)
   return -ENOMEM;

diff --git a/include/linux/sunrpc/rpc_pipe_fs.h b/include/linux/sunrpc/rpc_pipe_fs.h
index a0b9c46..328b3da 100644
--- a/include/linux/sunrpc/rpc_pipe_fs.h
+++ b/include/linux/sunrpc/rpc_pipe_fs.h
@@ -45,6 +45,7 @@ RPC_I(struct inode *inode)
 }

 extern struct vfsmount *get_rpc_pipefs(const char *path);
+extern void put_rpc_pipefs(struct vfsmount *rpcmount);
 extern int rpc_pipefs_add_destroy_cb(struct super_block *sb,
   void (*destroy_cb)(void *data), void *data);

diff --git a/net/sunrpc/rpc_pipe.c b/net/sunrpc/rpc_pipe.c
index 96973c0..82feacd 100644
--- a/net/sunrpc/rpc_pipe.c
+++ b/net/sunrpc/rpc_pipe.c
@@ -33,6 +33,7 @@
 #include <linux/sunrpc/cache.h>

 struct vfsmount *init_rpc_pipefs __read_mostly;
+static int init_rpc_pipefs_count;

 static struct file_system_type rpc_pipe_fs_type;

@@ -1005,22 +1006,35 @@ static int check_rpc_pipefs(struct vfsmount *mnt, void *arg)
  return 1;
 }
```

```
-struct vfsmount *get_rpc_pipefs(const char *p)
+static struct vfsmount *find_rpc_pipefs(void)
 {
- int error;
  struct vfsmount *rpcmount = ERR_PTR(-EINVAL);
- struct path path;
+ int err;

- if (!p) {
-  iterate_mounts(check_rpc_pipefs, &rpcmount,
-    current->nsproxy->mnt_ns->root);
+ iterate_mounts(check_rpc_pipefs, &rpcmount,
+   current->nsproxy->mnt_ns->root);

-  if (IS_ERR(rpcmount) && (current->nsproxy->mnt_ns ==
-     init_task.nsproxy->mnt_ns))
-   return mntget(init_rpc_pipefs);
+ if (!IS_ERR(rpcmount))
+  return rpcmount;

+ if (current->nsproxy->mnt_ns != init_task.nsproxy->mnt_ns)
   return rpcmount;
- }
+
+ err = simple_pin_fs(&rpc_pipe_fs_type, &init_rpc_pipefs,
+   &init_rpc_pipefs_count);
+ if (err)
+  return ERR_PTR(err);
+ return init_rpc_pipefs;
+}
+
+struct vfsmount *get_rpc_pipefs(const char *p)
+{
+ int error;
+ struct vfsmount *rpcmount = ERR_PTR(-EINVAL);
+ struct path path;
+
+ if (!p)
+  return find_rpc_pipefs();

  error = kern_path(p, LOOKUP_FOLLOW | LOOKUP_DIRECTORY, &path);
  if (error)
@@ -1033,6 +1047,15 @@ struct vfsmount *get_rpc_pipefs(const char *p)
 }
 EXPORT_SYMBOL_GPL(get_rpc_pipefs);

+void put_rpc_pipefs(struct vfsmount *rpcmount)
+{
```

```
+ if (rpcmount == init_rpc_pipefs)
+  simple_release_fs(&init_rpc_pipefs, &init_rpc_pipefs_count);
+ else
+  mntput(rpcmount);
+}
+EXPORT_SYMBOL_GPL(put_rpc_pipefs);
+
 struct destroy_cb {
  struct list_head list;
  void (*callback)(void *data);
@@ -1232,16 +1255,8 @@ int register_rpc_pipefs(void)
  if (err)
   goto destroy_cache;

- init_rpc_pipefs = kern_mount(&rpc_pipe_fs_type);
- if (IS_ERR(init_rpc_pipefs)) {
-  err = PTR_ERR(init_rpc_pipefs);
-  goto unregister_fs;
- }
-
  return 0;

-unregister_fs:
- unregister_filesystem(&rpc_pipe_fs_type);
 destroy_cache:
  kmem_cache_destroy(rpc_inode_cachep);
  return err;
--
1.7.3.4
```

Subject: Re: [PATCH v3 00/16] make rpc_pipefs be mountable multiple time
Posted by Rob Landley on Mon, 17 Jan 2011 12:30:16 GMT
View Forum Message <> Reply to Message

On 01/14/2011 07:48 AM, Kirill A. Shutemov wrote:
> Prepare nfs/sunrpc stack to use multiple instances of rpc_pipefs.
> Only for client for now.

Ok, Google is being really unhelpful here.

What is rpc_pipefs for? What uses it, and to do what exactly? Is it used by nfs server code, or by the client code, or both? Is it a way

for userspace to talk to the kernel, or for the kernel to talk to
itself? Is it used at mount time, or during filesystem operation?

I'm interested in giving this patch series a much more thorough review,
but I can't figure out what the subsystem it's modifying actually _is_.

(Maybe this is something to do with filesystems/nfs/rpc-cache.txt?)

Rob

_____
Containers mailing list
Containers@lists.linux-foundation.org
https://lists.linux-foundation.org/mailman/listinfo/containe rs

---

Subject: Re: [PATCH v3 00/16] make rpc_pipefs be mountable multiple time
Posted by Kirill A. Shutemov on Thu, 20 Jan 2011 11:35:52 GMT
View Forum Message <> Reply to Message

On Mon, Jan 17, 2011 at 06:30:16AM -0600, Rob Landley wrote:
> On 01/14/2011 07:48 AM, Kirill A. Shutemov wrote:
> > Prepare nfs/sunrpc stack to use multiple instances of rpc_pipefs.
> > Only for client for now.
>
> Ok, Google is being really unhelpful here.

It's better if you read the code. :)
>
> What is rpc_pipefs for? What uses it, and to do what exactly? Is it
> used by nfs server code, or by the client code, or both? Is it a way
> for userspace to talk to the kernel, or for the kernel to talk to
> itself? Is it used at mount time, or during filesystem operation?

Ok, It try to answer. Please correct me, if I'm wrong.

rpc_pipefs is a userland/kernel interface (I don't see kernel-kernel
usecases, but it's possible, I guess).
There is client dir (nfs/clntX) in rpc_pipefs for every sunrpc client.
Both client and server (see fs/nfsd/nfs4callback.c) can create sunrpc
client. So we rpc_pipefs on both side.
rpc_pipefs uses not only on mount time. See old idmapper, for example.

> I'm interested in giving this patch series a much more thorough review,
> but I can't figure out what the subsystem it's modifying actually _is_.
>
> (Maybe this is something to do with filesystems/nfs/rpc-cache.txt?)
>
> Rob

--
Kirill A. Shutemov

_____
Containers mailing list
Containers@lists.linux-foundation.org
https://lists.linux-foundation.org/mailman/listinfo/containe rs

---

Subject: Re: [PATCH v3 09/16] sunrpc: introduce rpc_pipefs_add_destroy_cb()
Posted by Kirill A. Shutemov on Thu, 20 Jan 2011 11:37:22 GMT
View Forum Message <> Reply to Message

On Fri, Jan 14, 2011 at 03:49:07PM +0200, Kirill A. Shutemov wrote:
> Add facility to do some action on destroying of rpc_pipefs superblock.
>
> Signed-off-by: Kirill A. Shutemov <kas@openvz.org>
> ---
> include/linux/sunrpc/rpc_pipe_fs.h | 3 ++
> net/sunrpc/rpc_pipe.c | 51 ++-
> 2 files changed, 52 insertions(+), 2 deletions(-)
>
> diff --git a/include/linux/sunrpc/rpc_pipe_fs.h b/include/linux/sunrpc/rpc_pipe_fs.h
> index b09bfa5..f5216f1 100644
> --- a/include/linux/sunrpc/rpc_pipe_fs.h
> +++ b/include/linux/sunrpc/rpc_pipe_fs.h
> @@ -46,6 +46,9 @@ RPC_I(struct inode *inode)
>
> extern struct vfsmount *init_rpc_pipefs;
>
> +extern int rpc_pipefs_add_destroy_cb(struct super_block *sb,
> + void (*destroy_cb)(void *data), void *data);
> +
> extern int rpc_queue_upcall(struct inode *, struct rpc_pipe_msg *);
>
> struct rpc_clnt;
> diff --git a/net/sunrpc/rpc_pipe.c b/net/sunrpc/rpc_pipe.c
> index 484c9a3..39511e3 100644
> --- a/net/sunrpc/rpc_pipe.c
> +++ b/net/sunrpc/rpc_pipe.c
> @@ -939,6 +939,31 @@ static const struct super_operations s_ops = {
>
> #define RPCAUTH_GSSMAGIC 0x67596969
>
> +struct destroy_cb {
> + struct list_head list;
> + void (*callback)(void *data);
> + void *data;

> +};
> +
> +int rpc_pipefs_add_destroy_cb(struct super_block *sb,
> + void (*destroy_cb)(void *data), void *data)
> +{
> + struct destroy_cb *dcb;
> + struct list_head *destroy_cb_list = sb->s_fs_info;
> +
> + dcb = kmalloc(sizeof(*dcb), GFP_KERNEL);
> + if (!dcb)
> + return -ENOMEM;
> +
> + dcb->callback = destroy_cb;
> + dcb->data = data;
> + INIT_LIST_HEAD(&dcb->list);
> + list_add(&dcb->list, destroy_cb_list);
> +
> + return 0;
> +}
> +EXPORT_SYMBOL_GPL(rpc_pipefs_add_destroy_cb);
> +
> /*
>   * We have a single directory with 1 node in it.
>   */
> @@ -1004,8 +1029,16 @@ rpc_fill_super(struct super_block *sb, void *data, int silent)
>   iput(inode);
>   return -ENOMEM;
>   }
> - if (rpc_populate(root, files, RPCAUTH_lockd, RPCAUTH_RootEOF, NULL))
> + /* List of destroy callbacks */
> + sb->s_fs_info = kmalloc(sizeof(struct list_head), GFP_KERNEL);
> + if (!sb->s_fs_info)
> + return -ENOMEM;
> + INIT_LIST_HEAD((struct list_head*) &sb->s_fs_info);

'&' is by mistake here.

> + if (rpc_populate(root, files, RPCAUTH_lockd, RPCAUTH_RootEOF, NULL)) {
> + kfree(sb->s_fs_info);
>   return -ENOMEM;
> + }
> +
>   return 0;
> }

--
 Kirill A. Shutemov

Subject: Re: [PATCH v3 00/16] make rpc_pipefs be mountable multiple time
Posted by Rob Landley on Thu, 20 Jan 2011 13:57:52 GMT
View Forum Message <> Reply to Message

On 01/20/2011 05:35 AM, Kirill A. Shutemov wrote:
> On Mon, Jan 17, 2011 at 06:30:16AM -0600, Rob Landley wrote:
>> On 01/14/2011 07:48 AM, Kirill A. Shutemov wrote:
>>> Prepare nfs/sunrpc stack to use multiple instances of rpc_pipefs.
>>> Only for client for now.
>>
>> Ok, Google is being really unhelpful here.
>
> It's better if you read the code. :)

I did. That doesn't necessarily help if I don't know what it's _trying_ to do and this implementation is just one small piece of a much larger mechanism.

>> What is rpc_pipefs for? What uses it, and to do what exactly? Is it
>> used by nfs server code, or by the client code, or both? Is it a way
>> for userspace to talk to the kernel, or for the kernel to talk to
>> itself? Is it used at mount time, or during filesystem operation?
>
> Ok, It try to answer. Please correct me, if I'm wrong.
>
> rpc_pipefs is a userland/kernel interface (I don't see kernel-kernel
> usecases, but it's possible, I guess).

Then why is the host context treated specially with an init_rcp_pipefs instance that isn't necessarily mounted anywhere? (How does userspace access it if it isn't mounted anywhere? Why doesn't the host context just use the same does-not-exist code as container context? Presumably there's a reason for this?)

I've also mounted NFS shares in test environments I never mounted rpc_pipefs in. (It's possible that the nfs.mount command was doing so for me, and apparently unmounting it again afterwards.) My test environment is exporting with the userspace NFS daemon so it's not using the kernel cacheing infrastructure.

> There is client dir (nfs/clntX) in rpc_pipefs for every sunrpc client.
> Both client and server (see fs/nfsd/nfs4callback.c) can create sunrpc
> client. So we rpc_pipefs on both side.

Ok, both client and server can create instances. And use them to do what?

I understand that the kernel needs to handle RPC calls to service NFS
requests, what I'm not figuring out is how userspace is involved after
the initial mount except on the other side of filesystem-agnostic system
calls.

> rpc_pipefs uses not only on mount time. See old idmapper, for example.

Um, does this sentence say that the old idmapper uses rpc_pipefs?
(Presumably not the kernel infrastructure controlled by
CONFIG_NFS_USE_NEW_IDMAPPER because you said it's not a kernel-kernel
communication mechanism... So you're saying that whatever userspace
infrastructure interacts with that will also depend on rpc_pipefs. But
the new infrastructure doesn't, which is why I need to look at the old?)

(And again, CONFIG_NFS_USE_NEW_IDMAPPER needs /sbin/request-key from the
keyutils package implying this is primarily for authentication, which
would explain why I haven't seen it in my simple test environment...)

Rob

_____
Containers mailing list
Containers@lists.linux-foundation.org
https://lists.linux-foundation.org/mailman/listinfo/containe rs

---

Subject: Re: [PATCH v3 00/16] make rpc_pipefs be mountable multiple time
Posted by Kirill A. Shutemov on Mon, 24 Jan 2011 23:55:45 GMT
View Forum Message <> Reply to Message

On Fri, Jan 14, 2011 at 03:48:58PM +0200, Kirill A. Shutemov wrote:
> Prepare nfs/sunrpc stack to use multiple instances of rpc_pipefs.
> Only for client for now.
>
> It's step forwart to get nfs work from container.

Any feedback?

> Changelog:
>
> v3:
> - rebase to the current Linus' tree (52cfd503ad)
> - rework get_rpc_pipefs() once again;
> - solve problem with rmmod sunrpc moudle;
> - free dns cache on killing rpc_pipefs superblock.
>

> v2:
> - one of rpc_create() calls was missed initially, fixed;
> - change logic for get_rpc_pipefs(NULL);
> - export get_rpc_pipefs() to be able to use from modules (tnx J. Bruce Field);
> - change "From:" and "Signed-off-by:" addresses.
>
> v1:
> - initial revision of the patchset.
>
> Kirill A. Shutemov (16):
>   sunrpc: mount rpc_pipefs on initialization
>   sunrpc: introduce init_rpc_pipefs
>   sunrpc: push init_rpc_pipefs up to rpc_create() callers
>   sunrpc: tag svc_serv with rpc_pipefs mount point
>   sunrpc: get rpc_pipefs mount point for svc_serv from callers
>   lockd: get rpc_pipefs mount point from callers
>   sunrpc: get rpc_pipefs mount point for rpcb_create[_local] from
>   callers
>   sunrpc: tag pipefs field of cache_detail with rpc_pipefs mount point
>   sunrpc: introduce rpc_pipefs_add_destroy_cb()
>   nfs: per-rpc_pipefs dns cache
>   Export iterate_mounts symbol to be able to use from sunrpc module.
>   sunrpc: introduce get_rpc_pipefs()
>   nfs: introduce mount option 'rpcmount'
>   sunrpc: make rpc_pipefs be mountable multiple times
>   sunrpc: remove global init_rpc_pipefs
>   Rework get_rpc_pipefs and introduce put_rpc_pipefs()
>
> fs/lockd/clntlock.c | 8 +-
> fs/lockd/host.c | 15 ++-
> fs/lockd/mon.c | 13 ++-
> fs/lockd/svc.c | 4 +-
> fs/namespace.c | 1 +
> fs/nfs/cache_lib.c | 18 +--
> fs/nfs/cache_lib.h | 3 +-
> fs/nfs/callback.c | 7 +-
> fs/nfs/callback.h | 3 +-
> fs/nfs/client.c | 45 ++++++-
> fs/nfs/dns_resolve.c | 137 +++++++++++++++++-----
> fs/nfs/dns_resolve.h | 15 +--
> fs/nfs/inode.c | 9 +-
> fs/nfs/internal.h | 10 ++-
> fs/nfs/mount_clnt.c | 1 +
> fs/nfs/namespace.c | 3 +-
> fs/nfs/nfs4namespace.c | 20 ++--
> fs/nfs/super.c | 20 +++
> fs/nfsd/nfs4callback.c | 5 +
> fs/nfsd/nfssvc.c | 20 +++-

> include/linux/lockd/bind.h | 3 +-
> include/linux/lockd/lockd.h | 4 +-
> include/linux/nfs_fs_sb.h | 1 +
> include/linux/sunrpc/cache.h | 9 +-
> include/linux/sunrpc/clnt.h | 5 +-
> include/linux/sunrpc/rpc_pipe_fs.h | 7 +-
> include/linux/sunrpc/svc.h | 9 +-
> net/sunrpc/cache.c | 16 ++-
> net/sunrpc/clnt.c | 19 ++--
> net/sunrpc/rpc_pipe.c | 235 ++----
> net/sunrpc/rpcb_clnt.c | 19 ++-
> net/sunrpc/svc.c | 52 +++++---
> 32 files changed, 549 insertions(+), 187 deletions(-)
>
> --
> 1.7.3.4
>

--
Kirill A. Shutemov

_____
Containers mailing list
Containers@lists.linux-foundation.org
https://lists.linux-foundation.org/mailman/listinfo/containe rs

---

Subject: Re: [PATCH v3 00/16] make rpc_pipefs be mountable multiple time
Posted by Rob Landley on Tue, 25 Jan 2011 01:53:04 GMT
View Forum Message <> Reply to Message

On 01/24/2011 05:55 PM, Kirill A. Shutemov wrote:
> On Fri, Jan 14, 2011 at 03:48:58PM +0200, Kirill A. Shutemov wrote:
>> Prepare nfs/sunrpc stack to use multiple instances of rpc_pipefs.
>> Only for client for now.
>>
>> It's step forwart to get nfs work from container.
>
> Any feedback?

I didn't see anything to object to, and I'll give a:

Reviewed-by: Rob Landley <rlandley@parallels.com>

To the one that you rewrote for me (um, 12/16). I still don't
understand at the design level why init_rpc_pipefs exists if containers
don't need it, but it seems to be implementing what it sets out to do.

But it doesn't apply to my test environment (nfsv3 unauthenticated), and

I'm still trying to finish up that before switching gears to look at v4.

Rob

_____

Containers mailing list
Containers@lists.linux-foundation.org
https://lists.linux-foundation.org/mailman/listinfo/containe rs