Subject: [PATCH 0/8] NFSd: precursor and cleanup patch set
Posted by Stanislav Kinsbursky on Wed, 28 Mar 2012 15:08:53 GMT
View Forum Message <> Reply to Message

This patch set prepares ground for making NFSd export and expkey caches
allocated and registered per network namespace context.
Mainly, it just replaces hard-coded pointers with dereferenced everywhere
required.

The following series consists of:

Stanislav Kinsbursky (8):
      nfsd: use passed cache_detail pointer expkey_parse()
      nfsd: add link to owner cache detail to svc_export structure
      nfsd: use cache detail pointer from svc_export structure on cache put
      nfsd: use exp_put() for svc_export_cache put
      nfsd: pass svc_export_cache pointer as private data to "exports" seq file ops
      nfsd: use hash table from cache detail in nfsd export seq ops
      nfsd: pass pointer to export cache down to stack wherever possible.
      nfsd: pass pointer to expkey cache down to stack wherever possible.

```
 fs/nfsd/export.c           |  102 +++++++++++++++++++++++++++++++++-------------------
 fs/nfsd/nfsctl.c           |   11 ++++-
 fs/nfsd/nfsfh.c            |    2 -
 include/linux/nfsd/export.h |    3 +
 4 files changed, 71 insertions(+), 47 deletions(-)
```

Subject: [PATCH 1/8] nfsd: use passed cache_detail pointer expkey_parse()
Posted by Stanislav Kinsbursky on Wed, 28 Mar 2012 15:09:01 GMT
View Forum Message <> Reply to Message

Using of hard-coded svc_expkey_cache pointer in expkey_parse() looks redundant.
Moreover, global cache will be replaced with per-net instance soon.

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

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

```
 fs/nfsd/export.c |    2 +-
 1 files changed, 1 insertions(+), 1 deletions(-)
```

```
diff --git a/fs/nfsd/export.c b/fs/nfsd/export.c
index 8e9689a..71c5ce3 100644
```

```
--- a/fs/nfsd/export.c
+++ b/fs/nfsd/export.c
@@ -163,7 +163,7 @@ static int expkey_parse(struct cache_detail *cd, char *mesg, int mlen)
  cache_flush();
  out:
  if (ek)
-  cache_put(&ek->h, &svc_expkey_cache);
+  cache_put(&ek->h, cd);
  if (dom)
   auth_domain_put(dom);
  kfree(buf);
```

Subject: [PATCH 2/8] nfsd: add link to owner cache detail to svc_export structure
Posted by Stanislav Kinsbursky on Wed, 28 Mar 2012 15:09:08 GMT
View Forum Message <> Reply to Message

Without info about owner cache datail it won't be able to find out, which
per-net cache detail have to be.

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

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

```
---
 fs/nfsd/export.c           |   10 +++++-----
 include/linux/nfsd/export.h |    1 +
 2 files changed, 6 insertions(+), 5 deletions(-)

diff --git a/fs/nfsd/export.c b/fs/nfsd/export.c
index 71c5ce3..99ea4c0 100644
--- a/fs/nfsd/export.c
+++ b/fs/nfsd/export.c
@@ -525,6 +525,7 @@ static int svc_export_parse(struct cache_detail *cd, char *mesg, int mlen)
   goto out1;

  exp.ex_client = dom;
+ exp.cd = cd;

  /* expiry */
  err = -EINVAL;
@@ -672,6 +673,7 @@ static void svc_export_init(struct cache_head *cnew, struct cache_head
*citem)
  new->ex_fslocs.locations = NULL;
  new->ex_fslocs.locations_count = 0;
  new->ex_fslocs.migrated = 0;
+ new->cd = item->cd;
 }
```

```
 static void export_update(struct cache_head *cnew, struct cache_head *citem)
@@ -739,8 +741,7 @@ svc_export_lookup(struct svc_export *exp)
  struct cache_head *ch;
  int hash = svc_export_hash(exp);

- ch = sunrpc_cache_lookup(&svc_export_cache, &exp->h,
-     hash);
+ ch = sunrpc_cache_lookup(exp->cd, &exp->h, hash);
  if (ch)
   return container_of(ch, struct svc_export, h);
  else
@@ -753,9 +754,7 @@ svc_export_update(struct svc_export *new, struct svc_export *old)
  struct cache_head *ch;
  int hash = svc_export_hash(old);

- ch = sunrpc_cache_update(&svc_export_cache, &new->h,
-     &old->h,
-     hash);
+ ch = sunrpc_cache_update(old->cd, &new->h, &old->h, hash);
  if (ch)
   return container_of(ch, struct svc_export, h);
  else
@@ -797,6 +796,7 @@ static svc_export *exp_get_by_name(svc_client *clp, const struct path
*path,

  key.ex_client = clp;
  key.ex_path = *path;
+ key.cd = &svc_export_cache;

  exp = svc_export_lookup(&key);
  if (exp == NULL)
diff --git a/include/linux/nfsd/export.h b/include/linux/nfsd/export.h
index f85308e..6445529 100644
--- a/include/linux/nfsd/export.h
+++ b/include/linux/nfsd/export.h
@@ -103,6 +103,7 @@ struct svc_export {
  struct nfsd4_fs_locations ex_fslocs;
  int   ex_nflavors;
  struct exp_flavor_info ex_flavors[MAX_SECINFO_LIST];
+ struct cache_detail *cd;
 };

 /* an "export key" (expkey) maps a filehandlefragement to an
```

Subject: [PATCH 3/8] nfsd: use cache detail pointer from svc_export structure on

cache put
Posted by Stanislav Kinsbursky on Wed, 28 Mar 2012 15:09:15 GMT
View Forum Message <> Reply to Message

Hard-coded pointer is redundant now and can be replaced.

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

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

---
 include/linux/nfsd/export.h | 2 +-
 1 files changed, 1 insertions(+), 1 deletions(-)

diff --git a/include/linux/nfsd/export.h b/include/linux/nfsd/export.h
index 6445529..485c2af 100644
--- a/include/linux/nfsd/export.h
+++ b/include/linux/nfsd/export.h
@@ -147,7 +147,7 @@ extern struct cache_detail svc_export_cache;

 static inline void exp_put(struct svc_export *exp)
 {
- cache_put(&exp->h, &svc_export_cache);
+ cache_put(&exp->h, exp->cd);
 }

 static inline void exp_get(struct svc_export *exp)

Subject: [PATCH 4/8] nfsd: use exp_put() for svc_export_cache put
Posted by Stanislav Kinsbursky on Wed, 28 Mar 2012 15:09:22 GMT
View Forum Message <> Reply to Message

This patch replaces cache_put() call for svc_export_cache by exp_put() call.

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

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

---
 fs/nfsd/export.c | 2 +-
 fs/nfsd/nfsfh.c | 2 +-
 2 files changed, 2 insertions(+), 2 deletions(-)

diff --git a/fs/nfsd/export.c b/fs/nfsd/export.c
index 99ea4c0..c20a405 100644
--- a/fs/nfsd/export.c
+++ b/fs/nfsd/export.c

```
@@ -1205,7 +1205,7 @@ static int e_show(struct seq_file *m, void *p)
  cache_get(&exp->h);
  if (cache_check(&svc_export_cache, &exp->h, NULL))
   return 0;
- cache_put(&exp->h, &svc_export_cache);
+ exp_put(exp);
  return svc_export_show(m, &svc_export_cache, cp);
 }

diff --git a/fs/nfsd/nfsfh.c b/fs/nfsd/nfsfh.c
index 68454e7..cc79300 100644
--- a/fs/nfsd/nfsfh.c
+++ b/fs/nfsd/nfsfh.c
@@ -636,7 +636,7 @@ fh_put(struct svc_fh *fhp)
 #endif
  }
  if (exp) {
-  cache_put(&exp->h, &svc_export_cache);
+  exp_put(exp);
   fhp->fh_export = NULL;
  }
  return;
```

Subject: [PATCH 5/8] nfsd: pass svc_export_cache pointer as private data to "exports" seq f

Posted by Stanislav Kinsbursky on Wed, 28 Mar 2012 15:09:29 GMT

View Forum Message <> Reply to Message

Global svc_export_cache cache is going to be replaced with per-net instance. So
prepare the ground for it.

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

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

```
---
 fs/nfsd/export.c |   18 ++++++++++-------
 fs/nfsd/nfsctl.c |   11 ++++++++-
 2 files changed, 21 insertions(+), 8 deletions(-)

diff --git a/fs/nfsd/export.c b/fs/nfsd/export.c
index c20a405..1495320 100644
--- a/fs/nfsd/export.c
+++ b/fs/nfsd/export.c
@@ -1029,13 +1029,14 @@ exp_pseudoroot(struct svc_rqst *rqstp, struct svc_fh *fhp)
 /* Iterator */
```

```
 static void *e_start(struct seq_file *m, loff_t *pos)
- __acquires(svc_export_cache.hash_lock)
+ __acquires(((struct cache_detail *)m->private)->hash_lock)
 {
  loff_t n = *pos;
  unsigned hash, export;
  struct cache_head *ch;
-
- read_lock(&svc_export_cache.hash_lock);
+ struct cache_detail *cd = m->private;
+
+ read_lock(&cd->hash_lock);
  if (!n--)
   return SEQ_START_TOKEN;
  hash = n >> 32;
@@ -1082,9 +1083,11 @@ static void *e_next(struct seq_file *m, void *p, loff_t *pos)
 }

 static void e_stop(struct seq_file *m, void *p)
- __releases(svc_export_cache.hash_lock)
+ __releases(((struct cache_detail *)m->private)->hash_lock)
 {
- read_unlock(&svc_export_cache.hash_lock);
+ struct cache_detail *cd = m->private;
+
+ read_unlock(&cd->hash_lock);
 }

 static struct flags {
@@ -1195,6 +1198,7 @@ static int e_show(struct seq_file *m, void *p)
 {
  struct cache_head *cp = p;
  struct svc_export *exp = container_of(cp, struct svc_export, h);
+ struct cache_detail *cd = m->private;

  if (p == SEQ_START_TOKEN) {
   seq_puts(m, "# Version 1.1\n");
@@ -1203,10 +1207,10 @@ static int e_show(struct seq_file *m, void *p)
  }

  cache_get(&exp->h);
- if (cache_check(&svc_export_cache, &exp->h, NULL))
+ if (cache_check(cd, &exp->h, NULL))
   return 0;
  exp_put(exp);
- return svc_export_show(m, &svc_export_cache, cp);
+ return svc_export_show(m, cd, cp);
 }
```

```
 const struct seq_operations nfs_exports_op = {
diff --git a/fs/nfsd/nfsctl.c b/fs/nfsd/nfsctl.c
index dee6c1b..9bc6f8c 100644
--- a/fs/nfsd/nfsctl.c
+++ b/fs/nfsd/nfsctl.c
@@ -127,7 +127,16 @@ static const struct file_operations transaction_ops = {

 static int exports_open(struct inode *inode, struct file *file)
 {
- return seq_open(file, &nfs_exports_op);
+ int err;
+ struct seq_file *seq;
+
+ err = seq_open(file, &nfs_exports_op);
+ if (err)
+  return err;
+
+ seq = file->private_data;
+ seq->private = &svc_export_cache;
+ return 0;
 }

 static const struct file_operations exports_operations = {
```

Subject: [PATCH 6/8] nfsd: use hash table from cache detail in nfsd export seq ops
Posted by Stanislav Kinsbursky on Wed, 28 Mar 2012 15:09:35 GMT
View Forum Message <> Reply to Message

Hard-code is redundant and will prevent from making caches per net ns.

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

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

```
---
 fs/nfsd/export.c |    3 +++
 1 files changed, 3 insertions(+), 0 deletions(-)

diff --git a/fs/nfsd/export.c b/fs/nfsd/export.c
index 1495320..9fe7156 100644
--- a/fs/nfsd/export.c
+++ b/fs/nfsd/export.c
@@ -1035,6 +1035,7 @@ static void *e_start(struct seq_file *m, loff_t *pos)
  unsigned hash, export;
  struct cache_head *ch;
  struct cache_detail *cd = m->private;
```

```
+ struct cache_head **export_table = cd->hash_table;

  read_lock(&cd->hash_lock);
  if (!n--)
@@ -1061,6 +1062,8 @@ static void *e_next(struct seq_file *m, void *p, loff_t *pos)
 {
  struct cache_head *ch = p;
  int hash = (*pos >> 32);
+ struct cache_detail *cd = m->private;
+ struct cache_head **export_table = cd->hash_table;

  if (p == SEQ_START_TOKEN)
   hash = 0;
```

Subject: [PATCH 7/8] nfsd: pass pointer to export cache down to stack wherever possible.

Posted by Stanislav Kinsbursky on Wed, 28 Mar 2012 15:09:42 GMT

View Forum Message <> Reply to Message

This cache will be per-net soon. And it's easier to get the pointer to desired
per-net instance only once and then pass it down instead of discovering it in
every place were required.

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

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

```
---
 fs/nfsd/export.c |   37 ++++++++++++++++++++++---------------
 1 files changed, 22 insertions(+), 15 deletions(-)

diff --git a/fs/nfsd/export.c b/fs/nfsd/export.c
index 9fe7156..c237b0a 100644
--- a/fs/nfsd/export.c
+++ b/fs/nfsd/export.c
@@ -785,8 +785,8 @@ exp_find_key(svc_client *clp, int fsid_type, u32 *fsidv, struct cache_req *reqp)
 }


-static svc_export *exp_get_by_name(svc_client *clp, const struct path *path,
-         struct cache_req *reqp)
+static svc_export *exp_get_by_name(struct cache_detail *cd, svc_client *clp,
+        const struct path *path, struct cache_req *reqp)
 {
  struct svc_export *exp, key;
  int err;
```

```
@@ -796,12 +796,12 @@ static svc_export *exp_get_by_name(svc_client *clp, const struct path
*path,

  key.ex_client = clp;
  key.ex_path = *path;
- key.cd = &svc_export_cache;
+ key.cd = cd;

  exp = svc_export_lookup(&key);
  if (exp == NULL)
   return ERR_PTR(-ENOMEM);
- err = cache_check(&svc_export_cache, &exp->h, reqp);
+ err = cache_check(cd, &exp->h, reqp);
  if (err)
   return ERR_PTR(err);
  return exp;
@@ -810,16 +810,17 @@ static svc_export *exp_get_by_name(svc_client *clp, const struct path
*path,
 /*
  * Find the export entry for a given dentry.
  */
-static struct svc_export *exp_parent(svc_client *clp, struct path *path)
+static struct svc_export *exp_parent(struct cache_detail *cd, svc_client *clp,
+         struct path *path)
 {
  struct dentry *saved = dget(path->dentry);
- svc_export *exp = exp_get_by_name(clp, path, NULL);
+ svc_export *exp = exp_get_by_name(cd, clp, path, NULL);

  while (PTR_ERR(exp) == -ENOENT && !IS_ROOT(path->dentry)) {
   struct dentry *parent = dget_parent(path->dentry);
   dput(path->dentry);
   path->dentry = parent;
-  exp = exp_get_by_name(clp, path, NULL);
+  exp = exp_get_by_name(cd, clp, path, NULL);
  }
  dput(path->dentry);
  path->dentry = saved;
@@ -834,13 +835,15 @@ static struct svc_export *exp_parent(svc_client *clp, struct path *path)
  * since its harder to fool a kernel module than a user space program.
  */
 int
-exp_rootfh(svc_client *clp, char *name, struct knfsd_fh *f, int maxsize)
+exp_rootfh(svc_client *clp, char *name,
+    struct knfsd_fh *f, int maxsize)
 {
  struct svc_export *exp;
  struct path  path;
```

```
  struct inode  *inode;
  struct svc_fh  fh;
  int    err;
+ struct cache_detail *cd = &svc_export_cache;

  err = -EPERM;
  /* NB: we probably ought to check that it's NUL-terminated */
@@ -853,7 +856,7 @@ exp_rootfh(svc_client *clp, char *name, struct knfsd_fh *f, int maxsize)
  dprintk("nfsd: exp_rootfh(%s [%p] %s:%s/%ld)\n",
    name, path.dentry, clp->name,
    inode->i_sb->s_id, inode->i_ino);
- exp = exp_parent(clp, &path);
+ exp = exp_parent(cd, clp, &path);
  if (IS_ERR(exp)) {
   err = PTR_ERR(exp);
   goto out;
@@ -875,7 +878,8 @@ out:
  return err;
 }

-static struct svc_export *exp_find(struct auth_domain *clp, int fsid_type,
+static struct svc_export *exp_find(struct cache_detail *cd,
+       struct auth_domain *clp, int fsid_type,
        u32 *fsidv, struct cache_req *reqp)
 {
  struct svc_export *exp;
@@ -883,7 +887,7 @@ static struct svc_export *exp_find(struct auth_domain *clp, int fsid_type,
  if (IS_ERR(ek))
   return ERR_CAST(ek);

- exp = exp_get_by_name(clp, &ek->ek_path, reqp);
+ exp = exp_get_by_name(cd, clp, &ek->ek_path, reqp);
  cache_put(&ek->h, &svc_expkey_cache);

  if (IS_ERR(exp))
@@ -926,12 +930,13 @@ struct svc_export *
 rqst_exp_get_by_name(struct svc_rqst *rqstp, struct path *path)
 {
  struct svc_export *gssexp, *exp = ERR_PTR(-ENOENT);
+ struct cache_detail *cd = &svc_export_cache;

  if (rqstp->rq_client == NULL)
   goto gss;

  /* First try the auth_unix client: */
- exp = exp_get_by_name(rqstp->rq_client, path, &rqstp->rq_chandle);
+ exp = exp_get_by_name(cd, rqstp->rq_client, path, &rqstp->rq_chandle);
  if (PTR_ERR(exp) == -ENOENT)
```

```
   goto gss;
  if (IS_ERR(exp))
@@ -943,7 +948,7 @@ gss:
  /* Otherwise, try falling back on gss client */
  if (rqstp->rq_gssclient == NULL)
   return exp;
- gssexp = exp_get_by_name(rqstp->rq_gssclient, path, &rqstp->rq_chandle);
+ gssexp = exp_get_by_name(cd, rqstp->rq_gssclient, path, &rqstp->rq_chandle);
  if (PTR_ERR(gssexp) == -ENOENT)
   return exp;
  if (!IS_ERR(exp))
@@ -955,12 +960,14 @@ struct svc_export *
 rqst_exp_find(struct svc_rqst *rqstp, int fsid_type, u32 *fsidv)
 {
  struct svc_export *gssexp, *exp = ERR_PTR(-ENOENT);
+ struct cache_detail *cd = &svc_export_cache;

  if (rqstp->rq_client == NULL)
   goto gss;

  /* First try the auth_unix client: */
- exp = exp_find(rqstp->rq_client, fsid_type, fsidv, &rqstp->rq_chandle);
+ exp = exp_find(cd, rqstp->rq_client, fsid_type,
+         fsidv, &rqstp->rq_chandle);
  if (PTR_ERR(exp) == -ENOENT)
   goto gss;
  if (IS_ERR(exp))
@@ -972,7 +979,7 @@ gss:
  /* Otherwise, try falling back on gss client */
  if (rqstp->rq_gssclient == NULL)
   return exp;
- gssexp = exp_find(rqstp->rq_gssclient, fsid_type, fsidv,
+ gssexp = exp_find(cd, rqstp->rq_gssclient, fsid_type, fsidv,
       &rqstp->rq_chandle);
  if (PTR_ERR(gssexp) == -ENOENT)
   return exp;
```

Subject: [PATCH 8/8] nfsd: pass pointer to expkey cache down to stack wherever possible.

Posted by Stanislav Kinsbursky on Wed, 28 Mar 2012 15:09:50 GMT

View Forum Message <> Reply to Message

This cache will be per-net soon. And it's easier to get the pointer to desired per-net instance only once and then pass it down instead of discovering it in every place were required.

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

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

---
 fs/nfsd/export.c |   32 ++++++++++++++++++++----------
 1 files changed, 16 insertions(+), 16 deletions(-)

diff --git a/fs/nfsd/export.c b/fs/nfsd/export.c
index c237b0a..429b2f5 100644
--- a/fs/nfsd/export.c
+++ b/fs/nfsd/export.c
@@ -71,9 +71,9 @@ static int expkey_upcall(struct cache_detail *cd, struct cache_head *h)
  return sunrpc_cache_pipe_upcall(cd, h, expkey_request);
 }

-static struct svc_expkey *svc_expkey_update(struct svc_expkey *new, struct svc_expkey *old);
-static struct svc_expkey *svc_expkey_lookup(struct svc_expkey *);
-static struct cache_detail svc_expkey_cache;
+static struct svc_expkey *svc_expkey_update(struct cache_detail *cd, struct svc_expkey *new,
+         struct svc_expkey *old);
+static struct svc_expkey *svc_expkey_lookup(struct cache_detail *cd, struct svc_expkey *);

 static int expkey_parse(struct cache_detail *cd, char *mesg, int mlen)
 {
@@ -131,7 +131,7 @@ static int expkey_parse(struct cache_detail *cd, char *mesg, int mlen)
  key.ek_fsidtype = fsidtype;
  memcpy(key.ek_fsid, buf, len);

- ek = svc_expkey_lookup(&key);
+ ek = svc_expkey_lookup(cd, &key);
  err = -ENOMEM;
  if (!ek)
   goto out;
@@ -145,7 +145,7 @@ static int expkey_parse(struct cache_detail *cd, char *mesg, int mlen)
  err = 0;
  if (len == 0) {
   set_bit(CACHE_NEGATIVE, &key.h.flags);
-  ek = svc_expkey_update(&key, ek);
+  ek = svc_expkey_update(cd, &key, ek);
   if (!ek)
    err = -ENOMEM;
  } else {
@@ -155,7 +155,7 @@ static int expkey_parse(struct cache_detail *cd, char *mesg, int mlen)

   dprintk("Found the path %s\n", buf);

-  ek = svc_expkey_update(&key, ek);
+  ek = svc_expkey_update(cd, &key, ek);
```

```
   if (!ek)
    err = -ENOMEM;
   path_put(&key.ek_path);
@@ -268,13 +268,12 @@ svc_expkey_hash(struct svc_expkey *item)
 }

 static struct svc_expkey *
-svc_expkey_lookup(struct svc_expkey *item)
+svc_expkey_lookup(struct cache_detail *cd, struct svc_expkey *item)
 {
  struct cache_head *ch;
  int hash = svc_expkey_hash(item);

- ch = sunrpc_cache_lookup(&svc_expkey_cache, &item->h,
-     hash);
+ ch = sunrpc_cache_lookup(cd, &item->h, hash);
  if (ch)
   return container_of(ch, struct svc_expkey, h);
  else
@@ -282,13 +281,13 @@ svc_expkey_lookup(struct svc_expkey *item)
 }

 static struct svc_expkey *
-svc_expkey_update(struct svc_expkey *new, struct svc_expkey *old)
+svc_expkey_update(struct cache_detail *cd, struct svc_expkey *new,
+    struct svc_expkey *old)
 {
  struct cache_head *ch;
  int hash = svc_expkey_hash(new);

- ch = sunrpc_cache_update(&svc_expkey_cache, &new->h,
-     &old->h, hash);
+ ch = sunrpc_cache_update(cd, &new->h, &old->h, hash);
  if (ch)
   return container_of(ch, struct svc_expkey, h);
  else
@@ -763,7 +762,8 @@ svc_export_update(struct svc_export *new, struct svc_export *old)


 static struct svc_expkey *
-exp_find_key(svc_client *clp, int fsid_type, u32 *fsidv, struct cache_req *reqp)
+exp_find_key(struct cache_detail *cd, svc_client *clp, int fsid_type,
+      u32 *fsidv, struct cache_req *reqp)
 {
  struct svc_expkey key, *ek;
  int err;
@@ -775,10 +775,10 @@ exp_find_key(svc_client *clp, int fsid_type, u32 *fsidv, struct
cache_req *reqp)
```

```
  key.ek_fsidtype = fsid_type;
  memcpy(key.ek_fsid, fsidv, key_len(fsid_type));

- ek = svc_expkey_lookup(&key);
+ ek = svc_expkey_lookup(cd, &key);
  if (ek == NULL)
   return ERR_PTR(-ENOMEM);
- err = cache_check(&svc_expkey_cache, &ek->h, reqp);
+ err = cache_check(cd, &ek->h, reqp);
  if (err)
   return ERR_PTR(err);
  return ek;
@@ -883,7 +883,7 @@ static struct svc_export *exp_find(struct cache_detail *cd,
        u32 *fsidv, struct cache_req *reqp)
 {
  struct svc_export *exp;
- struct svc_expkey *ek = exp_find_key(clp, fsid_type, fsidv, reqp);
+ struct svc_expkey *ek = exp_find_key(&svc_expkey_cache, clp, fsid_type, fsidv, reqp);
  if (IS_ERR(ek))
   return ERR_CAST(ek);
```

Subject: Re: [PATCH 0/8] NFSd: precursor and cleanup patch set
Posted by bfields on Thu, 29 Mar 2012 14:32:18 GMT

View Forum Message <> Reply to Message

On Wed, Mar 28, 2012 at 07:08:53PM +0400, Stanislav Kinsbursky wrote:
> This patch set prepares ground for making NFSd export and expkey caches
> allocated and registered per network namespace context.
> Mainly, it just replaces hard-coded pointers with dereferenced everywhere
> required.

Thanks. I'll be travelling next week and may not get to look at this
till I get back.

--b.

>
> The following series consists of:
>
> ---
>
> Stanislav Kinsbursky (8):
>   nfsd: use passed cache_detail pointer expkey_parse()
>   nfsd: add link to owner cache detail to svc_export structure
>   nfsd: use cache detail pointer from svc_export structure on cache put
>   nfsd: use exp_put() for svc_export_cache put
>   nfsd: pass svc_export_cache pointer as private data to "exports" seq file ops

> nfsd: use hash table from cache detail in nfsd export seq ops
> nfsd: pass pointer to export cache down to stack wherever possible.
> nfsd: pass pointer to expkey cache down to stack wherever possible.
>
>
> fs/nfsd/export.c | 102 ++++++++++++++++++++++++++++++----------------
> fs/nfsd/nfsctl.c | 11 ++++-
> fs/nfsd/nfsfh.c | 2 -
> include/linux/nfsd/export.h | 3 +
> 4 files changed, 71 insertions(+), 47 deletions(-)
>

Subject: Re: [PATCH 0/8] NFSd: precursor and cleanup patch set
Posted by Stanislav Kinsbursky on Thu, 29 Mar 2012 15:36:39 GMT
View Forum Message <> Reply to Message

> On Wed, Mar 28, 2012 at 07:08:53PM +0400, Stanislav Kinsbursky wrote:
>> This patch set prepares ground for making NFSd export and expkey caches
>> allocated and registered per network namespace context.
>> Mainly, it just replaces hard-coded pointers with dereferenced everywhere
>> required.
>
> Thanks. I'll be travelling next week and may not get to look at this
> till I get back.
>

Yep, sure. No rush here...

--
Best regards,
Stanislav Kinsbursky

Subject: Re: [PATCH 4/8] nfsd: use exp_put() for svc_export_cache put
Posted by bfields on Wed, 11 Apr 2012 14:16:14 GMT
View Forum Message <> Reply to Message

On Wed, Mar 28, 2012 at 07:09:22PM +0400, Stanislav Kinsbursky wrote:
> This patch replaces cache_put() call for svc_export_cache by exp_put() call.

Patch looks fine. Minor peeve: the changelog doesn't tell me anything I
couldn't figure out immediately from reading the patch. Tell me *why*
we're doing this ("Removing another hard-coded svc_export_cache will
simplify adding per-export caches later"), or if it's really totally
obvious I'd rather have an empty changelog than a useless one....

--b.

>
> Signed-off-by: Stanislav Kinsbursky <skinsbursky@parallels.com>
>
> Signed-off-by: Stanislav Kinsbursky <skinsbursky@parallels.com>
>
> ---
> fs/nfsd/export.c | 2 +-
> fs/nfsd/nfsfh.c | 2 +-
> 2 files changed, 2 insertions(+), 2 deletions(-)
>
> diff --git a/fs/nfsd/export.c b/fs/nfsd/export.c
> index 99ea4c0..c20a405 100644
> --- a/fs/nfsd/export.c
> +++ b/fs/nfsd/export.c
> @@ -1205,7 +1205,7 @@ static int e_show(struct seq_file *m, void *p)
>   cache_get(&exp->h);
>   if (cache_check(&svc_export_cache, &exp->h, NULL))
>   return 0;
> - cache_put(&exp->h, &svc_export_cache);
> + exp_put(exp);
>   return svc_export_show(m, &svc_export_cache, cp);
> }
>
> diff --git a/fs/nfsd/nfsfh.c b/fs/nfsd/nfsfh.c
> index 68454e7..cc79300 100644
> --- a/fs/nfsd/nfsfh.c
> +++ b/fs/nfsd/nfsfh.c
> @@ -636,7 +636,7 @@ fh_put(struct svc_fh *fhp)
> #endif
>   }
>   if (exp) {
> - cache_put(&exp->h, &svc_export_cache);
> + exp_put(exp);
>   fhp->fh_export = NULL;
>   }
>   return;
>

Subject: Re: [PATCH 4/8] nfsd: use exp_put() for svc_export_cache put
Posted by Stanislav Kinsbursky on Wed, 11 Apr 2012 14:41:39 GMT
View Forum Message <> Reply to Message

> On Wed, Mar 28, 2012 at 07:09:22PM +0400, Stanislav Kinsbursky wrote:

>> This patch replaces cache_put() call for svc_export_cache by exp_put() call.
>
> Patch looks fine. Minor peeve: the changelog doesn't tell me anything I
> couldn't figure out immediately from reading the patch. Tell me *why*
> we're doing this ("Removing another hard-coded svc_export_cache will
> simplify adding per-export caches later"), or if it's really totally
> obvious I'd rather have an empty changelog than a useless one....
>

Ok, will do so for further patches.

--
Best regards,
Stanislav Kinsbursky

Subject: Re: [PATCH 0/8] NFSd: precursor and cleanup patch set
Posted by bfields on Wed, 11 Apr 2012 16:05:23 GMT
View Forum Message <> Reply to Message

On Wed, Mar 28, 2012 at 07:08:53PM +0400, Stanislav Kinsbursky wrote:
> This patch set prepares ground for making NFSd export and expkey caches
> allocated and registered per network namespace context.
> Mainly, it just replaces hard-coded pointers with dereferenced everywhere
> required.
>
> The following series consists of:

OK, thanks, applying these all pending some testing. I'll try to get a
for-3.5 branch with this pushed out later today or tomorrow.

--b.

Subject: Re: [PATCH 0/8] NFSd: precursor and cleanup patch set
Posted by Stanislav Kinsbursky on Wed, 11 Apr 2012 16:06:00 GMT
View Forum Message <> Reply to Message

> On Wed, Mar 28, 2012 at 07:08:53PM +0400, Stanislav Kinsbursky wrote:
>> This patch set prepares ground for making NFSd export and expkey caches
>> allocated and registered per network namespace context.
>> Mainly, it just replaces hard-coded pointers with dereferenced everywhere
>> required.
>>
>> The following series consists of:
>

> OK, thanks, applying these all pending some testing. I'll try to get a
> for-3.5 branch with this pushed out later today or tomorrow.
>

Ok, thanks, Bruce.

--
Best regards,
Stanislav Kinsbursky
