Subject: Re: [PATCH 2/4] sysfs: Implement sysfs manged shadow directory support.
Posted by ebiederm on Tue, 31 Jul 2007 07:59:38 GMT
View Forum Message <> Reply to Message

Tejun Heo <teheo@suse.de> writes:

> Eric W. Biederman wrote:
>> What do we use inode->i_mutex for? I think we might be able
>> to kill that.
>>
>> I'm starting to wonder if we can completely remove sysfs
>> from grabbing inode->i_mutex.
>
> i_mutex is grabbed when dentry and inode locking requires it. It's not
> used to protect sysfs internal data structure anymore. I don't think we
> can remove i_mutex grabbing without violating dentry/inode locking rules.

Not entirely no. I think we can remove i_mutex from protecting dentry tree modifications.

>>>> At first glance sysfs_assoc_lock looks just as bad.
>>> I think sysfs_assoc_lock is okay. It's tricky tho. Why do you think
>>> it's bad?
>>
>> I'm still looking. I just have a weird vibe so far. sysfs_get_dentry
>> is really nasty with respect to locking.
>
> Yes, sysfs_get_dentry() is pretty hairy. I wish I could use
> path_lookup() there but can't allocate memory for path name because
> looking up must succeed when it's called from removal path if dentry
> already exists. Also, lookup_one_len_kern() bypasses security checks
> and there's no equivalent path_lookup() like function which does that.

We can use d_hash_and_lookup and that helps a lot. I have attached my in-progress rewrite of sysfs_get_dentry. It's a little less efficient but a whole lot easier to maintain.

> Locking rule aruond sysfs_assoc_lock is tricky. It's mainly used to
> avoid race condition between sysfs_d_iput() vs. dentry creation, node
> removal, etc. As long as sysfs_assoc_lock is held, sd->s_dentry can be
> dereferenced but you also need dcache_lock to determine whether the
> dentry is alive (dentry->d_inode != NULL) or in the process of being
> killed. There were two or three race conditions around dentry
> reclamation in the past and several discussion threads about them.

I think I have figured out how to safely remove s_dentry entirely from sysfs_dirent and that winds up removing a lot of subtle and nasty locking.

I'm hoping to have a good patch series after another couple of hours of work.

```
struct dentry *__sysfs_get_dentry(struct sysfs_dirent *sd, int create)
{
 struct sysfs_dirent *cur;
 struct dentry *parent_dentry, *dentry;
 struct qstr name;
 struct inode *inode;

 parent_dentry = NULL;
 dentry = dget(sysfs_sb->s_root);

 do {
  /* Find the first ancestor I have not looked up */
  cur = sd;
  while (cur->s_parent != dentry->d_fsdata)
   cur = cur->s_parent;

  /* look it up */
  dput(parent_dentry);
  parent_dentry = dentry;
  name.name = cur->s_name;
  name.len = strlen(cur->s_name);
  dentry = d_hash_and_lookup(parent_dentry, &name);
  if (dentry)
   continue;
  if (!create)
   goto out;
  dentry = d_alloc(parent_dentry, &name);
  if (!dentry) {
   dentry = ERR_PTR(-ENOMEM);
   goto out;
  }
  inode = sysfs_get_inode(cur);
  if (!inode) {
   dput(dentry);
   dentry = ERR_PTR(-ENOMEM);
   goto out;
  }
  d_instantiate(dentry, inode);
  sysfs_attach_dentry(cur, dentry);
 } while (cur != sd);

out:
 dput(parent_dentry);
 return dentry;
}
```

```
struct dentry *sysfs_get_dentry(struct sysfs_dirent *sd)
{
 struct dentry *dentry;

 mutex_lock(&sysfs_mutex);
 dentry = __sysfs_get_dentry(sd, 1);
 mutex_unlock(&sysfs_mutex);
 return dentry;
}
```

Eric