
Subject: [PATCH] ipc: Save the ipc namespace while reading proc files.

Posted by [ebiederm](#) on Tue, 06 Feb 2007 19:30:49 GMT

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

The problem we were assuming that current->nsproxy->ipc_ns would never change while someone has our file in /proc/sysvipc/ file open. Given that this can change with both unshare and by passing the file descriptor to another process that assumption is occasionally wrong.

Therefore this patch causes /proc/sysvipc/* to cache the namespace and increment it's count when we open the file and to decrement the count when we close the file, ensuring consistent operation with no surprises.

Signed-off-by: Eric W. Biederman <ebiederm@xmission.com>

```
ipc/util.c | 58 ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++-----
1 files changed, 45 insertions(+), 13 deletions(-)
```

```
diff --git a/ipc/util.c b/ipc/util.c
```

```
index a9b7a22..7fd10f1 100644
```

```
--- a/ipc/util.c
```

```
+++ b/ipc/util.c
```

```
@@ -739,14 +739,20 @@ int ipc_parse_version (int *cmd)
```

```
#endif /* __ARCH_WANT_IPC_PARSE_VERSION */
```

```
#ifdef CONFIG_PROC_FS
```

```
+struct ipc_proc_iter {
```

```
+ struct ipc_namespace *ns;
```

```
+ struct ipc_proc_iface *iface;
```

```
+};
```

```
+
```

```
static void *sysvipc_proc_next(struct seq_file *s, void *it, loff_t *pos)
```

```
{
```

```
- struct ipc_proc_iface *iface = s->private;
```

```
+ struct ipc_proc_iter *iter = s->private;
```

```
+ struct ipc_proc_iface *iface = iter->iface;
```

```
struct kern_ipc_perm *ipc = it;
```

```
loff_t p;
```

```
struct ipc_ids *ids;
```

```
- ids = current->nsproxy->ipc_ns->ids[iface->ids];
```

```
+ ids = iter->ns->ids[iface->ids];
```

```
/* If we had an ipc id locked before, unlock it */
```

```
if (ipc && ipc != SEQ_START_TOKEN)
```

```
@@ -773,12 +779,13 @@ static void *sysvipc_proc_next(struct seq_file *s, void *it, loff_t *pos)
```

```
*/
```

```

static void *sysvipc_proc_start(struct seq_file *s, loff_t *pos)
{
- struct ipc_proc_iface *iface = s->private;
+ struct ipc_proc_iter *iter = s->private;
+ struct ipc_proc_iface *iface = iter->iface;
  struct kern_ipc_perm *ipc;
  loff_t p;
  struct ipc_ids *ids;

- ids = current->nsproxy->ipc_ns->ids[iface->ids];
+ ids = iter->ns->ids[iface->ids];

  /*
   * Take the lock - this will be released by the corresponding
@@ -807,21 +814,23 @@ static void *sysvipc_proc_start(struct seq_file *s, loff_t *pos)
  static void sysvipc_proc_stop(struct seq_file *s, void *it)
  {
    struct kern_ipc_perm *ipc = it;
- struct ipc_proc_iface *iface = s->private;
+ struct ipc_proc_iter *iter = s->private;
+ struct ipc_proc_iface *iface = iter->iface;
    struct ipc_ids *ids;

    /* If we had a locked segment, release it */
    if (ipc && ipc != SEQ_START_TOKEN)
      ipc_unlock(ipc);

- ids = current->nsproxy->ipc_ns->ids[iface->ids];
+ ids = iter->ns->ids[iface->ids];
    /* Release the lock we took in start() */
    mutex_unlock(&ids->mutex);
  }

  static int sysvipc_proc_show(struct seq_file *s, void *it)
  {
- struct ipc_proc_iface *iface = s->private;
+ struct ipc_proc_iter *iter = s->private;
+ struct ipc_proc_iface *iface = iter->iface;

    if (it == SEQ_START_TOKEN)
      return seq_puts(s, iface->header);
@@ -836,22 +845,45 @@ static struct seq_operations sysvipc_proc_seqops = {
    .show = sysvipc_proc_show,
  };

- static int sysvipc_proc_open(struct inode *inode, struct file *file) {
+ static int sysvipc_proc_open(struct inode *inode, struct file *file)
+ {

```

```

    int ret;
    struct seq_file *seq;
+ struct ipc_proc_iter *iter;
+
+ ret = -ENOMEM;
+ iter = kmalloc(sizeof(*iter), GFP_KERNEL);
+ if (!iter)
+ goto out;

    ret = seq_open(file, &sysvipc_proc_seqops);
- if (!ret) {
- seq = file->private_data;
- seq->private = PDE(inode)->data;
- }
+ if (ret)
+ goto out_kfree;
+
+ seq = file->private_data;
+ seq->private = iter;
+
+ iter->iface = PDE(inode)->data;
+ iter->ns = get_ipc_ns(current->nsproxy->ipc_ns);
+out:
    return ret;
+out_kfree:
+ kfree(iter);
+ goto out;
+}
+
+static int sysvipc_proc_release(struct inode *inode, struct file *file)
+{
+ struct seq_file *seq = file->private_data;
+ struct ipc_proc_iter *iter = seq->private;
+ put_ipc_ns(iter->ns);
+ return seq_release_private(inode, file);
+}

static struct file_operations sysvipc_proc_fops = {
    .open = sysvipc_proc_open,
    .read = seq_read,
    .llseek = seq_lseek,
- .release = seq_release,
+ .release = sysvipc_proc_release,
};
#endif /* CONFIG_PROC_FS */
--
1.4.4.1.g278f

```

Containers mailing list
Containers@lists.osdl.org
<https://lists.osdl.org/mailman/listinfo/containers>
