Subject: [PATCH resend] proc: Fix proc_kill_inodes to kill dentries on all proc superblocks
Posted by ebiederm on Thu, 01 Nov 2007 15:39:45 GMT
View Forum Message <> Reply to Message

It appears we overlooked support for removing generic proc files
when we added support for multiple proc super blocks. Handle
that now.

Signed-off-by: Eric W. Biederman <ebiederm@xmission.com>
---
 fs/proc/generic.c | 38 ++++++++++++++++++++++++++----------
 fs/proc/internal.h | 2 ++
 fs/proc/root.c | 2 +-
 3 files changed, 24 insertions(+), 18 deletions(-)

diff --git a/fs/proc/generic.c b/fs/proc/generic.c
index 1bdb624..3906770 100644
--- a/fs/proc/generic.c
+++ b/fs/proc/generic.c
@@ -561,28 +561,32 @@ static int proc_register(struct proc_dir_entry * dir, struct proc_dir_entry
* dp
 static void proc_kill_inodes(struct proc_dir_entry *de)
 {
  struct list_head *p;
- struct super_block *sb = proc_mnt->mnt_sb;
+ struct super_block *sb;

  /*
   * Actually it's a partial revoke().
   */
- file_list_lock();
- list_for_each(p, &sb->s_files) {
- struct file * filp = list_entry(p, struct file, f_u.fu_list);
- struct dentry * dentry = filp->f_path.dentry;
- struct inode * inode;
- const struct file_operations *fops;
-
- if (dentry->d_op != &proc_dentry_operations)
-   continue;
- inode = dentry->d_inode;
- if (PDE(inode) != de)
-   continue;
- fops = filp->f_op;
- filp->f_op = NULL;
- fops_put(fops);
+ spin_lock(&sb_lock);
+ list_for_each_entry(sb, &proc_fs_type.fs_supers, s_instances) {

```
+  file_list_lock();
+  list_for_each(p, &sb->s_files) {
+   struct file * filp = list_entry(p, struct file, f_u.fu_list);
+   struct dentry * dentry = filp->f_path.dentry;
+   struct inode * inode;
+   const struct file_operations *fops;
+
+   if (dentry->d_op != &proc_dentry_operations)
+    continue;
+   inode = dentry->d_inode;
+   if (PDE(inode) != de)
+    continue;
+   fops = filp->f_op;
+   filp->f_op = NULL;
+   fops_put(fops);
+  }
+  file_list_unlock();
  }
- file_list_unlock();
+ spin_unlock(&sb_lock);
 }

 static struct proc_dir_entry *proc_create(struct proc_dir_entry **parent,
diff --git a/fs/proc/internal.h b/fs/proc/internal.h
index 1820eb2..1b2b6c6 100644
--- a/fs/proc/internal.h
+++ b/fs/proc/internal.h
@@ -78,3 +78,5 @@ static inline int proc_fd(struct inode *inode)
 {
  return PROC_I(inode)->fd;
 }
+
+extern struct file_system_type proc_fs_type;
diff --git a/fs/proc/root.c b/fs/proc/root.c
index ec9cb3b..1f86bb8 100644
--- a/fs/proc/root.c
+++ b/fs/proc/root.c
@@ -98,7 +98,7 @@ static void proc_kill_sb(struct super_block *sb)
  put_pid_ns(ns);
 }

-static struct file_system_type proc_fs_type = {
+struct file_system_type proc_fs_type = {
  .name  = "proc",
  .get_sb  = proc_get_sb,
  .kill_sb = proc_kill_sb,
--
1.5.3.rc6.17.g1911
```

Subject: Re: [PATCH resend] proc: Fix proc_kill_inodes to kill dentries on all proc superblocks
Posted by Pavel Emelianov on Thu, 01 Nov 2007 15:48:25 GMT
View Forum Message <> Reply to Message

Eric W. Biederman wrote:
> It appears we overlooked support for removing generic proc files
> when we added support for multiple proc super blocks. Handle
> that now.
>
> Signed-off-by: Eric W. Biederman <ebiederm@xmission.com>

AFAIS this is just making the kill for all the super blocks
we have.

Acked-by: Pavel Emelyanov <xemul@openvz.org>

> ---
> fs/proc/generic.c | 38 ++++++++++++++++++++++++++----------
> fs/proc/internal.h | 2 ++
> fs/proc/root.c | 2 +-
> 3 files changed, 31 insertions(+), 11 deletions(-)
>
> diff --git a/fs/proc/generic.c b/fs/proc/generic.c
> index 1bdb624..3906770 100644
> --- a/fs/proc/generic.c
> +++ b/fs/proc/generic.c
> @@ -561,28 +561,32 @@ static int proc_register(struct proc_dir_entry * dir, struct proc_dir_entry * dp
> static void proc_kill_inodes(struct proc_dir_entry *de)
> {
>   struct list_head *p;
> - struct super_block *sb = proc_mnt->mnt_sb;
> + struct super_block *sb;
>
>   /*
>   * Actually it's a partial revoke().
>   */
> - file_list_lock();
> - list_for_each(p, &sb->s_files) {
> - struct file * filp = list_entry(p, struct file, f_u.fu_list);

```
> -  struct dentry * dentry = filp->f_path.dentry;
> -  struct inode * inode;
> -  const struct file_operations *fops;
> -
> -  if (dentry->d_op != &proc_dentry_operations)
> -   continue;
> -  inode = dentry->d_inode;
> -  if (PDE(inode) != de)
> -   continue;
> -  fops = filp->f_op;
> -  filp->f_op = NULL;
> -  fops_put(fops);
> + spin_lock(&sb_lock);
> + list_for_each_entry(sb, &proc_fs_type.fs_supers, s_instances) {
> +  file_list_lock();
> +  list_for_each(p, &sb->s_files) {
> +   struct file * filp = list_entry(p, struct file, f_u.fu_list);
> +   struct dentry * dentry = filp->f_path.dentry;
> +   struct inode * inode;
> +   const struct file_operations *fops;
> +
> +   if (dentry->d_op != &proc_dentry_operations)
> +    continue;
> +   inode = dentry->d_inode;
> +   if (PDE(inode) != de)
> +    continue;
> +   fops = filp->f_op;
> +   filp->f_op = NULL;
> +   fops_put(fops);
> +  }
> +  file_list_unlock();
>   }
> - file_list_unlock();
> + spin_unlock(&sb_lock);
>  }
>
>  static struct proc_dir_entry *proc_create(struct proc_dir_entry **parent,
> diff --git a/fs/proc/internal.h b/fs/proc/internal.h
> index 1820eb2..1b2b6c6 100644
> --- a/fs/proc/internal.h
> +++ b/fs/proc/internal.h
> @@ -78,3 +78,5 @@ static inline int proc_fd(struct inode *inode)
>  {
>   return PROC_I(inode)->fd;
>  }
> +
> +extern struct file_system_type proc_fs_type;
> diff --git a/fs/proc/root.c b/fs/proc/root.c
```

> index ec9cb3b..1f86bb8 100644
> --- a/fs/proc/root.c
> +++ b/fs/proc/root.c
> @@ -98,7 +98,7 @@ static void proc_kill_sb(struct super_block *sb)
> put_pid_ns(ns);
> }
>
> -static struct file_system_type proc_fs_type = {
> +struct file_system_type proc_fs_type = {
> .name = "proc",
> .get_sb = proc_get_sb,
> .kill_sb = proc_kill_sb,

_____

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

---

Subject: Re: Re: [PATCH resend] proc: Fix proc_kill_inodes to kill dentries on all proc superblocks
Posted by Alexey Dobriyan on Thu, 01 Nov 2007 15:59:38 GMT
View Forum Message <> Reply to Message

On Thu, Nov 01, 2007 at 06:48:25PM +0300, Pavel Emelyanov wrote:
> Eric W. Biederman wrote:
> > It appears we overlooked support for removing generic proc files
> > when we added support for multiple proc super blocks. Handle
> > that now.
> >
> > Signed-off-by: Eric W. Biederman <ebiederm@xmission.com>
>
> AFAIS this is just making the kill for all the super blocks
> we have.
>
> Acked-by: Pavel Emelyanov <xemul@openvz.org>

bzzrt, this function is executed only for directories and symlinks,
obviously buggy (you can't flip ->f_op on live file) and is going away
after readdir/rmmod races for directories will be dealt the same way as
for regular file.

> > --- a/fs/proc/generic.c
> > +++ b/fs/proc/generic.c
> > @@ -561,28 +561,32 @@ static int proc_register(struct proc_dir_entry * dir, struct proc_dir_entry * dp
> > static void proc_kill_inodes(struct proc_dir_entry *de)
> > {

```
> >   struct list_head *p;
> > - struct super_block *sb = proc_mnt->mnt_sb;
> > + struct super_block *sb;
> >
> >   /*
> >    * Actually it's a partial revoke().
> >    */
> > - file_list_lock();
> > - list_for_each(p, &sb->s_files) {
> > -  struct file * filp = list_entry(p, struct file, f_u.fu_list);
> > -  struct dentry * dentry = filp->f_path.dentry;
> > -  struct inode * inode;
> > -  const struct file_operations *fops;
> > -
> > -  if (dentry->d_op != &proc_dentry_operations)
> > -   continue;
> > -  inode = dentry->d_inode;
> > -  if (PDE(inode) != de)
> > -   continue;
> > -  fops = filp->f_op;
> > -  filp->f_op = NULL;
> > -  fops_put(fops);
> > + spin_lock(&sb_lock);
> > + list_for_each_entry(sb, &proc_fs_type.fs_supers, s_instances) {
> > +  file_list_lock();
> > +  list_for_each(p, &sb->s_files) {
> > +   struct file * filp = list_entry(p, struct file, f_u.fu_list);
> > +   struct dentry * dentry = filp->f_path.dentry;
> > +   struct inode * inode;
> > +   const struct file_operations *fops;
> > +
> > +   if (dentry->d_op != &proc_dentry_operations)
> > +    continue;
> > +   inode = dentry->d_inode;
> > +   if (PDE(inode) != de)
> > +    continue;
> > +   fops = filp->f_op;
> > +   filp->f_op = NULL;
> > +   fops_put(fops);
> > +  }
> > +  file_list_unlock();
> >   }
> > - file_list_unlock();
> > + spin_unlock(&sb_lock);
> >  }
```

Yup, all this bogo-revoke is going away RSN. you just kicked my lazy ass.

Subject: Re: Re: [PATCH resend] proc: Fix proc_kill_inodes to kill dentries on all proc superblocks
Posted by ebiederm on Thu, 01 Nov 2007 17:07:18 GMT
View Forum Message <> Reply to Message

Alexey Dobriyan <adobriyan@sw.ru> writes:

> Yup, all this bogo-revoke is going away RSN. you just kicked my lazy ass.

As long as the issue is fixed I don't care.

Eric

Subject: Re: [PATCH resend] proc: Fix proc_kill_inodes to kill dentries on all proc superblocks
Posted by Sukadev Bhattiprolu on Thu, 01 Nov 2007 19:57:12 GMT
View Forum Message <> Reply to Message

Pavel Emelianov [xemul@openvz.org] wrote:
| Eric W. Biederman wrote:
| > It appears we overlooked support for removing generic proc files
| > when we added support for multiple proc super blocks. Handle
| > that now.
| >
| > Signed-off-by: Eric W. Biederman <ebiederm@xmission.com>
|
| AFAIS this is just making the kill for all the super blocks
| we have.
|
| Acked-by: Pavel Emelyanov <xemul@openvz.org>

Acked-by: Sukadev Bhattiprolu <sukadev@us.ibm.com>

|

```
| > ---
| >  fs/proc/generic.c  |   38 ++++++++++++++++++++++----------
| >  fs/proc/internal.h |    2 ++
| >  fs/proc/root.c     |    2 +-
| >  3 files changed, 24 insertions(+), 18 deletions(-)
| >
| > diff --git a/fs/proc/generic.c b/fs/proc/generic.c
| > index 1bdb624..3906770 100644
| > --- a/fs/proc/generic.c
| > +++ b/fs/proc/generic.c
| > @@ -561,28 +561,32 @@ static int proc_register(struct proc_dir_entry * dir, struct
proc_dir_entry * dp
| >  static void proc_kill_inodes(struct proc_dir_entry *de)
| >  {
| >   struct list_head *p;
| > - struct super_block *sb = proc_mnt->mnt_sb;
| > + struct super_block *sb;
| >
| >   /*
| >    * Actually it's a partial revoke().
| >    */
| > - file_list_lock();
| > - list_for_each(p, &sb->s_files) {
| > -  struct file * filp = list_entry(p, struct file, f_u.fu_list);
| > -  struct dentry * dentry = filp->f_path.dentry;
| > -  struct inode * inode;
| > -  const struct file_operations *fops;
| > -
| > -  if (dentry->d_op != &proc_dentry_operations)
| > -   continue;
| > -  inode = dentry->d_inode;
| > -  if (PDE(inode) != de)
| > -   continue;
| > -  fops = filp->f_op;
| > -  filp->f_op = NULL;
| > -  fops_put(fops);
| > + spin_lock(&sb_lock);
| > + list_for_each_entry(sb, &proc_fs_type.fs_supers, s_instances) {
| > +  file_list_lock();
| > +  list_for_each(p, &sb->s_files) {
| > +   struct file * filp = list_entry(p, struct file, f_u.fu_list);
| > +   struct dentry * dentry = filp->f_path.dentry;
| > +   struct inode * inode;
| > +   const struct file_operations *fops;
| > +
| > +   if (dentry->d_op != &proc_dentry_operations)
| > +    continue;
| > +   inode = dentry->d_inode;
```

```
| > +   if (PDE(inode) != de)
| > +    continue;
| > +   fops = filp->f_op;
| > +   filp->f_op = NULL;
| > +   fops_put(fops);
| > +  }
| > +  file_list_unlock();
| >   }
| > - file_list_unlock();
| > + spin_unlock(&sb_lock);
| >  }
| >
| >  static struct proc_dir_entry *proc_create(struct proc_dir_entry **parent,
| > diff --git a/fs/proc/internal.h b/fs/proc/internal.h
| > index 1820eb2..1b2b6c6 100644
| > --- a/fs/proc/internal.h
| > +++ b/fs/proc/internal.h
| > @@ -78,3 +78,5 @@ static inline int proc_fd(struct inode *inode)
| >  {
| >   return PROC_I(inode)->fd;
| >  }
| > +
| > +extern struct file_system_type proc_fs_type;
| > diff --git a/fs/proc/root.c b/fs/proc/root.c
| > index ec9cb3b..1f86bb8 100644
| > --- a/fs/proc/root.c
| > +++ b/fs/proc/root.c
| > @@ -98,7 +98,7 @@ static void proc_kill_sb(struct super_block *sb)
| >   put_pid_ns(ns);
| >  }
| >
| > -static struct file_system_type proc_fs_type = {
| > +struct file_system_type proc_fs_type = {
| >   .name  = "proc",
| >   .get_sb  = proc_get_sb,
| >   .kill_sb = proc_kill_sb,
```