
Subject: Re: [PATCH 2/2] Replace pid_t in autofs with struct pid reference

Posted by [serue](#) on Tue, 20 Mar 2007 20:15:48 GMT

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

Quoting Eric W. Biederman (ebiederm@xmission.com):

> "Serge E. Hallyn" <serue@us.ibm.com> writes:

>

> >

> > > Index: 2.6.20/fs/autofs4/waitq.c

> > > =====

> > > --- 2.6.20.orig/fs/autofs4/waitq.c

> > > +++ 2.6.20/fs/autofs4/waitq.c

> > > @@ -292,8 +292,8 @@ int autofs4_wait(struct autofs_sb_info *

> > > wq->ino = autofs4_get_ino(sbi);

> > > wq->uid = current->uid;

> > > wq->gid = current->gid;

> > > - wq->pid = current->pid;

> > > - wq->tgid = current->tgid;

> > > + wq->pid = pid_nr(task_pid(current));

> > > + wq->tgid = pid_nr(task_tgid(current));

> > > wq->status = -EINTR; /* Status return if interrupted */

> > > atomic_set(&wq->wait_ctr, 2);

> > > mutex_unlock(&sbi->wq_mutex);

> >

> > I have a concern with this bit as I my quick review said the wait queue
> > persists, and if so we should be cache the struct pid pointer, not the
> > pid_t value. Heck the whol pid_nr(task_xxx(current)) idiom I find very
> > suspicious.

> >

> > Based just on what I see right here I agree it seems like we would want
> > to store a ref to the pid, not store the pid_nr(pid) output, so in this
> > context it is suspicious.

>

> So that far we are in agreement.

So how about the following on top of the patch Cedric sent out?

Subject: [PATCH] autofs4: store struct pids in autofs_waitqs

From: Serge Hallyn <serue@us.ibm.com>

Date: 1174412305 -0500

Store struct pids in autofs_waitqs in place of pidnrs to prevent
pid overflow problems.

Signed-off-by: Serge E. Hallyn <serue@us.ibm.com>

```
fs/autofs4/autofs_i.h | 13 ++++++++--
fs/autofs4/waitq.c | 15 ++++++-----
2 files changed, 19 insertions(+), 9 deletions(-)
```

```
86a5866c7672b88d48380504977496d5637642ab
diff --git a/fs/autofs4/autofs_i.h b/fs/autofs4/autofs_i.h
index 3ccec0a..5d119e3 100644
```

```
--- a/fs/autofs4/autofs_i.h
+++ b/fs/autofs4/autofs_i.h
@@ -79,8 +79,8 @@ struct autofs_wait_queue {
    u64 ino;
    uid_t uid;
    gid_t gid;
-   pid_t pid;
-   pid_t tgid;
+   struct pid *pid;
+   struct pid *tgid;
    /* This is for status reporting upon return */
    int status;
    atomic_t wait_ctr;
@@ -228,5 +228,14 @@ out:
    return ret;
}
```

```
+static void autofs_free_wait_queue(struct autofs_wait_queue *wq)
+{
+   if (wq->pid)
+       put_pid(wq->pid);
+   if (wq->tgid)
+       put_pid(wq->tgid);
+   kfree(wq);
+}
```

```
+
void autofs4_dentry_release(struct dentry *);
extern void autofs4_kill_sb(struct super_block *);
```

```
diff --git a/fs/autofs4/waitq.c b/fs/autofs4/waitq.c
index 9857543..4a9ad9b 100644
```

```
--- a/fs/autofs4/waitq.c
+++ b/fs/autofs4/waitq.c
@@ -141,8 +141,8 @@ static void autofs4_notify_daemon(struct
    packet->ino = wq->ino;
    packet->uid = wq->uid;
    packet->gid = wq->gid;
-   packet->pid = wq->pid;
-   packet->tgid = wq->tgid;
+   packet->pid = pid_nr(wq->pid);
+   packet->tgid = pid_nr(wq->tgid);
    break;
```

```

}
default:
@@ -292,8 +292,8 @@ int autofs4_wait(struct autofs_sb_info *
    wq->ino = autofs4_get_ino(sbi);
    wq->uid = current->uid;
    wq->gid = current->gid;
- wq->pid = pid_nr(task_pid(current));
- wq->tgid = pid_nr(task_tgid(current));
+ wq->pid = get_pid(task_pid(current));
+ wq->tgid = get_pid(task_tgid(current));
    wq->status = -EINTR; /* Status return if interrupted */
    atomic_set(&wq->wait_ctr, 2);
    mutex_unlock(&sbi->wq_mutex);
@@ -360,8 +360,9 @@ int autofs4_wait(struct autofs_sb_info *
    status = wq->status;

    /* Are we the last process to need status? */
- if (atomic_dec_and_test(&wq->wait_ctr))
- kfree(wq);
+ if (atomic_dec_and_test(&wq->wait_ctr)) {
+ autofs_free_wait_queue(wq);
+ }

    return status;
}
@@ -390,7 +391,7 @@ int autofs4_wait_release(struct autofs_s
    wq->status = status;

    if (atomic_dec_and_test(&wq->wait_ctr)) /* Is anyone still waiting for this guy? */
- kfree(wq);
+ autofs_free_wait_queue(wq);
    else
        wake_up_interruptible(&wq->queue);

```

--

1.1.6

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