Subject: Re: [PATCH 2/2] Replace pid_t in autofs with struct pid reference
Posted by ebiederm on Fri, 16 Mar 2007 19:21:51 GMT
View Forum Message <> Reply to Message

Ian Kent <raven@themaw.net> writes:

> On Fri, 2007-03-16 at 15:44 +0100, Cedric Le Goater wrote:
>> > How about you send over the autofs4 bit and I'll have a look (the autofs
>> > patch looked fine). That would save me a bit of time and if there are
>> > any changes needed I can send an updated patch for you guys to review. I
>> > don't think autofs4 uses pids differently, in principle, than autofs so
>> > it "should" be straight forward.
>>
>> Here's the latest.
>
> That looks OK to me, assuming the "find_get_pid" and friends do what
> they suggest, but I'll give it a closer look tomorrow.
>
> A ref count is used here, what affect does that have on a thread (or
> process) that may go away (or be summarily killed) without umounting the
> mount?

Nothing.

The primary advantage is that you are pid wrap around safe as the struct
pid will never point to another process after one of those events occurs.

struct pid is a very small structure so not freeing it when the process
it originally referred to goes away is no big deal. Although not leaking
when you stop using it is still important.

The other big use of struct pid is that to get the user space pid value
you call pid_nr(). Depending on the pid namespace of the caller the return
value of pid_nr() can be different. So when you store a pid or pass a pid
between processes that should be done by passing a struct pid because those
processes could be in different pid namespaces.

```
>> 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.

Eric

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