Subject: Re: [PATCH] Consolidate sleeping routines in file locking code
Posted by Pavel Emelianov on Fri, 21 Sep 2007 06:57:56 GMT
View Forum Message <> Reply to Message

```
J. Bruce Fields wrote:
> On Thu, Sep 20, 2007 at 01:09:51PM +0400, Pavel Emelyanov wrote:
>> J. Bruce Fields wrote:
>>> On Tue, Sep 18, 2007 at 05:41:08PM +0400, Pavel Emelyanov wrote:
>>>> This is the next step in fs/locks.c cleanup before turning
>>>> it into using the struct pid *.
>>>>
>>>> This time I found, that there are some places that do a
>>>> similar thing - they try to apply a lock on a file and go
>>>> to sleep on error till the blocker exits.
>>>>
>>>> All these places can be easily consolidated, saving 28
>>>> lines of code and more than 600 bytes from the .text,
>>>> but there is one minor note.
>>> I'm not opposed to consolidating this code, but would it be possible to
>>> do so in a more straightforward way, without passing in a callback
>>> function?  E.g. a single __posix_lock_file_wait that just took an inode
>>> instead of a filp and called __posix_lock_file() could be called from
>>> both posix_lock_file_wait() and locks_mandatory_locked, right?
>> Well, the locks_mandatory_area() has to check for inode mode change
>> in my lock callback, the fcntl_setlk() has to call the vfs_lock_file,
>> and flock_lock_file_wait() has to call the flock_lock_file, so
>> I don't see the ways of having one routine to lock the file.
>>
>> If you don't mind, I'd port the patch with this approach (with the
>> "trylock" callback) on the latest Andrew's tree.
>
> OK.

:) Thanks.

>>>> The locks_mandatory_area() code becomes a bit different
>>>> after this patch - it no longer checks for the inode's
>>>> permissions change. Nevertheless, this check is useless
>>>> without my another patch that wakes the waiter up in the
>>>> notify_change(), which is not considered to be useful for
>>>> now.
>>> OK.  Might be better to submit this as a separate patch, though.
>> This one is already accepted, but I have just noticed that
>> the check for __mandatory_lock() in wait_event_interruptible
>> is ambiguous :(
>
> I'm not sure what you mean here....  Do you have a fix?
```

Well, I do, but this patch is already dropped from -mm.

> --b.
>