Subject: Re: [PATCH] Wake up mandatory locks waiter on chmod
Posted by Pavel Emelianov on Tue, 18 Sep 2007 06:36:32 GMT
View Forum Message <> Reply to Message

J. Bruce Fields wrote:
> On Mon, Sep 17, 2007 at 10:37:56AM +0400, Pavel Emelyanov wrote:
>> J. Bruce Fields wrote:
>>> Is there a small chance that a lock may be applied after this check:
>>>
>>>> + mandatory = (inode->i_flock && MANDATORY_LOCK(inode));
>>>> +
>>> but early enough that someone can still block on the lock while the file
>>> is still marked for mandatory locking? (And is the inode->i_flock check
>>> there really necessary?)
>> There is, but as you have noticed:
>
> OK, but why not just remove the inode->i_flock check there? I can't see
> how it helps anyway.
>
>>> Well, there are probably worse races in the mandatory locking code.
>> ...there are. The inode->i_lock is protected with lock_kernel() only
>> and is not in sync with any other checks for inodes. This is sad :(
>> but a good locking for locks is to be done...
>
> I would also prefer a locking scheme that didn't rely on the BKL. That
> said, except for this race:

I would as well :) But I don't know the locking code good enough to
start fixing. Besides, even if I send a patch series that handles this,
I don't think that anyone will accept it, due to "this changes too much
code", "can you prove you fixed all the places" and so on...

>>> (For example, my impression is that a mandatory lock can be applied just
>>> after the locks_mandatory_area() checks but before the io actually
>>> completes.)
>
> ... I'm not aware of other races in the existing file-locking code. It
> sounds like you might be. Could you give specific examples?

Well, there's a long standing BUG in leases code - when we made all the
checks in inserting lease, we call the locks_alloc_lock() and may fall
asleep. Bu after the wakeup nobody re-checks for the things to change.

I suspect there are other bad places.

> --b.
>