Ruby master - Feature #15006

[PATCH] io.c: use copy_file_range with every types of files

08/18/2018 03:33 AM - byroot (Jean Boussier)

Status:	Open
Priority:	Normal
Assignee:	
Target version:	

Description

Ref: https://bugs.ruby-lang.org/issues/13867

IO.copy_stream only attempt to use copy_file_range if the source is a regular file.

However from my understanding, contrary to sendfile and splice, copy_file_range has no file type restriction, it should be able to copy from and to sockets and pipes just fine.

It does have very optimized paths for regular files on specific file systems, but for other fd types it will fallback to do the copy using the page cache:

https://lwn.net/Articles/659523/

> If the function is absent or returns failure, the kernel will, if the COPY_FR_COPY flag is set, fall back to copying through the page cache.

So if it's available, it's preferable to nogvl_copy_stream_read_write.

GitHub PR: https://github.com/ruby/ruby/pull/1932

History

#1 - 08/20/2018 07:52 PM - normalperson (Eric Wong)

jean.boussier@gmail.com wrote:

> However from my understanding, contrary to sendfile and
> splice, copy_file_range has no file type restriction, it
> should be able to copy from and to sockets and pipes just
> fine.

That is incorrect, copy_file_range requires both source and
destination as regular files:

https://git.kernel.org/pub/scm/linux/kernel/git/torvalds/linux.git/tree/fs/read_write.c?h=v4.18#n1562

Did you try running "make test-all" with your patch?
It fails test/ruby/test_io.rb::test_copy_stream_megacontent_nonblock

#2 - 08/21/2018 11:38 AM - byroot (Jean Boussier)

> That is incorrect, copy_file_range requires both source and
> destination as regular files

Damn, I totally got mislead by that LWN article & the man page. I really searched for it but couldn't find an indication that only real files were accepted.

> Did you try running "make test-all" with your patch?

Unfortunately I don't have a Linux handy, but I should have used a VM, sorry about that.

In this case, how would you feel about another patch to add splice support to IO.copy_file?

My use case is that I need to efficiently write from a socket to a pipe, and sometimes the machine is CPU limited, so the performance tank when using write(read()).

Since IIRC you are the author of io_splice, you might have insights as of why IO.copy_stream doesn't try to use it?

#3 - 08/21/2018 07:52 PM - normalperson (Eric Wong)

jean.boussier@gmail.com wrote:

> In this case, how would you feel about another patch to add
> splice support to IO.copy_file?

Sure!

> My use case is that I need to efficiently write from a socket
> to a pipe, and sometimes the machine is CPU limited, so the
> performance tank when using write(read()).
>
> Since IIRC you are the author of io_splice, you might have
> insights as of why IO.copy_stream doesn't try to use it?

Laziness and caution:

In the past (Linux 2.6.1x and maybe 2.6.2x), splice could hang, leak memory or corrupt data. This was definitely a problem on CentOS 5.x around 2007/2008 and I had to reboot machines after trying it :x

And I hit problems with it even in 3.7.3:
https://lore.kernel.org/r/20130119044957.GA25395@dcvr.yhbt.net

Nowadays, it's probably OK :)

CentOS 5.x is no longer supported and I'm not sure if anybody still uses Linux 2.6.1x/2.6.2x. I guess 2.6.32+ is a safe target (CentOS 6.x+), but I haven't followed CentOS in a while, now; and I don't think any LTS 3.x+ kernels have unfixed problems.

Fwiw, I usually use non-blocking I/O with pipes, and IO.copy_stream isn't useful for non-blocking I/O. However, Feature #13618 could make IO.copy_stream for non-blocking pipes more compelling.

Files

copy-file-range.patch	1.56 KB	08/18/2018	byroot (Jean Boussier)