In Linux 4.5, the copy offloading feature with `copy_file_range(2)` was introduced. This patch enables `IO.copy_stream` to use it. If the "offload" keyword argument is specified, `IO.copy_stream` will try copy offloading.

```ruby
IO.copy_stream("src", "dst", offload: true)
```

If copy offloading is not available on the system, the "offload" option will be ignored.

When "src" and "dst" are not in the same filesystem, `copy_file_range(2)` must fail with EXDEV. In this case, `IO.copy_stream` will fallback to another method (sendfile(2) or read/write) silently.

It depends on the filesystem what offloading techniques will be used. Copy offloading is optional in `IO.copy_stream` because some techniques may change the current behavior.

Associated revisions

Revision 524e6608 - 10/27/2017 06:37 PM - normal

io.c: fix IO.copy_stream on O_APPEND destination on Linux

Linux `copy_file_range(2)` fails with EBADF if the destination FD has O_APPEND set. Preserve existing (Ruby <= 2.4) behavior by falling back to alternative copy mechanisms if this is the case (instead of raising Errno::EBADF).

- io.c (nogvl_copy_file_range): do not raise on O_APPEND dst
- test/ruby/test_io.rb (test_copy_stream_append): new test [Feature #13867]

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@60490 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision 60490 - 10/27/2017 06:37 PM - normal

io.c: fix IO.copy_stream on O_APPEND destination on Linux

Linux `copy_file_range(2)` fails with EBADF if the destination FD has O_APPEND set. Preserve existing (Ruby <= 2.4) behavior by falling back to alternative copy mechanisms if this is the case (instead of raising Errno::EBADF).

- io.c (nogvl_copy_file_range): do not raise on O_APPEND dst
- test/ruby/test_io.rb (test_copy_stream_append): new test [Feature #13867]

Revision 60490 - 10/27/2017 06:37 PM - normal

io.c: fix IO.copy_stream on O_APPEND destination on Linux

Linux `copy_file_range(2)` fails with EBADF if the destination FD has O_APPEND set. Preserve existing (Ruby <= 2.4) behavior by falling back to alternative copy mechanisms if this is the case (instead of raising Errno::EBADF).

- io.c (nogvl_copy_file_range): do not raise on O_APPEND dst
- test/ruby/test_io.rb (test_copy_stream_append): new test [Feature #13867]

Revision 60490 - 10/27/2017 06:37 PM - normal

io.c: fix IO.copy_stream on O_APPEND destination on Linux

Linux `copy_file_range(2)` fails with EBADF if the destination FD has O_APPEND set. Preserve existing (Ruby <= 2.4) behavior by falling back to alternative copy mechanisms if this is the case (instead of raising Errno::EBADF).

- io.c (nogvl_copy_file_range): do not raise on O_APPEND dst
- test/ruby/test_io.rb (test_copy_stream_append): new test [Feature #13867]

Revision 60490 - 10/27/2017 06:37 PM - normal

io.c: fix IO.copy_stream on O_APPEND destination on Linux

Linux `copy_file_range(2)` fails with EBADF if the destination FD has O_APPEND set. Preserve existing (Ruby <= 2.4) behavior by falling back to alternative copy mechanisms if this is the case (instead of raising Errno::EBADF).

- io.c (nogvl_copy_file_range): do not raise on O_APPEND dst
- test/ruby/test_io.rb (test_copy_stream_append): new test [Feature #13867]
has O_APPEND set. Preserve existing (Ruby <= 2.4) behavior by falling back to alternative copy mechanisms if this is the case (instead of raising Errno::EBADF).

- io.c (nogvl_copy_file_range): do not raise on O_APPEND dst
- test/ruby/test_io.rb (test_copy_stream_append): new test [Feature #13867]

**History**

### #1 - 09/05/2017 11:08 AM - normalperson (Eric Wong)

glass.saga@gmail.com wrote:

In linux 4.5, the copy offloading feature with copy_file_range(2) was introduced. This patch enables IO.copy_stream to use it.

Cool.

If "offload" keyword argument is specified, IO.copy_stream will try copy offloading.

Is a new keyword arg necessary? Since this is Linux-only, and we already do fstat tests on both FDs in the sendfile checking:

We can try using copy_file_range if both src and dst are both regular files with identical .st_dev, and fall back to using sendfile on ENOSYS.

Thanks.

### #2 - 10/20/2017 01:17 AM - shyouhei (Shyouhei Urabe)

We took a look at this issue at yesterday's developer meeting and agreed with Eric here; it can be automatically applied whenever possible. Seems there are no practical reasons to disable this feature.

### #3 - 10/21/2017 06:31 AM - Glass_saga (Masaki Matsushita)

- Assignee set to Glass_saga (Masaki Matsushita)
- Status changed from Open to Assigned
- File patch2.diff added

This patch makes IO.copy_stream use copy_file_range(2) without keyword arguments.

### #4 - 10/21/2017 02:54 PM - Glass_saga (Masaki Matsushita)

- Status changed from Assigned to Closed

commited in r60284.

### #5 - 10/27/2017 06:51 PM - normalperson (Eric Wong)

glass.saga@gmail.com wrote:

commited in r60284.

https://bugs.ruby-lang.org/issues/13867#change-67462

While working on r60490 to workaround O_APPEND; I noticed there's an unnecessary fstat call for dst_stat. I don't see how it is necessary to fstat dst_fd since copy_file_range will fail anyways on invalid FDs.

**Files**

<table>
<thead>
<tr>
<th>File</th>
<th>Size</th>
<th>Date</th>
<th>Author</th>
</tr>
</thead>
<tbody>
<tr>
<td>patch.diff</td>
<td>7.21 KB</td>
<td>09/05/2017</td>
<td>Glass_saga (Masaki Matsushita)</td>
</tr>
<tr>
<td>patch2.diff</td>
<td>3.35 KB</td>
<td>10/21/2017</td>
<td>Glass_saga (Masaki Matsushita)</td>
</tr>
</tbody>
</table>