Ruby master - Feature #16965

io.c: Unsupported copy_file_range() not detected properly on certain kernels

06/17/2020 10:04 PM - stanhu (Stan Hu)

Status:	Closed
Priority:	Normal
Assignee:	Glass_saga (Masaki Matsushita)
Target version:	

Description

In recent RedHat kernels (for example: RHEL 7.8 using kernel 3.10.0-1127.8.2.el7.x86_64), copy_file_range() may return EOPNOTSUP when Ruby attempts to call this in IO.copy_stream on an NFS mount. A simple FileUtils.copy_file will fail with Operation not supported - copy_file_range on these kernels.

This was possibly changed during a recent security release: https://access.redhat.com/errata/RHSA-2020:1465

Ruby's io.c detects whether copy_file_range() is defined, not whether it is actually supported. The following test program illustrates the hole in the detection mechanism:

```
#include <syscall.h>
#include <stdio.h>

#if defined __linux__ && defined __NR_copy_file_range
#  define USE_COPY_FILE_RANGE 1
#else
#  define USE_COPY_FILE_RANGE 0
#endif

int main()
{
  printf("copy_file_range? %d\n", USE_COPY_FILE_RANGE);
}
```

USE_COPY_FILE_RANGE gets set to 1 even in when the system call doesn't succeed.

I suggest a few improvements:

1. Use a compile-time test to verify that copy_file_range() can actually be executed.
2. Make it possible to disable USE_COPY_FILE_RANGE via a build option. Since the test in 1 could still pass if it is run on a Docker host that supports copy_file_range(), it would be helpful for us to manually disable it.

Reported by GitLab customers: https://gitlab.com/gitlab-org/gitlab/-/issues/218999

Associated revisions

Revision 93df3010 - 08/29/2020 06:38 AM - Glass_saga (Masaki Matsushita)

IO.copy_stream: handle ENOTSUP on copy_file_range(2)

fallback to other methods on ENOTSUP.
some RedHat kernels may return ENOTSUP on an NFS mount.
[Feature #16965]

History

#1 - 06/17/2020 10:10 PM - vo.x (Vit Ondruch)

This is tracked here:

https://bugzilla.redhat.com/show_bug.cgi?id=1840284

and considered Kernel issue. I think that in recent Rubies, there is compile-time test already.

#2 - 06/17/2020 10:30 PM - stanhu (Stan Hu)

Thanks for the link! As far as I can tell, the test in https://github.com/ruby/ruby/blob/41a4c80d284a24360d3a6c5b6e5ca408ccca6efc/io.c#L11007 is still the same as I described above.

#3 - 06/17/2020 11:41 PM - mame (Yusuke Endoh)

- Assignee set to Glass_saga (Masaki Matsushita)

I think that Ruby handles the ENOSYS appropriately and fall back to fcopyfile and sendfile:
https://github.com/ruby/ruby/blob/41a4c80d284a24360d3a6c5b6e5ca408ccca6efc/io.c#L11084

"Operation not supported" error is not ENOSYS but EOPNOTSUP, which Ruby does not handle. The issue is that the recent RedHat kernel may return EOPNOTSUP. The RedHat developers are aware of the incompatibility, and looks like they will change the return value to ENOSYS:
https://bugzilla.redhat.com/show_bug.cgi?id=1783554

So, please wait for their fix.

Aside from that, is it good for Ruby to rescue EOPNOTSUP for future robustness? Glass_saga (Masaki Matsushita)

#4 - 06/17/2020 11:42 PM - mame (Yusuke Endoh)

- Backport deleted (2.5: UNKNOWN, 2.6: UNKNOWN, 2.7: UNKNOWN)

- ruby -v deleted (ruby 2.6.6p146 (2020-03-31 revision 67876) [x86_64-linux])

- Tracker changed from Bug to Feature

#5 - 06/18/2020 05:44 AM - stanhu (Stan Hu)

- Description updated

#6 - 06/18/2020 06:54 AM - vo.x (Vit Ondruch)

I don't think EOPNOTSUP is documented error state, therefore I am against this. Not mentioning that it would need backport to older Rubies and what not. In the time this is going to be implemented in Ruby, updated RHEL kernel will be long released.

#7 - 08/29/2020 06:41 AM - Glass_saga (Masaki Matsushita)

- Status changed from Open to Closed

Applied in changeset git|93df3010482ad52e5ada2e416c996005da956e1e.

IO.copy_stream: handle ENOTSUP on copy_file_range(2)

fallback to other methods on ENOTSUP.
some RedHat kernels may return ENOTSUP on an NFS mount.
[Feature #16965]