Ruby - Bug #14968

[PATCH] io.c: make all pipes nonblocking by default

08/06/2018 04:46 AM - normalperson (Eric Wong)

Status:	Closed		
Priority:	Normal		
Assignee:			
Target version:			
ruby -v:		**Backport:**	2.3: UNKNOWN, 2.4: UNKNOWN, 2.5: UNKNOWN

Description

Crap, I always planned to have something like this for [Feature #13618] (auto-Fiber); but introducing a race condition for Timer-thread elimination in [Misc #14937] forced me to introduce this early.

Anyways, I might have to revert and reintroduce timer-thread if this change is unnacceptable :< (I HATE timer thread)

```
io.c: make all pipes nonblocking by default

All normal Ruby IO methods (IO#read, IO#gets, IO#write, ...) are
all capable of appearing to be "blocking" when presented with a
file description with the O_NONBLOCK flag set; so there is
little risk of incompatibility within Ruby-using programs.

The biggest compatibility risk is when spawning external
programs.  As a result, stdin, stdout, and stderr are now always
made blocking before exec-family calls.

Timer-thread elimination in https://bugs.ruby-lang.org/issues/14937
introduced a race condition in signal handling.  It is possible
to receive a signal inside BLOCKING_REGION right before read/write
syscalls.  If this patch cannot be accepted, I will have to revert
to reintroduce timer-thread and increase resource use (which
led to other failures in the past).  The race condition
introduced for [Misc #14937] led to rare CI failures on a few
tests:

- test/ruby/test_thread.rb (test_thread_timer_and_interrupt):
  http://www.rubyist.net/~akr/chkbuild/debian/ruby-trunk/log/20180805T080500Z.fail.html.gz

- test/ruby/test_io.rb (test_race_gets_and_close):
  http://ci.rvm.jp/results/trunk@P895/1190369

This change is ALSO necessary to take advantage of (proposed
lightweight concurrency (aka "auto-Fiber") or any similar
proposal: https://bugs.ruby-lang.org/issues/13618

TODO: all sockets and FIFOs non-blocking by default, too
```

Associated revisions

Revision 6a65f2b1e479f268489b51a004b6c153c634c68a - 11/22/2018 08:46 AM - Eric Wong

io + socket: make pipes and sockets nonblocking by default

All normal Ruby IO methods (IO#read, IO#gets, IO#write, ...) are
all capable of appearing to be "blocking" when presented with a
file description with the O_NONBLOCK flag set; so there is
little risk of incompatibility within Ruby-using programs.

The biggest compatibility risk is when spawning external
programs. As a result, stdin, stdout, and stderr are now always
made blocking before exec-family calls.

This change will make an event-oriented MJIT usable if it is
waiting on pipes on POSIX_like platforms.

It is ALSO necessary to take advantage of (proposed lightweight
concurrency (aka "auto-Fiber") or any similar proposal for
network concurrency: https://bugs.ruby-lang.org/issues/13618

Named-pipe (FIFO) are NOT yet non-blocking by default since
they are rarely-used and may introduce compatibility problems
and extra syscall overhead for a common path.

Please revert this commit if there are problems and if I am afk
since I am afk a lot, lately.

[ruby-core:89950] [Bug #14968]

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@65922 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision 6a65f2b1e479f268489b51a004b6c153c634c68a - 11/22/2018 08:46 AM - Eric Wong

io + socket: make pipes and sockets nonblocking by default

All normal Ruby IO methods (IO#read, IO#gets, IO#write, ...) are
all capable of appearing to be "blocking" when presented with a
file description with the O_NONBLOCK flag set; so there is
little risk of incompatibility within Ruby-using programs.

The biggest compatibility risk is when spawning external
programs. As a result, stdin, stdout, and stderr are now always
made blocking before exec-family calls.

This change will make an event-oriented MJIT usable if it is
waiting on pipes on POSIX_like platforms.

It is ALSO necessary to take advantage of (proposed lightweight
concurrency (aka "auto-Fiber") or any similar proposal for
network concurrency: https://bugs.ruby-lang.org/issues/13618

Named-pipe (FIFO) are NOT yet non-blocking by default since
they are rarely-used and may introduce compatibility problems
and extra syscall overhead for a common path.

Please revert this commit if there are problems and if I am afk
since I am afk a lot, lately.

[ruby-core:89950] [Bug #14968]

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@65922 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision 6a65f2b1 - 11/22/2018 08:46 AM - Eric Wong

io + socket: make pipes and sockets nonblocking by default

All normal Ruby IO methods (IO#read, IO#gets, IO#write, ...) are
all capable of appearing to be "blocking" when presented with a
file description with the O_NONBLOCK flag set; so there is
little risk of incompatibility within Ruby-using programs.

The biggest compatibility risk is when spawning external
programs. As a result, stdin, stdout, and stderr are now always
made blocking before exec-family calls.

This change will make an event-oriented MJIT usable if it is
waiting on pipes on POSIX_like platforms.

It is ALSO necessary to take advantage of (proposed lightweight
concurrency (aka "auto-Fiber") or any similar proposal for
network concurrency: https://bugs.ruby-lang.org/issues/13618

Named-pipe (FIFO) are NOT yet non-blocking by default since
they are rarely-used and may introduce compatibility problems
and extra syscall overhead for a common path.

Please revert this commit if there are problems and if I am afk
since I am afk a lot, lately.

[ruby-core:89950] [Bug #14968]

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@65922 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision 0bd8193eba5139812c18f779ba5831b3c7df01d7 - 11/22/2018 10:13 AM - Eric Wong

ext/socket/init.c (rsock_socket0): non-blocking for non-SOCK_NONBLOCK

We need to make sockets non-blocking for systems without
SOCK_CLOEXEC/SOCK_NONBLOCK macros at all.

[ruby-core:89965] [Bug #14968]

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@65925 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision 0bd8193eba5139812c18f779ba5831b3c7df01d7 - 11/22/2018 10:13 AM - Eric Wong

ext/socket/init.c (rsock_socket0): non-blocking for non-SOCK_NONBLOCK

We need to make sockets non-blocking for systems without
SOCK_CLOEXEC/SOCK_NONBLOCK macros at all.

[ruby-core:89965] [Bug #14968]

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@65925 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision 0bd8193e - 11/22/2018 10:13 AM - Eric Wong

ext/socket/init.c (rsock_socket0): non-blocking for non-SOCK_NONBLOCK

We need to make sockets non-blocking for systems without
SOCK_CLOEXEC/SOCK_NONBLOCK macros at all.

[ruby-core:89965] [Bug #14968]

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@65925 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision 0698c4969cc8688bc9979a8601656ccdc7a2e601 - 11/22/2018 08:02 PM - Eric Wong

socket: disable nonblocking-by-default on win32

Perhaps this fixes test failures reported by Greg and k0kubun.

However, the failure of certain tests to handle non-blocking I/O
seems to indicate pre-existing problems on win32 platforms.
Somebody knowledgeable about win32 should be able to fix it.

[ruby-core:89973] [ruby-core:89976] [ruby-core:89977] [Bug #14968]

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@65929 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision 0698c4969cc8688bc9979a8601656ccdc7a2e601 - 11/22/2018 08:02 PM - Eric Wong

socket: disable nonblocking-by-default on win32

Perhaps this fixes test failures reported by Greg and k0kubun.

However, the failure of certain tests to handle non-blocking I/O
seems to indicate pre-existing problems on win32 platforms.
Somebody knowledgeable about win32 should be able to fix it.

[ruby-core:89973] [ruby-core:89976] [ruby-core:89977] [Bug #14968]

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@65929 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision 0698c496 - 11/22/2018 08:02 PM - Eric Wong

socket: disable nonblocking-by-default on win32

Perhaps this fixes test failures reported by Greg and k0kubun.

However, the failure of certain tests to handle non-blocking I/O
seems to indicate pre-existing problems on win32 platforms.
Somebody knowledgeable about win32 should be able to fix it.

[ruby-core:89973] [ruby-core:89976] [ruby-core:89977] [Bug #14968]

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@65929 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision 8a233ea67b20910c6fc3429d6c30c628c1650255 - 11/22/2018 10:44 PM - Eric Wong

io.c: revalidate fptr->fd after rb_io_wait_readable

fptr->fd may become -1 while GVL is released in
rb_wait_for_single_fd, so we must check it after reacquiring
GVL. This should avoid EBADF errors exposed by making pipes
non-blocking by default:

http://ci.rvm.jp/results/trunk-test@ruby-sky3/1473710

[Bug #14968]

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@65931 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision 8a233ea67b20910c6fc3429d6c30c628c1650255 - 11/22/2018 10:44 PM - Eric Wong

io.c: revalidate fptr->fd after rb_io_wait_readable

fptr->fd may become -1 while GVL is released in
rb_wait_for_single_fd, so we must check it after reacquiring
GVL. This should avoid EBADF errors exposed by making pipes
non-blocking by default:

http://ci.rvm.jp/results/trunk-test@ruby-sky3/1473710

[Bug #14968]

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@65931 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision 8a233ea6 - 11/22/2018 10:44 PM - Eric Wong

io.c: revalidate fptr->fd after rb_io_wait_readable

fptr->fd may become -1 while GVL is released in
rb_wait_for_single_fd, so we must check it after reacquiring
GVL. This should avoid EBADF errors exposed by making pipes
non-blocking by default:

http://ci.rvm.jp/results/trunk-test@ruby-sky3/1473710

[Bug #14968]

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@65931 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision c0e20037f3fb16d75b55394e7bf0a1d3ef8b7b87 - 11/24/2018 08:23 AM - Eric Wong

io.c: wait on FD readability w/o GVL reacquisition

Since non-blocking I/O is the default after [Bug #14968],
we will hit it more often and cause more acquisition/release
of GVL to wait on single FD.

This also lets us avoid touching the temporal string locking
as much and lets us clean up some test changes made for
[Bug #14968]

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@65948 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision c0e20037f3fb16d75b55394e7bf0a1d3ef8b7b87 - 11/24/2018 08:23 AM - Eric Wong

io.c: wait on FD readability w/o GVL reacquisition

Since non-blocking I/O is the default after [Bug #14968],
we will hit it more often and cause more acquisition/release
of GVL to wait on single FD.

This also lets us avoid touching the temporal string locking
as much and lets us clean up some test changes made for
[Bug #14968]

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@65948 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision c0e20037 - 11/24/2018 08:23 AM - Eric Wong

io.c: wait on FD readability w/o GVL reacquisition

Since non-blocking I/O is the default after [Bug #14968],
we will hit it more often and cause more acquisition/release
of GVL to wait on single FD.

This also lets us avoid touching the temporal string locking
as much and lets us clean up some test changes made for
[Bug #14968]

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@65948 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision 608b9c29133e6d4c0822f1500e45c2a8073891be - 11/24/2018 07:59 PM - Eric Wong

io.c: disable nonblocking-by-default on win32 pipes

Lets admit Windows will always be too different from POSIX-like
platforms and non-blocking may never work as well or consistently.

[ruby-core:90042] [ruby-core:90044] [Bug #14968]

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@65962 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision 608b9c29133e6d4c0822f1500e45c2a8073891be - 11/24/2018 07:59 PM - Eric Wong

io.c: disable nonblocking-by-default on win32 pipes

Lets admit Windows will always be too different from POSIX-like
platforms and non-blocking may never work as well or consistently.

[ruby-core:90042] [ruby-core:90044] [Bug #14968]

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@65962 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision 608b9c29 - 11/24/2018 07:59 PM - Eric Wong

io.c: disable nonblocking-by-default on win32 pipes

Lets admit Windows will always be too different from POSIX-like
platforms and non-blocking may never work as well or consistently.

[ruby-core:90042] [ruby-core:90044] [Bug #14968]

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@65962 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision 9d74d402e15008c10f80e67595cc861c89a1636b - 11/29/2018 08:00 PM - Eric Wong

disable non-blocking pipes and sockets by default

There seems to be a compatibility problems with Rails +
Rack::Deflater; so we revert this incompatibility.

This effectively reverts r65922; but keeps the bugfixes to
better support non-blocking sockets and pipes for future use.

[Bug #15356] [Bug #14968]

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@66093 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision 9d74d402e15008c10f80e67595cc861c89a1636b - 11/29/2018 08:00 PM - Eric Wong

disable non-blocking pipes and sockets by default

There seems to be a compatibility problems with Rails +
Rack::Deflater; so we revert this incompatibility.

This effectively reverts r65922; but keeps the bugfixes to
better support non-blocking sockets and pipes for future use.

[Bug #15356] [Bug #14968]

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@66093 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision 9d74d402 - 11/29/2018 08:00 PM - Eric Wong

disable non-blocking pipes and sockets by default

There seems to be a compatibility problems with Rails +
Rack::Deflater; so we revert this incompatibility.

This effectively reverts r65922; but keeps the bugfixes to
better support non-blocking sockets and pipes for future use.

[Bug #15356] [Bug #14968]

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@66093 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

History

#1 - 08/06/2018 05:42 AM - normalperson (Eric Wong)

> Bug #14968: [PATCH] io.c: make all pipes nonblocking by default
> https://bugs.ruby-lang.org/issues/14968

Updated patch on top of r64203 since I restored timer-thread :<

https://80x24.org/spew/20180806053511.2421-1-e@80x24.org/raw

#2 - 08/06/2018 09:22 PM - normalperson (Eric Wong)

> > https://bugs.ruby-lang.org/issues/14968
>
> Updated patch on top of r64203 since I restored timer-thread :<
>
> https://80x24.org/spew/20180806053511.2421-1-e@80x24.org/raw

Greg, btw, does the above patch work on Windows? In fact, it
seems possible to implement io/nonblock for Windows if
the non-blocking flag can be cleared successfully on sockets and
pipes. I don't know why nobody did it before, though.

#3 - 08/06/2018 10:04 PM - MSP-Greg (Greg L)

@normalperson (Eric Wong)

The patch was fine, but the Appveyor build stopped.

Earlier, r64206 built fine, and I added the patch, which also built on that.

The build log is at
https://ci.appveyor.com/api/buildjobs/d0s8pbatmb7ax667/log

I can attach it if that would be helpful. Thanks, Greg

#4 - 08/06/2018 11:42 PM - normalperson (Eric Wong)

Greg.mpls@gmail.com wrote:

> The build log is at
> https://ci.appveyor.com/api/buildjobs/d0s8pbatmb7ax667/log
>
> I can attach it if that would be helpful. Thanks, Greg

Thanks, seeing the "ASYNC BUG" and EACCESS was enough.
I removed the error checking (not important in that code
path) and did some minor cleanups:

https://80x24.org/spew/20180806231256.14039-1-e@80x24.org/raw

#5 - 08/07/2018 02:44 AM - MSP-Greg (Greg L)

@normalperson (Eric Wong) Eric,

The zip of all the logs is at

https://ci.appveyor.com/api/buildjobs/pwyhyvkao2nssxy1/artifacts/zlogs_trunk_2018-08-07_64208.7z

All of the test-all failures show something similar to:

```
[Errno::ENOENT] exception expected, not.
Class: <Errno::EBADF>
```

The test-spec failures are similar. Test summary below:

```
13 Total Failures/Errors                          Build No 1032    Job Id pwyhyvkao2nssxy1
   ruby 2.6.0dev (2018-08-07 trunk 64208) [x64-mingw32]
   2018-08-07 02:29:51 UTC

test-all  19301 tests, 2250081 assertions, 5 failures, 0 errors, 107 skips, 107 skips shown

test-spec  3690 files, 28734 examples, 210530 expectations, 2 failures, 4 errors, 0 tagged
mspec      3690 files, 28736 examples, 210420 expectations, 2 failures, 4 errors, 0 tagged

test-basic test succeeded
btest      PASS all 1392 tests
```

#6 - 08/07/2018 04:12 AM - normalperson (Eric Wong)

Greg.mpls@gmail.com wrote:

> All of the test-all failures show something similar to:
>
> ```
> [Errno::ENOENT] exception expected, not.
> Class: <Errno::EBADF>
> ```

Thanks Greg. The following patch (on top of my previous one) might
fix it, but it's a bit dirty. No rush to try it.

```
--- a/process.c
+++ b/process.c
@@ -3454,7 +3454,11 @@ rb_execarg_run_options(const struct rb_execarg *eargp, struct rb_execarg *sargp,
rb_execarg_allocate_dup2_tmpbuf(sargp, RARRAY_LEN(ary));
 }
 }
-    stdfd_clear_nonblock();
+    {
+        int preserve = errno;
+        stdfd_clear_nonblock();
+        errno = preserve;
+    }

 return 0;
 }
```

> 13 Total Failures/Errors Build No 1032 Job Id pwyhyvkao2nssxy1

Anyways, it doesn't seem like the problems from moving from
blocking to non-blocking pipes by default are insurmountable
and it's another step towards getting lightweight concurrency
into Ruby.

#7 - 08/09/2018 08:30 AM - usa (Usaku NAKAMURA)

BTW, Windows does not have nonblocking pipe.
Yes, we may be able to write emulation layer with overlapped I/O functions, but it's very difficult.

#8 - 08/09/2018 08:52 AM - normalperson (Eric Wong)

usa@garbagecollect.jp wrote:

> BTW, Windows does not have nonblocking pipe.

Uh oh, really? I saw rb_w32_set_nonblock() has is_pipe() check
and {Get,Set}NamedPipeHandleState calls, already, so that's
something else?

> Yes, we may be able to write emulation layer with overlapped
> I/O functions, but it's very difficult.

Maybe it's not a big deal to not have non-blocking pipes.

Does win32 have TOCTTOU race in signal handling? All *nix
has this problem, but maybe it's a *nix-only problem.
I see you don't have ubf_list like thread_pthread.c has.

Thanks.

#9 - 08/09/2018 08:57 AM - usa (Usaku NAKAMURA)

> Uh oh, really? I saw rb_w32_set_nonblock() has is_pipe() check
> and {Get,Set}NamedPipeHandleState calls, already, so that's
> something else?

1. A pipe may be a named pipe, or an anonymous pipe :)
2. The code is not tested.
3. Microsoft says that "Note that nonblocking mode ... should not be used to achieve asynchronous input and output (I/O) with named pipes."

> Does win32 have TOCTTOU race in signal handling?

Maybe. But I should note that Windows does not have real signals :P

#10 - 08/10/2018 12:48 AM - shyouhei (Shyouhei Urabe)

I guess you are reinventing libuv?
It claims to support Windows.
I don't say we should use it but it might be worth looking at.

#11 - 08/10/2018 02:12 AM - normalperson (Eric Wong)

shyouhei@ruby-lang.org wrote:

> I guess you are reinventing libuv?
> It claims to support Windows.

I guess usa can take code from that project (MIT license)
for portability.

I'm not sure if non-blocking pipes makes much of a difference
for Windows since it doesn't have real signals; and most
Ruby API are still synchronous to users.

> I don't say we should use it but it might be worth looking at.

Right; event loop design pattern doesn't map to our VM.

#12 - 11/05/2018 11:33 PM - normalperson (Eric Wong)

> https://bugs.ruby-lang.org/issues/14968

Updated patch against r65553 with non-blocking socket support:

https://80x24.org/spew/20181105231324.24080-1-e@80x24.org/raw

Greg: It should be hiding errors for Win32 users, but I can't test.

Parallel MJIT, auto-fiber, UBF in thread_pthread will be much
easier if this change gets accepted.

Basically, auto-fiber/Thread::Light or any similar proposal
would be unusable for real-world apps without this change.
(users would be forced to add "sock.nonblock=true" to everything,
to take advantage of Thread::Light, so it's gross)

#13 - 11/06/2018 12:28 AM - MSP-Greg (Greg L)

@normalperson (Eric Wong) Eric,

The above patch (v5) passed when added to:

ruby 2.6.0dev (2018-11-06 trunk 65555) [x64-mingw32]

It's not uncommon for a few tests to fail during parallel testing, but normally all pass on retry. It occurs when Appveyor is 'busy'...

The following two tests failed in parallel:

TestIO#test_readpartial_lock = 0.26 s = F
TestJIT#test_inlined_undefined_ivar = 1.54 s = F

Thanks, Greg

#14 - 11/06/2018 01:03 AM - normalperson (Eric Wong)

Greg.mpls@gmail.com wrote:

> The above patch (v5) passed when added to:
>
> ruby 2.6.0dev (2018-11-06 trunk 65555) [x64-mingw32]
>
> It's not uncommon for a few tests to fail during parallel testing, but normally all pass on retry. It occurs when Appveyor is 'busy'...
>
> The following two tests failed in parallel:
>
> TestIO#test_readpartial_lock = 0.26 s = F

Thanks for the info!. Do you have any historical info about
the pass/fail rate of these tests in parallel?

Actually, I'm a little surprised test_readpartial_lock passes
on Linux. I think the test will need to be adjusted for
reliability.

> TestJIT#test_inlined_undefined_ivar = 1.54 s = F

No idea about that one. Thanks again.

#15 - 11/06/2018 04:14 AM - MSP-Greg (Greg L)

normalperson (Eric Wong) wrote:

> Do you have any historical info about the pass/fail rate of these tests in parallel?

Yes. Below is a summary of all builds from r64892 2018-10-01 thru r65558, except TestJIT. I assume TestIO & TestThread are the only tests possibly affected by anything you're working on...

Since the patch, TestIO#test_readpartial_lock has failed in parallel testing both times (r65555 & r65558).

```
ruby-loco - Parallel Test Failures - 124 builds

      Last
 Qty   Rev     ---------- Test ----------
  3   65495    TestGemRemoteFetcher#test_do_not_allow_invalid_client_cert_auth_connection
  2   65558    TestIO#test_readpartial_lock
  1   65526    TestIO#test_recycled_fd_close
  1   65424    TestIOWait#test_ready?
  1   65495    TestIOWait#test_wait
  2   65097    TestNetHTTPKeepAlive#test_keep_alive_server_close
  1   64896_b  TestThread#test_priority
  3   65317    TestThread#test_thread_interrupt_for_killed_thread
  2   65417    TestThreadQueue#test_queue_close_multi_multi
```

#16 - 11/08/2018 03:43 AM - normalperson (Eric Wong)

Greg.mpls@gmail.com wrote:

> Yes. Below is a summary of all builds from r64892 2018-10-01 thru r65558, except TestJIT. I assume TestIO & TestThread are the only tests possibly affected by anything you're working on...
>
> Since the patch, TestIO#test_readpartial_lock has failed in parallel testing both times (r65555 & r65558).

Thanks for that info; I don't know how test_readpartial_lock
would've passed on Windows at all, since it seems like IO#nonblock=
isn't available on Windows...

Here's a new version of the patch which adds a guard around
test_readpartial_lock and includes some test-spec fixes:

https://80x24.org/spew/20181108033144.11416-1-e@80x24.org/raw

It goes on top of r65619 (which is necessary for this patch
under rubyspec).

#17 - 11/08/2018 11:04 PM - MSP-Greg (Greg L)

@normalperson (Eric Wong) Eric,

Sorry for the delay. The patch passed. I applied it to r65635...

#18 - 11/22/2018 06:12 AM - ko1 (Koichi Sasada)

Honestly speaking, we are not sure we can introduce it into Ruby 2.6 (incompatibility, stability and so on).

So please commit it and let's try it.
If we find troubles we can't solve, we may revert them.

Thanks,
Koichi

#19 - 11/22/2018 08:47 AM - normalperson (Eric Wong)

- Status changed from Open to Closed

Applied in changeset trunk|r65922.

---

io + socket: make pipes and sockets nonblocking by default

All normal Ruby IO methods (IO#read, IO#gets, IO#write, ...) are
all capable of appearing to be "blocking" when presented with a
file description with the O_NONBLOCK flag set; so there is
little risk of incompatibility within Ruby-using programs.

The biggest compatibility risk is when spawning external
programs. As a result, stdin, stdout, and stderr are now always
made blocking before exec-family calls.

This change will make an event-oriented MJIT usable if it is
waiting on pipes on POSIX_like platforms.

It is ALSO necessary to take advantage of (proposed lightweight
concurrency (aka "auto-Fiber") or any similar proposal for
network concurrency: https://bugs.ruby-lang.org/issues/13618

Named-pipe (FIFO) are NOT yet non-blocking by default since
they are rarely-used and may introduce compatibility problems
and extra syscall overhead for a common path.

Please revert this commit if there are problems and if I am afk
since I am afk a lot, lately.

[ruby-core:89950] [Bug #14968]

#20 - 11/22/2018 09:04 AM - normalperson (Eric Wong)

ko1@atdot.net wrote:

> Honestly speaking, we are not sure we can introduce it into
> Ruby 2.6 (incompatibility, stability and so on).

Understood, but all sockets in 1.8 relied on non-blocking I/O, too.
Most scalability problems are focused on sockets, anyways.

Maybe we can limit the non-blocking to only sockets if
nonblocking pipes become a problem for some external process or
libraries. I also haven't touched FIFO, yet...

> So please commit it and let's try it.
> If we find troubles we can't solve, we may revert them.

OK r65922. Note: I am afk and offline much lately, so feel free
to revert fully or partially (e.g. revert pipe-only).

#21 - 11/22/2018 09:44 AM - k0kubun (Takashi Kokubun)

r65922 made Travis x86_64-darwin build fail https://travis-ci.org/ruby/ruby/jobs/458319172. Could you take a look at it?

```
  1) Failure:
TestSocket_BasicSocket#test_read_write_nonblock [/Users/travis/build/ruby/ruby/test/socket/test_basicsocket.rb
:163]:
Expected #<TCPSocket:fd 13, AF_INET6, ::1, 51219> to be nonblock?.

Finished tests in 292.162445s, 68.6262 tests/s, 7798.3158 assertions/s.
20050 tests, 2278375 assertions, 1 failures, 0 errors, 61 skips
```

#22 - 11/22/2018 10:22 AM - normalperson (Eric Wong)

takashikkbn@gmail.com wrote:

> r65922 made Travis x86_64-darwin build fail
> https://travis-ci.org/ruby/ruby/jobs/458319172. Could you take
> a look at it?

I assume below failure is all that is relevant, maybe r65925 fixes it.

Note: I don't use JavaScript regular basis (crap laptop and my
eyesight sucks too much for mainstream GUIs), so that Travis URL
has no info for me. AFAIK there is a way to get raw log from
Travis.

```
  1) Failure:
TestSocket_BasicSocket#test_read_write_nonblock [/Users/travis/build/ruby/ruby/test/socket/test_basicsocke
t.rb:163]:
Expected #<TCPSocket:fd 13, AF_INET6, ::1, 51219> to be nonblock?.
```

#23 - 11/22/2018 01:07 PM - k0kubun (Takashi Kokubun)

Thanks, that's fixed now.

#24 - 11/22/2018 02:29 PM - k0kubun (Takashi Kokubun)

Well, at the same time you seem to have introduced a bug on Windows. As of r65925 on AppVeyor of both mswin and MinGW, MinGW hangs on test-all's test_drbssl, and mswin hangs on some test (it doesn't produce helpful logs).

You can't see AppVeyor that requires JavaScript, but at least you would be able to see https://rubyci.org/logs/mswinci.japaneast.cloudapp.azure.com/vc12-x64/ruby-trunk/log/20181122T114828Z.fail.html.gz. Never mind about JIT's permission denied (known issue), but most of others seem to be related to your change.

Also in a later build (r65928 which looks irrelevant), Travis had another test failure on x86_64-darwin:

```
S.S....SFLeaked thread: TestIO#test_recycled_fd_close: #<Thread:0x00007f892c9c5c90@/Users/travis/build/ruby/ru
by/test/ruby/test_io.rb:3777 run>
#<Thread:0x00007f892c9c5c90@/Users/travis/build/ruby/ruby/test/ruby/test_io.rb:3777 run> terminated with excep
tion (report_on_exception is true):
Traceback (most recent call last):
 1: from /Users/travis/build/ruby/ruby/test/ruby/test_io.rb:3778:in `block (2 levels) in test_recycled_fd_clos
e'
```

```
/Users/travis/build/ruby/ruby/test/ruby/test_io.rb:3778:in `wait_readable': closed stream (IOError)
Finished thread: TestIO#test_reinitialize: #<Thread:0x00007f892c9c5c90@/Users/travis/build/ruby/ruby/test/ruby
/test_io.rb:3777 dead>
..S..S.S
Retrying...
#<Thread:0x00007fa31c1f7388@/Users/travis/build/ruby/ruby/test/ruby/test_io.rb:3777 run> terminated with excep
tion (report_on_exception is true):
Traceback (most recent call last):
 1: from /Users/travis/build/ruby/ruby/test/ruby/test_io.rb:3778:in `block (2 levels) in test_recycled_fd_clos
e'
/Users/travis/build/ruby/ruby/test/ruby/test_io.rb:3778:in `wait_readable': stream closed in another thread (I
OError)
  1) Failure:
TestIO#test_recycled_fd_close [/Users/travis/build/ruby/ruby/test/ruby/test_io.rb:3814]:
Expected /stream closed/ to match "closed stream".
```

#25 - 11/22/2018 03:15 PM - MSP-Greg (Greg L)

ruby-loco built r65909 green, and failed on r65922 and r65927.

Using the r65927 build locally, test_drb.rb freezes on various tests intermittently:

```
DRbTests::TestDRbAry#test_07_break_18
DRbTests::TestDRbCore#test_07_private_missing
DRbTests::TestDRbYield#test_01_one
```

EDIT: I was mistaken. test_drbssl.rb intermittently fails also. The one test I've seen fail is:
DRbTests::TestDRbSSLCore#test_07_send_missing

I'm trying to narrow the issue, but no luck yet... Greg

#26 - 11/22/2018 03:39 PM - MSP-Greg (Greg L)

Further testing (net, openssl, & socket):

```
  1) Error:
TestSocket#test_udp_recv_truncation:
Errno::EMSGSIZE: A message sent on a datagram socket was larger than the internal message buffer or some other
 network limit, or the buffer used to receive a datagram into was smaller than the datagram itself. - recvfrom
(2)
    C:/Greg/GitHub/ruby/test/socket/test_socket.rb:712:in `recv'
    C:/Greg/GitHub/ruby/test/socket/test_socket.rb:712:in `test_udp_recv_truncation'

  2) Error:
TestSocket#test_udp_recvmsg_truncation:
Errno::EMSGSIZE: A message sent on a datagram socket was larger than the internal message buffer or some other
 network limit, or the buffer used to receive a datagram into was smaller than the datagram itself. - recvmsg(
2)
    C:/Greg/Ruby99-x64_bad_27/lib/ruby/2.6.0/socket.rb:427:in `__recvmsg'
    C:/Greg/Ruby99-x64_bad_27/lib/ruby/2.6.0/socket.rb:427:in `recvmsg'
    C:/Greg/GitHub/ruby/test/socket/test_socket.rb:728:in `test_udp_recvmsg_truncation'

  3) Error:
TestSocket#test_udp_read_truncation:
Errno::EMSGSIZE: A message sent on a datagram socket was larger than the internal message buffer or some other
 network limit, or the buffer used to receive a datagram into was smaller than the datagram itself.
    C:/Greg/GitHub/ruby/test/socket/test_socket.rb:698:in `read'
    C:/Greg/GitHub/ruby/test/socket/test_socket.rb:698:in `test_udp_read_truncation'
```

Also, some of the TestNetHTTPS tests timeout.

All the above tests passed on r65909. Thanks, Greg

#27 - 11/22/2018 08:12 PM - normalperson (Eric Wong)

> https://bugs.ruby-lang.org/issues/14968#change-75083

Thanks for the reports, I'm trying r65929 to disable non-blocking-by-default
on Win32. However, this change may expose existing bugs:

> Also, some of the TestNetHTTPS tests timeout.

Maybe there is some deeper problem, because net/http and
net/protocol already relies heavily on nonblocking sockets...

#28 - 11/22/2018 08:32 PM - normalperson (Eric Wong)

> https://bugs.ruby-lang.org/issues/14968#change-75083
>
> Thanks for the reports, I'm trying r65929 to disable non-blocking-by-default
> on Win32. However, this change may expose existing bugs:

Greg: can you try reverting r65929 and applying the following
patch to check RB_WAITFD_PRI?

https://80x24.org/spew/20181122202628.2468-1-e@80x24.org/raw

I guess today is a holiday where you are, so no rush

Thanks.

#29 - 11/22/2018 10:59 PM - MSP-Greg (Greg L)

> holiday where you are

Yes, but I'm more or less not afk now. JFYI, unless I'm 'pinged', I don't receive messages via email even if I 'watch' the thread, which is a change from a few months ago...

I just built from r65930, and it passed, ruby Appveyor CI failed, but it was due to r65926. So r65929 seems ok.

Anyway, re the above msg, I'll check the patch.

#30 - 11/23/2018 12:41 AM - MSP-Greg (Greg L)

- File 14968_28_failures_errors.log added

@normalperson (Eric Wong) Eric,

Well, Appveyor testing can be intermittent. After r65931, the following issues in ruby Appveyor CI:

```
mswqin vc140
  1) Failure:
TestIO#test_readpartial_lock [C:/projects/ruby/test/ruby/test_io.rb:1366]:
RuntimeError expected but nothing was raised.


mingw
Retrying...
#<Thread:0x0000000005086a38@C:/projects/ruby/test/ruby/test_io.rb:1364 run> terminated with exception (report_
on_exception is true):
C:/projects/ruby/test/ruby/test_io.rb:1364:in `readpartial': stream closed in another thread (IOError)
 from C:/projects/ruby/test/ruby/test_io.rb:1364:in `block (2 levels) in test_readpartial_lock'

  1) Failure:
TestIO#test_readpartial_lock [C:/projects/ruby/test/ruby/test_io.rb:1366]:
RuntimeError expected but nothing was raised.
```

I've attached the test info from the ruby-loco build based on msg #28.

Thanks, Greg

#31 - 11/23/2018 04:39 AM - MSP-Greg (Greg L)

@normalperson (Eric Wong) Eric,

After a few more builds, the intermittent test failures seem to be:
TestIO#test_readpartial_lock
TestIO#test_recycled_fd_close

These often fail during parallel, occasionally fail during retry.

I've noticed that the bootstraptest suite seems to take about twice as long as before...

Thanks, Greg

#32 - 11/24/2018 08:52 AM - normalperson (Eric Wong)

Greg.mpls@gmail.com wrote:

> @normalperson (Eric Wong) Eric,
>
> After a few more builds, the intermittent test failures seem to be:
> TestIO#test_readpartial_lock

r65948 might help with that one (does IO#nonblock= work for you?)

> TestIO#test_recycled_fd_close

r65939, r65940 seem to have fixed the alerts I was getting on
this from Linux CI machines.

Fwiw, I consider r65931, r65937, r65939, r65940 and r65948 worthwhile
bugfixes and improvements even if r65922 and r65925 to change the
default gets reverted.

> These often fail during parallel, occasionally fail during retry.
>
> I've noticed that the bootstraptest suite seems to take about
> twice as long as before...

Do you have any more details on the slowdown (e.g. profiling info?)

Maybe r65948 can help since it reduces GVL activity.

Otherwise, I think we can accept win32 and POSIX-like platforms
are too different and deserve different defaults
(Same as r65929 for socket):

https://80x24.org/spew/20181124083700.25685-1-e@80x24.org/raw

#33 - 11/24/2018 05:11 PM - MSP-Greg (Greg L)

@normalperson (Eric Wong) Eric,

> Do you have any more details on the slowdown (e.g. profiling info?)

I pointed it out thinking it might help with the issues. I concentrated more on determining what's failing in test-all.

> Otherwise, I think we can accept win32 and POSIX-like platforms are too different and deserve different defaults
> (Same as r65929 for socket):

That may be the outcome. You've taken a lot of time with all the code & testing you've done, so thank you.

JFYI, there are currently several tests in test_io.rb & test_readpartial.rb that are freezing. Summary below:

```
test-all failure, errors, & stops

ruby 2.6.0dev (2018-11-24 trunk 65950) [x64-mingw32]

The following tests froze

test/ruby/test_io.rb                TestIO
S  test_cross_thread_close_fd
S  test_read_buffer_error
S  test_read_unlocktmp_ensure
S  test_readpartial_buffer_error
S  test_readpartial_locktmp
S  test_readpartial_unlocktmp_ensure
```

```
S  test_recycled_fd_close

test/ruby/test_readpartial.rb      TestReadPartial
S  test_open_pipe
S  test_with_stdio

After commenting out the above tests, here are the errors in three test files.
I believe the ruby test-all folder passes otherwise.

  1) Failure:
TestThreadQueue#test_thr_kill [ruby/test/ruby2/test_thread_queue.rb:153]:
only 0/250 done in 60 seconds.

  2) Error:
TestIO#test_race_closed_stream:
Timeout::Error: execution of assert_separately expired timeout (10 sec)
pid 8996 exit 0
|

    ruby/test/ruby2/test_io.rb:3591:in `test_race_closed_stream'

  3) Error:
TestIO#test_cross_thread_close_stdio:
Timeout::Error: execution of assert_separately expired timeout (10 sec)
pid 6044 exit 0
|

    ruby/test/ruby2/test_io.rb:2946:in `test_cross_thread_close_stdio'

  4) Error:
TestIO#test_closed_stream_in_rescue:
Timeout::Error: execution of assert_separately expired timeout (10 sec)
pid 7816 exit 0
|

    ruby/test/ruby2/test_io.rb:3685:in `test_closed_stream_in_rescue'
```

#34 - 11/24/2018 06:02 PM - MSP-Greg (Greg L)

@normalperson (Eric Wong) Eric,

Using ruby 2.6.0dev (2018-11-24 trunk 65959) [x64-mingw32], applied the patch in msg 32, passed all tests.

bootstraptest suite is back to mid 40's, it had jumped to mid to high 80's.

Thanks, Greg

#35 - 11/24/2018 08:12 PM - normalperson (Eric Wong)

Greg.mpls@gmail.com wrote:

> Using ruby 2.6.0dev (2018-11-24 trunk 65959) [x64-mingw32], applied the patch in msg 32, passed all tests.

Thanks, committed as r65962

> bootstraptest suite is back to mid 40's, it had jumped to mid to high 80's.

I guess select() overhead is really high on Windows?
(it's not great on Linux, either, but we can use poll()/ppoll())

#36 - 11/29/2018 09:22 PM - normalperson (Eric Wong)

ko1@atdot.net wrote:

> Honestly speaking, we are not sure we can introduce it into
> Ruby 2.6 (incompatibility, stability and so on).

Doesn't look like we can.

> So please commit it and let's try it.
> If we find troubles we can't solve, we may revert them.

Reverted in r66093 because of [Bug #15356]

However, I kept the bugfixes and some infrastructure
changes in this patch. So maybe in the future we can
enable nonblock on a per-EC basis.

#37 - 12/07/2018 07:22 AM - normalperson (Eric Wong)

> https://bugs.ruby-lang.org/issues/14968

So I'm hoping this can be revisited in the 2.7 timeframe even
if I'm not around.

IMHO it's too late for 2.6, now, but [Bug #15356] was resolved
and it doesn't seem like this had anything to do with it after
all.

#38 - 06/20/2019 11:09 PM - ioquatix (Samuel Williams)

Eric do you mind explaining where we got to with this and how to move forward?

#39 - 06/24/2019 01:53 AM - normalperson (Eric Wong)

samuel@oriontransfer.net wrote:

> Eric do you mind explaining where we got to with this and how
> to move forward?

I don't think this patch is doable given potential incompatibilities
(and I fixed the problem with timer-thread before 2.6 release).

AFAIK there's also a major performance regression on Windows.

I suggest supporting "nonblock: true" keyword:

```
IO.pipe(nonblock: true)
IO.popen(..., nonblock: true)
```

> https://bugs.ruby-lang.org/issues/14968#change-78763

I don't have much time to pay attention to Ruby anymore, but
feel free to Cc: me.

#40 - 12/06/2020 06:38 AM - ioquatix (Samuel Williams)

Eric, we implemented the vast majority of this PR but our current implementation makes all I/O non-blocking by default. You laid the ground work for this and made my job much easier. Given that Windows has limited support/performance issues, most I/O is not non-blocking by default on this platform.

Files

0001-io.c-make-all-pipes-nonblocking-by-default.patch	12.1 KB	08/06/2018	normalperson (Eric Wong)
14968_28_failures_errors.log	23.5 KB	11/23/2018	MSP-Greg (Greg L)