Ruby master - Bug #4558

TestSocket#test_closed_read fails after r31230

04/07/2011 12:53 AM - nagachika (Tomoyuki Chikanaga)

Status:	Closed		
Priority:	Normal		
Assignee:	kosaki (Motohiro KOSAKI)		
Target version:	1.9.3		
ruby -v:	-	**Backport:**	

Description

=begin
After r31230, make test-all reports a failure in test_socket.rb @Mac OS X 10.6.6

```
% make test-all TESTS=../ruby/test/socket/test_socket.rb
./miniruby -I../ruby/lib -I. -I.ext/common  ../ruby/tool/runruby.rb --extout=.ext  -- "../ruby/test/runner.rb" --ruby="./miniruby -I../ruby/lib -I. -I.ext/common  ../ruby/tool/runruby.rb --extout=.ext  --" ../ruby/test/socket/test_socket.rb
Run options: "--ruby=./miniruby -I../ruby/lib -I. -I.ext/common  ../ruby/tool/runruby.rb --extout=.ext  --"

# Running tests:

....F................

Finished tests in 1.611761s, 13.6497 tests/s, 40.9490 assertions/s.

1) Failure:
test_closed_read(TestSocket) [/ruby/test/socket/test_socket.rb:428]:
[ruby-core:35203]
[IOError] exception expected, not
Class: Errno::EBADF
Message: <"Bad file descriptor">
---Backtrace---
/ruby/test/socket/test_socket.rb:422:in readline'
/ruby/test/socket/test_socket.rb:422:inblock in test_closed_read'
```

```
22 tests, 66 assertions, 1 failures, 0 errors, 0 skips
make: *** [yes-test-all] Error 1
```

=end

Related issues:

Related to Ruby master - Bug #4527: [PATCH] IO#close releases GVL if possible	**Closed**	**03/26/2011**
Related to Ruby master - Feature #4570: [PATCH v2] io.c (rb_io_close): releas...	**Closed**	**04/12/2011**
Related to Ruby master - Bug #4390: TCPSocket#readline doesn't raise if the s...	**Closed**	**02/12/2011**

History

#1 - 04/07/2011 10:23 PM - kosaki (Motohiro KOSAKI)

- ruby -v changed from ruby 1.9.3dev (2011-04-05 trunk 31241) [x86_64-darwin10.6.0] to -

=begin

Bug #4558: TestSocket#test_closed_read fails after r31230
http://redmine.ruby-lang.org/issues/4558

I think current rb_io_close() is broken. We have to call rb_thread_fd_close()
before releasing GVL.

Eric, Am I missing something?
=end

#2 - 04/08/2011 04:23 AM - normalperson (Eric Wong)

=begin
KOSAKI Motohiro kosaki.motohiro@gmail.com wrote:

> ---
>
> > Bug #4558: TestSocket#test_closed_read fails after r31230
> > http://redmine.ruby-lang.org/issues/4558
>
> I think current rb_io_close() is broken. We have to call rb_thread_fd_close()
> before releasing GVL.
>
> Eric, Am I missing something?

You are correct.

I can't reproduce the test failure on x86_64-linux but the
following patch should fix a race condition:

diff --git a/io.c b/io.c
index 7ce7148..b79cc5e 100644
--- a/io.c
+++ b/io.c
@@ -3685,8 +3685,8 @@ rb_io_close(VALUE io)
if (fptr->fd < 0) return Qnil;

```
  fd = fptr->fd;
```

- rb_io_fptr_cleanup(fptr, FALSE);   rb_thread_fd_close(fd);

- rb_io_fptr_cleanup(fptr, FALSE);

  if (fptr->pid) {
  rb_syswait(fptr->pid);

--
Eric Wong
=end

#3 - 04/09/2011 01:23 AM - kosaki (Motohiro KOSAKI)

=begin
2011/4/8 Eric Wong normalperson@yhbt.net:

> KOSAKI Motohiro kosaki.motohiro@gmail.com wrote:
>
> > ---
> >
> > > Bug #4558: TestSocket#test_closed_read fails after r31230
> > > http://redmine.ruby-lang.org/issues/4558
> >
> > I think current rb_io_close() is broken. We have to call rb_thread_fd_close()
> > before releasing GVL.
> >
> > Eric, Am I missing something?
>
> You are correct.
>
> I can't reproduce the test failure on x86_64-linux but the
> following patch should fix a race condition:
>
> diff --git a/io.c b/io.c
> index 7ce7148..b79cc5e 100644
> --- a/io.c
> +++ b/io.c
> @@ -3685,8 +3685,8 @@ rb_io_close(VALUE io)
> Â Â if (fptr->fd < 0) return Qnil;
>
> Â Â fd
> =end

#4 - 04/09/2011 09:15 AM - nagachika (Tomoyuki Chikanaga)

- Target version set to 1.9.3

- ruby -v changed from - to ruby 1.9.3dev (2011-04-05 trunk 31241) [x86_64-darwin10.6.0]

=begin
Hi,

I applied Eric's patch, but TestSocket#test_closed_read still report same failure.

I can reproduce EBADF with following script.

r, w = IO.pipe
read_thread = Thread.new{ r.read(1) }
sleep(0.1) until read_thread.stop?
r.close
read_thread.join

=end

#5 - 04/09/2011 12:23 PM - normalperson (Eric Wong)

=begin
Tomoyuki Chikanaga redmine@ruby-lang.org wrote:

> ruby -v changed from - to ruby 1.9.3dev (2011-04-05 trunk 31241) [x86_64-darwin10.6.0]
>
> Hi,
>
> I applied Eric's patch, but TestSocket#test_closed_read still report same failure.
>
> I can reproduce EBADF with following script.

Thanks for testing, think I have a better fix below (supercedes my
original fix)

Also pushed to the "io-close-fixes" branch of git://bogomips.org/ruby.git

From d5f9659ea9c2e8e0ed67544ed35afef4ca2bb3c5 Mon Sep 17 00:00:00 2001
From: Eric Wong normalperson@yhbt.net
Date: Thu, 7 Apr 2011 19:25:20 +0000
Subject: [PATCH] io.c (rb_io_close): ensure IOError for cross-thread closes

We need to inform threads to stop operations on the FD before
closing it and also invalidate the fd member of the rb_io_t
struct for other threads to properly raise IOError.

FDs may be created and destroyed without the GVL, so
rb_thread_fd_close() may be improperly hitting the wrong
threads/FDs if we close() before notifying and in the worst case
or threads will end up reading/writing to an unexpected FD.

ref: [ruby-core:35631]
ref: http://redmine.ruby-lang.org/issues/4558

---

io.c | 25 ++++++++++++++++++-------
test/ruby/test_io.rb | 39 +++++++++++++++++++++++++++++++++++++++
2 files changed, 57 insertions(+), 7 deletions(-)

diff --git a/io.c b/io.c
index 7ce7148..5d37b7f 100644
--- a/io.c
+++ b/io.c
@@ -3504,6 +3504,7 @@ maygvl_close(int fd, int keepgvl)
if (keepgvl)
return close(fd);

- rb_thread_fd_close(fd);
  /*

- close() may block for certain file types (NFS, SO_LINGER sockets,
- inotify), so let other threads run. @@ -3525,6 +3526,8 @@ maygvl_fclose(FILE *file, int keepgvl) if (keepgvl) return fclose(file);

- rb_thread_fd_close(fileno(file));
  +
  return (int)rb_thread_blocking_region(nogvl_fclose, file, RUBY_UBF_IO, 0);
  }

@@ -3555,24 +3558,35 @@ fptr_finalize(rb_io_t *fptr, int noraise)
}
}
if (IS_PREP_STDIO(fptr) || fptr->fd <= 2) {

- int fd = fptr->fd; +
- fptr->stdio_file = 0;
- fptr->fd = -1;
- rb_thread_fd_close(fd);   goto skip_fd_close;   }   if (fptr->stdio_file) {
- FILE *stdio_file = fptr->stdio_file; +
- fptr->stdio_file = 0;
- fptr->fd = -1; +   /* fptr->stdio_file is deallocated anyway   * even if fclose failed. */
- if ((maygvl_fclose(fptr->stdio_file, noraise) < 0) && NIL_P(err))
- if ((maygvl_fclose(stdio_file, noraise) < 0) && NIL_P(err))   err = noraise ? Qtrue : INT2NUM(errno);   }   else if (0 <= fptr->fd) {
- int fd = fptr->fd;
- fptr->fd = -1; +   /* fptr->fd may be closed even if close fails.   * POSIX doesn't specify it.   * We assumes it is closed. */
- if ((maygvl_close(fptr->fd, noraise) < 0) && NIL_P(err))
- if ((maygvl_close(fd, noraise) < 0) && NIL_P(err))   err = noraise ? Qtrue : INT2NUM(errno);   } skip_fd_close:
- fptr->fd = -1;

- fptr->stdio_file = 0;
  fptr->mode &= ~(FMODE_READABLE|FMODE_WRITABLE);

  if (!NIL_P(err) && !noraise) {
  @@ -3668,7 +3682,6 @@ VALUE
  rb_io_close(VALUE io)
  {
  rb_io_t *fptr;

- int fd;
  VALUE write_io;
  rb_io_t *write_fptr;

@@ -3684,9 +3697,7 @@ rb_io_close(VALUE io)
if (!fptr) return Qnil;
if (fptr->fd < 0) return Qnil;

- fd = fptr->fd;   rb_io_fptr_cleanup(fptr, FALSE);

- rb_thread_fd_close(fd);

  if (fptr->pid) {
  rb_syswait(fptr->pid);
  diff --git a/test/ruby/test_io.rb b/test/ruby/test_io.rb
  index 6b8e6b5..3d086b3 100644
  --- a/test/ruby/test_io.rb
  +++ b/test/ruby/test_io.rb
  @@ -1809,4 +1809,43 @@ End
  Process.waitpid2(pid)
  end
  end
  +

- def test_cross_thread_close_fd

- with_pipe do |r,w|

- [redacted]

- [redacted]

- [redacted]

- [redacted]
- [redacted]
- [redacted]
- [redacted]

  +
- [redacted]
- [redacted]
- [redacted]
- [redacted]
- end
- end
  +
- def test_cross_thread_close_stdio
- with_pipe do |r,w|
- [redacted]
- [redacted]
- [redacted]
- [redacted]
- [redacted]
- [redacted]
- [redacted]
- [redacted]
- [redacted]
- [redacted]
- [redacted]
- [redacted]
- [redacted]
- [redacted]
- [redacted]
- [redacted]
- end
- rescue NotImplementedError
- end

  ## end

  Eric Wong
  =end

#6 - 04/09/2011 12:23 PM - normalperson (Eric Wong)

=begin

KOSAKI Motohiro kosaki.motohiro@gmail.com wrote:

> After while thinking, I conclude I was wrong. If rb_io_fptr_cleanup()
> raise a exception, We don't have to kill other threads. So, now I'm
> incline to revert r31230. Hmm...

I think we can still fix the issues revolving around r31230. We've
already found at least two (this and #4555) bugs. As for #4555
(connect() failing on EINTR) it was a bug anyways, but just very hard to
expose before.

--
Eric Wong
=end

#7 - 04/09/2011 12:29 PM - normalperson (Eric Wong)

=begin
Eric Wong normalperson@yhbt.net wrote:

> Thanks for testing, think I have a better fix below (supercedes my
> original fix)
>
> Also pushed to the "io-close-fixes" branch of git://bogomips.org/ruby.git

Oops, I broke test/ruby/test_thread.rb in the atexit handlers :x
I squashed the following change and forcibly repushed:

diff --git a/io.c b/io.c
index 5d37b7f..65e7693 100644
--- a/io.c
+++ b/io.c
@@ -3562,7 +3562,8 @@ fptr_finalize(rb_io_t *fptr, int noraise)

```
    fptr->stdio_file = 0;
    fptr->fd = -1;
```

- rb_thread_fd_close(fd);
- if (!noraise)
- rb_thread_fd_close(fd);   goto skip_fd_close;   }   if (fptr->stdio_file) { -- Eric Wong =end

#8 - 04/09/2011 06:14 PM - nagachika (Tomoyuki Chikanaga)

=begin
Hi Eric,

These patches seem good and after applying them, make test-all passes all tests except a test for parallel_test in my environment. thanks!
=end

#9 - 04/10/2011 04:23 PM - normalperson (Eric Wong)

=begin
Tomoyuki Chikanaga redmine@ruby-lang.org wrote:

> These patches seem good and after applying them, make test-all passes
> all tests except a test for parallel_test in my environment. thanks!

Thanks for reporting and helping us track it down.
Can somebody please commit my patches?

--
Eric Wong
=end

#10 - 04/11/2011 09:53 PM - kosaki (Motohiro KOSAKI)

- Status changed from Open to Closed

=begin
r31230 was revered by r31261.

=end

#11 - 04/11/2011 10:23 PM - kosaki (Motohiro KOSAKI)

- ruby -v changed from ruby 1.9.3dev (2011-04-05 trunk 31241) [x86_64-darwin10.6.0] to -

=begin

Subject: [PATCH] io.c (rb_io_close): ensure IOError for cross-thread closes

We need to inform threads to stop operations on the FD before
closing it and also invalidate the fd member of the rb_io_t
struct for other threads to properly raise IOError.

FDs may be created and destroyed without the GVL, so
rb_thread_fd_close() may be improperly hitting the wrong
threads/FDs if we close() before notifying and in the worst case
or threads will end up reading/writing to an unexpected FD.

ref: [ruby-core:35631]

ref: http://redmine.ruby-lang.org/issues/4558

Â io.c Â Â Â Â Â Â Â Â Â | Â 25 +++++++++++++++++++-------
Â test/ruby/test_io.rb | Â 39 +++
Â 2 files changed, 57 insertions(+), 7 deletions(-)

diff --git a/io.c b/io.c
index 7ce7148..5d37b7f 100644
--- a/io.c
+++ b/io.c
@@ -3504,6 +3504,7 @@ maygvl_close(int fd, int keepgvl)
Â Â if (keepgvl)
Â Â Â Â return close(fd);

- Â Â rb_thread_fd_close(fd);
  Â Â /*
  Â Â Â * close() may block for certain file types (NFS, SO_LINGER sockets,
  Â Â Â * inotify), so let other threads run.
  @@ -3525,6 +3526,8 @@ maygvl_fclose(FILE *file, int keepgvl)
  Â Â if (keepgvl)
  Â Â Â Â return fclose(file);

- Â Â rb_thread_fd_close(fileno(file));
  +
  Â Â return (int)rb_thread_blocking_region(nogvl_fclose, file, RUBY_UBF_IO, 0);
  Â }

@@ -3555,24 +3558,35 @@ fptr_finalize(rb_io_t *fptr, int noraise)
Â Â Â Â }
Â Â }
Â Â if (IS_PREP_STDIO(fptr) || fptr->fd <
=end

#12 - 04/12/2011 10:55 AM - usa (Usaku NAKAMURA)

- Category changed from core to test

- Status changed from Closed to Assigned

- Assignee set to kosaki (Motohiro KOSAKI)

=begin
I have no opinion about this topic, but the test code which was checked in at r31260 by kosaki-san is platform dependent.
It blocks on Windows, and stops all tests.

I request to revert it.
Or, please explain grounds from which this test should be accepted as behavior of ruby.

=end

#13 - 04/12/2011 11:29 AM - normalperson (Eric Wong)

=begin

I consider either Errno::EBADF or IOError to be acceptable.

The main thing I care about is I/O for pipes/sockets being interruptable
(I only work on *nix).

By the way, test/socket/test_socket.rb has had a similar test for months
(since r30852).   It does have a Timeout wrapping it, so maybe that is
needed (but you'd still get an error). Maybe just disabling this test
for Windows platforms would be acceptable?

Eventually I would like to get rid of places where we call select()
before doing (any) I/O across the board (ref: [ruby-core:35586]) if
possible.

=end

#14 - 04/12/2011 10:23 PM - kosaki (Motohiro KOSAKI)

=begin
2011/4/12 redmine@ruby-lang.org:

> Issue #4558 has been updated by Usaku NAKAMURA.
>
> Category changed from core to test
> Status changed from Closed to Assigned
> Assignee set to Motohiro KOSAKI
>
> I have no opinion about this topic, but the test code which was checked in at r31260 by kosaki-san is platform dependent.
> It blocks on Windows, and stops all tests.
>
> I request to revert it.
> Or, please explain grounds from which this test should be accepted as behavior of ruby.

I succuseeded to reporoduce this issue. On win32, IO.close() cause hang-up.
So, I think we have to discuss two thing.
1) Why close() makes hang-up? Is it acceptable behavior?
2) At [ruby-core:35203], We decided IO.close() raise exception to
othread threads
and then they should wake up as ruby-1.8.
Should we think win32 is exception for this rule?

In the other words, We have to decided rb_thread_io_blocking_region()
spec. Otherwise
we can't make any testcase. ;-)
=end

#15 - 04/12/2011 10:23 PM - kosaki (Motohiro KOSAKI)

=begin

> Issue #4558 has been updated by Eric Wong.
>
> I consider either Errno::EBADF or IOError to be acceptable.

Hmm...
I can't agree this. If EBADF can be observed, we can observe completely
unrelated file when a fd number was recycled just after close.

> The main thing I care about is I/O for pipes/sockets being interruptable
> (I only work on *nix).
>
> By the way, test/socket/test_socket.rb has had a similar test for months
> (since r30852). Â It does have a Timeout wrapping it, so maybe that is
> needed (but you'd still get an error). Â Maybe just disabling this test
> for Windows platforms would be acceptable?

Hmm...
If windows can't implement close() case, I doubt r30852 is correct fix.
Is this really worth that *nix and windows have different spec?

> Eventually I would like to get rid of places where we call select()
> before doing (any) I/O across the board (ref: [ruby-core:35586]) if

> possible.

this makes sense to me.
=end

#16 - 04/13/2011 01:29 PM - usa (Usaku NAKAMURA)

=begin
Hello,

In message "[ruby-core:35725] Re: [Ruby 1.9 - Bug #4558][Assigned] TestSocket#test_closed_read fails after r31230"
on Apr.12,2011 21:31:46, kosaki.motohiro@gmail.com wrote:

>> Or, please explain grounds from which this test should be accepted as behavior of ruby.
>
> I succuseeded to reporoduce this issue. On win32, IO.close() cause hang-up.
> So, I think we have to discuss two thing.
> 1) Why close() makes hang-up? Is it acceptable behavior?

MSVCRT's fds have their own locks.
MSVCRT locks fds when accessing them -- reading, writing,
closing, etc.
The author of MSVCRT obviously intended the behavior, I think.

> 2) At [ruby-core:35203], We decided IO.close() raise exception to
> othread threads
> and then they should wake up as ruby-1.8.
> Should we think win32 is exception for this rule?

I see. Hmm...

Is the behavior that close() doesn't block and the I/O operations
of other threads are interrupted defind by posix or some specifications?
We found this problem in Windows this time, but might there be
other platforms which have same problem?

Regards,
--
U.Nakamura usa@garbagecollect.jp
=end

#17 - 04/13/2011 09:23 PM - kosaki (Motohiro KOSAKI)

=begin
Hi

2011/4/13 U.Nakamura usa@garbagecollect.jp:

> Hello,
>
> In message "[ruby-core:35725] Re: [Ruby 1.9 - Bug #4558][Assigned] TestSocket#test_closed_read fails after r31230"
> Â Â on Apr.12,2011 21:31:46, kosaki.motohiro@gmail.com wrote:
>
>>> Or, please explain grounds from which this test should be accepted as behavior of ruby.
>>
>> I succuseeded to reporoduce this issue. On win32, IO.close() cause hang-up.
>> So, I think we have to discuss two thing.
>> Â 1) Why close() makes hang-up? Is it acceptable behavior?
>
> MSVCRT's fds have their own locks.
> MSVCRT locks fds when accessing them -- reading, writing,
> closing, etc.
> The author of MSVCRT obviously intended the behavior, I think.

ok, I see.

>> Â 2) At [ruby-core:35203], We decided IO.close() raise exception to
>> othread threads

Â Â Â and then they should wake up as ruby-1.8.
Â Â Â Should we think win32 is exception for this rule?

I see. Â Hmm...

Is the behavior that close() doesn't block and the I/O operations
of other threads are interrupted defind by posix or some specifications?

No. It's purely implementation defined.

We found this problem in Windows this time, but might there be
other platforms which have same problem?

It's possible.
So, now I'm incline to revert r30852.

nobu, What do you think?
=end

#18 - 04/14/2011 05:23 AM - normalperson (Eric Wong)

=begin
KOSAKI Motohiro kosaki.motohiro@gmail.com wrote:

Issue #4558 has been updated by Eric Wong.

I consider either Errno::EBADF or IOError to be acceptable.

Hmm...
I can't agree this. If EBADF can be observed, we can observe completely
unrelated file when a fd number was recycled just after close.

Actually, I expect EBADF make any read/write-retry loop stop
immediately, but yes, exposing IOError to user is better.

Hmm...
If windows can't implement close() case, I doubt r30852 is correct fix.
Is this really worth that *nix and windows have different spec?

If r30852 is reverted, Linux (and maybe other *nix) will still break
threads out of blocking read/write with EBADF and
rb_io_wait_*able(fptr->fd) will raise IOError as long as we
assign fptr->fd = -1 before the actual close() call in IO#close.

Maybe Windows (and possibly other OS) will be forced to call do_select()
to implement behavior consistent with Linux.

On a related note, r30852 also has the problem with making IO#close an
O(n) operation since it needs to scan through all threads (and I'd like
to run thousands of threads :>).

--
Eric Wong
=end

#19 - 04/15/2011 06:23 AM - normalperson (Eric Wong)

=begin
KOSAKI Motohiro kosaki.motohiro@gmail.com wrote:

KOSAKI Motohiro kosaki.motohiro@gmail.com wrote:

Issue #4558 has been updated by Eric Wong.
following scenario should be happen too.

CPU1 CPU2 CPU3

open() -> 5
close(5)
open() -> 5
read(5) -> success, but read different data.

OK, I'm starting to think there's no safe way to handle these
situations, especially with MVM on the horizon. Just telling users to
stop doing close() in a different thread is probably the best way to
go...

Hmm...
If windows can't implement close() case, I doubt r30852 is correct fix.
Is this really worth that *nix and windows have different spec?

If r30852 is reverted, Linux (and maybe other *nix) will still break
threads out of blocking read/write with EBADF and
rb_io_wait_*able(fptr->fd) will raise IOError as long as we
assign fptr->fd = -1 before the actual close() call in IO#close.

Maybe Windows (and possibly other OS) will be forced to call do_select()
to implement behavior consistent with Linux.

???
I'm sorry, I haven't catch your point.
Which issue is solved by calling do_select()?

It can reduce the likelyhood of read() being uninterruptable since
do_select() will sleep in 100ms intervals to check for interrupts on
win. There's still a small chance of blocking read() since select()
can have false positives and other threads could've drained the data...

On a related note, r30852 also has the problem with making IO#close an
O(n) operation since it needs to scan through all threads (and I'd like
to run thousands of threads :>).

I have no opinion. I like faster software, but I haven't seen close
makes performance bottleneck.

Contrived test case, but it gets worse as nr_thread increases:

```
# ruby 1.9.2p180 (2011-02-18 revision 30909) [x86_64-linux]
0.010000   0.000000   0.010000 (  0.007288)

# ruby 1.9.3dev (2011-04-14 trunk 31267) [x86_64-linux]
0.030000   0.000000   0.030000 (  0.033881)
-----------------------------8< -------------------------
require "benchmark"
nr_thread = 1_000
nr_close = 1_000

threads = nr_thread.times.map { Thread.new { sleep } }
puts(Benchmark.measure do
nr_close.times do
File.open(**FILE**).close
end
end)
threads.each { |thr| thr.run }.each { |thr| thr.join }
--
Eric Wong
=end
```

#20 - 06/11/2011 04:20 PM - kosaki (Motohiro KOSAKI)

- Status changed from Assigned to Closed

1) Windows test case failure have already been fixed.
2) "release GVL on close" is now discussing #4570.

So, we can close this ticket.