Ruby master - Bug #13632

Not processable interrupt queue for a thread after it's notified that FD is closed in some other thread.

06/05/2017 04:16 PM - nvashchenko (Nikolay Vashchenko)

Status:	Closed		
Priority:	Normal		
Assignee:			
Target version:			
ruby -v:	2.2.x, 2.3.x, 2.4.x, 2.5.x	**Backport:**	2.2: UNKNOWN, 2.3: DONE, 2.4: DONE

Description

In the bugfix for https://bugs.ruby-lang.org/issues/13076 has been introduced another bug, caused by a busy waiting in rb_notify_fd_close method, while the FD is being released. During this waiting, it pumps huge amounts of the ruby_error_stream_closed errors into thread's interrupt queue, which almost all stay there unprocessed. It can be up for several hundred of exceptions in the queue, depending on circumstances.

```
  a = []
  t = []
  10.times do
    r,w = IO.pipe
    a << [r,w]
    t << Thread.new do
      while r.gets
      end rescue IOError
      # Interrupt queue is full and all IO is broken
     Thread.current.pending_interrupt? # Expected to be false, because it's already rescued
      IO.sysopen ("/dev/tty")                   # Expected not to throw an error. On each such cal
l, it dequeues the next item from interrupt queue until there's none
    end
  end
  a.each do |r,w|
    w.puts "test"
    w.close
    r.close
  end
  t.each do |th|
    th.join
  end
```

Output:

```
Traceback (most recent call last):
    1: from test2.rb:9:in `block (2 levels) in <main>'
test2.rb:9:in `sysopen': stream closed in another thread (IOError)
```

Associated revisions

Revision 59fb9297 - 06/06/2017 12:13 AM - normal

IO#close: do not enqueue redundant interrupts

Enqueuing multiple errors for one event causes spurious errors
down the line, as reported by Nikolay Vashchenko in
https://bugs.ruby-lang.org/issues/13632

- thread.c (rb_notify_fd_close): do not enqueue multiple interrupts [ruby-core:81581] [Bug #13632]
- test/ruby/test_io.rb (test_single_exception_on_close): new test based on script from Nikolay

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@59020 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision 59020 - 06/06/2017 12:13 AM - normalperson (Eric Wong)

IO#close: do not enqueue redundant interrupts

Enqueuing multiple errors for one event causes spurious errors
down the line, as reported by Nikolay Vashchenko in

https://bugs.ruby-lang.org/issues/13632

- thread.c (rb_notify_fd_close): do not enqueue multiple interrupts [ruby-core:81581] [Bug #13632]
- test/ruby/test_io.rb (test_single_exception_on_close): new test based on script from Nikolay

Revision 59020 - 06/06/2017 12:13 AM - normal

IO#close: do not enqueue redundant interrupts

Enqueuing multiple errors for one event causes spurious errors
down the line, as reported by Nikolay Vashchenko in
https://bugs.ruby-lang.org/issues/13632

- thread.c (rb_notify_fd_close): do not enqueue multiple interrupts [ruby-core:81581] [Bug #13632]
- test/ruby/test_io.rb (test_single_exception_on_close): new test based on script from Nikolay

Revision 59020 - 06/06/2017 12:13 AM - normal

IO#close: do not enqueue redundant interrupts

Enqueuing multiple errors for one event causes spurious errors
down the line, as reported by Nikolay Vashchenko in
https://bugs.ruby-lang.org/issues/13632

- thread.c (rb_notify_fd_close): do not enqueue multiple interrupts [ruby-core:81581] [Bug #13632]
- test/ruby/test_io.rb (test_single_exception_on_close): new test based on script from Nikolay

Revision 10421fb1 - 06/06/2017 10:55 PM - normal

IO#close: do not enqueue redundant interrupts (take #2)

Enqueuing multiple errors for one event causes spurious errors
down the line, as reported by Nikolay Vashchenko in
https://bugs.ruby-lang.org/issues/13632

This should fix bad interactions with test_race_gets_and_close
in test/ruby/test_io.rb since we ensure rb_notify_fd_close
continues returning the busy flag after enqueuing the interrupt.

Backporting changes to 2.4 and earlier releases will be more
challenging...

- thread.c (rb_notify_fd_close): do not enqueue multiple interrupts [ruby-core:81581] [Bug #13632]
- test/ruby/test_io.rb (test_single_exception_on_close): new test based on script from Nikolay

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@59028 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision 59028 - 06/06/2017 10:55 PM - normalperson (Eric Wong)

IO#close: do not enqueue redundant interrupts (take #2)

Enqueuing multiple errors for one event causes spurious errors
down the line, as reported by Nikolay Vashchenko in
https://bugs.ruby-lang.org/issues/13632

This should fix bad interactions with test_race_gets_and_close
in test/ruby/test_io.rb since we ensure rb_notify_fd_close
continues returning the busy flag after enqueuing the interrupt.

Backporting changes to 2.4 and earlier releases will be more
challenging...

- thread.c (rb_notify_fd_close): do not enqueue multiple interrupts [ruby-core:81581] [Bug #13632]
- test/ruby/test_io.rb (test_single_exception_on_close): new test based on script from Nikolay

Revision 59028 - 06/06/2017 10:55 PM - normal

IO#close: do not enqueue redundant interrupts (take #2)

Enqueuing multiple errors for one event causes spurious errors
down the line, as reported by Nikolay Vashchenko in
https://bugs.ruby-lang.org/issues/13632

This should fix bad interactions with test_race_gets_and_close
in test/ruby/test_io.rb since we ensure rb_notify_fd_close

continues returning the busy flag after enqueuing the interrupt.

Backporting changes to 2.4 and earlier releases will be more
challenging...

- thread.c (rb_notify_fd_close): do not enqueue multiple interrupts [ruby-core:81581] [Bug #13632]
- test/ruby/test_io.rb (test_single_exception_on_close): new test based on script from Nikolay

Revision 59028 - 06/06/2017 10:55 PM - normal

IO#close: do not enqueue redundant interrupts (take #2)

Enqueuing multiple errors for one event causes spurious errors
down the line, as reported by Nikolay Vashchenko in
https://bugs.ruby-lang.org/issues/13632

This should fix bad interactions with test_race_gets_and_close
in test/ruby/test_io.rb since we ensure rb_notify_fd_close
continues returning the busy flag after enqueuing the interrupt.

Backporting changes to 2.4 and earlier releases will be more
challenging...

- thread.c (rb_notify_fd_close): do not enqueue multiple interrupts [ruby-core:81581] [Bug #13632]
- test/ruby/test_io.rb (test_single_exception_on_close): new test based on script from Nikolay

Revision 651990fa - 07/07/2017 02:03 AM - usa (Usaku NAKAMURA)

This backport of r58812 is necessary to ease backporting r59028,
which fixes a real bug.

- thread.c (struct waiting_fd): declare
  (rb_thread_io_blocking_region): use on-stack list waiter
  (rb_notify_fd_close): walk vm->waiting_fds instead
  (call_without_gvl): remove old field setting
  (th_init): ditto
  [Feature #9632]

- vm_core.h (typedef struct rb_vm_struct): add waiting_fds list

- (typedef struct rb_thread_struct): remove waiting_fd field
  (rb_vm_living_threads_init): initialize waiting_fds list

This should fix bad interactions with test_race_gets_and_close
in test/ruby/test_io.rb since we ensure rb_notify_fd_close
continues returning the busy flag after enqueuing the interrupt.

- thread.c (rb_notify_fd_close): do not enqueue multiple interrupts
  [ruby-core:81581] [Bug #13632]

- test/ruby/test_io.rb (test_single_exception_on_close):
  new test based on script from Nikolay

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/branches/ruby_2_3@59274 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision 59274 - 07/07/2017 02:03 AM - usa (Usaku NAKAMURA)

This backport of r58812 is necessary to ease backporting r59028,
which fixes a real bug.

- thread.c (struct waiting_fd): declare
  (rb_thread_io_blocking_region): use on-stack list waiter
  (rb_notify_fd_close): walk vm->waiting_fds instead
  (call_without_gvl): remove old field setting
  (th_init): ditto
  [Feature #9632]

- vm_core.h (typedef struct rb_vm_struct): add waiting_fds list

- (typedef struct rb_thread_struct): remove waiting_fd field
  (rb_vm_living_threads_init): initialize waiting_fds list

This should fix bad interactions with test_race_gets_and_close
in test/ruby/test_io.rb since we ensure rb_notify_fd_close
continues returning the busy flag after enqueuing the interrupt.

- thread.c (rb_notify_fd_close): do not enqueue multiple interrupts
  [ruby-core:81581] [Bug #13632]

- test/ruby/test_io.rb (test_single_exception_on_close):
  new test based on script from Nikolay

Revision 27251312 - 07/08/2017 02:21 AM - nagachika (Tomoyuki Chikanaga)

merge revision(s) 58284,58812,59028: [Backport #13632]

```
    vm_core.h: ruby_error_stream_closed

    * vm_core.h (ruby_special_exceptions): renamed
      ruby_error_closed_stream as ruby_error_stream_closed, like the
      message.
    speed up IO#close with many threads

    Today, it increases IO#close performance with many threads:

      Execution time (sec)
      name            trunk   after
      vm_thread_close 4.276   3.018

      Speedup ratio: compare with the result of `trunk' (greater is better)
      name            after
      vm_thread_close 1.417

    This speedup comes because rb_notify_fd_close only scans threads
    inside rb_thread_io_blocking_region, not all threads in the VM.

    In the future, this type data structure may allow us to notify
    waiters of multiple FDs on a single thread (when using
    Fibers).

    * thread.c (struct waiting_fd): declare
      (rb_thread_io_blocking_region): use on-stack list waiter
      (rb_notify_fd_close): walk vm->waiting_fds instead
      (call_without_gvl): remove old field setting
      (th_init): ditto
    * vm_core.h (typedef struct rb_vm_struct): add waiting_fds list
    * (typedef struct rb_thread_struct): remove waiting_fd field
      (rb_vm_living_threads_init): initialize waiting_fds list

    I am now kicking myself for not thinking about this 3 years ago
    when I introduced ccan/list in [Feature #9632] to optimize this
    same function :<
    IO#close: do not enqueue redundant interrupts (take #2)

    Enqueuing multiple errors for one event causes spurious errors
    down the line, as reported by Nikolay Vashchenko in
    https://bugs.ruby-lang.org/issues/13632

    This should fix bad interactions with test_race_gets_and_close
    in test/ruby/test_io.rb since we ensure rb_notify_fd_close
    continues returning the busy flag after enqueuing the interrupt.

    Backporting changes to 2.4 and earlier releases will be more
    challenging...

    * thread.c (rb_notify_fd_close): do not enqueue multiple interrupts
      [ruby-core:81581] [Bug #13632]
    * test/ruby/test_io.rb (test_single_exception_on_close):
      new test based on script from Nikolay
```

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/branches/ruby_2_4@59286 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision 59286 - 07/08/2017 02:21 AM - nagachika (Tomoyuki Chikanaga)

merge revision(s) 58284,58812,59028: [Backport #13632]

```
vm_core.h: ruby_error_stream_closed

* vm_core.h (ruby_special_exceptions): renamed
  ruby_error_closed_stream as ruby_error_stream_closed, like the
  message.
speed up IO#close with many threads

Today, it increases IO#close performance with many threads:

  Execution time (sec)
  name            trunk   after
  vm_thread_close 4.276   3.018

  Speedup ratio: compare with the result of `trunk' (greater is better)
  name            after
  vm_thread_close 1.417

This speedup comes because rb_notify_fd_close only scans threads
inside rb_thread_io_blocking_region, not all threads in the VM.

In the future, this type data structure may allow us to notify
waiters of multiple FDs on a single thread (when using
Fibers).

* thread.c (struct waiting_fd): declare
  (rb_thread_io_blocking_region): use on-stack list waiter
  (rb_notify_fd_close): walk vm->waiting_fds instead
  (call_without_gvl): remove old field setting
  (th_init): ditto
* vm_core.h (typedef struct rb_vm_struct): add waiting_fds list
* (typedef struct rb_thread_struct): remove waiting_fd field
  (rb_vm_living_threads_init): initialize waiting_fds list

I am now kicking myself for not thinking about this 3 years ago
when I introduced ccan/list in [Feature #9632] to optimize this
same function :<
IO#close: do not enqueue redundant interrupts (take #2)

Enqueuing multiple errors for one event causes spurious errors
down the line, as reported by Nikolay Vashchenko in
https://bugs.ruby-lang.org/issues/13632

This should fix bad interactions with test_race_gets_and_close
in test/ruby/test_io.rb since we ensure rb_notify_fd_close
continues returning the busy flag after enqueuing the interrupt.

Backporting changes to 2.4 and earlier releases will be more
challenging...

* thread.c (rb_notify_fd_close): do not enqueue multiple interrupts
  [ruby-core:81581] [Bug #13632]
* test/ruby/test_io.rb (test_single_exception_on_close):
  new test based on script from Nikolay
```

History

#1 - 06/05/2017 04:20 PM - nvashchenko (Nikolay Vashchenko)

- Description updated

#2 - 06/05/2017 04:20 PM - nvashchenko (Nikolay Vashchenko)

- Description updated

#3 - 06/05/2017 10:03 PM - normalperson (Eric Wong)

sir.nickolas@gmail.com wrote:

> Bug #13632: Not processable interrupt queue for a thread after it's notified that FD is closed in some other thread.
> https://bugs.ruby-lang.org/issues/13632

Investigating. I've been looking at IO close notification for
Feature #13618 anyways.

#4 - 06/06/2017 12:13 AM - Anonymous

- Status changed from Open to Closed

Applied in changeset trunk|r59020.

IO#close: do not enqueue redundant interrupts

Enqueuing multiple errors for one event causes spurious errors
down the line, as reported by Nikolay Vashchenko in
https://bugs.ruby-lang.org/issues/13632

- thread.c (rb_notify_fd_close): do not enqueue multiple interrupts [ruby-core:81581] [Bug #13632]
- test/ruby/test_io.rb (test_single_exception_on_close): new test based on script from Nikolay

#5 - 06/06/2017 12:21 AM - normalperson (Eric Wong)

sir.nickolas@gmail.com wrote:

> https://bugs.ruby-lang.org/issues/13632

r59020 should fix it trivially in trunk.

Backporting to <= 2.4 is only a little different due to the
data structure change:

```
s/wfd->fd = -1/th->waiting_fd = -1/
https://80x24.org/spew/20170606001646.22889-1-e@80x24.org/raw
```

Thanks for the report!

#6 - 06/06/2017 04:26 AM - ko1 (Koichi Sasada)

- Description updated

#7 - 06/07/2017 08:13 PM - normalperson (Eric Wong)

Eric Wong normalperson@yhbt.net wrote:

> sir.nickolas@gmail.com wrote:
>
> > https://bugs.ruby-lang.org/issues/13632
> >
> > r59020 should fix it trivially in trunk.

Make that r59028 :x r59020 interacted badly with r57422

> Backporting to <= 2.4 is only a little different due to the
> data structure change:

r59028 backporting is more difficult.

Below are links to backported patches from trunk to fix [Bug #13632] for
Ruby 2.4 maintenance branches and earlier. I mainly wanted to
backport r59028, but that depends on r58812 (originally intended
as a pure performance optimization).

I'd rather not change r59028 to something unrecognizable from
what is in trunk for backporting, as that might negatively
impact future backports.

r58812: speed up IO#close with many threads
https://80x24.org/spew/20170607195901.18958-2-e@80x24.org/raw
r59028: IO#close: do not enqueue redundant interrupts (take #2)
https://80x24.org/spew/20170607195901.18958-3-e@80x24.org/raw

```
test/ruby/test_io.rb | 22 ++++++++++++++++++++
thread.c             | 33 ++++++++++++++++++++++++++---------
vm.c                 |  1 -
vm_core.h            |  4 ++--
```

4 files changed, 48 insertions(+), 12 deletions(-)

#8 - 06/29/2017 05:05 PM - usa (Usaku NAKAMURA)

- Backport changed from 2.2: UNKNOWN, 2.3: UNKNOWN, 2.4: UNKNOWN to 2.2: UNKNOWN, 2.3: UNKNOWN, 2.4: REQUIRED

#9 - 07/07/2017 02:07 AM - usa (Usaku NAKAMURA)

- Backport changed from 2.2: UNKNOWN, 2.3: UNKNOWN, 2.4: REQUIRED to 2.2: UNKNOWN, 2.3: DONE, 2.4: REQUIRED

#10 - 07/08/2017 02:21 AM - nagachika (Tomoyuki Chikanaga)

- Backport changed from 2.2: UNKNOWN, 2.3: DONE, 2.4: REQUIRED to 2.2: UNKNOWN, 2.3: DONE, 2.4: DONE

ruby_2_4 r59286 merged revision(s) 58284,58812,59028.

#11 - 07/17/2017 04:22 AM - nvashchenko (Nikolay Vashchenko)

I created a gem with a temporary workaround for versions 2.2.7, 2.3.4 and 2.4.1 to help folks until versions with backports are released:
https://rubygems.org/gems/stopgap_13632