Ruby master - Bug #4555

[PATCH] ext/socket/init.c: rsock_connect retries on interrupt

04/06/2011 08:13 AM - normalperson (Eric Wong)

Status:	Closed		
Priority:	Normal		
Assignee:	kosaki (Motohiro KOSAKI)		
Target version:	2.0.0		
ruby -v:	-	**Backport:**	

Description

=begin
Otherwise I get the following error in test/openssl/test_ssl.rb:

test_verify_result(OpenSSL::TestSSL):
Errno::EINTR: Interrupted system call - connect(2)
/tmp/ruby/test/openssl/test_ssl.rb:338:in initialize'
/tmp/ruby/test/openssl/test_ssl.rb:338:innew'
/tmp/ruby/test/openssl/test_ssl.rb:338:in block in test_verify_result'
/tmp/ruby/test/openssl/test_ssl.rb:117:incall'
/tmp/ruby/test/openssl/test_ssl.rb:117:in start_server'
/tmp/ruby/test/openssl/test_ssl.rb:330:intest_verify_result'

This bug is made more noticeable by r31230, though it always
existed before.

Fwiw, I think all the wait_connectable() logic incorrectly uses
the except_fds parameter of select() and rb_io_wait_writable()
should be used here...

=end

Associated revisions

Revision 49b4510c - 05/01/2011 03:38 PM - kosaki (Motohiro KOSAKI)

- ext/socket/init.c (rsock_connect): add to care EINTR. based on a patch from Eric Wong at [ruby-core:35621][Bug #4555]

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@31405 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision 31405 - 05/01/2011 03:38 PM - kosaki (Motohiro KOSAKI)

- ext/socket/init.c (rsock_connect): add to care EINTR. based on a patch from Eric Wong at [ruby-core:35621][Bug #4555]

Revision 31405 - 05/01/2011 03:38 PM - kosaki (Motohiro KOSAKI)

- ext/socket/init.c (rsock_connect): add to care EINTR. based on a patch from Eric Wong at [ruby-core:35621][Bug #4555]

Revision 31405 - 05/01/2011 03:38 PM - kosaki (Motohiro KOSAKI)

- ext/socket/init.c (rsock_connect): add to care EINTR. based on a patch from Eric Wong at [ruby-core:35621][Bug #4555]

Revision 31405 - 05/01/2011 03:38 PM - kosaki (Motohiro KOSAKI)

- ext/socket/init.c (rsock_connect): add to care EINTR. based on a patch from Eric Wong at [ruby-core:35621][Bug #4555]

Revision 31405 - 05/01/2011 03:38 PM - kosaki (Motohiro KOSAKI)

- ext/socket/init.c (rsock_connect): add to care EINTR. based on a patch from Eric Wong at [ruby-core:35621][Bug #4555]

Revision 31405 - 05/01/2011 03:38 PM - kosaki (Motohiro KOSAKI)

- ext/socket/init.c (rsock_connect): add to care EINTR. based on a patch from Eric Wong at [ruby-core:35621][Bug #4555]

Revision d3818d6b - 05/31/2011 12:11 AM - yugui (Yuki Sonoda)

merges r31405 from trunk into ruby_1_9_2.

- ext/socket/init.c (rsock_connect): add to care EINTR. based on a patch from Eric Wong at [ruby-core:35621][Bug #4555]

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/branches/ruby_1_9_2@31829 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

History

#1 - 04/08/2011 09:31 AM - naruse (Yui NARUSE)

- Status changed from Open to Assigned

- Assignee set to kosaki (Motohiro KOSAKI)

- Priority changed from 5 to Normal

=begin

=end

#2 - 04/11/2011 11:08 PM - kosaki (Motohiro KOSAKI)

=begin
Hi

I'm not convinced why we can safely call select() when we get EINTR. IOW,
Why don't you choose following patch?

Index: ext/socket/init.c

--- ext/socket/init.c (revision 31258)
+++ ext/socket/init.c (working copy)
@@ -383,6 +383,12 @@
status = (int)BLOCKING_REGION_FD(func, &arg);
if (status < 0) {
switch (errno) {

- case EINTR: +#if defined(ERESTART)
- case ERESTART: +#endif
- continue; + case EAGAIN: #ifdef EINPROGRESS case EINPROGRESS:

=end

#3 - 04/12/2011 01:04 AM - normalperson (Eric Wong)

=begin
Your patch looks reasonable to me, but maybe some platforms break under it...

I was trying to emulate rb_io_wait_writable() logic which calls
rb_thread_fd_writable() (which wraps select() if there are multiple threads).
Maybe avoiding select() in all EINTR cases will work, I am under the impression
a lot of the select()-wrapping calls in Ruby are relics of the old green threading
and can be removed.

=end

#4 - 05/01/2011 01:23 PM - kosaki (Motohiro KOSAKI)

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

> Your patch looks reasonable to me, but maybe some platforms break under it...

Can you please clarify this? Which platform break and why?

> I was trying to emulate rb_io_wait_writable() logic which calls
> rb_thread_fd_writable() (which wraps select() if there are multiple threads).

> Maybe avoiding select() in all EINTR cases will work, I am under the impression
> a lot of the select()-wrapping calls in Ruby are relics of the old green threading
> and can be removed.

I missed your point. If anyone set non-blocking flag and call socket#connect,
the connect call in rsock_connect() can return non-blocking related error. We
have to care it even though we only have single thread. Am I missing something?

#5 - 05/01/2011 02:29 PM - normalperson (Eric Wong)

KOSAKI Motohiro kosaki.motohiro@gmail.com wrote:

> > Your patch looks reasonable to me, but maybe some platforms break under it...
>
> Can you please clarify this? Which platform break and why?

Nevermind, see below:

> > I was trying to emulate rb_io_wait_writable() logic which calls
> > rb_thread_fd_writable() (which wraps select() if there are multiple threads).
> > Maybe avoiding select() in all EINTR cases will work, I am under the impression
> > a lot of the select()-wrapping calls in Ruby are relics of the old green threading
> > and can be removed.
>
> I missed your point. If anyone set non-blocking flag and call socket#connect,
> the connect call in rsock_connect() can return non-blocking related error. We
> have to care it even though we only have single thread. Am I missing something?

You're correct. I was mistaken and thought rsock_connect() was intended
to be retrying and emulate a blocking operation (like IO#write even if
O_NONBLOCK is set)

--
Eric Wong

#6 - 05/02/2011 12:39 AM - kosaki (Motohiro KOSAKI)

- Status changed from Assigned to Closed

r31405

Files

0001-ext-socket-init.c-rsock_connect-retries-on-interrupt.patch	1.48 KB	04/06/2011	normalperson (Eric Wong)