Ruby master - Feature #5101

allow optional timeout for TCPSocket.new

07/27/2011 08:59 AM - normalperson (Eric Wong)

Status:	Closed	
Priority:	Normal	
Assignee:	akr (Akira Tanaka)	
Target version:	2.0.0	

Description

Proposed API would be:

TCPSocket.new(remote_host, remote_port,
connect_timeout: 0.5, local_host: nil, local_port: nil)

Or:

TCPSocket.new(remote_host, remote_port, connect_timeout: 0.5)

For the common case.

This would only timeout for establishing the TCP connection, not DNS
resolution. DNS resolution can be covered by Feature #5100.

The existing form for the (rarely-used) local_host/local_port args will
still be supported for backwards compatibility:

TCPSocket.new(remote_host, remote_port, local_host=nil, local_port=nil)

The current construct for doing a non-blocking connect with timeout
is too verbose:

```
addr = Socket.pack_sockaddr_in(9418, "127.0.0.1")
s = Socket.new(:AF_INET, :SOCK_STREAM, 0)
begin
  s.connect_nonblock(addr)
rescue Errno::EINPROGRESS
  IO.select(nil, [s], nil, 0.5) or raise Timeout::Error
end
```

And could be replaced with:

```
TCPSocket.new("127.0.0.1", 9418, connect_timeout: 0.5)
```

I am not sure what exception TCPSocket.new should return. Timeout::Error
seems like a reasonable choice...

History

#1 - 07/27/2011 04:13 PM - naruse (Yui NARUSE)

- Status changed from Open to Assigned

- Assignee set to akr (Akira Tanaka)

#2 - 07/27/2011 11:53 PM - normalperson (Eric Wong)

Eric Wong normalperson@yhbt.net wrote:

> Proposed API would be:
>
> TCPSocket.new(remote_host, remote_port,
> connect_timeout: 0.5, local_host: nil, local_port: nil)

Btw, I'll gladly provide patches + tests if the API is approved or a better one is suggested :>

#3 - 07/28/2011 03:23 AM - normalperson (Eric Wong)

Eric Wong normalperson@yhbt.net wrote:

> Proposed API would be:
>
> TCPSocket.new(remote_host, remote_port,
> connect_timeout: 0.5, local_host: nil, local_port: nil)

An alternative (or complimentary) API could also be to add a new class
method that only starts the TCP connection, but returns the socket
on EINPROGRESS:

TCPSocket.start(remote_host, remote_port,
local_host=nil, local_port=nil) => tcp_socket

This means the user would have to wait for writability on the socket
returned by this method (e.g. with IO.select).

I've already implemented something similar for kgio:
http://bogomips.org/kgio/Kgio/TCPSocket.html#method-c-start

.. and am of course happy to provide patches for Ruby itself if
ruby-core agrees it's a good API to have :>

--
Eric Wong

#4 - 07/28/2011 09:23 PM - akr (Akira Tanaka)

2011/7/27 Eric Wong normalperson@yhbt.net:

> Proposed API would be:
>
> TCPSocket.new(remote_host, remote_port,
> connect_timeout: 0.5, local_host: nil, local_port: nil)
>
> Or:
>
> TCPSocket.new(remote_host, remote_port, connect_timeout: 0.5)

How about Socket class?
You can implement timeout on Socket.

I think you need full power of socket API,
so you should use low level bindings for socket API i.e. Socket class.

Note that Socket is not so cumbersome since Ruby 1.9.2.
--
Tanaka Akira

#5 - 07/29/2011 01:59 AM - normalperson (Eric Wong)

Tanaka Akira akr@fsij.org wrote:

> 2011/7/27 Eric Wong normalperson@yhbt.net:
>
>> Proposed API would be:
>>
>> TCPSocket.new(remote_host, remote_port,
>> connect_timeout: 0.5, local_host: nil, local_port: nil)
>>
>> Or:
>>
>> TCPSocket.new(remote_host, remote_port, connect_timeout: 0.5)
>
> How about Socket class?
> You can implement timeout on Socket.
>
> I think you need full power of socket API,

> so you should use low level bindings for socket API i.e. Socket class.
>
> Note that Socket is not so cumbersome since Ruby 1.9.2.

Like my original example in the ticket?

```
addr = Socket.pack_sockaddr_in(9418, "127.0.0.1")
s = Socket.new(:AF_INET, :SOCK_STREAM, 0)
begin
  s.connect_nonblock(addr)
rescue Errno::EINPROGRESS
  IO.select(nil, [s], nil, 0.5) or raise Timeout::Error
end
```

I can only seem to shorten it to the following, is there a better way?
I can't find it looking through ext/socket/...

```
addr = Addrinfo.tcp("127.0.0.1", 9418)
s = Socket.new(:AF_INET, :SOCK_STREAM)
begin
  s.connect_nonblock(addr)
rescue Errno::EINPROGRESS
  IO.select(nil, [s], nil, 0.5) or raise Timeout::Error
end
```

Anyways I would like to be able to implement open_timeout in net/http
without using threads (with timeout.rb). Also see
http://redmine.ruby-lang.org/issues/5100 for resolv.rb timeouts.

--
Eric Wong

#6 - 07/30/2011 12:53 AM - akr (Akira Tanaka)

2011/7/29 Eric Wong normalperson@yhbt.net:

> Like my original example in the ticket?
>
> addr = Socket.pack_sockaddr_in(9418, "127.0.0.1")
> s = Socket.new(:AF_INET, :SOCK_STREAM, 0)
> begin
> s.connect_nonblock(addr)
> rescue Errno::EINPROGRESS
> IO.select(nil, [s], nil, 0.5) or raise Timeout::Error
> end
>
> I can only seem to shorten it to the following, is there a better way?
> I can't find it looking through ext/socket/...
>
> addr = Addrinfo.tcp("127.0.0.1", 9418)
> s = Socket.new(:AF_INET, :SOCK_STREAM)
> begin
> s.connect_nonblock(addr)
> rescue Errno::EINPROGRESS
> IO.select(nil, [s], nil, 0.5) or raise Timeout::Error
> end

I expected such code in client socket library because
socket library doesn't provide timeout feature (and Timeout exception class).

> Anyways I would like to be able to implement open_timeout in net/http
> without using threads (with timeout.rb). Also see

I tested net/http using Socket class as follows.
This doesn't cause any problems with test-all.

Index: lib/net/http.rb
===
--- lib/net/http.rb   (revision 32734)
+++ lib/net/http.rb   (working copy)
@@ -763,7 +763,7 @@

```
  def connect
    D "opening connection to #{conn_address()}..."
```

- s = timeout(@open_timeout) { TCPSocket.open(conn_address(), conn_port()) }
- s = timeout(@open_timeout) { Socket.tcp(conn_address(), conn_port()) }   D "opened"   if use_ssl?   ssl_parameters = Hash.new

I remember multiple IP addresses issue.
The behavior of timeout is not clear if two or more IP addresses are
given for a hostname.
--
Tanaka Akira

#7 - 08/01/2011 10:23 PM - akr (Akira Tanaka)

2011/7/30 Tanaka Akira akr@fsij.org:

> I expected such code in client socket library because
> socket library doesn't provide timeout feature (and Timeout exception class).

On second thought, Errno::ETIMEDOUT can be a candidate.

> I remember multiple IP addresses issue.
> The behavior of timeout is not clear if two or more IP addresses are
> given for a hostname.

This is still a problem.
--
Tanaka Akira

#8 - 08/02/2011 05:29 AM - normalperson (Eric Wong)

Tanaka Akira akr@fsij.org wrote:

> 2011/7/30 Tanaka Akira akr@fsij.org:
>
> > I expected such code in client socket library because
> > socket library doesn't provide timeout feature (and Timeout exception class).
>
> On second thought, Errno::ETIMEDOUT can be a candidate.

OK; sounds good.

> > I remember multiple IP addresses issue.
> > The behavior of timeout is not clear if two or more IP addresses are
> > given for a hostname.
>
> This is still a problem.

I only want this timeout to affect connect() time, DNS timeout can
be separate (Feature #5100). I don't know what to do about multiple
IP addresses, it may be better on a case-by-case basis or until total
time runs out.

Maybe even have an option to do a non-blocking connect() to all IPs
simultaneously and use the first one that returns writability
with IO.select:

# This may make some server/network admins unhappy and even be
# considered abusive and get clients banned, but it could be useful
# in some situations; too
sockets = addresses.map do |addr|
s = Socket.new(:AF_INET, :SOCK_STREAM)
begin
s.connect_nonblock(addr)
rescue Errno::EINPROGRESS
end
s
end
ready = IO.select(nil, sockets, nil, 0.5) or raise Errno::ETIMEDOUT

```
keep = nil
ready[1].each do |sock|
if keep
sock.close
else
begin
do_something(sock)
keep = sock
rescue
sock.close
end
end
end
do_more_things(keep)
```

--
Eric Wong

#9 - 08/02/2011 07:23 AM - akr (Akira Tanaka)

2011/8/2 Eric Wong normalperson@yhbt.net:

> I only want this timeout to affect connect() time, DNS timeout can
> be separate (Feature #5100). I don't know what to do about multiple
> IP addresses, it may be better on a case-by-case basis or until total
> time runs out.

I see.

If N IP address is obtained, N * connect_timeout seconds may be required for
timeout.

> Maybe even have an option to do a non-blocking connect() to all IPs
> simultaneously and use the first one that returns writability
> with IO.select:

Clealy, it should be optional.
--
Tanaka Akira

#10 - 08/03/2011 12:42 PM - akr (Akira Tanaka)

- File socket-tcp-connect-timeout.patch added

I made a pactch for connect_timeout for Socket.tcp.

with timeout:

```
% time ./ruby -rsocket -e 'Socket.tcp("192.0.2.1", 80, :connect_timeout=>1)'
/home/akr/ruby/tst1/lib/ruby/1.9.1/socket.rb:48:in rescue in connect_internal': Connection timed out - user specified timeout (Errno::ETIMEDOUT)
from /home/akr/ruby/tst1/lib/ruby/1.9.1/socket.rb:44:inconnect_internal'
from /home/akr/ruby/tst1/lib/ruby/1.9.1/socket.rb:108:in connect'
from /home/akr/ruby/tst1/lib/ruby/1.9.1/socket.rb:277:inblock in tcp'
from /home/akr/ruby/tst1/lib/ruby/1.9.1/socket.rb:181:in each'
from /home/akr/ruby/tst1/lib/ruby/1.9.1/socket.rb:181:inforeach'
from /home/akr/ruby/tst1/lib/ruby/1.9.1/socket.rb:267:in tcp'
from -e:1:in'
./ruby -rsocket -e 'Socket.tcp("192.0.2.1", 80, :connect_timeout=>1)'  0.12s user 0.00s system 10% cpu 1.126 total
```

without timeout:

```
% time ./ruby -rsocket -e 'Socket.tcp("192.0.2.1", 80)'
/home/akr/ruby/tst1/lib/ruby/1.9.1/socket.rb:56:in connect': Connection timed out - connect(2) (Errno::ETIMEDOUT)
from /home/akr/ruby/tst1/lib/ruby/1.9.1/socket.rb:56:inconnect_internal'
from /home/akr/ruby/tst1/lib/ruby/1.9.1/socket.rb:108:in connect'
from /home/akr/ruby/tst1/lib/ruby/1.9.1/socket.rb:277:inblock in tcp'
from /home/akr/ruby/tst1/lib/ruby/1.9.1/socket.rb:181:in each'
from /home/akr/ruby/tst1/lib/ruby/1.9.1/socket.rb:181:inforeach'
from /home/akr/ruby/tst1/lib/ruby/1.9.1/socket.rb:267:in tcp'
from -e:1:in'
./ruby -rsocket -e 'Socket.tcp("192.0.2.1", 80)'  0.12s user 0.00s system 0% cpu 3:09.51 total
```

#11 - 08/05/2011 11:59 AM - normalperson (Eric Wong)

Akira Tanaka akr@fsij.org wrote:

File socket-tcp-connect-timeout.patch added

Looks good to me, thanks!

#12 - 08/10/2011 09:55 PM - akr (Akira Tanaka)

- File socket-tcp-connect-timeout-2.patch added

I updated the patch for document and argument checking.
I'll commit it later.

#13 - 08/10/2011 10:21 PM - akr (Akira Tanaka)

- Status changed from Assigned to Closed

#14 - 08/11/2011 05:33 AM - normalperson (Eric Wong)

- File 0001-test-socket-test_socket.rb-add-test-for-Socket.tcp-w.patch added

Thanks! I've attached a test case for test_socket.rb to test this feature.

Files

socket-tcp-connect-timeout.patch	3.7 KB	08/03/2011	akr (Akira Tanaka)
socket-tcp-connect-timeout-2.patch	6.58 KB	08/10/2011	akr (Akira Tanaka)
0001-test-socket-test_socket.rb-add-test-for-Socket.tcp-w.patch	1.72 KB	08/11/2011	normalperson (Eric Wong)