Ruby master - Bug #18940
Ruby Ractor fails with IOError when handling higher concurrency

07/26/2022 06:33 PM - brodock (Gabriel Mazetto)

Status: Open  
Priority: Normal  
Assignee:  
Target version:  
ruby -v: ruby 3.2.0dev (2022-07-26T16:40:03Z master 3b1ed03d8c) [arm64-darwin21]  
Backport: 2.7: UNKNOWN, 3.0: UNKNOWN, 3.1: UNKNOWN

Description
Reproduction server:

```ruby
require 'socket'

# Set based on CPU count
CONCURRENCY = 8

server = TCPServer.new(8080)
workers = CONCURRENCY.times.map do
  Ractor.new do
    loop do
      # receive TCPSocket
      session = Ractor.recv

      request = session.gets
      puts request

      session.print "HTTP/1.1 200\r\n"
      session.print "Content-Type: text/html\r\n"
      session.print "\r\n"
      session.print "Hello world! Current time is #{Time.now}"
      session.close
    end
  end
end

loop do
  conn, _ = server.accept
  # pass TCPSocket to one of the workers
  workers.sample.send(conn, move: true)
end

run apache benchmark against code above:

ab -n 20000 -c 20 http://localhost:8080/

or run using hey (https://github.com/rakyll/hey):

hey -n 20000 -c 20 http://localhost:8080/

you should see something like this on the benchmark tool side:

Summary:
  Total: 32.9538 secs
  Slowest: 2.6317 secs
  Fastest: 0.0002 secs
  Average: 0.0331 secs
  Requests/sec: 606.9098
```
Response time histogram:

- 0.000 [1] |
- 0.263 [16968] |*
- 0.527 [1740] |
- 0.790 [0] |
- 1.053 [0] |
- 1.316 [0] |
- 1.579 [0] |
- 1.842 [0] |
- 2.105 [20] |
- 2.369 [0] |
- 2.632 [6] |

Latency distribution:
- 10% in 0.0008 secs
- 25% in 0.0010 secs
- 50% in 0.0012 secs
- 75% in 0.0016 secs
- 90% in 0.0075 secs
- 95% in 0.3101 secs
- 99% in 0.3175 secs

Details (average, fastest, slowest):
- DNS+dialup: 0.0322 secs, 0.0002 secs, 2.6317 secs
- DNS-lookup: 0.0006 secs, 0.0000 secs, 0.0127 secs
- req write: 0.0001 secs, 0.0000 secs, 0.0095 secs
- resp wait: 0.0007 secs, 0.0000 secs, 0.0140 secs
- resp read: 0.0001 secs, 0.0000 secs, 0.0088 secs

Status code distribution:
- [200] 18735 responses

Error distribution:

and this on the ruby process:

...

GET / HTTP/1.1
GET / HTTP/1.1
#<Thread:0x000000000010b6e8 run> terminated with exception (report_on_exception is true):
  ractor.rb:21:in `write': GET / HTTP/1.1
  uninitialized stream (IOError)
    from ractor.rb:21:in `print'
    from ractor.rb:21:in `block (3 levels) in <main>''
    from ractor.rb:11:in `loop'
    from ractor.rb:11:in `block (2 levels) in <main>'
GET / HTTP/1.1
GET / HTTP/1.1