Ruby - Feature #14268

[PATCH] net/protocol: optimize large read case

01/02/2018 08:53 AM - normalperson (Eric Wong)

Status:	Closed
Priority:	Normal
Assignee:	naruse (Yui NARUSE)
Target version:	

Description

```
Any comment?  Will commit in a few days if none.

net/protocol: optimize large read case

There are several places where rbuf_consume is called with
@rbuf.size as its length arg; simplify that case by avoiding
the slow String#slice! operation in favor of a lightweight
replacement.

The following script exhibits reduced memory usage and
runtimes using the time(1) command:

 2.9s =>  2.6s
 70MB => 12 MB

---------
require 'net/http'
require 'digest/md5'
Thread.abort_on_exception = true
s = TCPServer.new('127.0.0.1', 0)
len = 1024 * 1024 * 1024
th = Thread.new do
  c = s.accept
  c.readpartial(16384)
  c.write("HTTP/1.0 200 OK\r\nContent-Length: #{len}\r\n\r\n")
  IO.copy_stream('/dev/zero', c, len)
  c.close
end

addr = s.addr
Net::HTTP.start(addr[3], addr[1]) do |http|
  http.request_get('/') do |res|
    dig = Digest::MD5.new
    res.read_body { |buf|
      dig.update(buf)
      # String#clear is important to reduce malloc overhead,
      # but most Ruby programmers don't do this :<
      buf.clear
    }
    puts dig.hexdigest
  end
end
----------

* lib/net/protocol (rbuf_consume): optimize for @rbuf.size == len
```

Associated revisions

Revision d9beb7690fb88734de52c67ab030cf0f864a2762 - 01/05/2018 02:22 AM - Eric Wong

net/protocol: optimize large read case

There are several places where rbuf_consume is called with
@rbuf.size as its length arg; simplify that case by avoiding
the slow String#slice! operation in favor of a lightweight

replacement.

The following script exhibits reduced memory usage and
runtimes using the time(1) command:

```
    2.9s =>  2.6s
    70MB => 12 MB
```

```
require 'net/http'
require 'digest/md5'
Thread.abort_on_exception = true
s = TCPServer.new('127.0.0.1', 0)
len = 1024 * 1024 * 1024
th = Thread.new do
c = s.accept
c.readpartial(16384)
c.write("HTTP/1.0 200 OK\r\nContent-Length: #{len}\r\n\r\n")
IO.copy_stream('/dev/zero', c, len)
c.close
end
```

```
addr = s.addr
Net::HTTP.start(addr[3], addr[1]) do |http|
http.request_get('/') do |res|
dig = Digest::MD5.new
res.read_body { |buf|
dig.update(buf)
# String#clear is important to reduce malloc overhead,
# but most Ruby programmers don't do this :<
buf.clear
}
puts dig.hexdigest
end
end
```

- lib/net/protocol (rbuf_consume): optimize for @rbuf.size == len
  [Feature #14268]

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@61602 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision d9beb769 - 01/05/2018 02:22 AM - Eric Wong

net/protocol: optimize large read case

There are several places where rbuf_consume is called with
@rbuf.size as its length arg; simplify that case by avoiding
the slow String#slice! operation in favor of a lightweight
replacement.

The following script exhibits reduced memory usage and
runtimes using the time(1) command:

```
    2.9s =>  2.6s
    70MB => 12 MB
```

```
require 'net/http'
require 'digest/md5'
Thread.abort_on_exception = true
s = TCPServer.new('127.0.0.1', 0)
len = 1024 * 1024 * 1024
th = Thread.new do
c = s.accept
c.readpartial(16384)
c.write("HTTP/1.0 200 OK\r\nContent-Length: #{len}\r\n\r\n")
IO.copy_stream('/dev/zero', c, len)
c.close
end
```

addr = s.addr
Net::HTTP.start(addr[3], addr[1]) do |http|
http.request_get('/') do |res|
dig = Digest::MD5.new
res.read_body { |buf|
dig.update(buf)
 # String#clear is important to reduce malloc overhead,
 # but most Ruby programmers don't do this :<
buf.clear
}
puts dig.hexdigest
end
end

- lib/net/protocol (rbuf_consume): optimize for @rbuf.size == len
  [Feature #14268]

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@61602 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

History

#1 - 01/05/2018 02:22 AM - Anonymous

- Status changed from Open to Closed

Applied in changeset trunk|r61602.

---

net/protocol: optimize large read case

There are several places where rbuf_consume is called with
@rbuf.size as its length arg; simplify that case by avoiding
the slow String#slice! operation in favor of a lightweight
replacement.

The following script exhibits reduced memory usage and
runtimes using the time(1) command:

```
2.9s =>  2.6s
70MB => 12 MB
```

---

require 'net/http'
require 'digest/md5'
Thread.abort_on_exception = true
s = TCPServer.new('127.0.0.1', 0)
len = 1024 * 1024 * 1024
th = Thread.new do
c = s.accept
c.readpartial(16384)
c.write("HTTP/1.0 200 OK\r\nContent-Length: #{len}\r\n\r\n")
IO.copy_stream('/dev/zero', c, len)
c.close
end

addr = s.addr
Net::HTTP.start(addr[3], addr[1]) do |http|
http.request_get('/') do |res|
dig = Digest::MD5.new
res.read_body { |buf|
dig.update(buf)
 # String#clear is important to reduce malloc overhead,
 # but most Ruby programmers don't do this :<
buf.clear

```
}
puts dig.hexdigest
end
end
```

- lib/net/protocol (rbuf_consume): optimize for @rbuf.size == len
  [Feature #14268]

Files

0001-net-protocol-optimize-large-read-case.patch	1.74 KB	01/02/2018	normalperson (Eric Wong)