Ruby - Bug #15356

Rack::Deflater on Rails responds with no data on Ruby 2.6.0

11/28/2018 07:23 PM - zunda (zunda an)

Status:	Closed		
Priority:	Normal		
Assignee:			
Target version:			
ruby -v:	ruby 2.6.0dev (2018-11-28 trunk 66081) [x86_64-linux]	**Backport:**	2.4: UNKNOWN, 2.5: UNKNOWN

Description

I'd like to report that a Rails app with Rack::Deflater enabled responds with a zero-length body on Ruby 2.6.0, to a request with Accept-encoding: gzip request header. It seems to happen at least on x86_64-linux (Ubuntu 18.04.1; see below for procedure for a reproduction) as well as on universal.x86_64-darwin17 (macOS High Sierra 10.13.6; tested about a half year ago). I suspect some kind of change in behavior of pipe or named pipe is causing this.

1: Prepare a Rails app, enable Rack::Deflater, and run the server

```
$ rbenv install 2.6.0-dev
$ mkdir ruby26-rails-deflater
$ cd ruby26-rails-deflater
$ echo 2.6.0-dev > .ruby-version
$ ruby --version
ruby 2.6.0dev (2018-11-28 trunk 66081) [x86_64-linux]
$ cat <<_END > Gemfile
source 'https://rubygems.org'
gem 'rails', '~> 5.2.1'
_END
$ bundle install --path=vendor/bundle
$ bundle exec rails new test-deflater
$ cd test-deflater
$ sed -ie '/config.load_defaults 5.2/a config.middleware.use Rack::Deflater' \
config/application.rb
$ bundle exec rails s
```

2: Send a request

From another terminal, run the curl command. The 0 in the last line represents the first chunked response body with zero-length. I expect to see the "Yay! You're on Rails!" response body.

```
$ curl -H 'Accept-encoding: gzip' -v http://localhost:3000 --raw
* Rebuilt URL to: http://localhost:3000/
*   Trying 127.0.0.1...
* TCP_NODELAY set
* Connected to localhost (127.0.0.1) port 3000 (#0)
> GET / HTTP/1.1
> Host: localhost:3000
> User-Agent: curl/7.58.0
> Accept: */*
> Accept-encoding: gzip
>
< HTTP/1.1 200 OK
< X-Frame-Options: SAMEORIGIN
< X-XSS-Protection: 1; mode=block
< X-Content-Type-Options: nosniff
< X-Download-Options: noopen
< X-Permitted-Cross-Domain-Policies: none
< Referrer-Policy: strict-origin-when-cross-origin
< Content-Type: text/html; charset=utf-8
< Vary: Accept-Encoding
< Content-Encoding: gzip
```

```
< ETag: W/"cd54cb5f8d93b7800b9e76460c5fe915"
< Cache-Control: max-age=0, private, must-revalidate
< Content-Security-Policy: script-src 'unsafe-inline'; style-src 'unsafe-inline'
< X-Request-Id: e671eca8-f6ff-4599-b241-ea00fd864f56
< X-Runtime: 0.033015
< Transfer-Encoding: chunked
<
0

* Connection #0 to host localhost left intact
```

Puma logs doesn't show anything out of ordinary:

```
Started GET "/" for 127.0.0.1 at 2018-11-28 09:09:44 -1000
Processing by Rails::WelcomeController#index as */*
  Rendering /home/zunda/.rbenv/versions/2.6.0-dev/lib/ruby/gems/2.6.0/gems/railties-5.2.1.1/lib/ra
ils/templates/rails/welcome/index.html.erb
  Rendered /home/zunda/.rbenv/versions/2.6.0-dev/lib/ruby/gems/2.6.0/gems/railties-5.2.1.1/lib/rai
ls/templates/rails/welcome/index.html.erb (5.1ms)
Completed 200 OK in 17ms (Views: 10.8ms | ActiveRecord: 0.0ms)
```

Puma responds as expected without compression:

```
$ curl -s http://localhost:3000 | tail -8
      <p class="version">
        <strong>Rails version:</strong> 5.2.1.1<br />
        <strong>Ruby version:</strong> 2.6.0 (x86_64-linux)
      </p>
    </section>
  </div>
</body>
</html>
```

Associated revisions

Revision 9d74d402e15008c10f80e67595cc861c89a1636b - 11/29/2018 08:00 PM - Eric Wong

disable non-blocking pipes and sockets by default

There seems to be a compatibility problems with Rails +
Rack::Deflater; so we revert this incompatibility.

This effectively reverts r65922; but keeps the bugfixes to
better support non-blocking sockets and pipes for future use.

[Bug #15356] [Bug #14968]

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@66093 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision 9d74d402e15008c10f80e67595cc861c89a1636b - 11/29/2018 08:00 PM - Eric Wong

disable non-blocking pipes and sockets by default

There seems to be a compatibility problems with Rails +
Rack::Deflater; so we revert this incompatibility.

This effectively reverts r65922; but keeps the bugfixes to
better support non-blocking sockets and pipes for future use.

[Bug #15356] [Bug #14968]

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@66093 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision 9d74d402 - 11/29/2018 08:00 PM - Eric Wong

disable non-blocking pipes and sockets by default

There seems to be a compatibility problems with Rails +
Rack::Deflater; so we revert this incompatibility.

This effectively reverts r65922; but keeps the bugfixes to

better support non-blocking sockets and pipes for future use.

[Bug #15356] [Bug #14968]

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@66093 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision cc9b57a2b4b6ddcf52502b3b1e827fcc0173ff7a - 12/07/2018 07:09 AM - Eric Wong

zlib (gzfile_write_raw): do not resize string after .write call

Apparently, a component of Rails implements a buffering .write
method which keeps the String buffer around and makes it unsafe
for us to clear it after calling .write.

This caused Rack::Deflater to give empty results when enabled.

Fortunately, per r61631 / a55abcc0ca6f628fc05304f81e5a044d65ab4a68,
this misguided optimization was only worth a small (0.5MB) savings
and we still benefit from the majority of the memory savings in
that change.

Thanks to zunda for the bug report.

[ruby-core:90133] [Bug #15356]

Fixes: r61631 (commit a55abcc0ca6f628fc05304f81e5a044d65ab4a68)
("zlib: reduce garbage on gzip writes (deflate)")

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@66268 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision cc9b57a2b4b6ddcf52502b3b1e827fcc0173ff7a - 12/07/2018 07:09 AM - Eric Wong

zlib (gzfile_write_raw): do not resize string after .write call

Apparently, a component of Rails implements a buffering .write
method which keeps the String buffer around and makes it unsafe
for us to clear it after calling .write.

This caused Rack::Deflater to give empty results when enabled.

Fortunately, per r61631 / a55abcc0ca6f628fc05304f81e5a044d65ab4a68,
this misguided optimization was only worth a small (0.5MB) savings
and we still benefit from the majority of the memory savings in
that change.

Thanks to zunda for the bug report.

[ruby-core:90133] [Bug #15356]

Fixes: r61631 (commit a55abcc0ca6f628fc05304f81e5a044d65ab4a68)
("zlib: reduce garbage on gzip writes (deflate)")

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@66268 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision cc9b57a2 - 12/07/2018 07:09 AM - Eric Wong

zlib (gzfile_write_raw): do not resize string after .write call

Apparently, a component of Rails implements a buffering .write
method which keeps the String buffer around and makes it unsafe
for us to clear it after calling .write.

This caused Rack::Deflater to give empty results when enabled.

Fortunately, per r61631 / a55abcc0ca6f628fc05304f81e5a044d65ab4a68,
this misguided optimization was only worth a small (0.5MB) savings
and we still benefit from the majority of the memory savings in
that change.

Thanks to zunda for the bug report.

[ruby-core:90133] [Bug #15356]

Fixes: r61631 (commit a55abcc0ca6f628fc05304f81e5a044d65ab4a68)
("zlib: reduce garbage on gzip writes (deflate)")

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@66268 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

History

#1 - 11/29/2018 08:46 AM - zunda (zunda an)

- Description updated

Clarify what I think of causing the problem.

#2 - 11/29/2018 10:42 AM - normalperson (Eric Wong)

zundan@gmail.com wrote:

> Clarify what I think of causing the problem.

> I'd like to report that a Rails app with Rack::Deflater
> enabled responds with a zero-length body on Ruby 2.6.0, to a
> request with Accept-encoding: gzip request header. It seems
> to happen at least on x86_64-linux (Ubuntu 18.04.1; see
> below for procedure for a reproduction) as well as on
> universal.x86_64-darwin17 (macOS High Sierra 10.13.6; tested
> about a half year ago).

So you're saying this is not a new problem...

> I suspect some kind of change in
> behavior of pipe or named pipe is causing this.

...Because the only recent change for pipe was enabling
non-blocking by default on *nix; and that only
happened about a week ago on r65922 [Bug #14968]

Anyways, you can try using blocking pipes (one-line change
below), but I suspect the problem is some bad interaction with
Rails and Rack::Deflater (Rack::Deflater does some weird stuff
with block captures)

```
diff --git a/io.c b/io.c
index d59bde93cf..e87e507540 100644
--- a/io.c
+++ b/io.c
@@ -138,7 +138,7 @@ off_t __syscall(quad_t number, ...);
 #if defined(_WIN32)
 #  define RUBY_PIPE_NONBLOCK_DEFAULT    (0)
 #elif defined(O_NONBLOCK)
-#  define RUBY_PIPE_NONBLOCK_DEFAULT    (O_NONBLOCK)
+#  define RUBY_PIPE_NONBLOCK_DEFAULT    (0)
 #else /* any platforms where O_NONBLOCK does not exist? */
 #  define RUBY_PIPE_NONBLOCK_DEFAULT    (0)
 #endif
```

Fwiw, I don't know Rails well, but I use Rack::Deflater
a bunch with Rack (not-Rails) apps on Ruby trunk.

#3 - 11/29/2018 06:40 PM - zunda (zunda an)

Thanks! The patch actually fixed the problem.

With 2.6.0-dev pointing to the binary built as below, the command curl -H 'Accept-encoding: gzip' -v http://localhost:3000 --raw prints the HTML data as expected.

```
$ cat io.patch
diff --git a/io.c b/io.c
index d59bde93cf..e87e507540 100644
--- a/io.c
+++ b/io.c
@@ -138,7 +138,7 @@ off_t __syscall(quad_t number, ...);
 #if defined(_WIN32)
 #  define RUBY_PIPE_NONBLOCK_DEFAULT    (0)
 #elif defined(O_NONBLOCK)
-#  define RUBY_PIPE_NONBLOCK_DEFAULT    (O_NONBLOCK)
```

```
+#  define RUBY_PIPE_NONBLOCK_DEFAULT    (0)
 #else /* any platforms where O_NONBLOCK does not exist? */
 #  define RUBY_PIPE_NONBLOCK_DEFAULT    (0)
 #endif
```

```
$ mv ~/.rbenv/versions/2.6.0-dev ~/.rbenv/versions/2.6.0-dev-vanila
$ rbenv install --patch 2.6.0-dev < io.patch
Cloning https://github.com/ruby/ruby.git...
Installing ruby-trunk...
patching file io.c
$ ruby --version
ruby 2.6.0dev (2018-11-29 trunk 66092) [x86_64-linux]
```

> So you're saying this is not a new problem...

First time I noticed this was running a Rails app on 2.6.0-preview1 on Heroku. Since then, I've been trying to see what was going on from time to time but I couldn't figure out how Rails and Rack::Deflater interacts with each other as you wrote. Sorry being slow in reporting it here. I might also have to note that the I do NOT see the problem on 2.5.3.

> I use Rack::Deflater a bunch with Rack (not-Rails) apps on Ruby trunk.

Right. I couldn't reproduce the problem on non-Rails Rack apps.

#4 - 11/29/2018 07:22 PM - normalperson (Eric Wong)

zundan@gmail.com wrote:

> Thanks! The patch actually fixed the problem.

OK, that is surprising...

> > So you're saying this is not a new problem...
>
> First time I noticed this was running a Rails app on 2.6.0-preview1 on Heroku. Since then, I've been trying to see what was going on from time to time but I couldn't figure out how Rails and Rack::Deflater interacts with each other as you wrote. Sorry being slow in reporting it here. I might also have to note that the I do NOT see the problem on 2.5.3.

So 2.6.0-preview1 was back in February; and the
non-blocking-by-default change wasn't even proposed, yet.
So yes, I'm not confident there's another bug, here.

Anyways, testing to revert nonblocking-by-default:

https://80x24.org/spew/20181129190533.13377-1-e@80x24.org/raw

And will probably commit once I iron out the tests.

#5 - 11/29/2018 08:00 PM - normalperson (Eric Wong)

- Status changed from Open to Closed

Applied in changeset trunk|r66093.

---

disable non-blocking pipes and sockets by default

There seems to be a compatibility problems with Rails +
Rack::Deflater; so we revert this incompatibility.

This effectively reverts r65922; but keeps the bugfixes to
better support non-blocking sockets and pipes for future use.

[Bug #15356] [Bug #14968]

#6 - 12/01/2018 08:42 PM - normalperson (Eric Wong)

> https://bugs.ruby-lang.org/issues/15356

Btw, I commited as r66093 (git 9d74d402)
Can you confirm current trunk is good for you?

Thanks

#7 - 12/07/2018 01:07 AM - zunda (zunda an)

Sorry for the delay in getting back to this. Just for your information, unfortunately, I noticed this problem coming back on ruby 2.6.0rc1 (2018-12-06 trunk 66253) [x86_64-linux]. Below is an example Rails app:
https://github.com/zunda/mastodon/tree/ce66a61796921449beddd14464698749b81b2217 deployed to Heroku.

$ curl https://zundan-mastodon-staging-pr-4.herokuapp.com/about

shows the expected HTML code but

$ curl -H 'Accept-encoding: gzip' https://zundan-mastodon-staging-pr-4.herokuapp.com/about

shows nothing.

I'll try to make time to dig this further but I'm not sure how quickly I'd be able to do so.

#8 - 12/07/2018 07:22 AM - normalperson (Eric Wong)

OK, I'm pretty sure r66268 fixes it. It's still my fault, not
because of non-blocking I/O, but because of a misguided zlib
optimization, so I'm still a moron :<

Fortunately the damage was limited and the majority of the zlib
memory optimization that went into r61631
(commit a55abcc0ca6f628fc05304f81e5a044d65ab4a68) was preserved.

#9 - 12/07/2018 11:49 AM - dm1try (Dmitry Dedov)

normalperson (Eric Wong) wrote:

> OK, I'm pretty sure r66268 fixes it. It's still my fault, not
> because of non-blocking I/O, but because of a misguided zlib
> optimization, so I'm still a moron :<

though this patch leads to described behavior, it looks like the actual problem can be solved by putting Rack::Deflater to the proper place in middleware stack:

```
config.middleware.insert_after ActionDispatch::Static, Rack::Deflater
```

as the position on the bottom of the stack anyway leads to the wrong behavior(ex. different ETag is calculated for the same body on each request because ETag middleware is called after Deflater and Deflater uses timestamps for packing)

#10 - 12/07/2018 01:33 PM - normalperson (Eric Wong)

me@dmitry.it wrote:

> though this patch leads to described behavior, it looks like
> the actual problem can be solved by putting Rack::Deflater
> to the proper place in middleware stack:

Asking users to change app config to migrate from 2.5 to 2.6 is
not acceptable to me. For every person who takes time to report
an incompatibility, there'll be more who move away from Ruby,
instead.

#11 - 12/07/2018 08:57 PM - zunda (zunda an)

Thanks so much for both of you! I've finally dug into this a little bit further.

First, I could confirm that the problem I first reported here for the simple Rails app with just Rack::Deflater added is fixed on ruby 2.6.0dev (2018-12-08 trunk 66278) [x86_64-linux]. On rc1 (ruby 2.6.0rc1 (2018-12-06 trunk 66253) [x86_64-linux]), Rack::Deflater needs to be moved to the proper place like below for the client to receive gzipped response as expected.

Second, the sample app (Mastodon) runs as expected with moving Rack::Deflater to the proper place on Heroku with ruby 2.6.0rc1 (2018-12-06 trunk 66253) [x86_64-linux].