

Redmine - Defect #33752

Uploading a big file fails with NoMemoryError

2020-07-20 08:59 - Karel Pičman

Status:	Closed	Start date:	
Priority:	Normal	Due date:	
Assignee:	Go MAEDA	% Done:	0%
Category:	Attachments	Estimated time:	0.00 hour
Target version:	4.1.4	Affected version:	
Resolution:			
Description Uploading of a file bigger than available RAM fails with <i>no memory</i> error. I've found the reason in <i>request.raw_post</i> which is <i>String</i> and doesn't respond to <i>:read</i> . Consequently, the whole file is read into memory. The attached patch simply replaces <i>raw_post</i> with <i>body</i> which is type of <i>StringIO</i> that provides <i>read</i> method.			
Related issues: Related to Redmine - Defect #35715: File upload fails when run with uWSGI Closed			

Associated revisions

Revision 20993 - 2021-05-11 07:35 - Go MAEDA

Uploading a big file fails with NoMemoryError (#33752).

Patch by Karel Pičman and Pavel Rosický.

Revision 20994 - 2021-05-12 04:34 - Go MAEDA

Merged r20993 from trunk to 4.2-stable (#33752).

Revision 20995 - 2021-05-12 04:36 - Go MAEDA

Merged r20993 from trunk to 4.1-stable (#33752).

History

#1 - 2020-07-20 18:11 - Go MAEDA

Redmine originally used *request.body*, but changed to use *request.raw_post* in [r9652](#) in order to fix an error regarding fastcgi ([#10832](#)). The attached *big_files_upload.diff* reverts [r9652](#).

I think the patch can be merged to the core if the patch works fine with fastcgi.

#2 - 2020-07-21 00:39 - Pavel Rosický

- *File attachments_test.rb.patch* added

Hi @Go MAEDA,
in my opinion, the fix in [#10832](#) is wrong. It breaks the concept of streaming file uploads
<https://github.com/redmine/redmine/blob/master/app/models/attachment.rb#L126>

however, the proposed patch breaks FCGI. I've attached a test case.

unlike other request handlers, CGI uses a rewindable input, that doesn't support the *#size* method. In theory, the only way how to determine the file size is to read the content first. The content should be streamed into a tempfile not into a memory.
https://github.com/rack/rack/blob/master/lib/rack/rewindable_input.rb

the second option is to rely on the CONTENT-LENGTH header, but it might not be accurate or it could be faked.

after some investigation, I found that Rails actually relies on the header
https://github.com/rails/rails/blob/master/actionpack/lib/action_dispatch/http/request.rb#L327

this means that FCGI is currently broken if an invalid CONTENT-LENGTH is provided. Note that most web-browsers and curl do send this header by default.

I'm wondering why anyone wants to use FCGI these days, they're definitely better options. We should try to fix it if possible, but the original patch [#10832](#) broke other websevers that behave correctly.

#3 - 2020-07-21 09:42 - Karel Pičman

- File `file_size.diff` added

I think that the problem with the size can be solved as suggested by Pavel by getting the size after data are stored into the filesystem. See the patch.

#4 - 2020-07-27 16:11 - Go MAEDA

- Target version set to Candidate for next minor release

#5 - 2020-08-19 00:30 - Go MAEDA

A test is broken after applying [big_files_upload.diff](#) and [file_size.diff](#).

```
Failure:
Redmine::ApiTest::AttachmentsTest#test_POST_/uploads.xml_should_return_errors_if_file_is_too_big [test/integration/api_test/attachments_test.rb:207]:
Expected response to be a <422: Unprocessable Entity>, but was a <201: Created>
Response body: <?xml version="1.0" encoding="UTF-8"?><upload><id>24</id><token>24.1d1801f753ccd9fa57966c46f360585caf83337a394a5f238d4e4e7d6005788d</token></upload>.
Expected: 422
Actual: 201
```

```
bin/rails test test/integration/api_test/attachments_test.rb:204
```

#6 - 2020-08-24 12:27 - Pavel Rosický

- File `10-patches.rb.patch` added

there're validations for size before the file is actually uploaded.

let's choose a simpler approach. These patches should be committed:

[big_files_upload.diff](#)
[10-patches.rb.patch](#)
[attachments_test.rb.patch](#)

it passes all tests.

#7 - 2021-04-03 18:56 - Pavel Rosický

@Go MAEDA may I ask for a review? it's a simple change.

#8 - 2021-04-07 06:15 - Go MAEDA

- Target version changed from Candidate for next minor release to 5.0.0

#9 - 2021-04-20 07:07 - Jens Krämer

+1, that may really help installations on VPS or other small-memory machines. The rack patch for FCGI support has been merged already (not released unfortunately, appears it'll be in Rack 3.0.0)

#10 - 2021-05-08 09:58 - Go MAEDA

- File `33752.patch` added

- Subject changed from Uploading big files to Uploading a big file fails with NoMemoryError

- Target version changed from 5.0.0 to 4.1.4

Pavel Rosický wrote:

let's choose a simpler approach. These patches should be committed:

[big_files_upload.diff](#)
[10-patches.rb.patch](#)
[attachments_test.rb.patch](#)

Thank you. I have merged the three patches and fixed RuboCop offenses: [33752.patch](#)

#11 - 2021-05-11 07:35 - Go MAEDA

- Status changed from New to Resolved

- Assignee set to Go MAEDA

Committed the patch. Thank you.

#12 - 2021-05-12 04:36 - Go MAEDA

- *Status changed from Resolved to Closed*

#13 - 2021-05-31 06:49 - Go MAEDA

- *Tracker changed from Patch to Defect*

#14 - 2021-08-11 21:30 - Holger Just

- *Related to Defect #35715: File upload fails when run with uWSGI added*

Files

big_files_upload.diff	594 Bytes	2020-07-20	Karel Pičman
attachments_test.rb.patch	1.22 KB	2020-07-20	Pavel Rosický
file_size.diff	1.16 KB	2020-07-21	Karel Pičman
10-patches.rb.patch	621 Bytes	2020-08-24	Pavel Rosický
33752.patch	2.36 KB	2021-05-08	Go MAEDA