An Experiment with Improved File Upload Handling in CF 8

Posted At : September 12, 2007 1:42 PM | Posted By : Steven Erat
Related Categories: Java, Adobe, ColdFusion

ColdFusion 8 Application Server offers an important improvement regarding memory utilization during the uploading of large files via the CFFILE tag. This entry will offer an experimental observation to demonstrate the improvement in CF 8, but first I'll expand upon each of the related settings and provide some usage scenarios.

Request Throttle Settings
The ColdFusion 7.01 Administrator introduced new file upload settings to better control file uploads. The broadest setting is called Request Throttle Memory, with a default of 200MB, where its purpose is to regulate the cumulative impact of all concurrent large file uploads on the ColdFusion server. The adjacent setting for Request Throttle Threshold, default of 4MB, is the minimum size of file uploads for which the throttle should regulate, anything below the threshold is ignored by the throttle. Further up on the Settings page is also the new field Maximum Size of Post Data, with a default of 100MB.

These default settings for ColdFusion 7.0x would throttle all concurrent file uploads exceeding 4MB each such that a single request attempting to upload more than 100MB of multipart/form-data would be terminated by a PostSizeLimitExceededException (written to the J2EE server's out or event log).

```
error Post Size exceeds the maximum limit.
coldfusion.filter.RequestThrottleFilter$PostSizeLimitExceededException:
 Post Size exceeds the maximum limit.
at coldfusion.filter.RequestThrottleFilter.invoke (RequestThrottleFilter.java:102)
at coldfusion.CfmServlet.service (CfmServlet.java:107)
```

Moreover, if the total cumulative size of all file uploads were close to but less than the default 200MB Request Throttle Memory maximum limit, then the next file upload to run concurrently that attempts to exceed the max throttle limit for all requests would be queued until enough of the earlier file uploads completed.

In the odd case where the Request Throttle Memory value were set lower than the Maximum Size of Post Data limit, then a single request attempting to post more than the Request Throttle Memory would terminate with a MemoryUnavailableException (written to the browser and to the underlying J2EE server's out or event logs).

```
error Memory required (104856464 bytes) exceeds the maximum allowed memory.
coldfusion.util.MemorySemaphore$MemoryUnavailableException:
```

```
 Memory required (104856464 bytes) exceeds the maximum allowed memory.
at coldfusion.util.MemorySemaphore.acquire (MemorySemaphore.java:75)
at coldfusion.filter.RequestThrottleFilter.invoke (RequestThrottleFilter.java:112)
at coldfusion.CfmServlet.service (CfmServlet.java:107)
```

In ColdFusion MX servers 7.0x and earlier, the server would read multipart/form-data content into memory during the upload process. The JVM heap usage would grow in proportion to the content length of all concurrent file uploads, a scenario which could easily drive ColdFusion to begin logging the java.lang.OutOfMemoryError exception and sending 500 Null responses back to the browser. It should be noted, however, that when content length of the file upload exceeds the Maximum Size of Post Data or Request Throttle Memory size that the data is not read into memory and ColdFusion MX 6/7 memory utilization is not affected.

Improved Memory Utilization in CF8
ColdFusion 8 has dramatically improved the performance of large file uploads in this regard. No longer will the JVM's heap grow significantly during file uploads, and in my observations there was hardly a change in memory usage even during multiple large file uploads with CF8. Of minor note, all the related settings are now grouped together on the Settings page under the heading Request Size Limits.

Ben Forta **recently blogged about this** to note that the improvement was due to a change in the GetHTTPRequestData() function, so if you care at all about the "content" key of the result structure from that function, then you should read his entry.

A File Upload Experiment
To test this change I created a file upload form and generated 100MB text file to exercise it. I used ColdFusion MX 7.02 and ColdFusion 8 on the same Windows XP machine having the same relevant settings:

- **Settings**
- 512m Max Heap Size
- Maximum Size of Post Data - 100 MB
- Request Throttle Threshold - 4 MB
- Request Throttle Memory - 400 MB

On another client machine I opened the file upload form in 5 tabs within Firefox and prepared each form to upload the 100 MB text file. Then I started the first test by running only CFMX 7.02 while noting the Task Manager columns for Mem Usage and VM Size. I then quickly hit submit on each of the 5 file upload forms.

Assuming sufficient heap space in CF7, I expected that 4 uploads would begin (100 MB * 4 uploads

- **CFMX 7.02 Results**

- Mem Usage and VM Size before test: 65 MB, 78 MB
- Mem Usage and VM Size after test: 433 MB, 575 MB
- Mem Usage and VM Size 5 minutes after test: 5 MB, 470 MB
- Mem Usage and VM Size 15 minutes after test: 6 MB, 69 MB
- Total Successful File Uploads: 4 out of 5
- Errors: java.lang.OutOfMemoryError occured once
- Other: a partial file upload of 65 MB in side was found in the temporary directory SERVER-INF empwwwroot-tmp

Next, ColdFusion 7 was stopped and CF 8 was started. The same test of 5 concurrent uploads was performed.

- **ColdFusion 8 Results**
- Mem Usage and VM Size before test: 24 MB, 276 MB
- Mem Usage and VM Size after test: 41 MB, 276 MB
- Mem Usage and VM Size 15 minutes after test: 5 MB, 275 MB
- Total Successful File Uploads: 5 out of 5
- Errors: none

(The initial CF 8 memory footprint was higher than that of CF 7 because the CF 8 server was configured with a large test suite that was not being used, although the CF 8 server did have to load a lot of dependencies upon start up causing the higher footprint. The CF 7 server was a base installation without further configuration.)

Summary
The example above demonstrates that ColdFusion 8 was able to successfully handle 5 concurrent file uploads of 100 MB each with only slight increase in memory usage. This is compared to CFMX 7.02 which quickly consumed the maximum amount of memory and even with file upload request queuing it was not able to complete all 5 file uploads.

Server Monitor
Memory usage for file uploads can also be observed with the new CF 8 Server Monitor. Without enabling any of the three monitor types, the panel Statistics > Request Statistics > Request Memory Throttle will automatically collect data.

: **Related Information**

- **Uploading Large Files To ColdFusion 8**, Ben Forta
- **Limiting the Size of a POST Request**, Mark Kruger
- **ColdFusion 8 Documentation**