Dynamic Compression on ColdFusion 9 and IIS7

Posted At : September 24, 2010 4:32 PM | Posted By : Mark Kruger
Related Categories: Coldfusion Optimization, Hosting and Networking

Maybe your already know that web servers can compress outgoing content. Compressed content arrives at the browser which decompresses it and is able to render it. This is all generally seamless to the user and results in a more effective use of bandwidth. Now, compressing static files (like .html files) is a no brainer for web servers. They simply pre-compress the files and store them in a file cache somewhere. When the original file is called for the web server serves up the compressed file instead.

Dynamic files are more problematic. There's no correlation between the file name and the buffered output of a ColdFusion page for example. Consider search results. One user might receive 10 results and another user might receive 10 completely different results. Still another user might receive 100 results. How is the *web server* supposed to compress that data? Like your app server it does it "on the fly". It waits for ColdFusion to return the response buffer, compresses the file *in memory* (as I understand it) and then outputs the buffer to the browser. At least that's the way it works in theory. In practice you might find that ColdFusion 9 and IIS 7 don't quite have this figured out yet.

Before I give you the blow-by-blow (and thankfully a solution) I want to make it clear that this problem and solution come to me by way of my good friend and colleague Vlad Friedman of **Edgeweb Hosting**. EdgeWeb consistently receives the highest possible reviews from its customers and Vlad is one of the brightest folks I know in our corner of the IT world. Now let's talk about our little problem shall we?

The Issue: IIS Dynamic Compression Won't "Take"

On the surface this seems pretty easy. The MS **technote** indicates this is a matter of simply going into the conveniently (and unnecessarily) redesigned IIS manager, finding the "compression" icon in the features and selecting "enable dynamic content compression". What could be easier right? As an experiment, go ahead and try it and see if it is successful (you can use this handy **Online Tool** to check your site). In *many* cases the content will still show as "uncompressed". IIS just refuses to compress the output buffer.

Now if you were to stumble onto this problem on your own without the Muse (or Vlad) to guide you, you would undoubtedly spend many fruitless hours adding and removing various modules to try and get compression working. Fortunately, Vlad (and presumably a 6 pack of Red Bull) was already up all night figuring it out for you and he's authorized the Muse to give you the solution forthwith (that means right away).

The Fix

It turns out that IIS cares about *the order in which the various modules are loaded*. Vlad discovered this by finding a Win2008 IIS7 server that was correctly compressing dynamic content and comparing the config files with the problem server. What he found was a curiosity. In the file "applicationhost.config" in the /windows/sytem32/inetserv/config folder he found the following XML on the "known good" server:

```
<modules>
<add name="HttpCacheModule" lockItem="true" />
<add name="DefaultDocumentModule" lockItem="true" />
<add name="DirectoryListingModule" lockItem="true" />
<add name="IsapiFilterModule" lockItem="true" />
<add name="ProtocolSupportModule" lockItem="true" />
<add name="DynamicCompressionModule" lockItem="true" />
<add name="StaticCompressionModule" lockItem="true" />
<add name="HttpRedirectionModule" lockItem="true" />
     ....
</modules>
```

On the "bad" server it looked like this:

```
<modules>
<add name="HttpCacheModule" lockItem="true" />
<add name="DynamicCompressionModule" lockItem="true" />
<add name="StaticCompressionModule" lockItem="true" />
<add name="DefaultDocumentModule" lockItem="true" />
<add name="DirectoryListingModule" lockItem="true" />
<add name="IsapiFilterModule" lockItem="true" />
<add name="ProtocolSupportModule" lockItem="true" />
<add name="HttpRedirectionModule" lockItem="true" />
    ...
</modules>
```

See any difference?

It turns out that the ***order in which these modules are loaded*** makes a difference in how compression is supported for dynamic content. All Vlad had to do to "fix" his ailing server was to move the lines containing:

```
<add name="DynamicCompressionModule" lockItem="true" />
<add name="StaticCompressionModule" lockItem="true" />
```

...below the line containing:

```
<add name="ProtocolSupportModule" lockItem="true" />
```

He then restarted IIS and his compression problem was solved.

Steps in Order

For those of you who need a list of steps here it is in order:

1. Find the file "applicationhost.config" located at c:\windows\system32\inetserv\config and open for editing (make sure and back it up first!).
2. Search for the string *DynamicCompressionModule* - you should find the staticCompression Module immediately adjacent to it
3. Move the two lines:

   ```
   <add name="DynamicCompressionModule" lockItem="true" />
        <add name="StaticCompressionModule" lockItem="true" />
   ```

   Below the line

```
<add name="ProtocolSupportModule" lockItem="true" />
```

4. Save the file, restart IIS and try your test again.

All you IIS gurus out there let me know any tips on the issue. The Muse is always happy to add to the collective hive knowledge.