Too many files causing RAM pressure? (a.k.a. DynCache to the rescue!)

Arvind Shyamsundar 20 Mar 2014 5:28 AM 0

Readers of this blog might recall a previous post which described one impact of having too many similarly named files in one folder in NTFS. It turns out that the 8.3 naming convention is not the only thing you need to worry about when you have very large amounts of (smaller) files in the same volume.

Today I was called in to assist with a performance issue on a server. The only visible symptom of the problem was excessive RAM utilization on the server. The interesting aspect was that no specific user-mode process was consuming that RAM, so we were wondering where it came from. If this server was running SQL Server, for example, the 'ghost' utilization could be due to locked pages (which do not show up in Task Manager) but that was not the case here.

Analysis

So we ran the RAMMap utility, we found that the usage for **MetaFile** was a substantial percentage of the total RAM usage. From this AskPerf blog post you can see what MetaFile is all about:

> *"Metafile is part of the system cache and consists of NTFS metadata. NTFS metadata includes the MFT as well as the other various NTFS metadata files (see How NTFS Works for more details, and of course Windows Internals is a great reference). In the MFT each file attribute record takes 1k and each file has at least one attribute record. Add to this the other NTFS metadata files and you can see why the Metafile category can grow quite large on servers with lots of files."*

The next step therefore was to cross-check how large the MFT was in reality. The easy way to do this is to utilize the command given below:

> **fsutil fsinfo ntfsinfo <drive letter>**

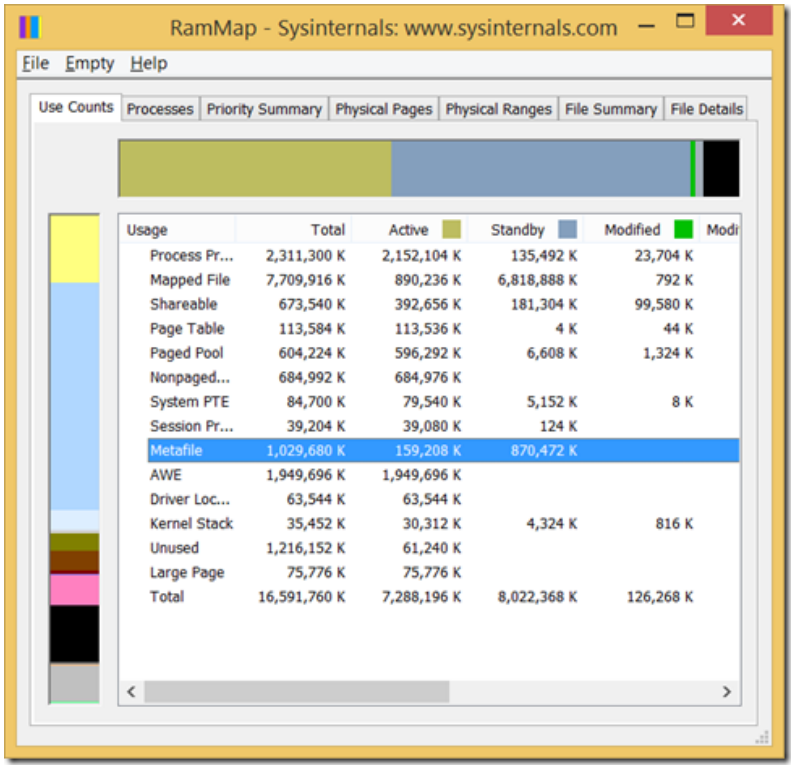
A sample output is given below (from my own laptop)

```
C:\>fsutil fsinfo ntfsinfo c:
NTFS Volume Serial Number :       0x8a40c9ee40c9e0d5
NTFS Version   :                  3.1
LFS Version    :                  2.0
Number Sectors :                  0x000000003a2007ff
Total Clusters :                  0x00000000074400ff
Free Clusters  :                  0x0000000000ab1f84
Total Reserved :                  0x000000000002ef10
Bytes Per Sector  :               512
Bytes Per Physical Sector :       4096
Bytes Per Cluster :               4096
Bytes Per FileRecord Segment    : 1024
Clusters Per FileRecord Segment : 0
Mft Valid Data Length :           0x0000000030240000
Mft Start Lcn  :                  0x00000000000c0000
Mft2 Start Lcn :                  0x0000000000000002
Mft Zone Start :                  0x0000000004d58da0
Mft Zone End   :                  0x0000000004d655c0
Resource Manager Identifier :     96CC88FE-5621-11E3-AF31-3C970EA47926
```

In this output, the "Mft Valid Data Length" gives us an indicator of how many bytes are used by the MFT. In the above case for example it equates to around 770MB:

> 0x0000000030240000 / (1024*1024) = 770.25 MB

Just as a curiosity, if I run RAMMap in my laptop (Windows 8.1), here is what I see, you can see a rough alignment with the above computed number.



FYI, in the server that I was looking at in the real world, the size of the MFT was actually 1.5 times the amount of RAM on the box

Mitigation

Since the server in question was running Windows Server 2008 R2 SP1, we recommended the usage of the DynCache service sample, which would automatically control the size of the system cache based on system memory notifications.

SQL Server FileStream

Practically, this issue would also apply when you use SQL FileStream to store a very large number of blobs in the NTFS file system. In such cases, here are my recommendations:

1. Consider setting the SQL Server Database Engine to use 'locked pages' with 'max server memory' set appropriately.
2. If the OS is Windows 2008 R2 or below, you may additionally consider using the DynCache service to mitigate the effect that the large MFT will have.

Other notes

If you anticipate huge numbers of files to be stored on the file system, keep in mind that each file record will take up around 1KB. That means 100 million files will take up close to 100GB worth of MFT storage!

On Windows 2008 R2 and below be aware of issues like the one described in KB article 967351 and install the updated version of NTFS.sys accordingly.

In extreme cases, if you want to achieve optimal performance without compromising on system cache memory utilization, be aware that the system RAM sizing must be done accordingly keeping in mind the very large MFT which might result.

References

You may want to read these articles for more information:

- You experience performance issues in applications and services when the system file cache consumes most of the physical RAM
- Too much cache?
- Windows 7 and Windows Server 2008 R2: Do you still need the Microsoft Windows Dynamic Cache Service?
- NTDebugging blog post about DynCache
- How NTFS Works: Local File Systems

Spread the word!

Comments