Preliminary notes on analyzing Disk and File I/O performance with ETW traces

devblogs.microsoft.com/oldnewthing/20201125-00

November 25, 2020

Raymond Chen

Event Tracing for Windows (ETW) is a powerful tool for blah blah blah. (Sometimes I get tired of writing the introduction.)

Here's a little diagram of how I/O happens. This diagram is not 100% accurate, but it'll do for the purpose of today's discussion.

Application.exe
↓
File System
↓
Minifilters
↓
Anti-malware
↓
luafv.sys
↓
fileinfo.sys★
↓
↓
Disk cache
↓
volsnap.sys
↓
fvevol.sys
↓
Volume Manager
↓
Partition Manager★
↓
Disk drivers

↓
Hard drive
When the application initiates an I/O, the request goes to the file system. The request then passes through various minifilters (some of which are shown here), and then enters the I/O subsystem.

It's possible that the request can be satisfied from the disk cache, in which case the I/O is completed immediately.

If we have a cache miss, then the I/O continues through some more drivers. Here, I've shown volsnap, the Volume Snapshot driver (we'll learn more about this later), and fvevol, which is where BitLocker encryption and decryption happens. (The letters FVE stand for Full Volume Encryption.)

The request eventually reaches the Volume Manager, the Partition Manager, the disk drivers, and finally hits the physical hard drive.

When the I/O completes, the results are returned upward through the diagram back to the application.

I put stars on fileinfo.sys and the Partition Manager because those are the components which are responsible for generating the `FILE_IO` and `DISK_IO` ETW events, respectively.

Already, we've learned a bunch of stuff:

- Time spent by anti-malware is not counted by either the `FILE_IO` or `DISK_IO` events, since it happens either before the logging components log the start of the I/O, or after they have logged the end of the I/O.
- Time spent in the file system, System Restore, and BitLocker are not counted by the `DISK_IO` events for the same reason.
- If an I/O is satisfied from the disk cache, then there will be no corresponding disk I/O event.
- If multiple file I/O operations are batched together, they will show up as only one disk I/O event.
- In general, file I/O events vastly out number disk I/O events.

In the disk I/O data, there is a column called *IO Type*. There are three I/O types at the disk level: Read, Write, and Flush. Read and Write are self-explanatory. Flush occurs when the operating system commits all pending data to the hard disk, including telling the hard drive to flush its own internal caches. Flushes are generally bad for performance because they cause everything to come to a halt until the flush operation is complete, but flushes are often necessary to ensure disk coherency.

Hard drive manufacturers are sneaky, and when the operating system tells them to flush, they often don't actually flush. They just say, "Yeah, I flushed the data," and make a note to themselves, "Y'know, I probably should flush the data at some point before anybody notices that I've been lying to them all this time." This can lead to strange things like a subsequent Read operation taking a very long time. It's not that the read itself was inherently slow. It's just that the read happened to occur while the drive was busy "paying back its flush debt", so it got stuck behind a physical flush operation.

One of the fun graphs to look at is the Disk Offset graph. It's under the Disk Usage category. This graph shows a dot for each I/O issued to the hard drive, with time on the *x*-axis and the disk offset (distance from start of the disk) on the *y*-axis. The dots are connected with lines, giving you a visualization of the movement of the disk head (assuming rotational media).

Sometimes you'll see what appears to be the disk head vibrating back and forth rapidly between two locations on the disk. That may not be what's actually happening. Hard drives often have more than one head, so what you could be seeing is one head doing work at one part of the disk, and another head doing work at a different part of the disk. The graph doesn't realize this, so it looks like there's a single poor disk head running back and forth between two spots on the disk.

On the other hand, if your hard drive has only one head, then it really is bouncing back and forth like crazy.

On the Disk Offset graph, flushes are marked with vertical red lines. You can see the points at which everything on the disk came to a stop.

Bruce Dawson has a nice picture of the disk offset graph. He also describes what each of the columns in the event log means, so I'll defer to his write-up. Points of interest include the difference between Disk Service Time and I/O Time.

Next time, I'll look at the mysterious *System* process and why it gets blamed for so much I/O.

Additional resources

- Defrag Tools Episode 44: WPT – Disk I/O Analysis. Particularly interesting is the discussion of QD/I and QD/C (queue depth at initiation and queued depth at complete), which give you a glimpse into how loaded the hard drive is.
- Defrag Tools Episode 45: WPT – File & Registry Analysis

Raymond Chen

Follow