Realtime publishers

Why Windows Defragmentation Isn't Enough

The Essentials Series

Better than Defragmentation: Fragmentation Prevention

sponsored by

Diskeeper corporation

Greg Shields

Better than Defragmentation: Fragmentation Prevention 1

Fragmentation Prevention: A Primer 1

Defragmentation Represents a Cost 2

Defragmentation…Without Defragmentation 3

Copyright Statement

Better than Defragmentation: Fragmentation Prevention

You know that fragmentation is a natural byproduct of Windows file system operations. You also know that Microsoft includes a defragmentation tool inside Windows to alleviate some of fragmentation's problems. With this knowledge comes unnerving questions: Wouldn't defragmentation itself impact performance, particularly as disk drives grow larger? Wouldn't the native tool's defragmentation pass approach be improved through a proactive alternative? Wouldn't Windows perform better if fragmentation never occurred in the first place?

These questions are all valid. Find their answers in this article's explanation of fragmentation prevention. Considered the logical next step in solving the fragmentation problem, fragmentation prevention represents a proactive approach to ensuring that files are always laid contiguously on the disk as they're written. With prevention, you eliminate the intrinsic limitations of the defragmentation pass, simply because you eliminate the pass altogether. The result is a computer system that is always running at peak performance because its files always exist without fragments.

Fragmentation Prevention: A Primer

Understanding how fragmentation prevention works is best begun by looking at why files fragment. File fragmentation occurs at the point a file must be written to disk. When that file is written, the Windows operating system (OS) looks for an available bit of space on which to write it. Fragmentation occurs when that bit of space isn't large enough to fit the whole file.

Figure 1 shows how this process occurs across two units of time from top to bottom. Similar to Figure 2 in the previous article, the white bars represent a disk drive, with the sectors on that drive going from left to right. At the first unit of time, File A has already been written to an available portion of the disk. As you can see, the disk is relatively empty at this point. File A has plenty of space in which to write itself; so as a result, it gets written without being fragmented.

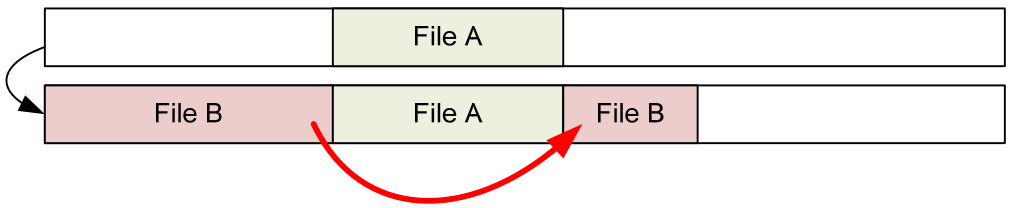

Figure 1: When File B must expand in size, it must fragment to the next available area of storage space.

The problem occurs when File B then needs to be written. Due to the position of File A, File B is forced to be splintered into two pieces: one that resides "before" File A with its second half residing "after" File A. This splintering of the file is considered fragmentation.

This example makes the problem appear exceptionally simple, but this scenario happens repeatedly and rapidly as a computer system operates. It happens as files are created, deleted, and modified. Considering the large number of files in the Windows OS plus the even larger number of temporary and other files that are constantly being created and deleted, it is easy to see how this process quickly gets out of hand.

Fragmentation prevention takes a much different approach to solving the problem than does defragmentation. Fragmentation prevention improves upon a process that Windows already employs to mitigate fragmentation. Windows itself temporarily places an amount of "padding" at the end of a file as it is being written or modified. This padding is placed on the disk as a small amount of extra space, which ideally would enable the file to grow without fragmenting.

Yet although this extra padding is a great solution when its size is carefully calculated, you can see how a miscalculation can create problems. Not providing enough padding means that growing files will be forced to fragment. Providing too much padding unnecessarily consumes disk space, until that unused temporary padding is released back to the file system for reuse.

Fragmentation prevention solutions expand upon Windows' native capabilities by adding intelligence to the calculation process. By watching and learning the write behavior of different applications, such a solution can make superior judgments about how much padding to set aside. Essentially, by using a smarter calculation, the ability to prevent fragmentation can be greatly enhanced.

Defragmentation Represents a Cost

Fragmentation prevention's proactive approach solves "the natural byproduct of Windows file system operations" by eliminating altogether the byproduct: file fragments. The outward simplicity of how it works highlights the invisible costs that might not have been calculated in using its alternative. Think for a minute about the costs your enterprise might be paying:

- **The cost of performance.** This cost has been highlighted throughout this Essentials Series. Preventing fragments from ever occurring eliminates the resource consumption required for defragmentation activities. In effect, fragmentation prevention gives back needy resources that would otherwise be consumed during important nighttime activities. This is particularly important for servers in the data center that are generally tasked with other resource-intensive activities outside normal work hours.

- **The cost of resource scheduling.** There is also a cost to the scheduling of fragmentation activities itself. Integrating fragmentation prevention with background scheduling ensures that its activities consume only unused system resources. This further reduces the impact of the just-in-time fragmentation prevention activity.
- **The cost of free space.** Full defragmentation with the native Windows tool normally requires 15% free space to run a full defragmentation pass. On systems with less than 15% of free space, the native tool will typically only run a "partial defragmentation." This limitation creates a cost by requiring servers and desktops to reserve that amount of free disk space, essentially wasting this disk space for more productive uses.
- **The cost of free space fragmentation.** Lastly is the hidden danger in defragmentation's hidden cost: free space fragmentation. Recall that the Windows file system will continue to create fragments as files are created, modified, and deleted. That constant fragment creation also creates fragments of free space, making future file writes even more fragmented. Fragmented free space forces more fragments, which limits performance and exacerbates the fragmentation problem.

Defragmentation…Without Defragmentation

It should be obvious at this point that fragmentation prevention presents a compelling alternative to defragmentation's limitations. With it, fragments never occur on a system, being handled as files are written. Fragmentation prevention also tackles four of the major costs associated with defragmentation: performance, resource scheduling, free space, and free space fragmentation. It accomplishes its task by eliminating the defragmentation process, ensuring that computer systems are always running with best performance.

However, all this being said, even the very best fragmentation prevention solution won't be a good fit for business enterprises without enterprise feature sets. Just what are those feature sets? What additional capabilities must a fragmentation prevention solution incorporate to provide best value for your organization? That's the topic of this series' third and final article.