Realtime publishers

Solving the Storage Challenge Across Platforms: Transparent Compression for Windows Operating Systems

sponsored by

Dan Sullivan

Copyright Statement

Compression Technologies: Distinguishing Factors

Data compression is an important and well-studied problem. One of the earliest compression algorithms, the Lempel-Ziv algorithm, was invented in the 1970s. These compression algorithms, and others like it, became increasingly important as use of the Internet grew. Compression technologies are now commonly used throughout IT deployments.

Providing Compression: Different Options

Today, there are a variety of compression algorithms available, and these are implemented in many forms. The best combination of compression algorithm and implementation model will depend on your requirements.

Compression Applications

Some users opt for compression programs that are run as desktop or command-line applications. These are useful, for example, when someone needs to send a large file using email or when someone wants to archive a large file without taking up too much space. These programs enable users to apply compression when it is needed on an ad hoc basis.

Operating System Compression

Another way to employ compression is by incorporating it in the operating system (OS). This method makes the compression and decompression operations transparent to the user. For example, a marketing professional might write a long document describing a new marketing strategy. The document might include images as well as text. Instead of requiring the user to compress the document each time it is saved and then decompressing each time it is opened, the OS could automatically compress and decompress the file.

An advantage of incorporating compression at the OS level is that it allows all programs that read and save data to take advantage of compression. Users do not need to remember to compress files each time they are saved. An OS-based approach means any application on the system, such as word processors, spreadsheets, and other productivity tools, can leverage the advantage of compression without having to implement it in each application.

Comparing Compression Solutions

Compression solutions offer different strengths and weaknesses. Depending on the compression technology, users will see differences in both efficiency and effectiveness. When considering compression technologies for the enterprise, it is important to understand the various distinguishing factors. Three factors to consider are:

- Compression efficiency and effectiveness
- Ease of use and deployment
- Implementation factors

Although the reasons enterprises require compression may vary, these three factors should be considered in virtually all cases.

Compression Efficiency and Effectiveness

The most obvious variable when looking at a compression solution is the input-to-compression ratio. Seeing how much a compression algorithm can shrink the size of data is an easy and effective way to judge the merit of a compression solution—although this is not the only factor to consider.

In general, the best algorithms for business data will shrink data without data loss. This scenario is known as lossless compression (see Figure 1). Business users should be able recreate the original data from a compressed version of their data. For example, if financial data is compressed, then every time it is read, it should be decompressed exactly as it was in the original data.

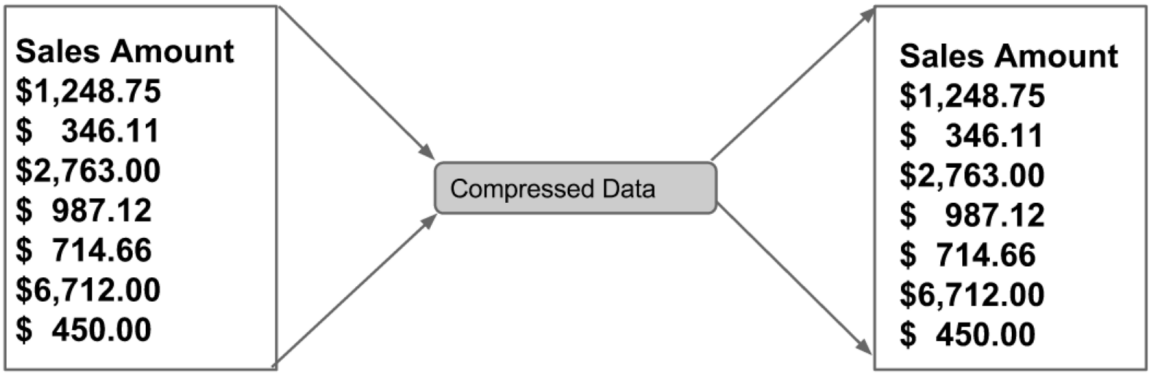

Figure 1: Lossless compression ensures that after data is compressed, it can be restored to its exact, original form.

In contrast, some compression technologies will shrink data more but lose pieces of the original data; this type is also known as lossy compression. Lossy compression is best suited for media-centric data, such as image and video compression. In these cases, some data loss is acceptable. However, business data does not have this luxury. Every piece of business data is crucial to the health of the data set as a whole; as such, lossless compression will always be the best option.

Ease of Use and Deployment

Using compression should not put unnecessary burdens on users or systems administrators. Users, for example, should not have to take unusual steps to take advantage of compression. It should be essentially transparent from a user's perspective. The only difference users should notice is that they have more free storage than they might otherwise expect.

Similarly, compression systems should operate with minimal systems administration after deployment. Ideally, deployment will be fairly painless, too. Compression tools that may be installed using software distribution tools will further reduce the workload on administrators.

Implementation Factors

The effectiveness of disk read operations is close behind input-to-compression ratio in terms of importance. Disk read speeds are always accelerated when compression is used, even on very fast SSD drives and slower mechanical drives. However, when compressing data to a disk, there will always be a certain amount of overhead with write operations.

The more data read with a single disk seek operation, the better. This indicates the compression technology is working efficiently by not wasting valuable time and computing power. Specifically, it keeps the CPU busy when data is readily available instead of having to wait for read operations to complete.

A high-quality compression algorithm should be able to complete operations quickly as well as effectively. Business operations require speed, and compression technology should be no different. The speed of decompression operations is also important. Ultimately, compression effectiveness hinges on both how it treats data and how the data moves from a compressed to a decompressed state.

In addition to how a compression algorithm works on your data, it is important to look at how the solution handles usability and deployment. A high level of transparency relative to users and business applications is vital to a high-quality compression solution. User applications should function using compressed data without change, and any compressed data should be transparent to the application itself. Having to reconfigure application user access settings significantly reduces the value of a compression solution.

Compression should also be part of OS operations, not application operations. For example, when opening a compressed text file in a word processor, such as Microsoft Word, the software should not need to perform the decompression or need to interact with the compressed file version at all. Compression interactions should occur using the underlying OS. This quality makes room for applications that do not have the ability to interact with compressed data.

Similarly, the best compression solution should have support for common user operations, including content copying and pasting. These functions might seem basic, but they are vital nonetheless. In addition, the solution should support file and directory manipulation.

Finally, the ability to implement a compression solution involves several factors, including multithreading, downgrade and upgrade compression, benchmarking, and 32-bit or 64-bit support. Multithreading is simply how many compression operations can be performed in parallel. High multithreading capabilities significantly increase the value of a compression solution, as long as it doesn't sacrifice compression performance.

Benchmarking Compression

Vendors often provide benchmarking metrics to help potential customers understand the performance characteristics of an application. This setup is the same for compression tools. However, you should consider performing your own benchmarks. There are two reasons to do so.

Although valid, the benchmarks provided by a vendor may not be representative of the type of data you will compress. If benchmarks are performed on database files that contain mostly numeric data and your data is primarily text documents, you might find different compression ratios or time to compress.

Also, implementations may vary. A benchmark performed on a 64-bit OS on a server with several cores, SSDs, and a large amount of memory will likely have different performance characteristics than a server running a 32-bit OS, little RAM, and older disk drives. Performing your own benchmarks will help ensure the results are relevant to your particular needs.

Different Algorithms Mean Different Results

Different compression algorithms and their implementations will provide different performance characteristics. Some may compress quickly but not compress file sizes as well as another implementation that compresses slowly and yields significantly smaller files. Compression should be seamless for end users and demand little in the way of systems administration. Also, consider how well compression implementation takes advantage of your device resources by leveraging multi-threading and other performance-improving techniques. Finally, consider performing your own benchmarks to ensure the results reflect what you can expect with your use cases.