F5 White Paper

Understanding Advanced Data Compression

Nearly all WAN optimization appliances store and use previously transferred network data to achieve high compression ratios, while leveraging advanced compression routines to improve application performance. How they achieve these gains, and the limitations of certain routines, vary widely and can significantly affect the improvements and benefits associated with WAN application delivery services.

by Lori MacVittie
Senior Technical Marketing Manager, Application Services

Contents

Introduction

The increasingly distributed nature of users and the prevalence of teleworkers, coupled with emerging application deployment models that leverage external cloud computing, introduce additional stress on existing network connections in the form of more data being exchanged more often. Employee productivity can be dramatically affected by slow networks that result in poorly performing applications. Business continuity plans—no matter how carefully thought out and implemented—can go awry when backups fail to complete, take more time than expected, and cause some applications to go unprotected.

Organizations have turned to WAN optimization to combat the challenges of assuring application performance and help ensure timely transfer of large data sets across constrained network links. Many WAN optimization solutions are focused wholly on network-layer optimizations and operate based on rigid configurations. Not only are these solutions inflexible, but they also fail to include optimizations that can further enhance the performance of applications commonly delivered over WAN links.

> "... within 12 months roughly half of [168 enterprise IT professionals who were surveyed] will be using WAN optimization technology to help them to successfully deliver applications to branch offices. The technologies that they will use include techniques such as compression, deduplication, caching, quality of service, and protocol acceleration."
>
> Source: "Keys to Unlocking IT Value Through WAN Optimization," Dr. Jim Metzler

Implementation Approaches

Packets versus Sessions

To date, most network compression systems have been packet-based. Packet-based compression systems buffer packets destined for a remote network with a decompressor. These packets are compressed either one at a time or as a group and then sent to the decompressor where the process is reversed (see Figure 1). Packet-based compression has been available for many years and can be found in routers and VPN clients.

Packet-based compression systems have additional problems. When compressing packets, these systems must choose between writing small packets to the network and performing additional work to aggregate and encapsulate multiple packets. Neither option produces optimal results. Writing small packets to the network increases TCP/IP header overhead, while aggregating and encapsulating packets adds encapsulation headers to the stream.

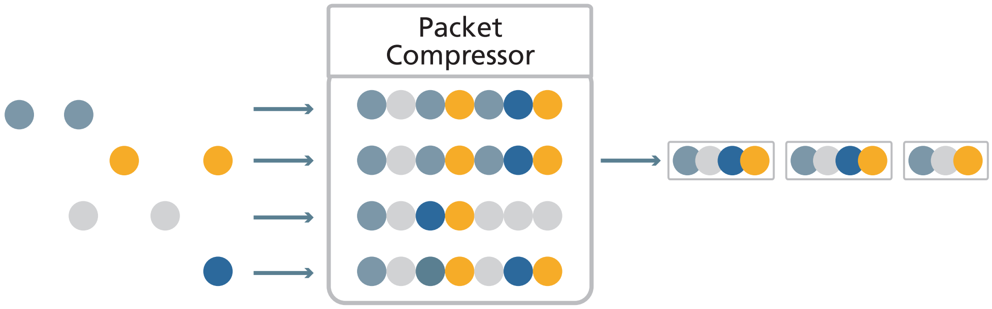

Figure 1: Packet-based compression

Unlike previous compression solutions, F5® BIG-IP® Local Traffic Manager™ (LTM) and BIG-IP® Application Acceleration Manager™ (AAM) operates at the session layer (Figure 2). This enables BIG-IP AAM to apply compression across a completely homogenous data set while addressing all application types, resulting in higher compression ratios than comparable packet-based systems.

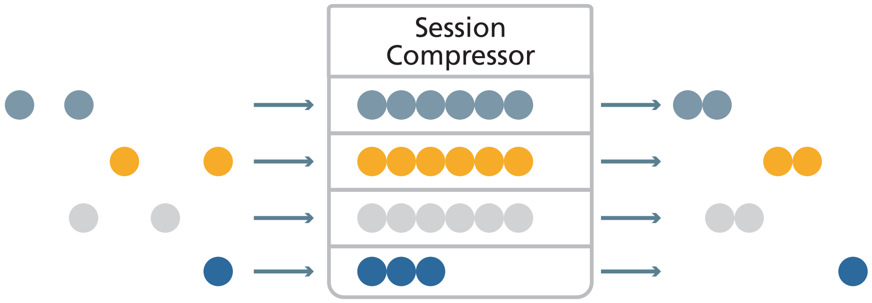

Figure 2: Session-based compression

Furthermore, by operating at the session layer, packet boundary and repacketization problems are eliminated. Session layer compression enables a BIG-IP AAM-enabled BIG-IP LTM device to easily find matches in data streams that at Layer 3 might be many bytes apart, but at Layer 5 are contiguous. System throughput is also increased when compression is performed at the session layer through the elimination of the encapsulation stage.

Dictionary Size

One limitation all compression routines have in common is limited storage space. Some routines, such as those used by GNUzip (gzip), store as little as 64 kilobytes (KBs) of data. Others techniques, such as disk-based compression systems, can store as much as 1 terabyte of data. To understand the impact of dictionary size, a basic understanding of cache management is required.

Similar to requests to a website, not all bytes transferred on the network repeat with the same frequency. Some byte patterns occur with great frequency because they are part of a popular document or common network protocol. Other byte patterns occur only once and are never repeated again. The relationship between frequently repeating byte sequences and less frequently repeating ones is seen in both Zipf's and Heaps' laws.

Zipf's and Heaps' laws are linguistics-derived mathematical equations used to predict the repetitiveness of a vocabulary subset in a finite text. Both laws are applicable outside linguistics to describe observed patterns of repetitiveness in data. Both are often used in data deduplication and compression algorithms as aids to predict and optimize the elimination of repeating byte patterns.

Heaps' Law

Heaps' law states that the number of unique words (V) in a collection with N words is approximately Sqrt[N]. A plot graph of data that exhibits Heaps' Law will have a slope of approximately 0.5.

Zipf's Law

Zipf's law provides a mathematical formula for determining the frequency distribution of words in a language.

r = rank of a word

N = total number of words in the collection (not number of unique words)

r * freq(r) = A * N

Zipf's law states that the frequency of any word in a collection is inversely proportional to its rank in the frequency table. The most frequent word will occur twice as often as the second most frequent, and so on. A plot graph of data that exhibits Zipf's law will have a slope of -1.

All modern, dictionary-based compression systems leverage uneven distribution by storing more frequently accessed data and discarding less frequently accessed data. Through this type of optimization, a dictionary that stores less than 10 percent of all the byte patterns can achieve a hit ratio well in excess of 50 percent. The effect of this uneven distribution of byte patterns is evident in the effectiveness of common compression programs. For example, while gzip stores only 64 KB of history, it averages approximately 64 percent compression. However, bzip2 stores between 100 KB and 900 KB of history and averages 66 percent compression. The reason gzip and bzip2 perform so well despite lacking a substantial data store is that the most frequently occurring sequences of bytes represent the majority of bytes on a network.

Blocks versus Bytes

Block-based systems, such as Riverbed Technology's Steelhead appliances, store segments of previously transferred data flowing across the WAN. When these blocks are encountered a second time, references to the blocks are transmitted to the remote appliance, which then reconstructs the original data.

A critical shortcoming of block-based systems is that repetitive data almost never is exactly the length of a block. As a result, matches are almost always only partial matches, which leave some of the repetitive data uncompressed. Figure 3 illustrates what happens when a system using a 256-byte block size attempts to compress 512 bytes of data.

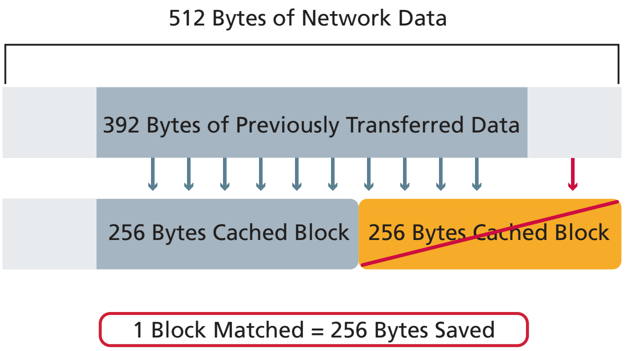
512 Bytes of Network Data
392 Bytes of Previously Transferred Data
256 Bytes Cached Block
256 Bytes Cached Block
1 Block Matched = 256 Bytes Saved

Figure 3: Block-based data reduction

Similar to Riverbed's approach of using previously transferred data to reduce network utilization, BIG-IP LTM with BIG-IP AAM builds a dictionary of previously transferred bytes using the F5 Transparent Data Reduction™ (TDR) feature. Unlike the Steelhead appliances, though, BIG-IP LTM and BIG-IP AAM matches and sends references with byte-level granularity. Figure 4 illustrates how BIG-IP AAM addresses the same 512 bytes of data.

Unlike block-based systems, the entire repeating pattern is matched and compressed by BIG-IP AAM. In the previous examples, instead of matching only 256 bytes of data, BIG-IP LTM and BIG-IP AAM matched and reduced all 392 bytes of repetitive data. This level of granularity enables BIG-IP AAM to achieve greater levels of compression than competing block-based systems—not only on documents, but also on application layer protocol headers.

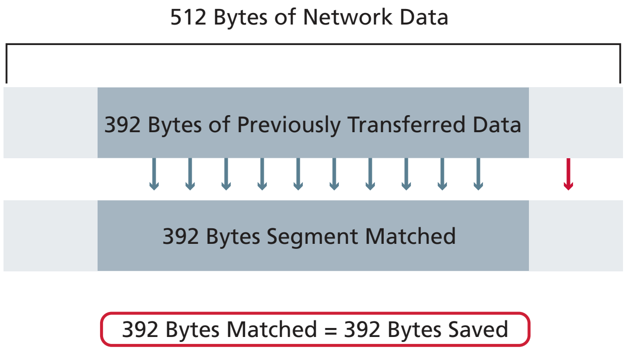
512 Bytes of Network Data
392 Bytes of Previously Transferred Data
392 Bytes Segment Matched
392 Bytes Matched = 392 Bytes Saved

Figure 4: Transparent data reduction

Static versus Adaptive Compression

Most compression capabilities on WAN optimization devices are statically configured. This means the algorithm, whether optimal for the network link and conditions or not, is always applied to the data being transferred across the WAN. Unique to F5 devices is symmetric adaptive compression, which automatically picks the right compression algorithm to maximize compression while maintaining high throughput. This feature is native to F5 TMOS® architecture and is part of a larger symmetric optimization feature set known as iSession®.

As noted in Figure 5, the performance of compression algorithms varies greatly; furthermore, performance is highly dependent on the type of data being exchanged. Symmetric adaptive compression automatically selects a high-compression codec for slow link speeds; it will never select a compression codec that is too slow for the link. It also includes a CPU saver mode for data that is known not to compress well. This feature is advantageous to organizations that have multiple WAN links with varying speeds: CPU saver mode minimizes concern over less-than-ideal WAN optimization that can result from differences in WAN characteristics.

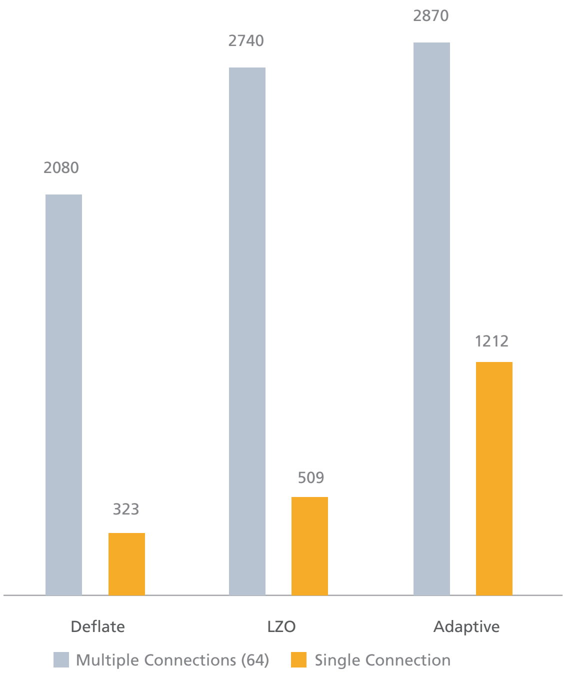

Figure 5: Comparison of compression algorithm throughput performance with BIG-IP version 8900

Application versus Network

By virtue of their beginnings as network-focused solution sets, WAN optimization solutions have traditionally focused on the network. These solutions optimize a few application layer protocols, but those protocols are generally focused on the transfer of large data sets from shared file systems such as Common Internet File System (CIFS), Microsoft's file access protocol, and Samba.

BIG-IP LTM and BIG-IP AAM provide specific policies for file sharing across CIFS, to optimize traffic between servers running Microsoft Exchange Server and clients running Microsoft Office Outlook, and for optimizing web applications. These optimization policies reduce chattiness of the protocols and add web application–specific acceleration options that can improve response time and overall performance of applications delivered via the WAN. These optimizations and acceleration techniques are possible because of TMOS, which enables WAN optimization and application acceleration solutions to share a unified internal architecture. This architecture enhances the ability to apply multiple techniques to the same data, ensuring it performs as well as possible.

Does Throughput Matter?

While achieving a high compression ratio is vital to improving application performance on networks with limited bandwidth, system throughput also plays an important role. The performance gains from a given compression technology can be assessed by considering the technology's expected compression ratio, the device's peak compression throughput, and the network bandwidth. If the compression ratio is too low, the network will remain saturated and performance gains will be minimal. Similarly, if compression speed is too low, the compressor will become the bottleneck.

TDR, as implemented in BIG-IP LTM and BIG-IP AAM, has been optimized to maintain high throughput. While the Riverbed Steelhead 5520 peaks at 540 Mbps, BIG-IP LTM and BIG-IP AAM can sustain speeds of up to 10,000 Mbps with a single appliance (BIG-IP 8900). When TDR is coupled with symmetric adaptive compression capabilities, BIG-IP LTM and BIG-IP AAM can sustain up to 10,600 Mbps with the same single appliance.

Conclusion

Achieving substantial application performance gains through compression requires a good compression algorithm and a system architecture that is designed for performance. The compression system must precisely match repetitive patterns to achieve high compression ratios. When possible, the most efficient compression algorithm based on the network link should be applied automatically. This system must manage stored data and incoming application traffic to maximize effectiveness, and it should optimize and accelerate the performance of applications commonly accessed via a WAN link (see Figure 6). Finally, this system must do all this quickly to minimize latency and continue to fill the network.

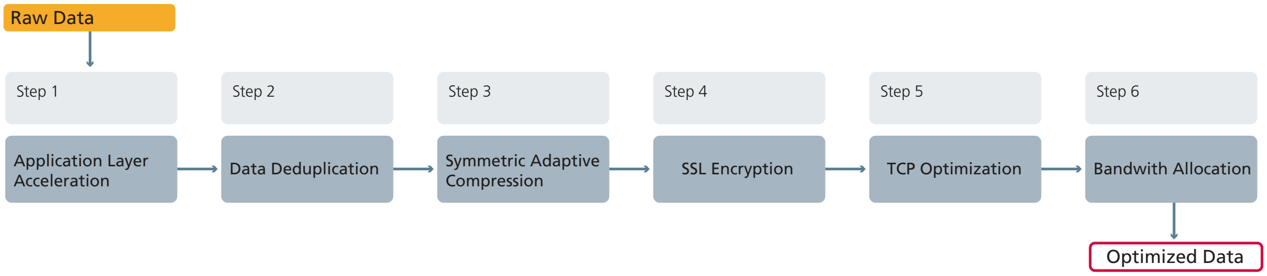

Figure 6: How BIG-IP LTM with BIG-IP AAM optimizes applications and data transfers

BIG-IP LTM and BIG-IP AAM and the TDR feature were designed from the ground up to meet these demands that a system not only provide significant compression to improve data transfer rates, but also simultaneously accelerate and optimize applications delivered over the WAN. By leveraging the capabilities afforded by deployment on a unified application delivery platform, BIG-IP LTM and BIG-IP AAM can apply compression algorithms dynamically, optimize and accelerate web application and email access, reduce bandwidth utilization, and minimize the time required to transfer large data sets across constrained WAN links.

F5 Networks, Inc. 401 Elliott Avenue West, Seattle, WA 98119 888-882-4447 www.f5.com

F5 Networks, Inc.
Corporate Headquarters
info@f5.com

F5 Networks
Asia-Pacific
apacinfo@f5.com

F5 Networks Ltd.
Europe/Middle-East/Africa
emeainfo@f5.com

F5 Networks
Japan K.K.
f5j-info@f5.com

©2013 F5 Networks, Inc. All rights reserved. F5, F5 Networks, and the F5 logo are trademarks of F5 Networks, Inc. in the U.S. and in certain other countries. Other F5 trademarks are identified at f5.com. Any other products, services, or company names referenced herein may be trademarks of their respective owners with no endorsement or affiliation, express or implied, claimed by F5.