

Understanding Advanced Data Compression

Overview

Nearly all WAN optimization appliances leverage advanced compression routines to improve application performance. At a high level, they all store and leverage previously transferred network data to achieve high compression ratios. When examined further though, how they achieve these gains, and their resulting limitations, varies widely.

Packets vs. Sessions

Most network compression systems to date have been packet-based. Packet-based compression systems buffer packets destined for a remote network with a decompressor. These packets are then compressed either one at a time or as a group and then sent to the decompressor where the process is reversed. Packet-based compression has been available for many years and can be found in routers, VPN clients, and Juniper Networks® WX and WXC appliances.

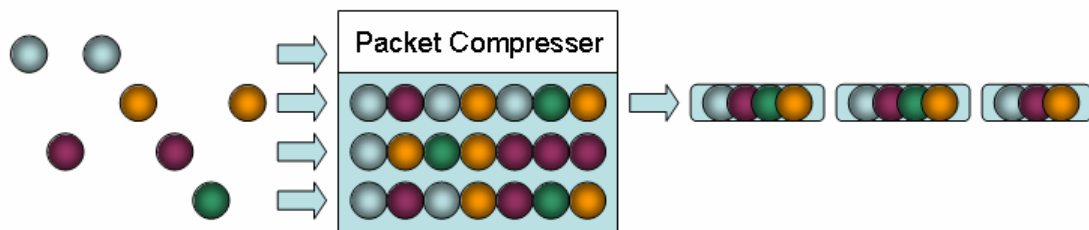


Figure 1: *Packet-based compression*

The primary issue with packet-based compression is that it mixes multiple different data types together when compressing. All compression routines achieve greater levels of compression when operating on homogenous data. When presented with heterogeneous data, such as a collection of packets from multiple different protocols, compression ratios fall dramatically.

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. In addition, aggregating and encapsulating packets adds encapsulation headers to the stream.

Unlike previous compression solutions, the F5 WANJet appliance operates at the session layer. This allows it to apply compression across a completely homogenous data set while addressing all application types. This results in higher compression ratios than comparable packet-based systems.

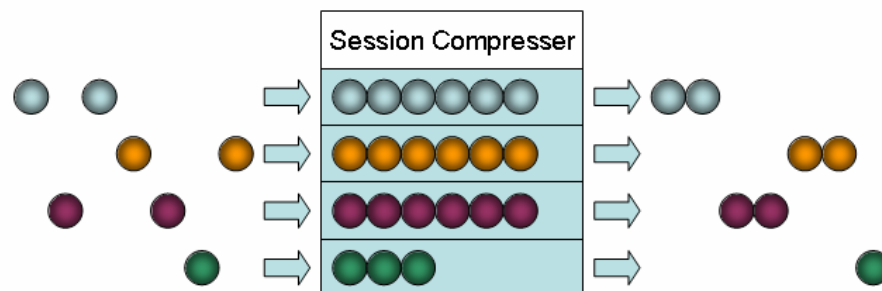


Figure 2: *Session-based Compression*

Furthermore, by operating at the session layer, packet boundary and repacketization problems are eliminated. This allows the WANJet device to easily find matches in data streams which at layer 3 may 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 Gzip, store as little as 64kb of data. Others techniques, such as disk based compression systems, may store as much as a terabyte of data. In order to understand the effect of dictionary size, a basic understanding of cache management is required.

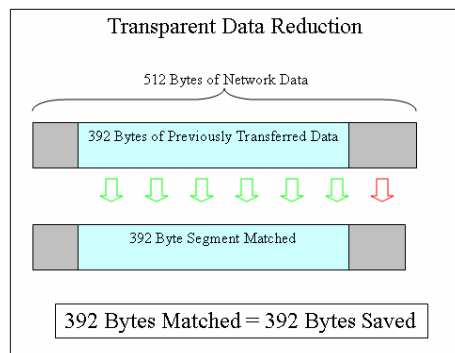
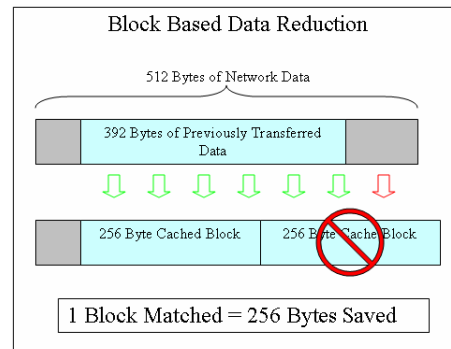
Similar to requests to a web site, not all bytes transferred on the network repeat with the same frequency. Some byte patterns occur with great frequency because they are part of a popular document or common network protocol. Other byte patterns occur only once and are never repeated again. The relationship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

All modern dictionary-based compression systems leverage this 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% of all the byte patterns can achieve a hit ratio well in excess of 50%. The effect of this uneven distribution of byte patterns is evident in the effectiveness of common compression programs. Gzip stores only 64kb of history yet averages approximately 64% compression. Bzip2, which stores between 100kb and 900kb of history, averages 66% 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 vs. Bytes

Block-based systems, such as Juniper Networks WXC and 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 leaving some of the repetitive data uncompressed. The diagram to the right details what happens when a system using a 256 byte block size attempts to compress 512 bytes of data.



Similar to Riverbed and Juniper's approach of using previously transferred data to reduce network utilization, the F5 WANJet builds a dictionary of previously transferred bytes using TDR (Transparent Data Reduction) technology. Unlike the WXC and Steelhead appliances though, the WANJet matches and sends references with byte level granularity. The diagram to the left illustrates how WANJet address the same 512 bytes of data.



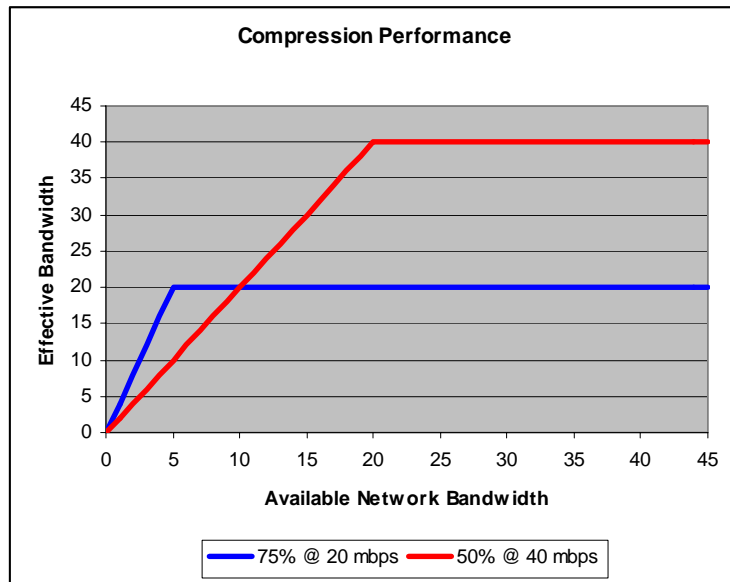
Unlike block-based systems, the entire repeating pattern is matched and compressed by the WANJet. In the previous examples, instead of matching only 256 bytes of data, the WANJet is able to match and reduce all 392 bytes of repetitive data. This level of granularity allows the WANJet to achieve greater levels of compression than competing block-based systems on not only documents, but application layer protocol headers as well.

Does Throughput Matter?

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

In the following graph, the compressor with the slightly better compression ratio (75%) only outperforms the faster routine for link speeds below 10 Mbps. The reason for this is that at speeds greater than 5 Mbps, well below the 20 Mbps mark, the slower compressing routine is unable to process enough data to continue to fully utilize the network.

TDR (Transparent Data Reduction), as implemented in the F5 WANJet, has been optimized to maintain high throughput. While the Juniper WXC 500 peaks at 20 Mbps and the Riverbed Steelhead 5010 peaks at 45 Mbps, the WANJet can sustain speeds of up to 400 Mbps with a single appliance. This level of performance allows the WANJet to deliver 8x performance gains on T3 networks – a feat impossible for both WXC and Steelhead appliances.



Summary

Achieving substantial application performance gains through compression requires not only a good compression algorithm, but an entire system architecture designed for performance. The compression system must precisely match repetitive patterns to achieve high compression ratios. It must manage both stored data and incoming application traffic to maximize effectiveness. Finally, it must do all this quickly to minimize latency and continue to fill the network. F5 Networks' WANJet and TDR (Transparent Data Reduction) were designed from the ground up to meet these demands.