IBM Storage & SDI

TS7700 R 4.1.2 Compression and Performance

Khanh Ly

July 18, 2018

Overview

- TS7700 R 4.1.2 major offerings:
  - Software Compression
  - 16Gb FICON support

Background

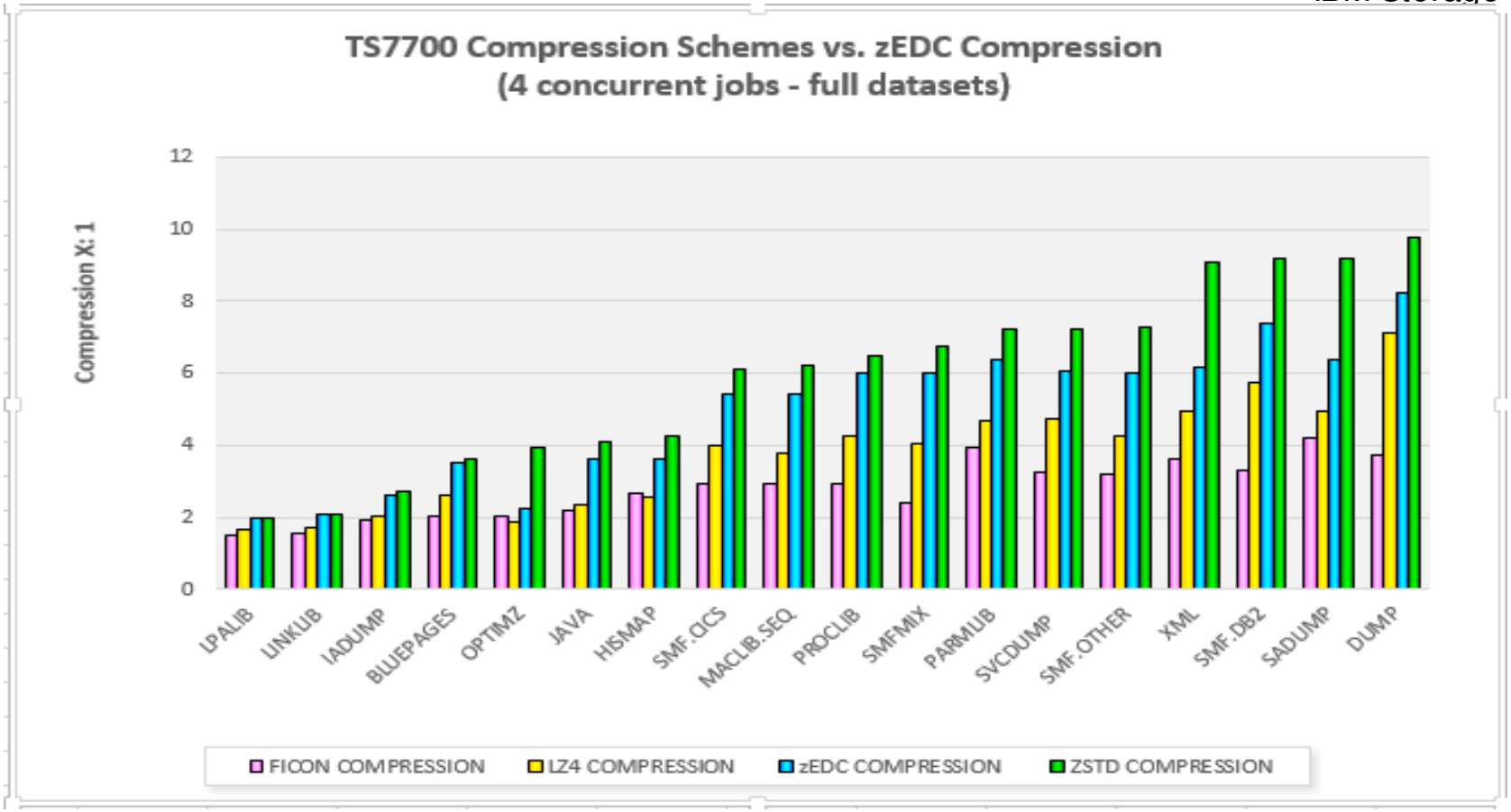
- TS7700 has supported a form of ALDC compression in the FICON adapters since its first release
- The FICON adapter compression is an older algorithm that produces lower than average compression results.
- zEDC compression in the z13 and z14 hosts is an alternative to device based compression, but it requires extra expense and is limited to certain applications.
- With release R 4.1.2, the TS7700 introduced two new software based compression types: LZ4 and ZSTD.
- Of these two compression methods:
  - LZ4 – Fastest, minimal CPU usage, improved compression
  - ZSTD – Fast, higher CPU usage, highest compression

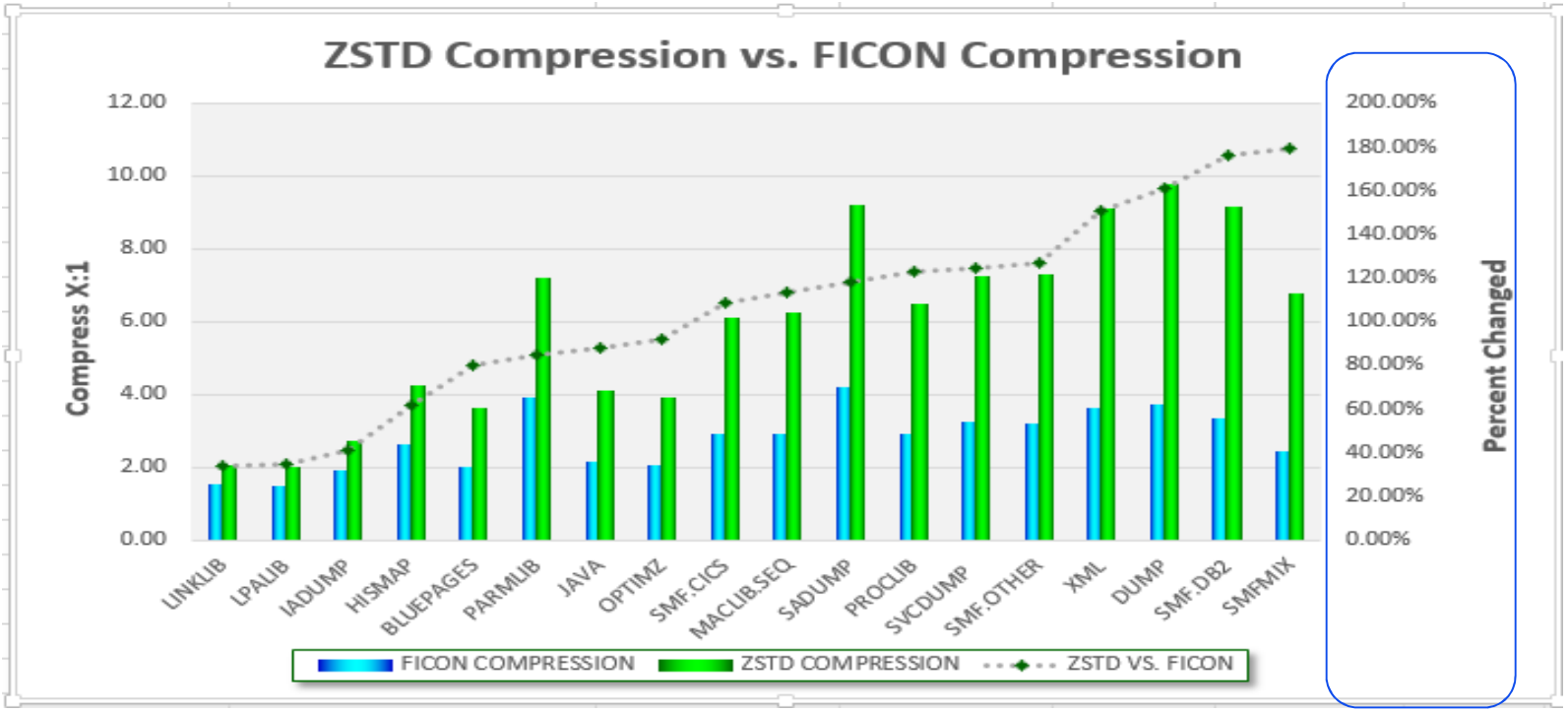
Testing:

Using the same workload patterns used by the zEDC IBM performance team, we compared the compression efficiency of:

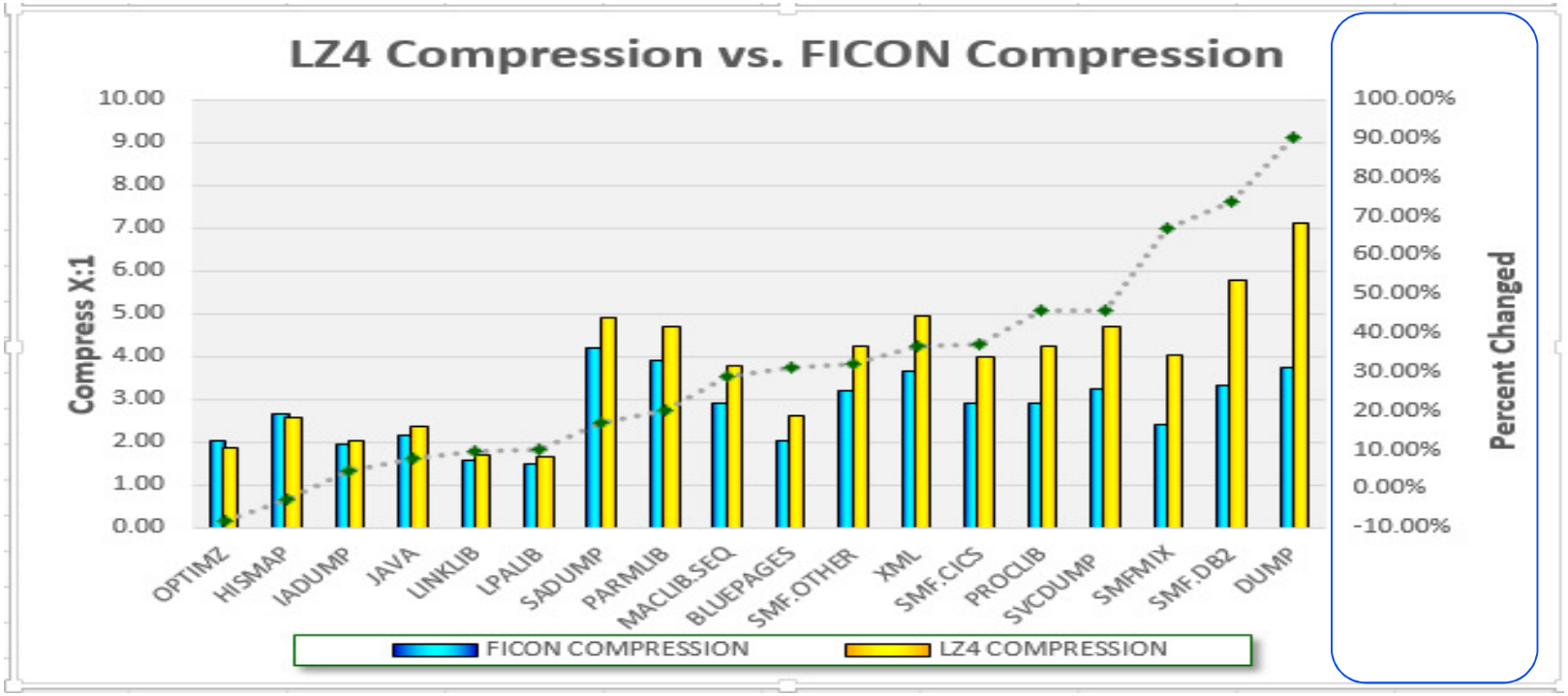
- zEDC
- FICON
- LZ4
- ZSTD

The results showed that ZSTD excelled in all scenarios.





Based on the different workloads, our tests show that ZSTD compresses data more efficiently than FICON with improvements from +33% to +179%. ZSTD delivers high compression ratio at the expense of higher CPU utilization. As a result, ZSTD is generally not recommended for TS7700 with Power 7 server (models VEB/V07).



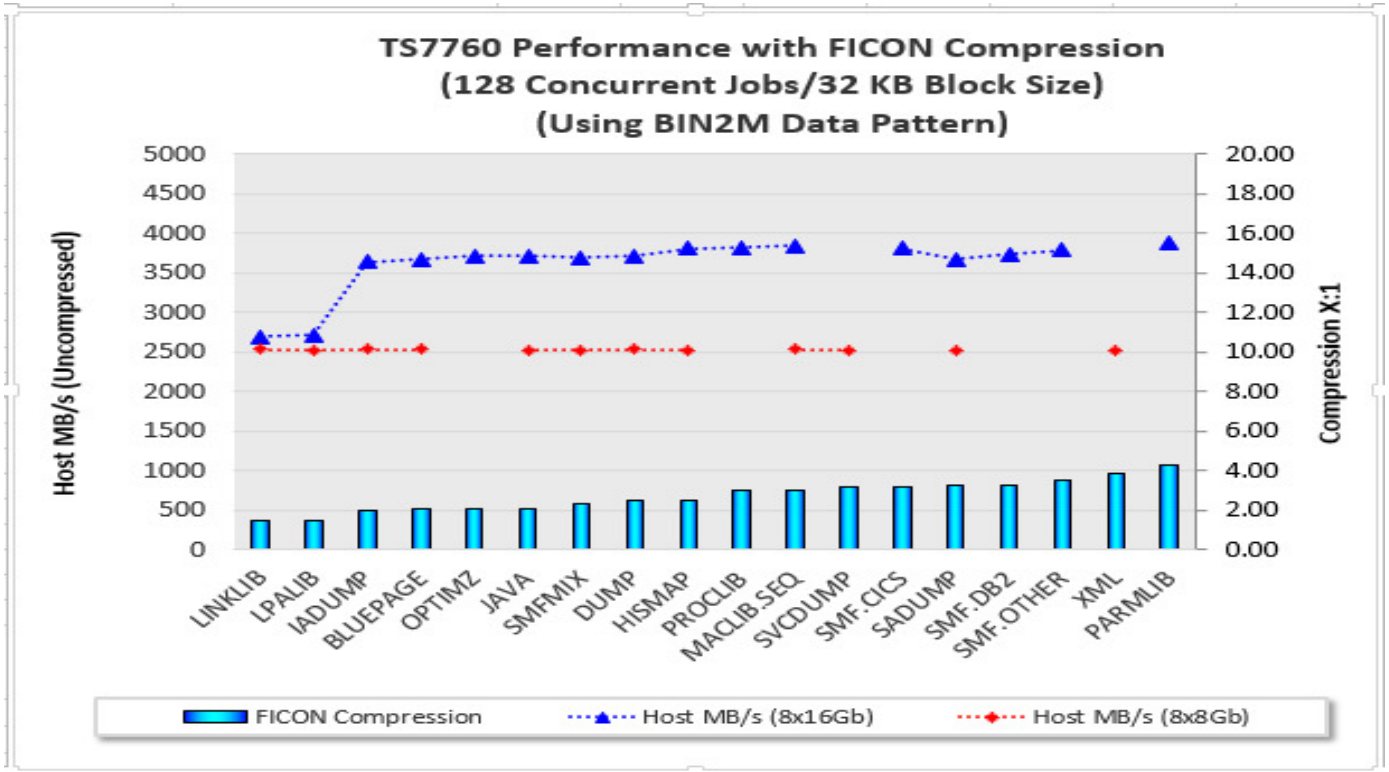
Of the 18 tests, LZ4 exceeds FICON in 16 tests (from +4% to +90%). FICON yields higher compression ratios in only 2 instances (-2% and -8%). In general, LZ4 provides decent compression ratios without really stressing out the CPU on the TS7700.

Best Practices: Which one should I use?

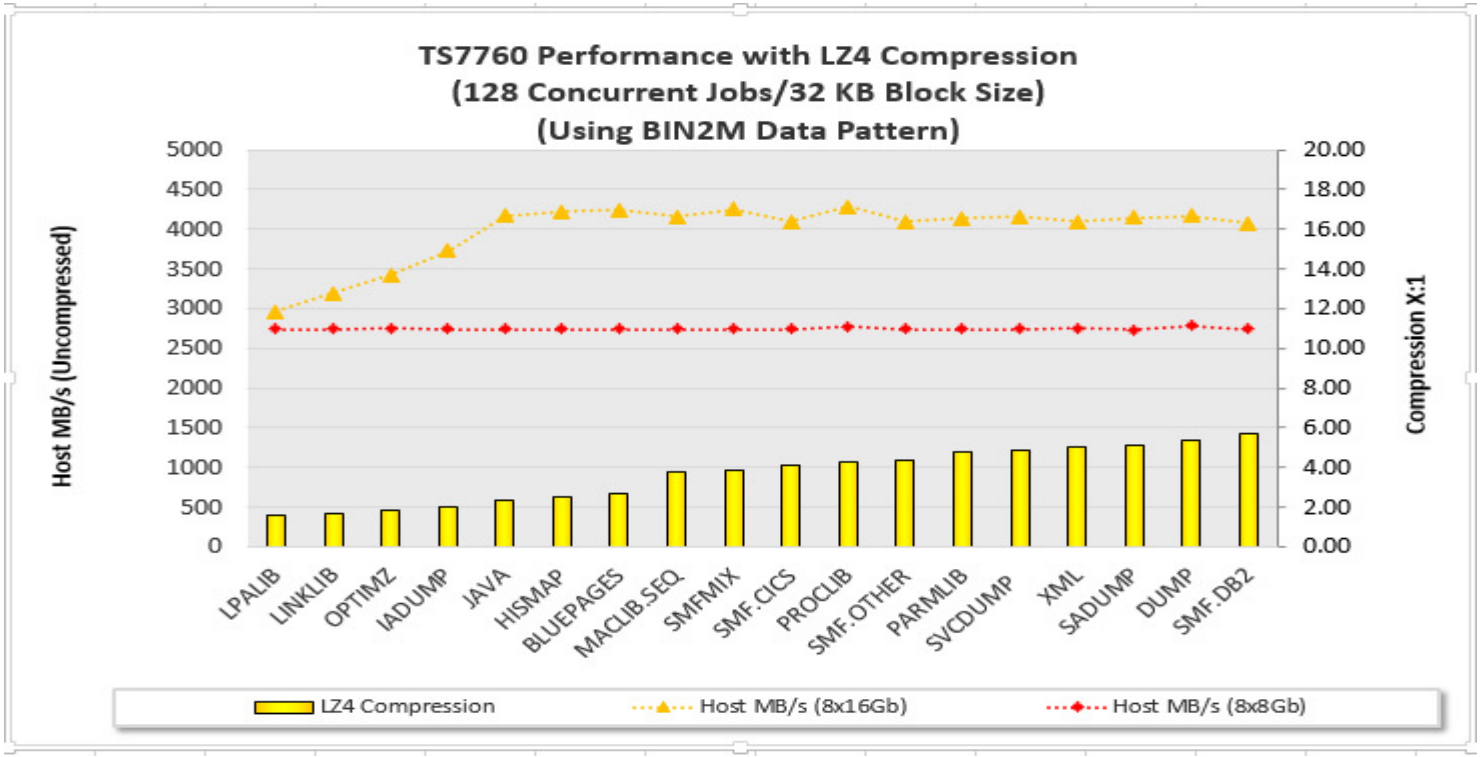
- If the most compression is the goal, ZSTD would be the go-to algorithm. But performance is also a variable that customers need to take into account.
- The following tests evaluate the performance of the 3 different compression schemes.

New Performance Data Patterns

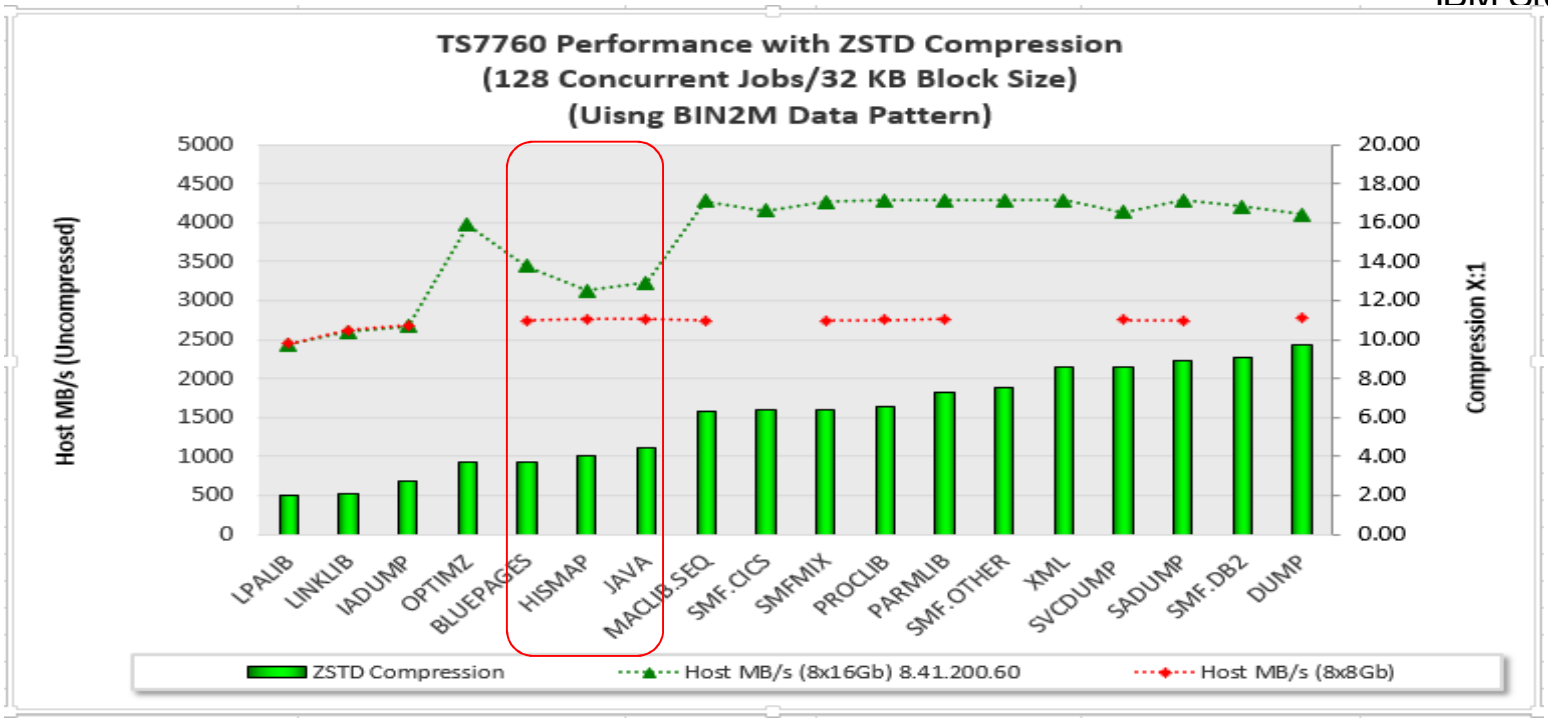
- In our compression tests with zEDC data, we ran with only 4 concurrent jobs to avoid contention between Host and DASD where the data was stored.
- To really stress the TS7700, we need many concurrent jobs (128 in current TS7700 performance benchmark workload), but having them all pull data from DASD would introduce a bottleneck. Therefore, we need smaller pattern files that produce similar compression results to the large datasets.
- Our DFSms expert extracted a 2 MB block from each zEDC test dataset. The 2 MB block had a similar compression result as the original dataset. We called these BIN2M and used them to test the performance of the 18 different data patterns.



Data is sorted based on compression ratio. TS7760 with 8x8Gb FICON channels easily hit the adapter compression limit of 2500 MB/s. TS7760 with 8x16Gb FICON channels resulted in host data rates greater than 3600MB/s except for BIN2M patterns which had a compression rate below 1.5:1.



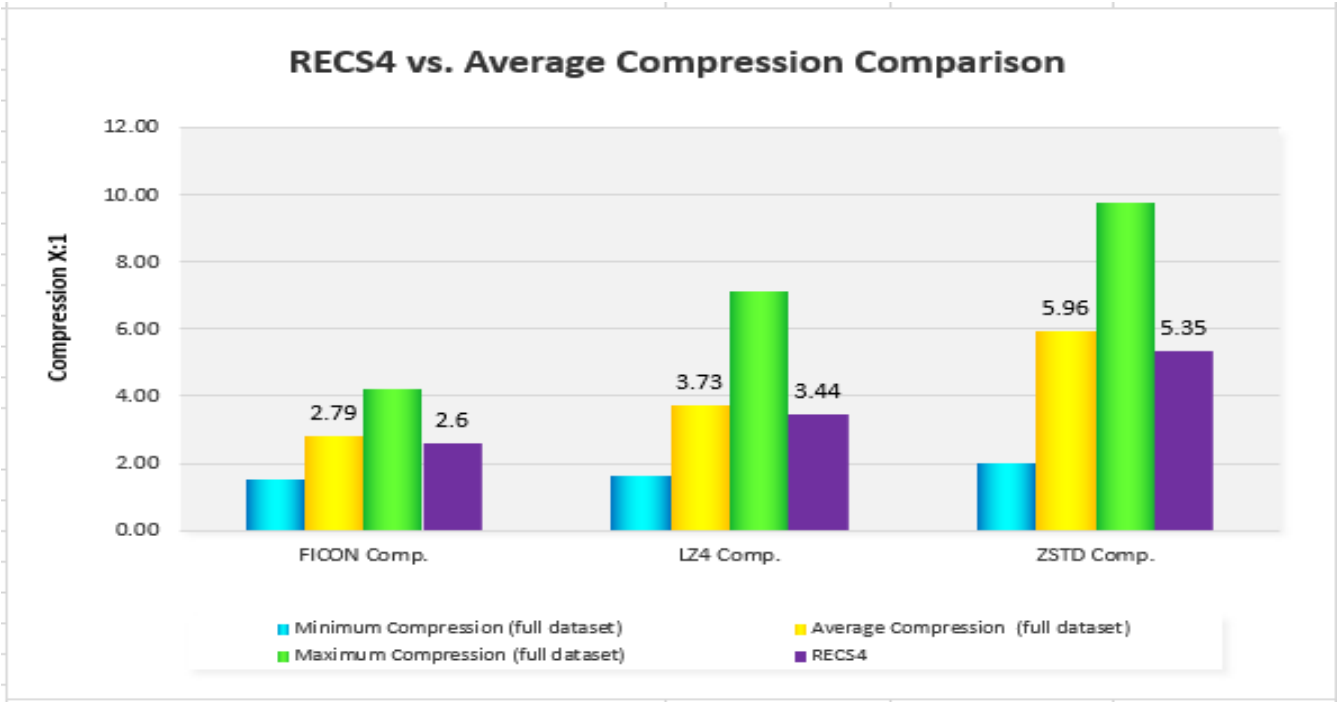
Data is sorted based on compression ratio. TS7760 with 8x8Gb FICON channels topped out at over 2700 MB/s (no longer gated by FICON compression ASIC). TS7760 with 8x16Gb FICON channels resulted in host data rates greater than 4000 MB/s except for those BIN2M patterns with a compression ratio less than 2:1.



ZSTD consumes more CPU than LZ4 and FICON. As a result, performance can change based on compression ratio and the CPU resources required to compress the data. In our tests, high CPU usage caused a performance impact for those workloads in red. It would be better to use LZ4 for workloads that produce similar results to the three circled BIN2M patterns.

Two New Performance Benchmark Data Patterns

- Since day one, we have selected two data patterns for TS7700 Performance benchmark workloads:
  - VT2P7 (FICON compressed @ 2.66:1) – for general testing.
  - VT1P0 (FICON compressed @ 1:1) -- for cache-stressed testing in standalone configuration
- Today with software compression, VT2P7 and VT1P0 produce unrealistic high compression ratios. We had to create two new more realistic data pattern for TS7700 performance benchmark workloads:
  - RECS4 (near average compression results) – for general testing.
  - A00 -- for 1:1 cache-stressed testing in standalone configuration
- For customer workloads that don't compress, it's best to disable compaction via z/OS DATACLASS so that the overhead of attempting compression can be avoided.
  - Previously compressed data
  - Encrypted data
  - LZ4 and ZSTD will detect expansion and store data at 1:1, but at a cost… ZSTD costing more.



Our DFSms expert extracted four consecutive 32KB blocks from each of the 18 BIN2Ms and created a data pattern called RECS4. RECS4 compression ratio was very close to the average compression ratio of the 18 original datasets.

A00 data pattern

Completely random data pattern aiming to stress the TS7700 cache and compression algorithm.

- FICON compression shows 0.89:1 (expansion)
- LZ4 compression shows 1:1 (expansion automatically avoided)
- ZSTD compression shows 1:1 (expansion automatically avoided)

TS7700 Performance Workloads and Measurements

TS770 Performance Workloads and Measurements

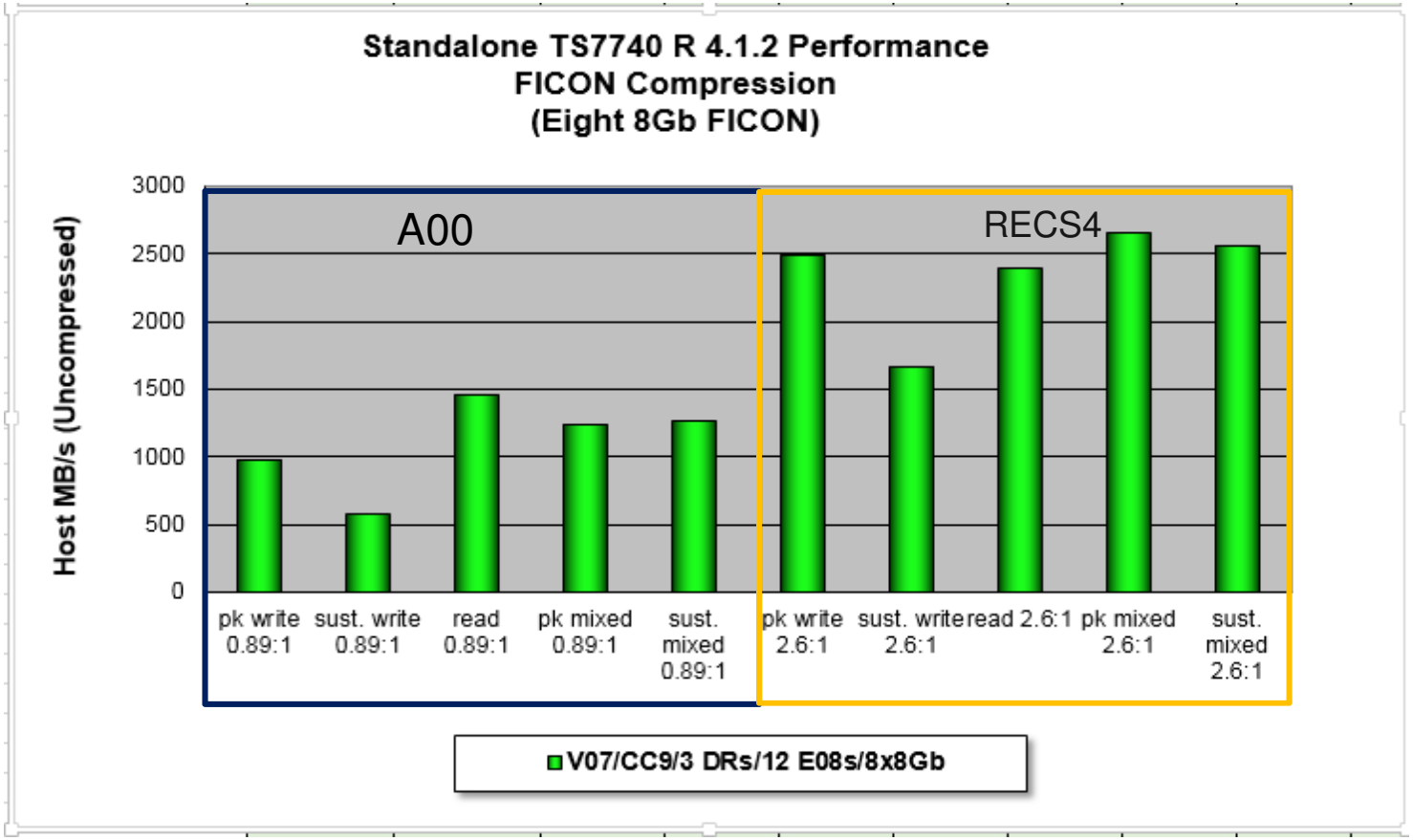
- Unless otherwise stated, all runs were made with 128 concurrent jobs. Each job wrote/read a volume (2GB after compression at the TS7700), using a 32768 byte block size, QSAM BUFFNO=20, and eight 16Gb (or 8Gb) FICON channels from a z13 LPAR.
- Clusters are located at zero or nearly zero distance to each other in laboratory setup.
- All runs were made using tuning values: DCT=125, PMPIOR=3600, PMTHLVL=4000, ICOPYT=ENABLED, Reclaim=disabled, LINKSPEED=1000. Number of premigration drives per pool=10.

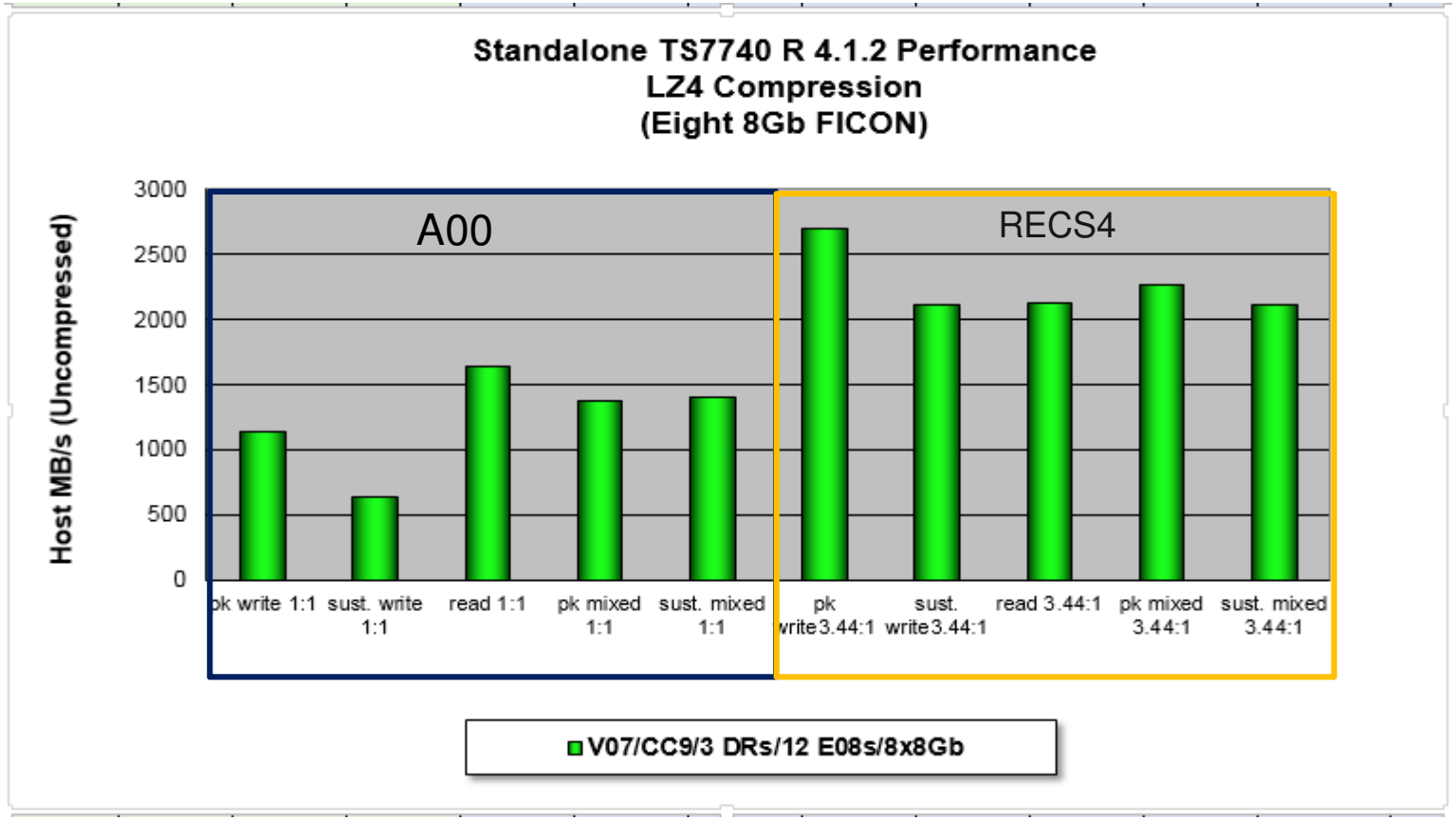
TS770 Performance Metrics

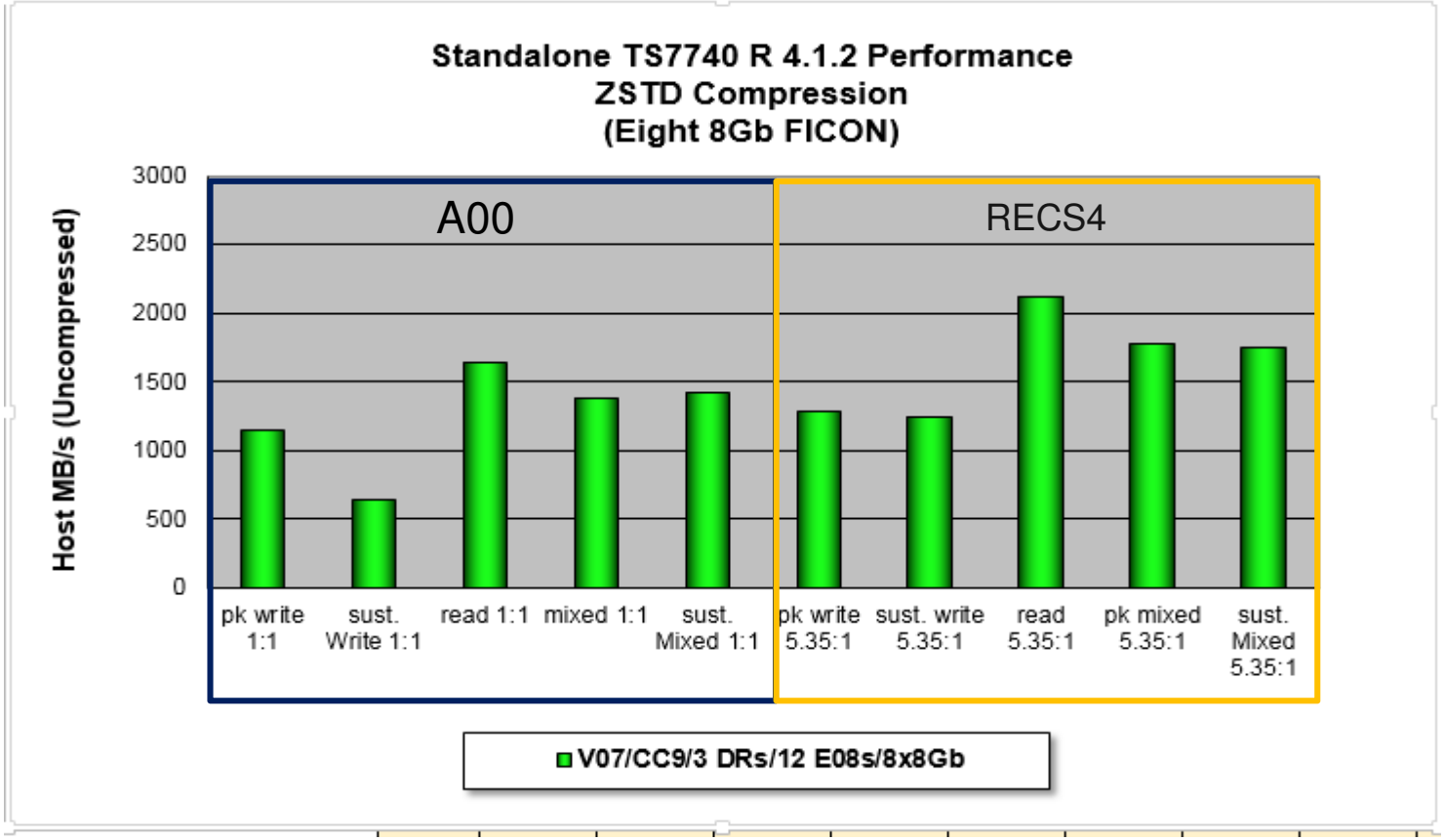
- **Peak Write Rate**: host rate at which data can be written into the DASD cache with no required physical tape drive activity.
- **Sustained Write Rate**: rate at which data can be written into the DASD cache with equivalent concurrent destaging to physical tape drives.
- **Read Hit Rate**: rate at which data can be read from the DASD cache.
- **Grid Deferred Copy Mode**: copy to other Hydra does not have to complete prior to job end (DD copy mode).
- **Grid Immediate Copy Mode**: copy to other Hydra must complete prior to job end (RR copy mode).
- **Grid Sync Copy Mode**: tape synchronization up to SYNC level granularity across two clusters within a grid (SS copy mode).
- **Grid Copy Rate**: rate at which data is copied to other Hydra.

Standalone TS7740 Performance

(Using A00 and RECS4 Data Patterns)





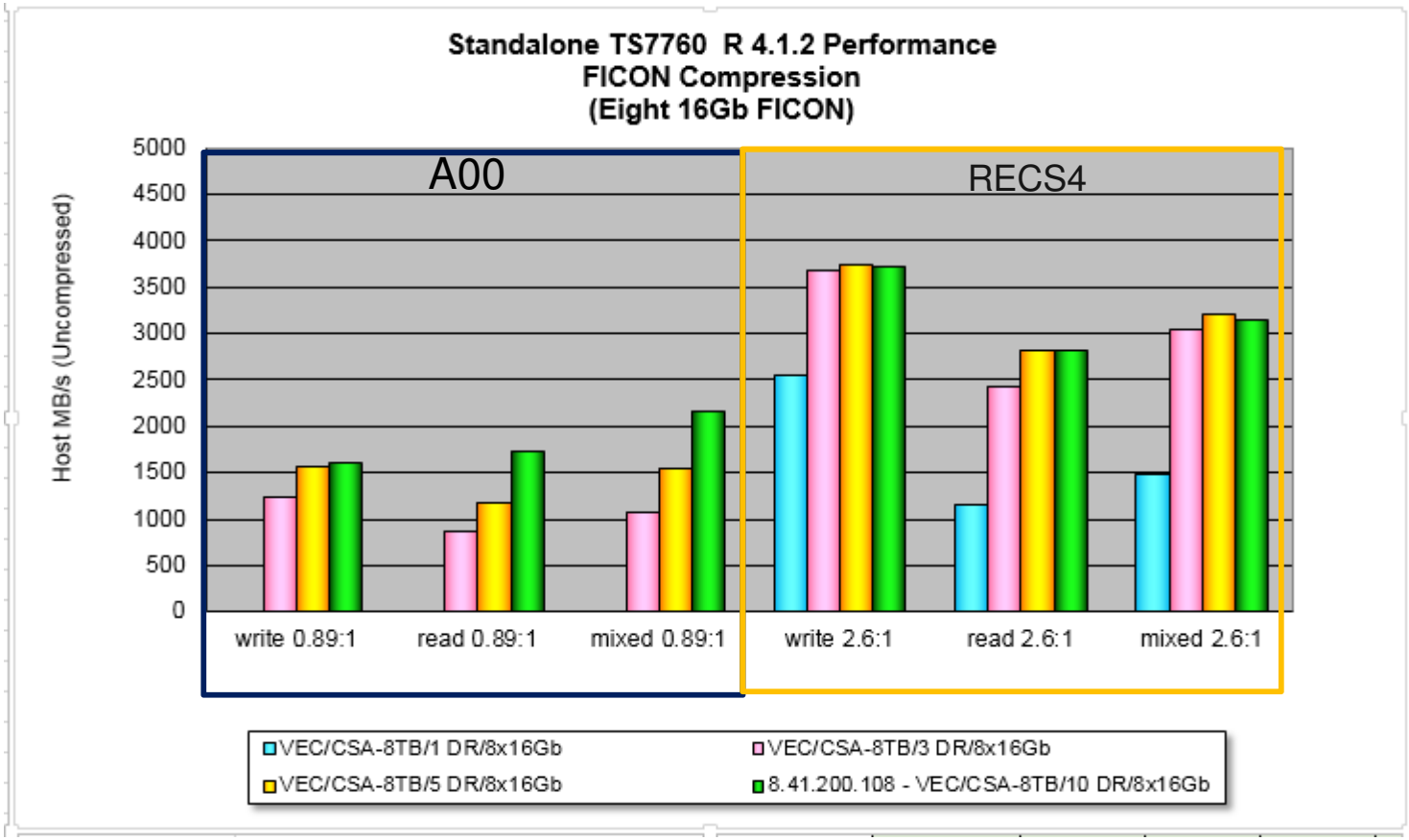


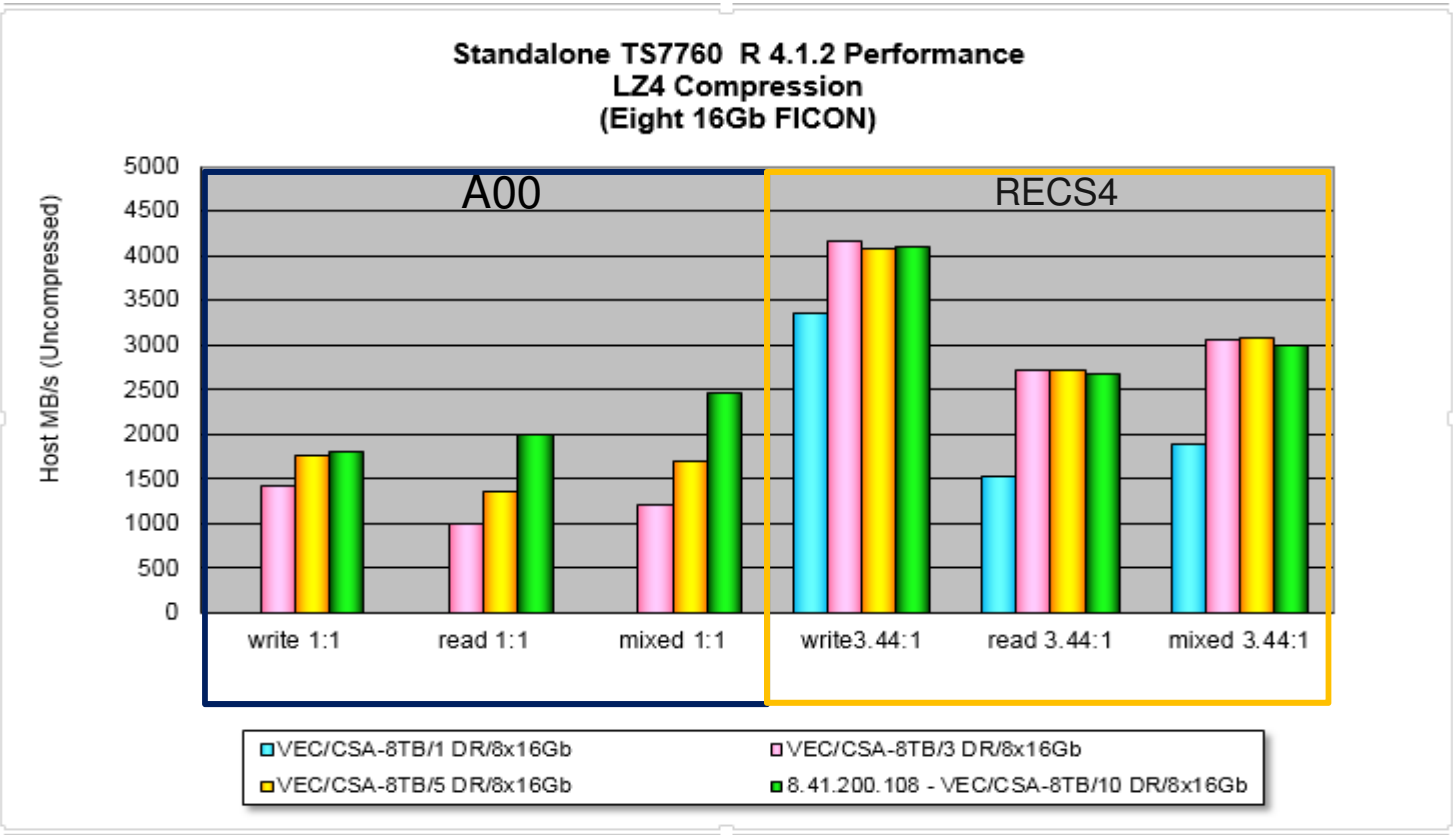
Discussion on the stand alone TS7740 results

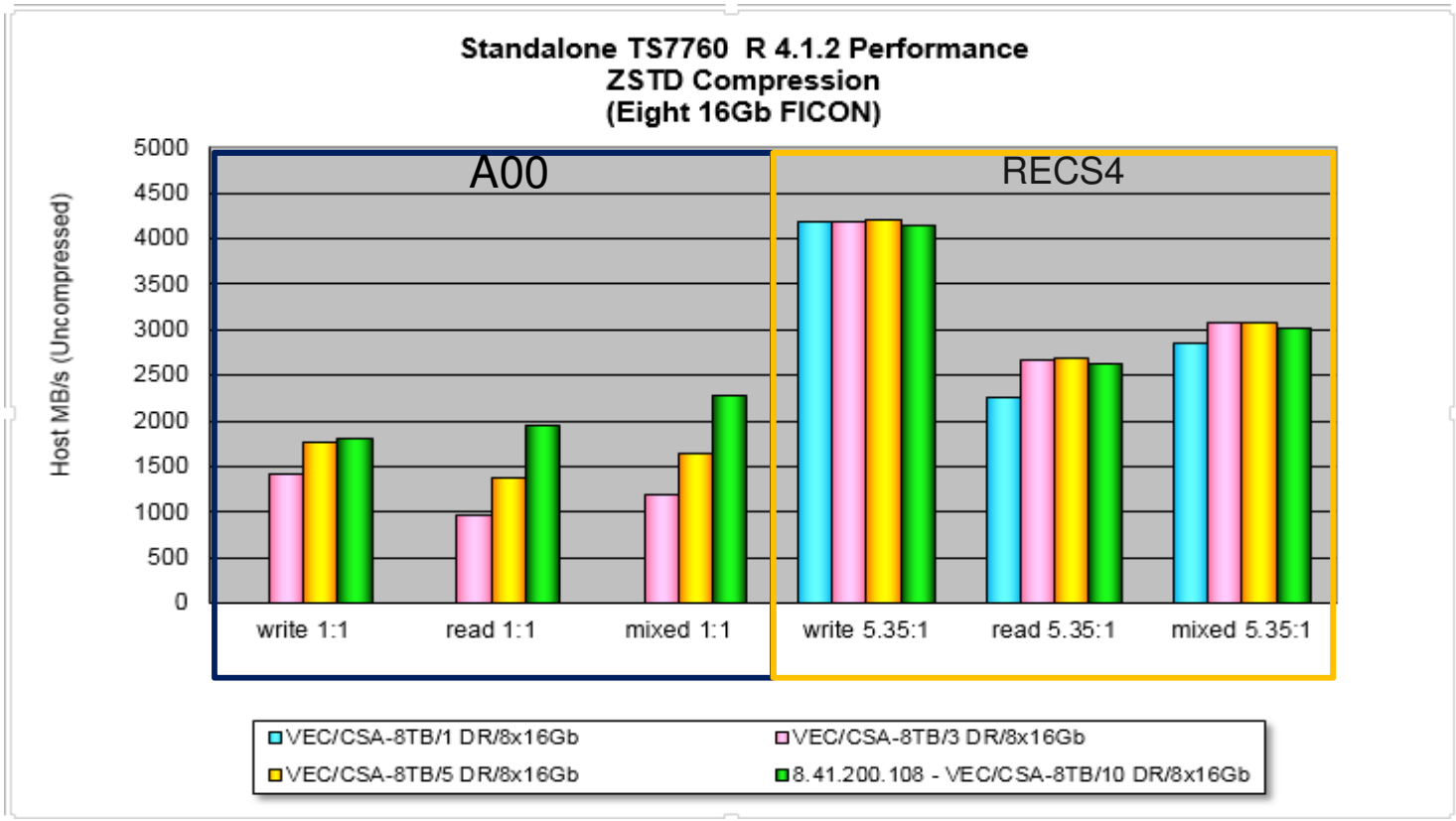
- For workloads that don’t compress, the performance between LZ4 and ZSTD compression methods were equal. The performance was lower for FICON compression due to data expansion.
- For those workloads which do compress, in general, LZ4 excelled in write performance. FICON, in read performance. ZSTD showed poor read and write performance due to high demands on CPU resources (Power 7 server).

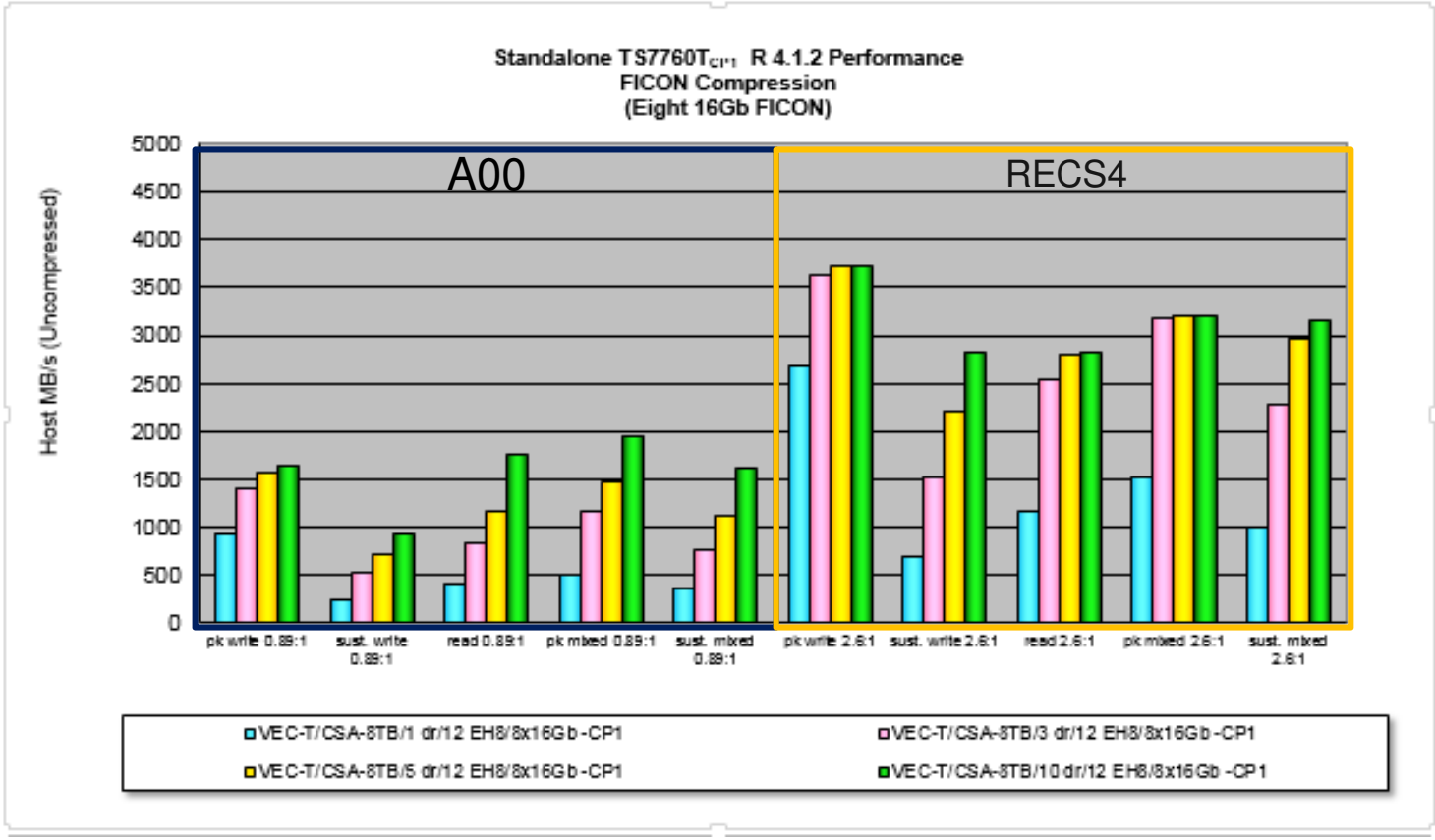
Standalone TS7760(T) Performance

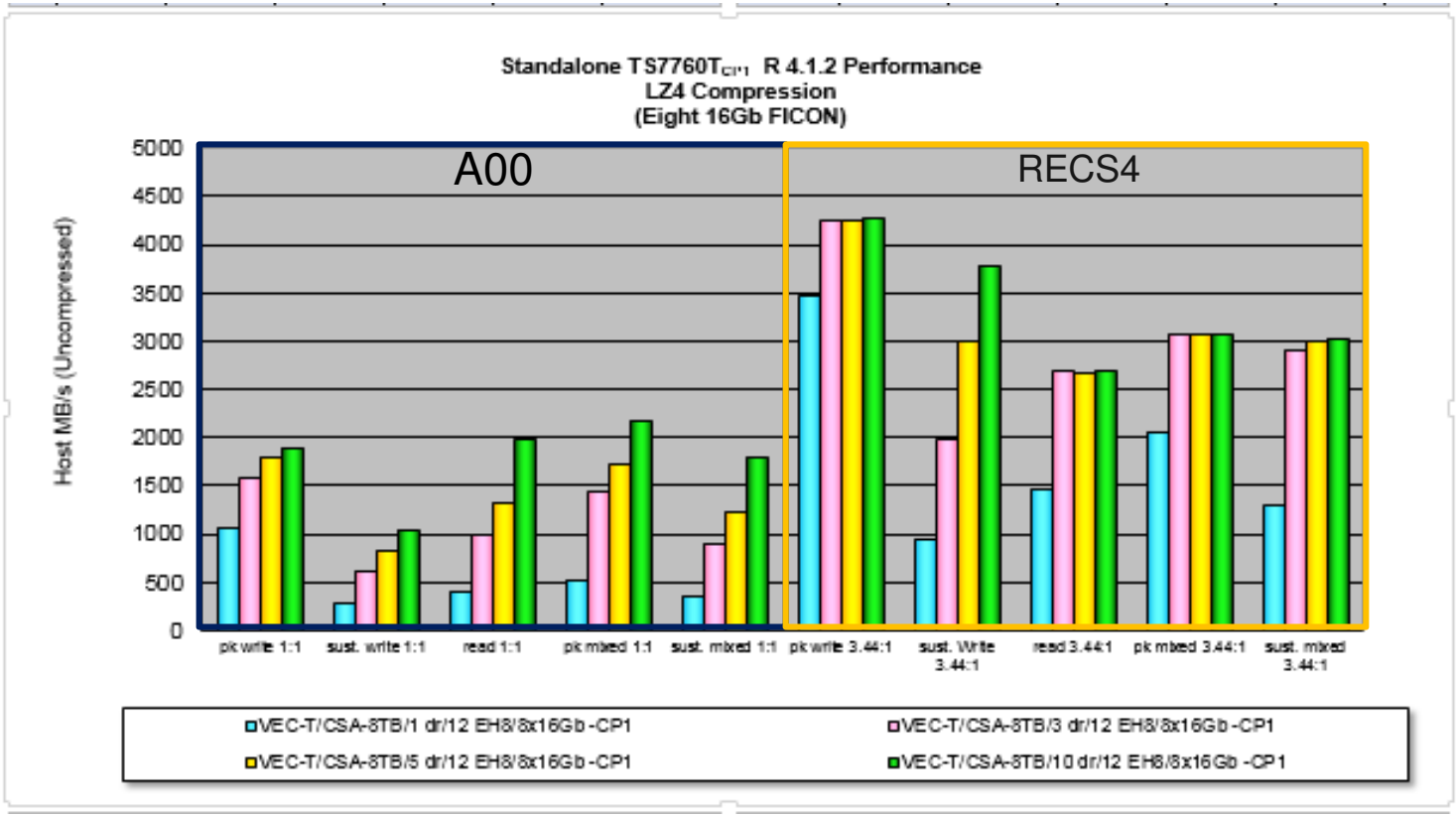
(Using A00 and RECS4 Data Patterns)

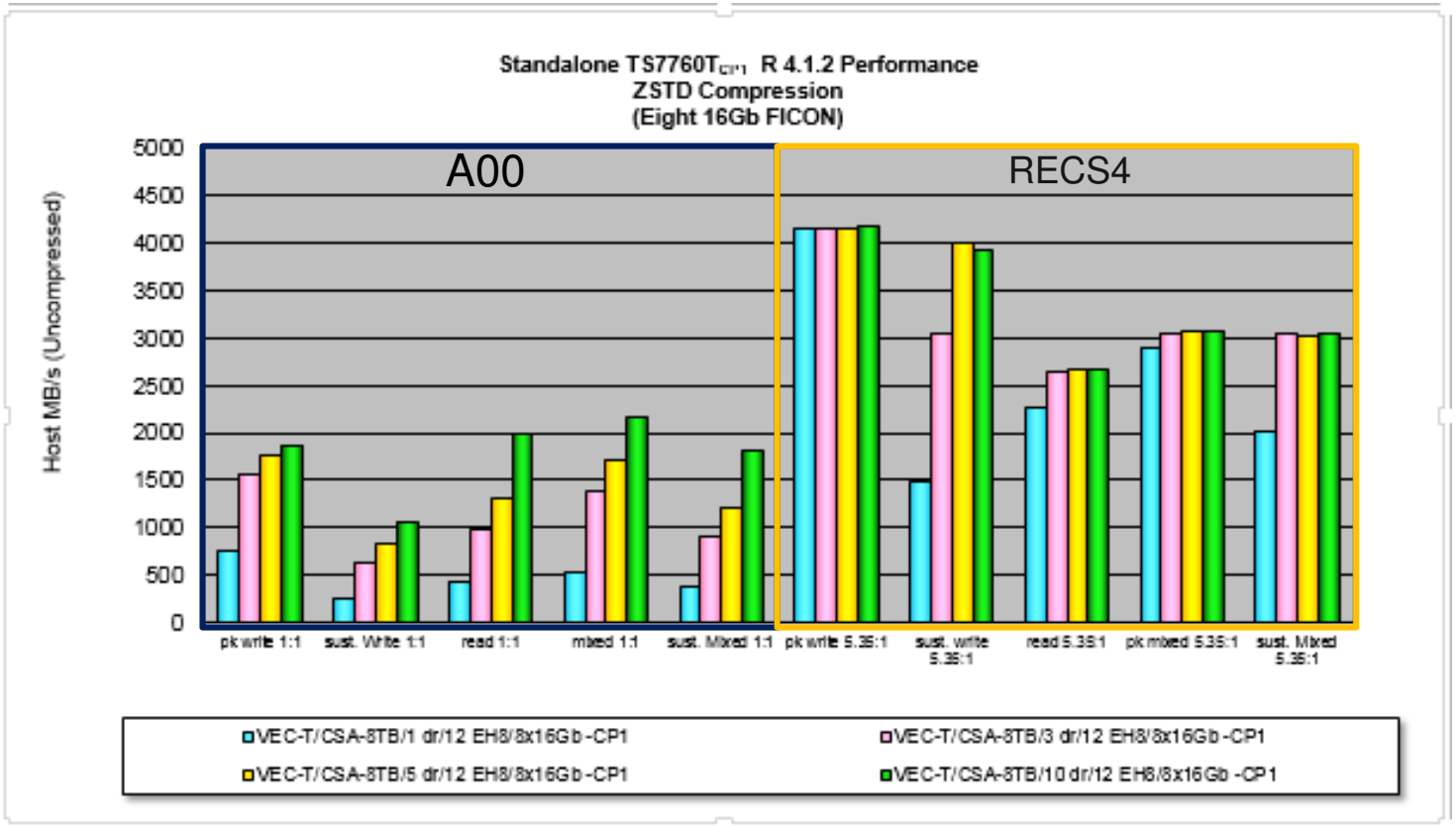










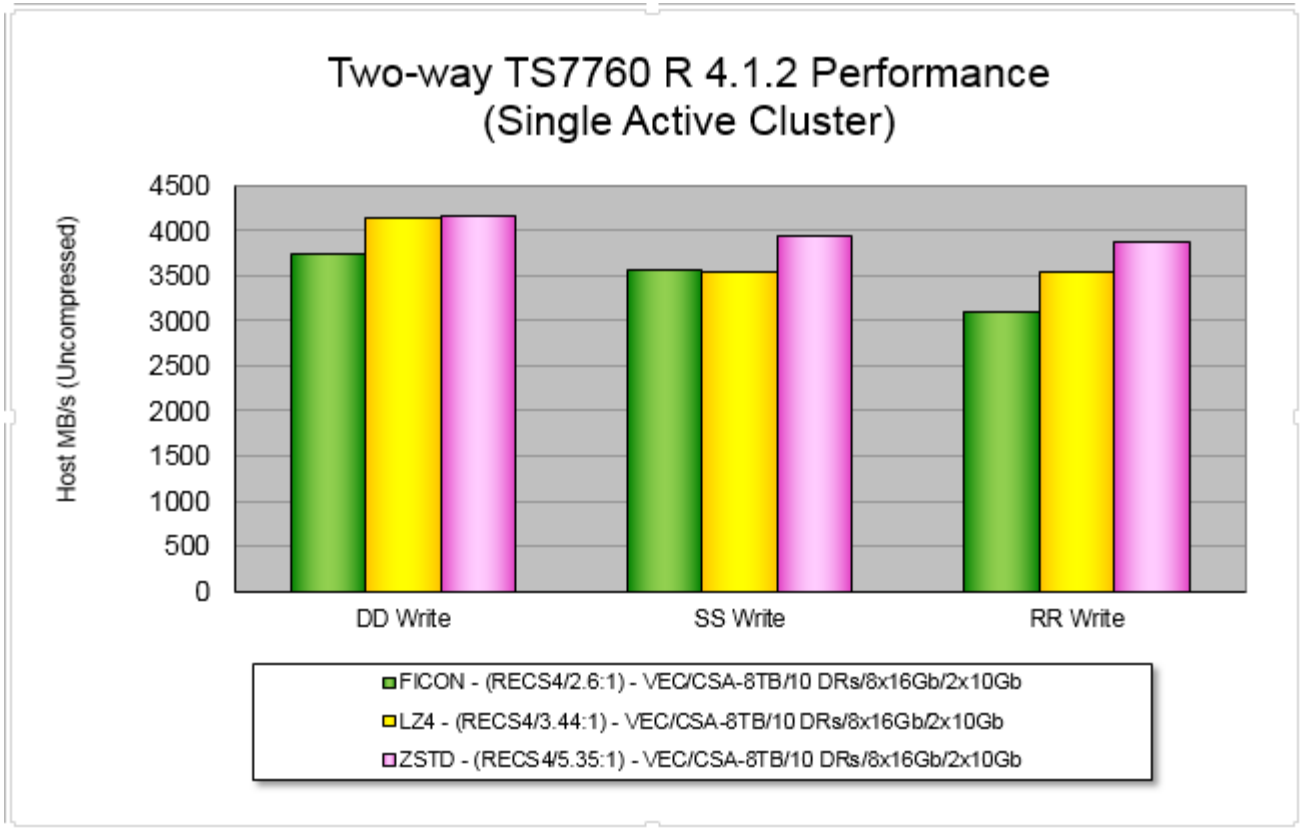


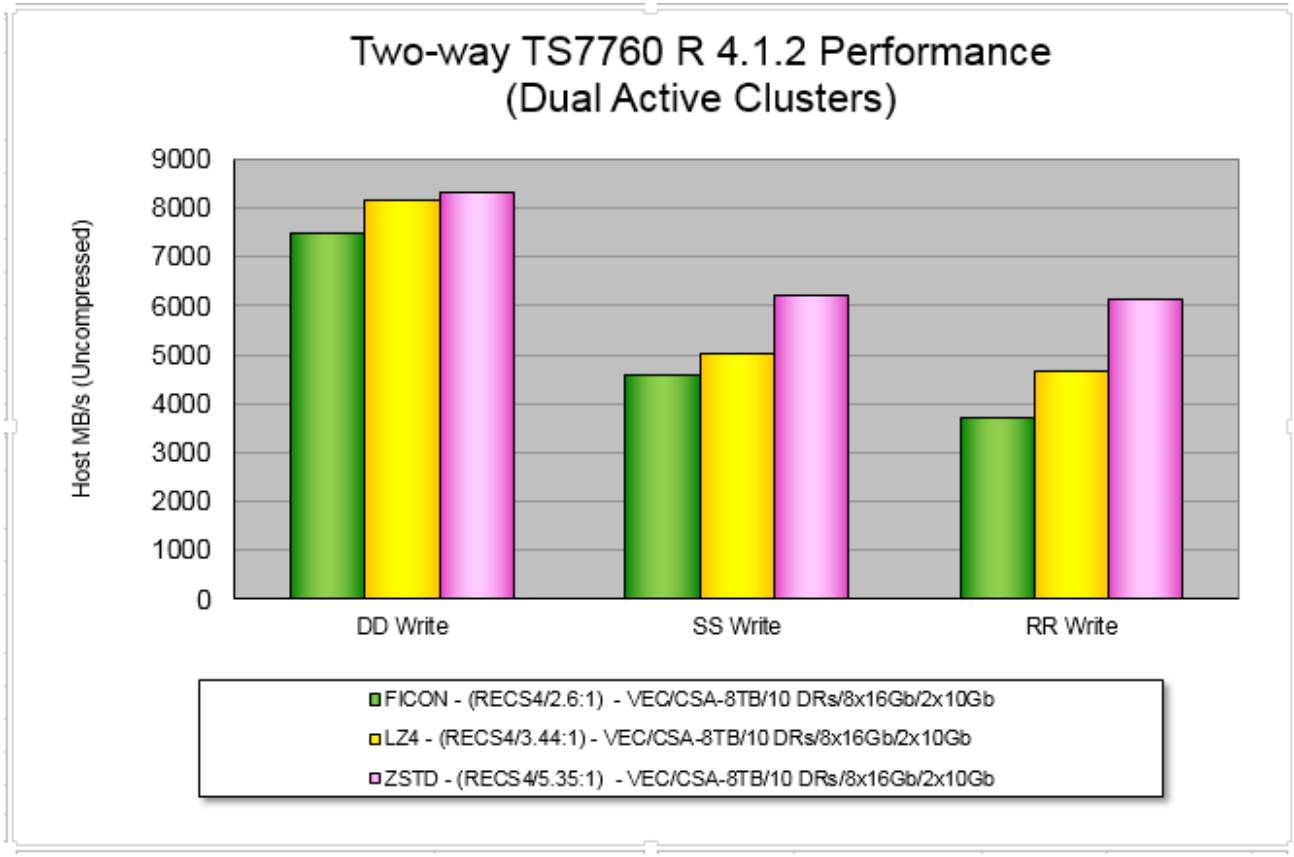
Discussion on the stand alone TS7760(T) results

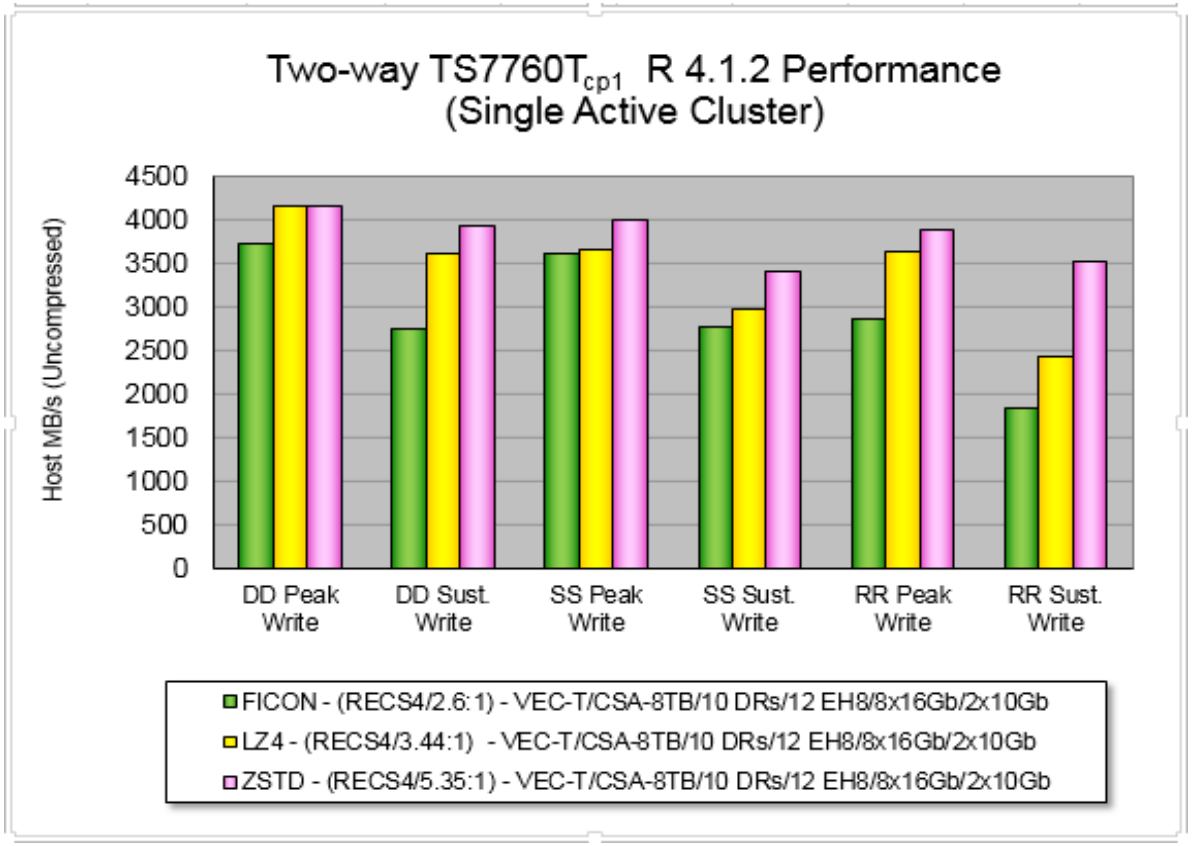
- For workloads that don't compress, the performance between LZ4 and ZSTD compression methods were equal. The performance was lower for FICON compression due to data expansion.
- For those workloads which do compress, the CPU overhead was less of a factor for the TS7760. FICON was the slowest performer in all cases. When cache is the bottleneck, ZSTD and LZ4 performed fairly equal. When cache is the bottleneck, the algorithm which compresses the best will provide the highest performance. This is why ZSTD was able to achieve over 4,000MB/s in peak write with only one drawer.

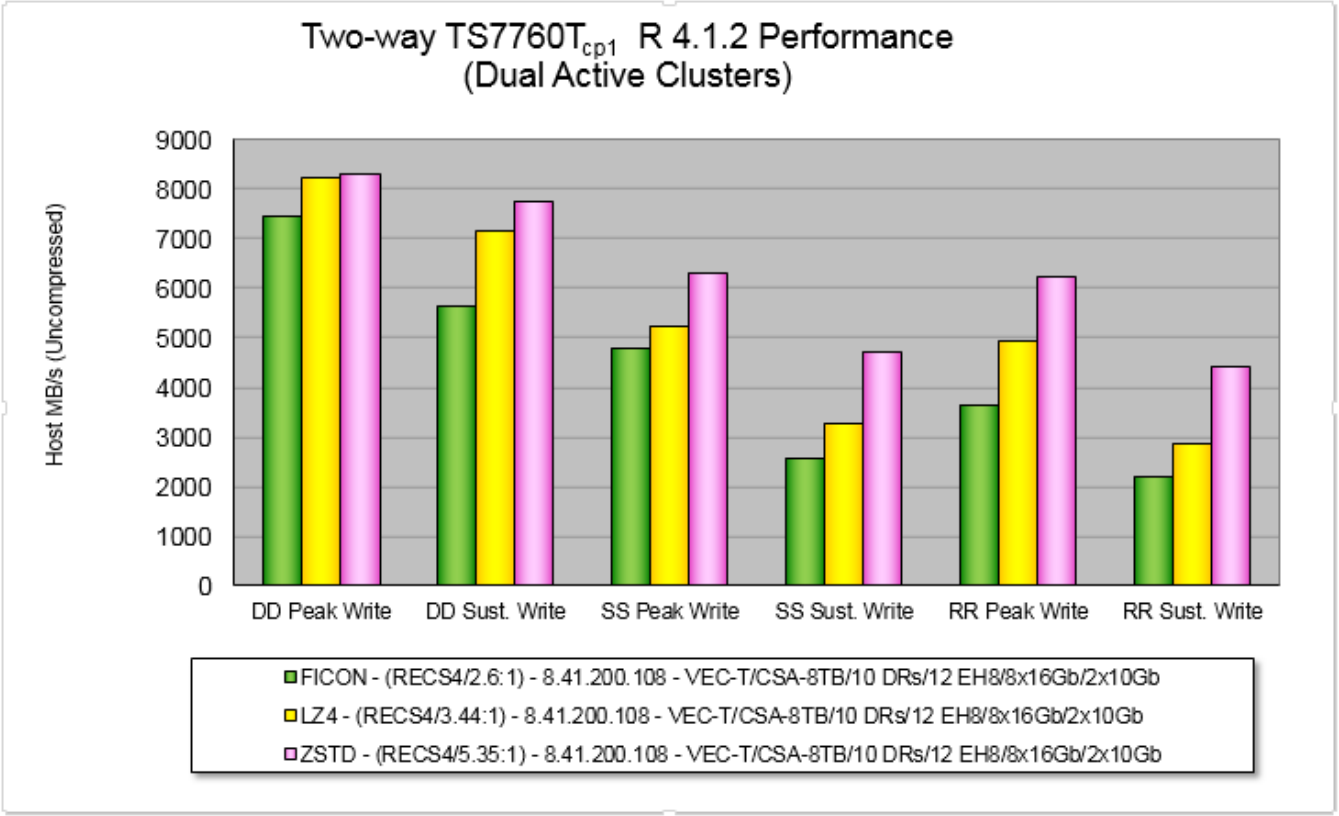
TS7700 Grid Performance
(8 x 16Gb FICON)

(Using RECS4 Data Patterns)



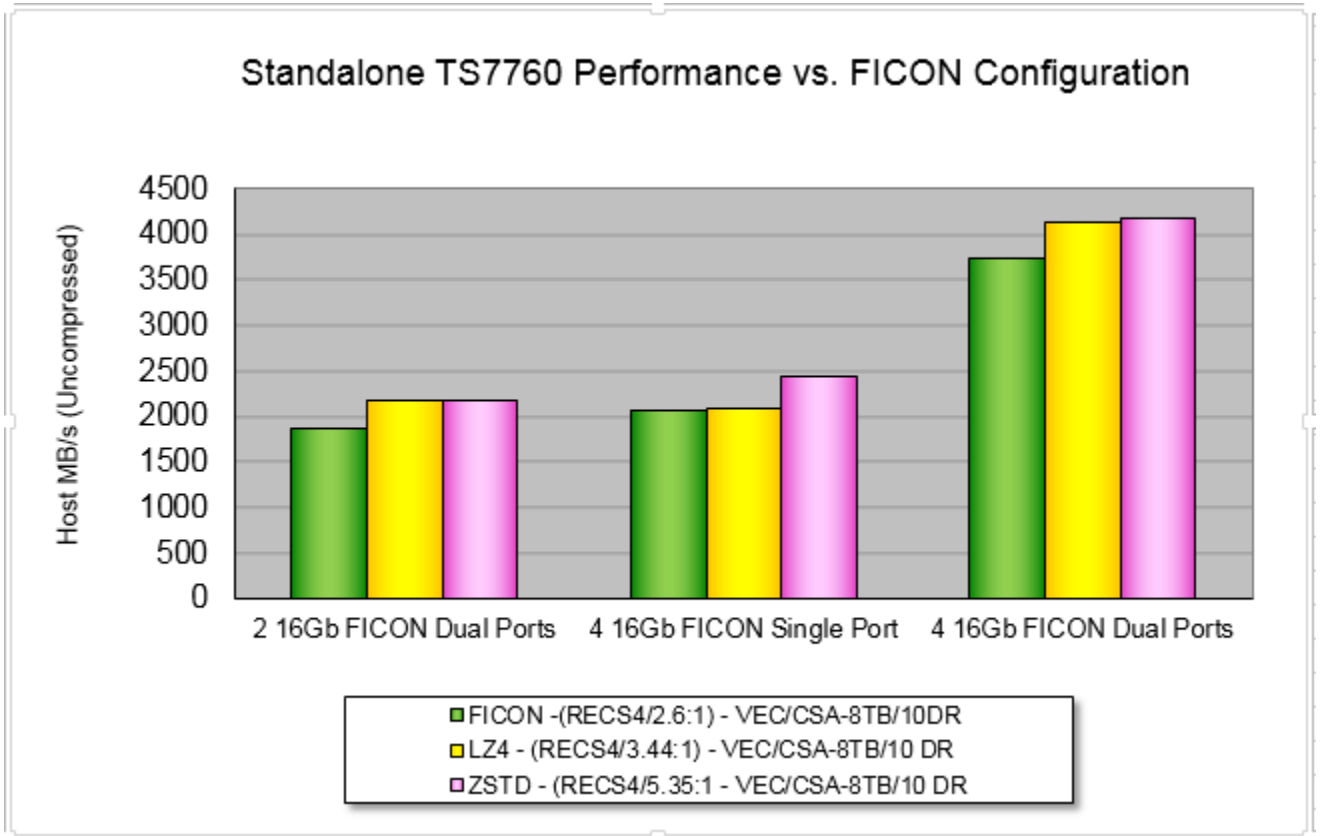






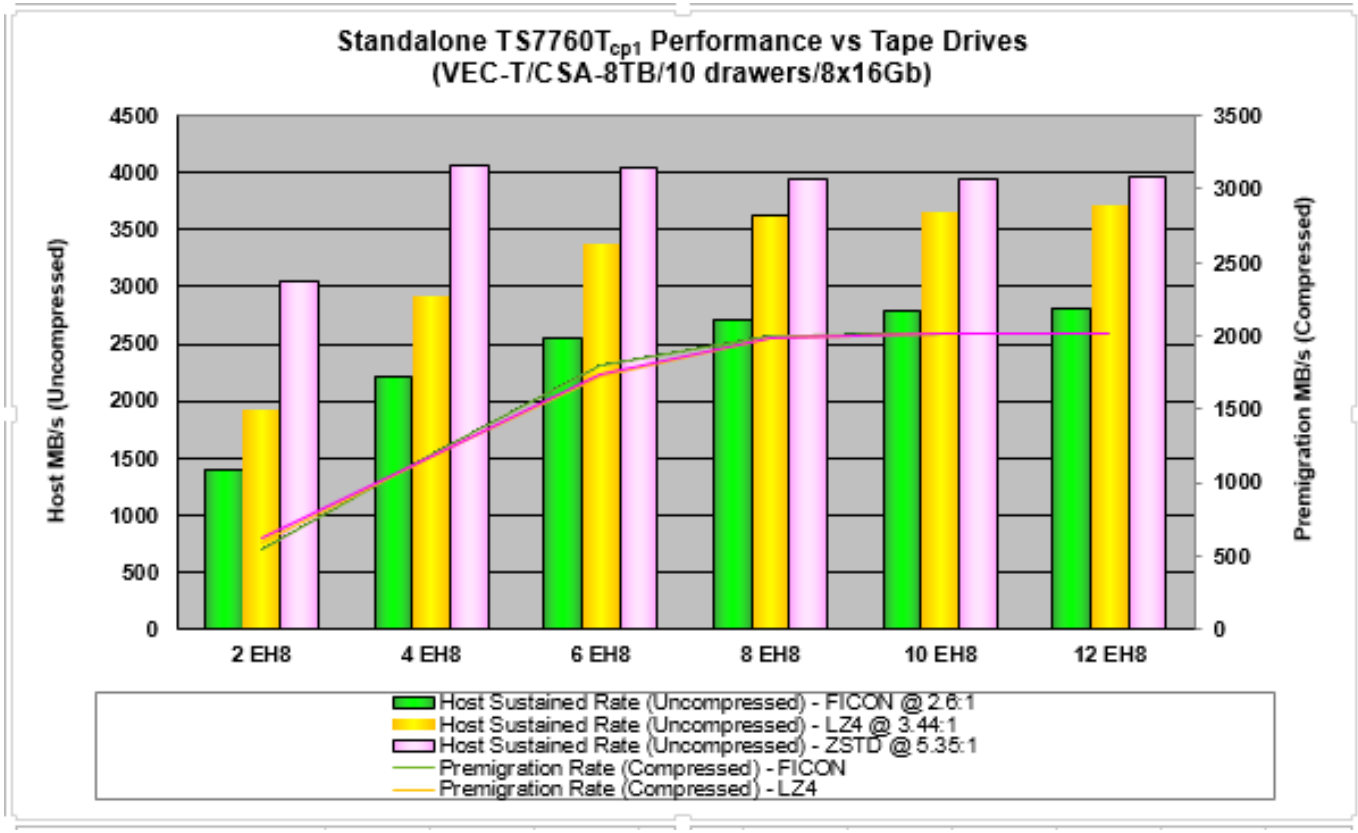
Standalone TS7760 Write Performance vs. FICON Configuration

(Using RECS4 Data Patterns)



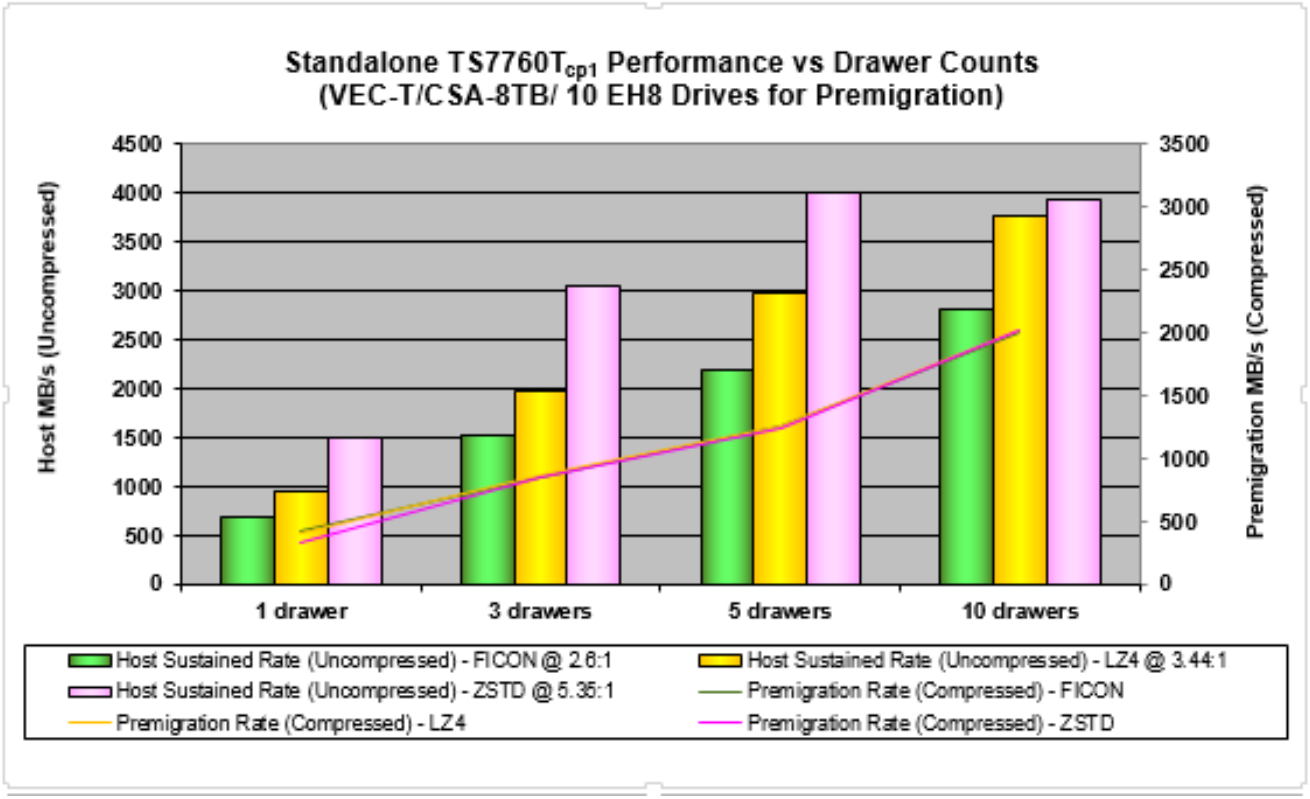
Standalone TS7760T$_{cp1}$ Performance vs. Tape Drives

(Using RECS4 Data Patterns)



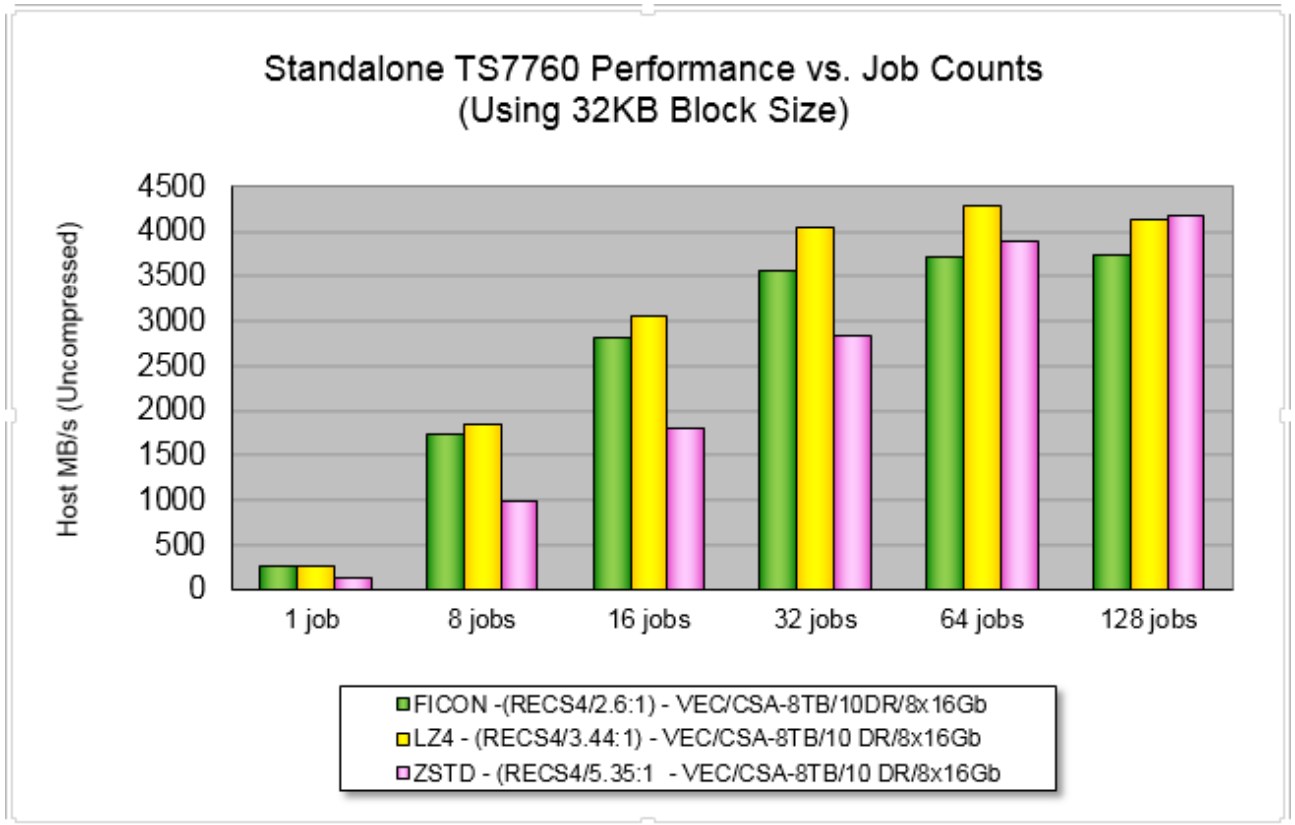
Standalone TS7760T$_{cp1}$ Performance vs. Drawer Counts

(Using RECS4 Data Patterns)



Standalone TS7760 Write Performance vs. Concurrent Job Counts

(Using RECS4 Data Patterns)



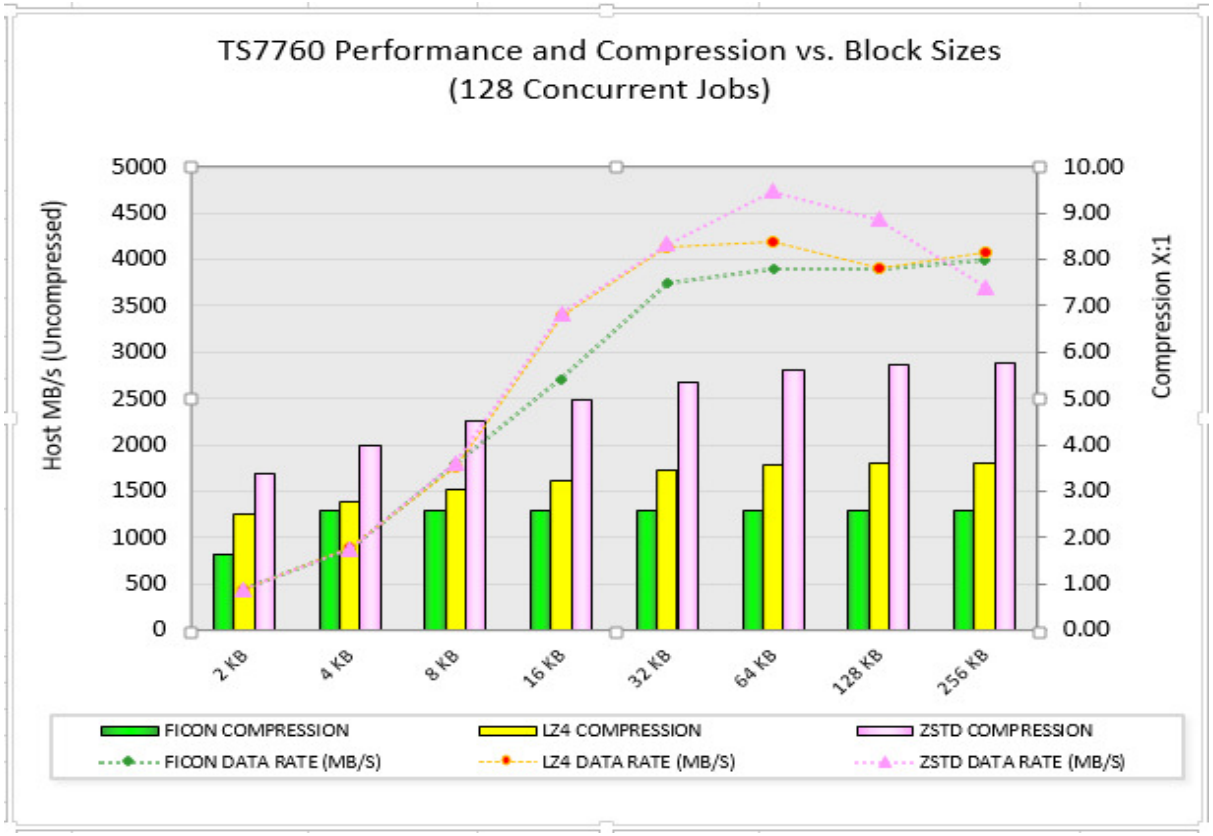
Discussion on the job count results with TS7760 and 32KB block size

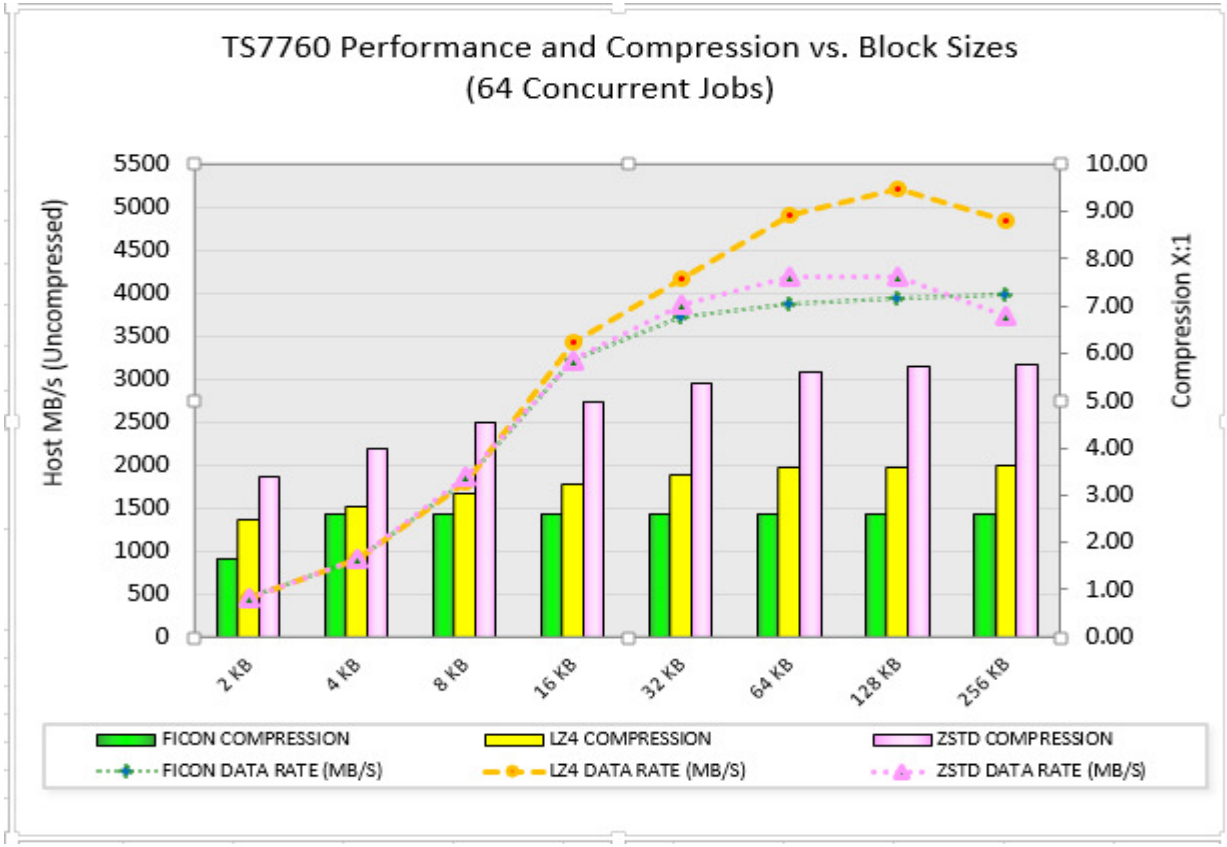
- The results may be surprising as to why the ZSTD workloads at smaller job counts are slower. This is a direct result of the overhead the ZSTD algorithm incurs. Assuming the solution is not disk cache bound, a single stream LZ4/FICON workload can run up to two times faster than a single stream ZSTD workload.
- If enough concurrent jobs are running, the cumulative performance of ZSTD equals LZ4 given the improved compression creates less overhead for data movement through the TS7700.
- For batch periods where very few active devices run in parallel, going with LZ4 would be ideal if performance is most important. If compression is more important for these lower active device periods, then ZSTD would be a better choice. You can also mix workloads so some use LZ4 while other use ZSTD.
- If many workloads are expected to run in parallel, then ZSTD is usually the best choice for both cumulative performance and compression. If some of these many jobs require higher performance, they can choose LZ4 so that they can run independently faster, but with less compression.
- Keep in mind that these comparisons are all relative to a maximum achievable throughput. If your configuration is not expected to exceed certain threshold (e.g. throughput increment limited), the difference between LZ4 and ZSTD may irrelevant.
- Very small drawer configurations can also produce results where lower job counts will actually run faster with ZSTD given the improved compression puts less strain on the limited disk cache.

Standalone TS7760

Performance vs. Concurrent Job Counts and Block Sizes

(Using RECS4 Data Patterns)





Thank You.

IBM Storage & SDI