NVMesh ELastic Erasure Coded Tiered (ELECT) Volumes

Summary of Design and PoC Phases

Technical White Paper

Table of Contents

Introduction **4**

Architectural Foundations **4**

Benefits of NVMesh EC 4

Node and Drive Failure Protection *4*

Scalability *4*

Elastic Logical Volume Management *5*

Support for Heterogeneous Hardware *5*

ELECT Volume Requirements 5

Performant Random Read Operations *5*

Conserving Write Endurance of QLC drives *5*

Design **5**

A Multi-Tiered Volume 6

Metadata Handling 6

MetaData Volumes *7*

Cache Handling 7

Copy-on-Write (CoW) Cache Collision Solution 7

New Capabilities 8

Endurance-Sensitive Volume Layout *8*

Fine-Granularity Writes to RAID-1 Volume *8*

Proof of Concept **8**

Limitations 8

Setup 9

Tests 9

Single Thread Sequential 512KB Writes *10*

Multiple Sequential 512KB Writes *11*

Multiple 4KB Read Streams *11*

Multiple 64KB Read Streams *12*

Writing During Multiple 64KB Read Stream *12*

4KB Read Latency During Writes *13*

File System Level Test - Multi-Stream Copying of Many Small Files *13*

Discussion 14

Comparison to Solutions w/o Optane Drives *14*

QLC Endurance Improvement *15*

Summary **15**

1 Introduction

There are two outliers in the spectrum of persistent storage drives: high-performance, low-latency Optane drives and read-intensive, low-cost and high-capacity QLC drives with PLC drives on the horizon. Currently, Excelero is concentrated on the mainstream TLC drives for NVMesh development. These drives provide the balance between sufficiently good write performance, excellent read performance and competitive cost structure, and therefore are suitable for most tasks.

However, the unique properties of the combination of Optane and QLC drives can be very effectively used for some workloads especially the ones that require massive random read capacity and ability to sustain bursts of writes. The modern Deep Learning training phase is a great example of such a workload.

2 Architectural Foundations

Intel QLC drives provide an excellent opportunity to create a cost efficient and, at the same time, performant and scalable solution when combined with NVMesh Distributed Erasure Coding (EC). This was the main driving force behind the ELECT project. The chosen approach preserves the main advantages of NVMesh's distributed EC while allowing it to use QLC drives. NVMesh EC uses TLC drives with long blocks (4096+8B) support to store metadata in a persistent way without any need for additional hardware. Intel QLC drives don't support long blocks. Therefore, we use QLC drives to store the data and Optane drives for the metadata. This design does not fork the subsequent development of the TLC-based EC code and allows for the two to co-exist. This approach will be described in the Design section below.

2.1 Benefits of NVMesh EC

We ensure that the combination of NVMesh EC and QLC/Optane preserve the benefits of the former while retaining the better economics of the latter.

2.1.1 Node and Drive Failure Protection

Unlike most of the solutions in the market, NVMesh EC is fully distributed so it can protect from multiple node failures as well as from multiple drive failures. This allows it to be used in converged, disaggregated and mixed deployments without requiring dedicated storage servers.

2.1.2 Scalability

NVMesh was built from the outset with scalability in mind. Both control and management planes are distributed. There is no cross-talk between the clients. The client side of NVMesh implements the data services but remains stateless, keeping state on the target side. The NVMesh protocol allows bypassing the CPUs on the NVMesh targets ensuring unlimited and highly efficient scalability with each client talking only to the targets holding pieces of volumes this client is attached to. Targets communicate with relevant peers to implement a highly-available and scalable control plane.

NVMesh has been successfully deployed in clusters with hundreds of physical servers and thousands of virtualized ones demonstrating linear scalability and almost perfect resource utilization.

2.1.3 Elastic Logical Volume Management

NVMesh allows carving logical volumes out of a physical NVMe pool according to various parameters related to failure domains and separation criteria. For example, it is possible to create an EC volume such that no two parts of it will reside on the same server or no more than two drives will reside in the same power zone. On the other hand, thanks to a flexible policy, NVMesh enables creating small footprint, single target volumes protecting only from drive failures.

Adding Optane drives into the mix allows using them not only as a metadata store for the ELECT volumes, but also to create super low latency volumes carved out of Optane drives.

2.1.4 Support for Heterogeneous Hardware

Being a purely software solution, NVMesh supports heterogeneous servers, drives and network equipment in the same cluster. This allows customers to start small and grow the storage solution according to the growing needs of a customer, which is especially relevant in such a dynamic and growing field as Machine Learning, while retaining the same beneficial price-performance ratio.

2.2 ELECT Volume Requirements

To make ELECT a natural enhancement of NVMesh, the same architectural principles have been employed. For example, stateless clients and efficient scalability mean there is no stateful client caching or passing of information between clients to coordinate cache use. On the other hand, incorporation of the new media requires special design considerations which we explore in this section.

2.2.1 Performant Random Read Operations

We already covered the exciting applications of QLC-based all flash solutions in the realms of AI and Analytics due to the excellent random read characteristics of QLC drives. The proposed architecture should expose these performance capabilities as much as possible while introducing the required data protection.

2.2.2 Conserving Write Endurance of QLC drives

The main limitation of QLC drives is their low write endurance. Therefore, one of the main emphasis of the design is to conserve write endurance and lower the Write Amplification Factor (WAF) of the solution.

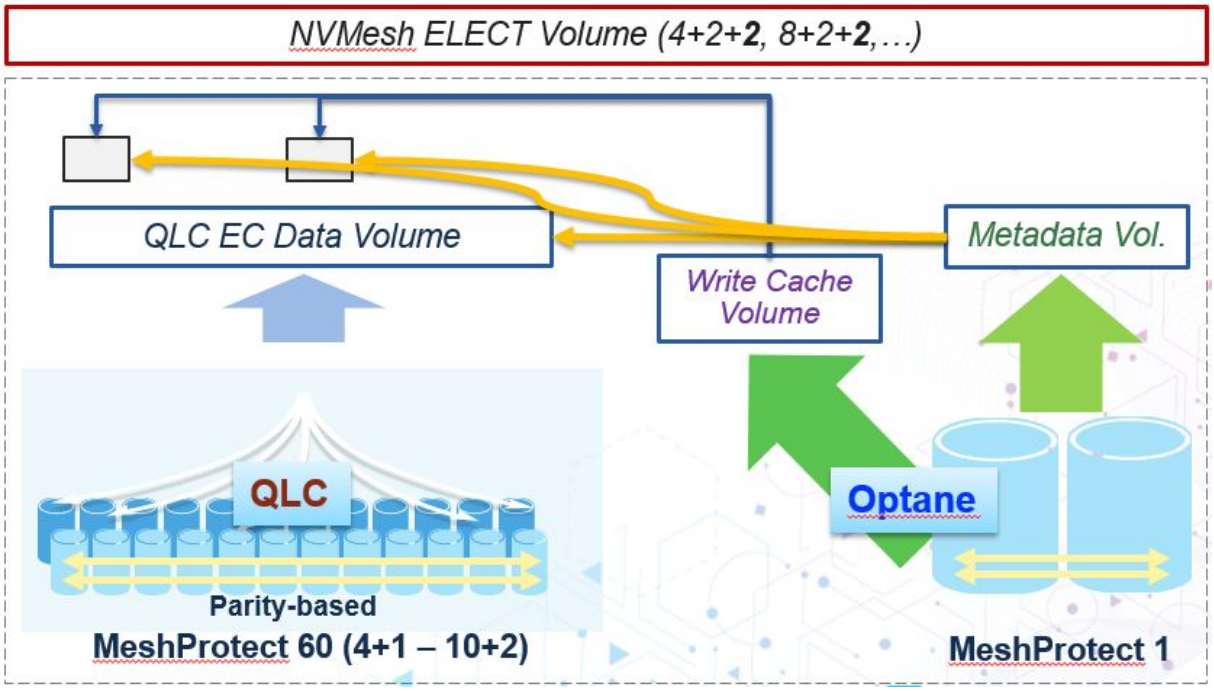
3 Design

The current NVMesh EC volume implementation does not require or assume a write cache or a layer to absorb and combine write operations. As is, this won't work well for QLC-backed volumes as it may generate small scattered writes that are detrimental for QLC drives. QLC write endurance characteristics require writing in large blocks and in as sequential a manner as possible. Therefore, it is imperative to have a write cache layer that will aggregate writes before writing them to QLC drives. As NVMesh does not support thin provisioning yet, this should work well for high locality write

workloads. These constitute the majority of high performance write scenarios with transactional databases being the only real exception.

3.1 A Multi-Tiered Volume

To conserve development efforts and leverage NVMesh's product quality, we chose to implement the write cache layer as a separate RAID-1 volume based on Optane drives while the QLC part is implemented on an EC volume with separated metadata. Therefore, the ELECT volume itself is represented by a new NVMesh entity called a Multi-Tiered Volume. The following diagram presents this solution.



3.2 Metadata Handling

Next we describe how the different types of existing and new metadata are handled. There are four types of metadata maintained in NVMesh in the 64-bits of metadata space attached to each page for 4KB+8B drives as well as in the separate Journal partition on each drive supporting an EC volume, as follows:

End-to-end Data Integrity Checking (EDIC) - this metadata plays a role similar to T10-DIF, to help protect the data from data rot and to enable End-to-End Data Integrity Checking.

Dirty Bit Persistency - this metadata keeps track of the blocks to recover when the system is required to write on a degraded volume.

Transactional Data Journal - this is used to store data being written before it is committed to an EC Volume blockset, i.e.a unit under a single lock. In this design, the Optane-based Write Cache volume performs this role so we don't need this metadata.

Bidirectional Journal-to-Data Links - these are used to connect the journaled transaction with the destination of the corresponding Write operation for roll-forward during recovery. In the presented design, reads go to the Optane-based cache until the data is fully written to the underlying blockset. Thus, we avoid using these links and can always roll-forward from the cache.

Reads and writes into NVMesh volumes are broken into single blockset operations. Therefore, the only metadata we keep from the NVMesh EC is much smaller than a 4KB page per full blockset of 1MB (in case of an 8+2+2 configuration) or 512KB (in the 4+2+2 case). The metadata required for bookkeeping of the Multi-Tiered Volumes fits into a single metadata page and requires only a single access to the Metadata Volume per read operation and per write to the Write Cache. The background destage operations writing the blocksets to the QLC drives will need to perform an additional write operation, but these are not on the critical synchronous path.

3.2.1 MetaData Volumes

In order to support fast, scalable, multi-client and coherent access to the metadata, we decided to implement it as a RAID-1 NVMesh volume on the Optane drives naming it the MetaData Volume or MDV. The 4KB-based access pattern to this volume will require a new feature to get the maximum performance in case of simultaneous access from multiple clients. This feature is discussed below in the New Capabilities section.

3.3 Cache Handling

We design a scalable system for multi-client high-performance scenarios. As shown in the Multi-Tiered Volume design diagram, the Write Cache allocations for all clients are parts of a Write Cache Volume or WCV which is a regular RAID-1 or RAID-10 NVMesh volume. Any client attaching to an ELECT volume will use a “sub-volume”,essentially an extent, carved out from this volume. A server side mechanism persists these allocations, similar to the mechanism used for the journal range allocation persistence used in regular NVMesh EC volumes. This volume can be extended like any other regular NVMesh volume providing more Write Cache space in case there are a large number of clients accessing ELECT volumes.

3.3.1 Copy-on-Write (CoW) Cache Collision Solution

Preventing inter-client communication for real scalability is a staple of the NVMesh architecture. So, we faced a dilemma on how to approach the write cache collision issue without breaking the scalability. Write cache collision happens when a write to a blockset, the basic locking or handling unit of the underlying QLC EC volume, was committed to client A’s write cache area, but not yet destaged to the QLC drives and the same blockset is being written at the same time from client B. For true performance at scale, we do not want to serialize multi-client writes on destage. However rare such a scenario is, such an approach is against NVMesh architecture principles.

So the solution is that when client B discovers a write cache collision from the MDV page, it performs a regular read of the current data,actually from client A’s write cache, and transfers ownership of the whole new data for this blockset to itself. When the time comes to destage the data, client B will destage both the data originally written to client A’s write cache area and client B’s new data.

Since blocksets are small, 1MB in the 8+2+2 case, this scenario should be very rare in any real world usage, but the approach to it demonstrates the flexibility of NVMesh's design with its stateless client/stateful servers approach for scalable and performant storage.

3.4 New Capabilities

There are new capabilities required from different volumes that together constitute a Multi-Tiered Volume:

3.4.1 Endurance-Sensitive Volume Layout

The current NVMesh EC implementation is focused on minimizing the need for reads during write operations aligned with RAID parameters to get consistently low latency even when writing to a distributed EC volume. he volume layout is row-first, effectively ensuring that a 32KB write to an EC 8+2 volume will be written to all drives without the need to read existing data at all. However, this approach does not work well for QLC drives where the additional latency of read operations is masked by the Write Cache layer and so is of less importance than large write IO operations in case of non-full destage from the Write Cache.

On the other hand, from the write endurance perspective, it is very important to perform such a 32KB write to one or two drives as 1x32KB or 2x16KB writes since this improves the internal Write Amplification Factor (WAF) of the drives. Therefore, together with ELECT volumes, we have developed the ability to specify the height of an EC stripe in 4KB pages in NVMesh EC adjusting the layout to the particular drive WAF characteristics.

3.4.2 Fine-Granularity Writes to RAID-1 Volume

As mentioned above, clients access the MDV with a 4KB granularity pretty much randomly. Since we leave the NVMesh locking mechanism untouched, this means the operations lock 32 metadata pages with a single lock. So we need to make sure that the duration of this lock is minimal. Taking into account that these are metadata operations usually meant to update several fields in a page, we designed a server-side operation which performs the required update under lock and releases the lock without passing the data back and forth between the client and the server under lock. The updated data is returned to the client for bookkeeping and metadata caching.

4 Proof of Concept

We developed a limited-scope version of ELECT volume support to verify and demonstrate the design and to be able to share its benefits, limitations and roadmap with selected partners to align our development plans with the field.

4.1 Limitations

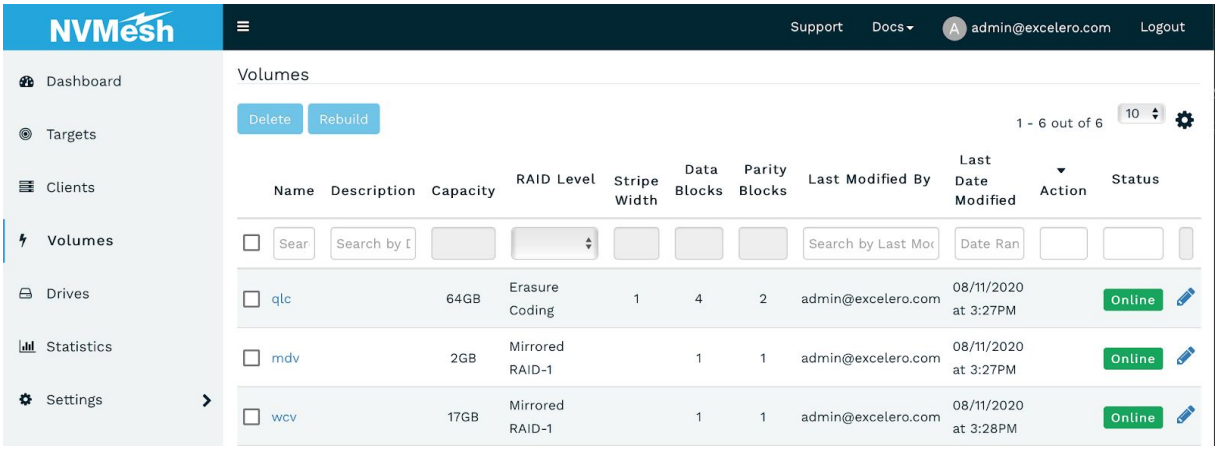
The PoC version has several limitations, which need to be taken into account when assessing the testing results.

Single client: We did not implement the CoW mechanism to cope with multiple clients so all the tests are performed from a single client, actually from a converged client.

Separate metadata areas for existing and new types of metadata: The existing NVMesh EC metadata (the two types of it we keep for ELECT) was not merged with the Write Cache bookkeeping metadata designed for ELECT volumes, which affects write performance since we need to write to two different pages in the MDV.

No endurance-sensitive volume layout for QLC EC volumes: This means we cannot demonstrate on the current PoC the efficient, WAF-sensitive partial destage from the Write Cache Volume. Instead, all writes are performed for the full blocksets (512KB ones in case of 4+2+2 which is what is tested in the PoC setup) and so the missing data is read from the QLC drives before destaging to ensure drive endurance is preserved.

No management support for ELECT volumes: All the underlying NVMesh volumes are created manually and connected together manually in the NVMesh client itself. Eventually, the user will only see the ELECT volume itself, but in the PoC we are able to observe all 3 NVMesh volumes comprising the tested ELECT volume:



4.2 Setup

For the tests below we used an Intel server based on the S2600WFT (Wolf Pass) board with dual Intel Xeon 6226R Gold CPUs and 6 Intel D5-P4320 7.68TB QLC drives and 2 Intel DC-P4800X Optane drives in a 4+2+2 configuration.

The NVMesh version used for the PoC was based on NVMesh 2.0.3 with the corresponding additions to support multi-tiered ELECT volumes, as described above.

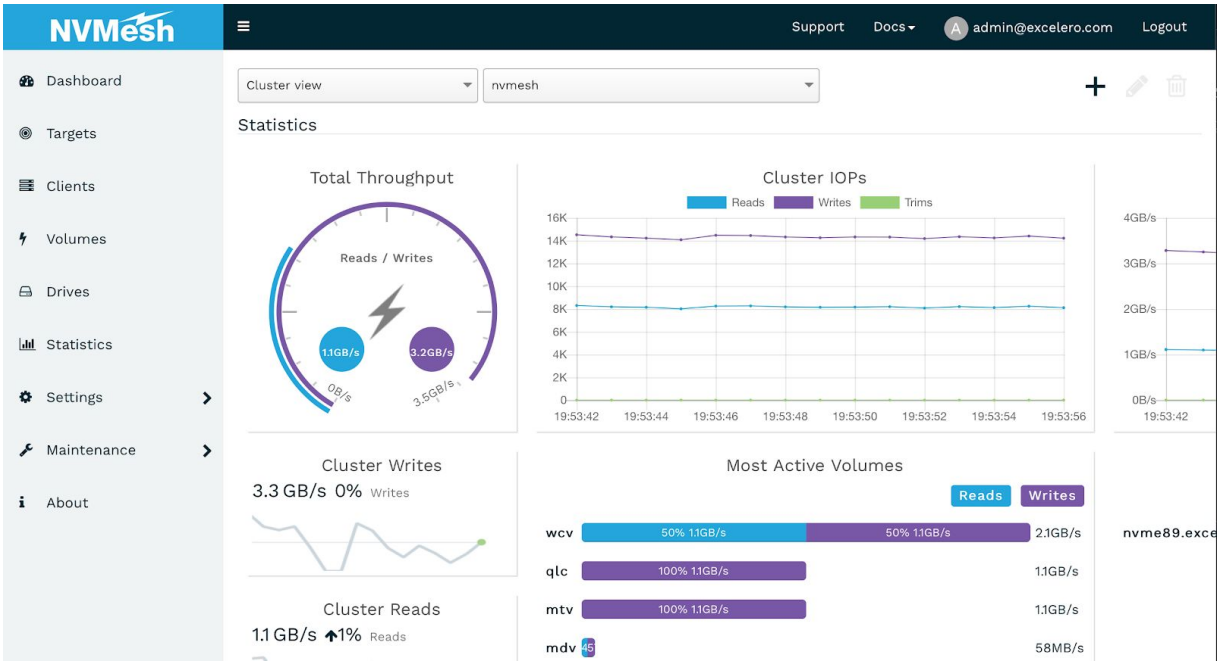
4.3 Tests

In the context of this PoC, we concentrated on block level tests to show that the results adhere to the design and behave according to the performance of the underlying drives and to the limitations of the PoC code. In addition we conducted a file system level test to show advantages of ELECT volumes for a typical AI small file dataset case.

In all the tests, the destage code is tweaked to run all the time to demonstrate the worst case scenario in which the write operations are extended beyond the burst capacity of the write cache.

Single Thread Sequential 512KB Writes

We start from a single sequential write test of 512KB (full blocksets). This test should provide a baseline for the write performance and demonstrate consistent performance:

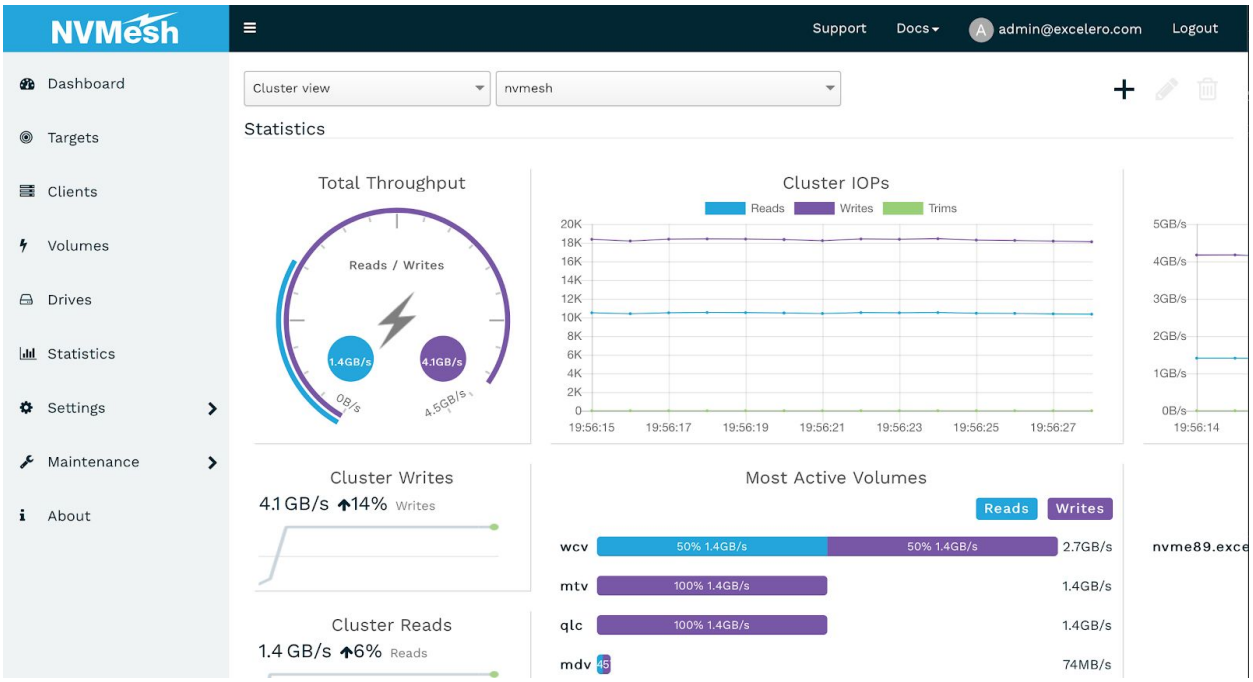


The ability to observe the underlying volume provides clear demonstration that the data written to the ELECT volume ‘mtv’ is being written to the Write Cache volume ‘wcv’ first and then is being read from it to be destaged to the QLC EC volume ‘qlc’. The metadata access is very small compared to the data operations.

The average latency is under 0.5 msec and is very consistent. The bandwidth is consistent as well during the whole test.

Multiple Sequential 512KB Writes

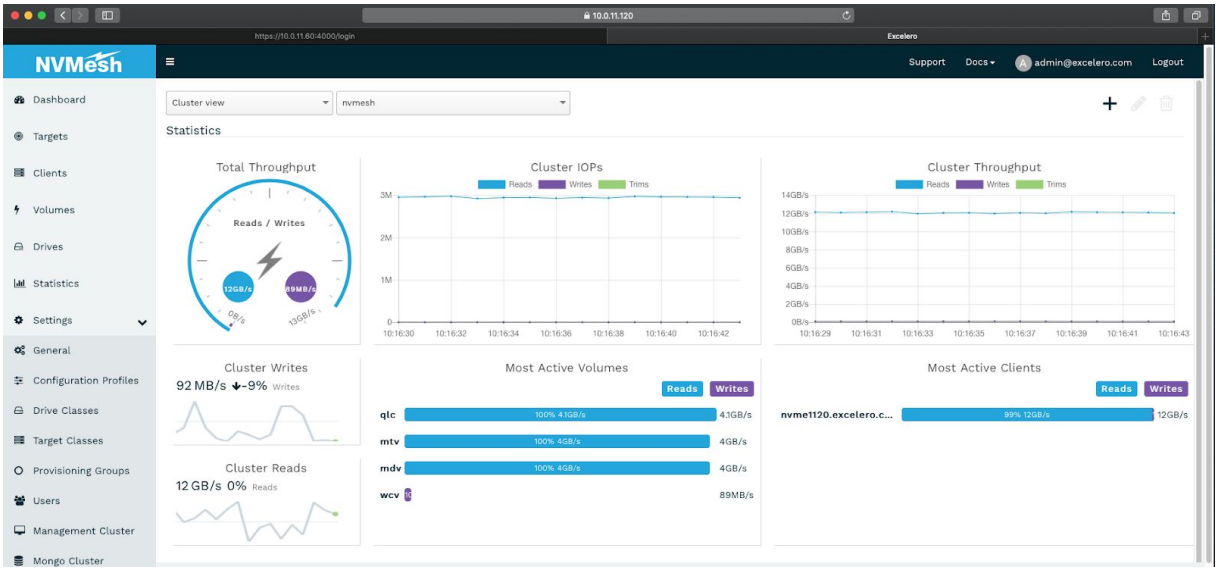
This is very similar to how real world applications work on top of a cached file system.



The limitation on bandwidth comes from the Optane drives which are limited to 1.4 GB/s. Once again the performance is very consistent.

Multiple 4KB Read Streams

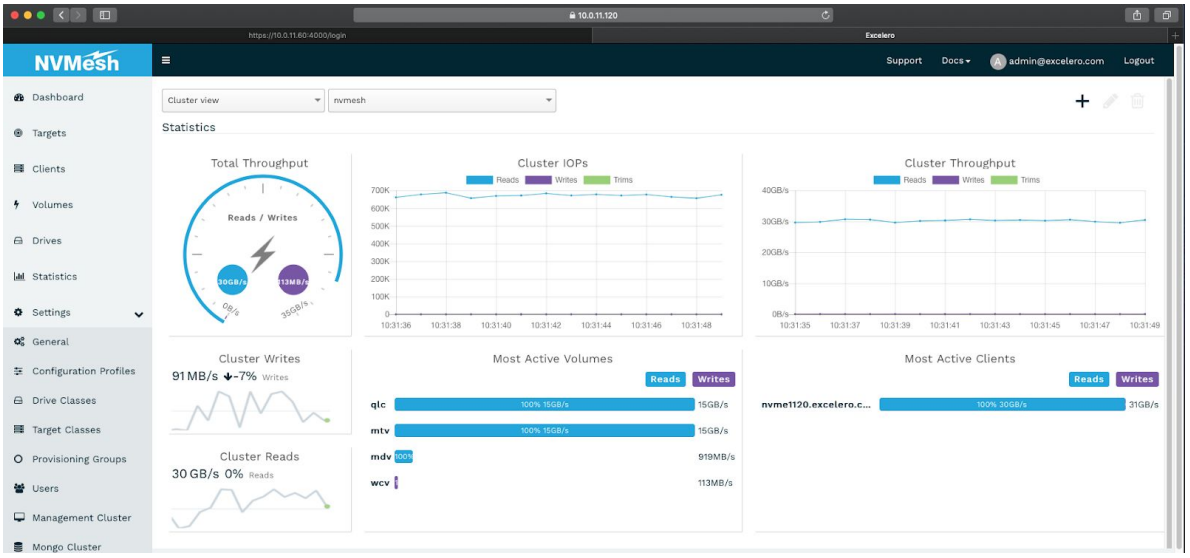
With the current generation of Optane drives, their ~500K 4K read IOPS per drive (or 1M from RAID-1 volume) limits the read performance of this scenario to 1M reads from the ELECT volume and this is exactly what we demonstrate in this random read test.



The next generation of Optane drives drastically raises the 4K IOPS to over 1.6M per drive. Since 6 QLC drives provide much more IOPS than this, in the GA solution we should expect excellent random read performance even from the smaller setups.

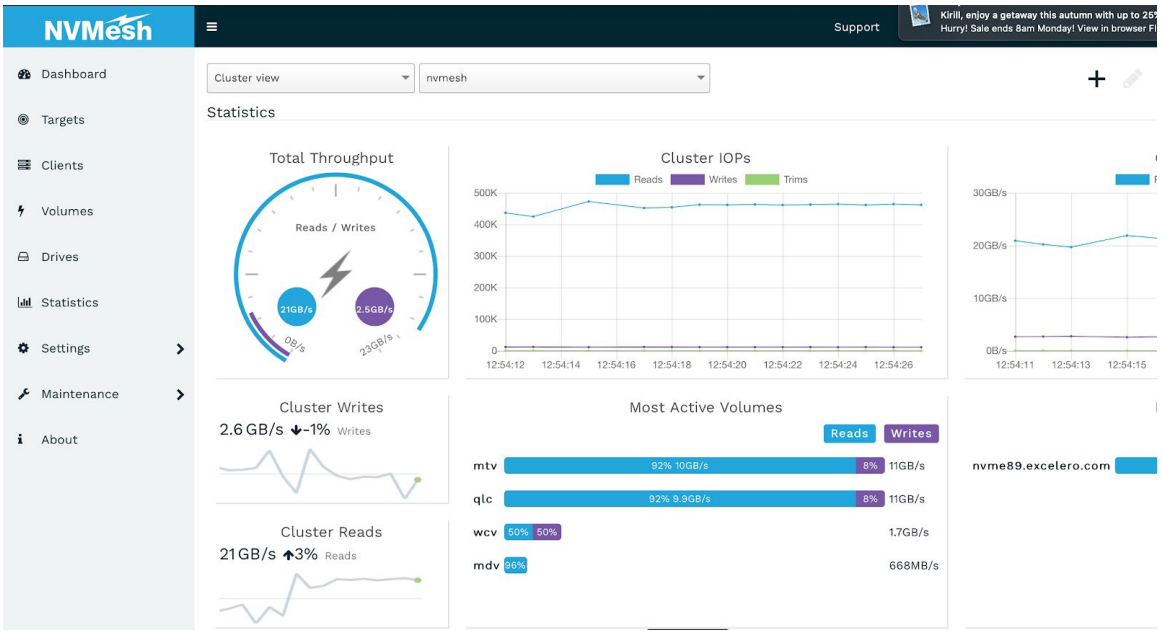
Multiple 64KB Read Streams

When increasing the size of a read IO to 64KB, the performance limit shifts from the Optane to the QLC drive. This is clearly seen in this test where we're able to get a sustained bandwidth of 15GB/s from 6 QLC drives, very close to their random read specification.



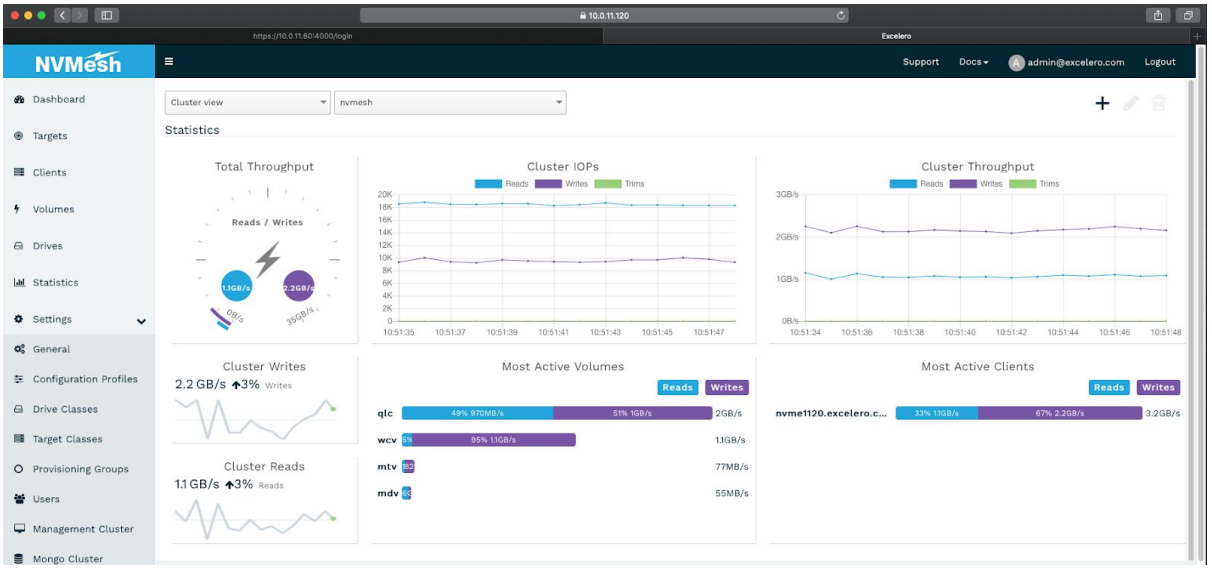
Writing During Multiple 64KB Read Stream

We combined the previous test with a single sequential write stream to see how write pressure affects read bandwidth and latency. The following snapshot demonstrates that while the writes reached over 1GB/s reads reduced only by 30% to 10GB/s and the read latency remained <1ms.



4KB Read Latency During Writes

This is one of the most widely used tests in the industry, assessing how write pressure affects latency of single read threads, to ensure that reads won't wait for writes to complete so the transactional latency, and therefore the number of transactions per seconds, won't be affected by writes from another client.

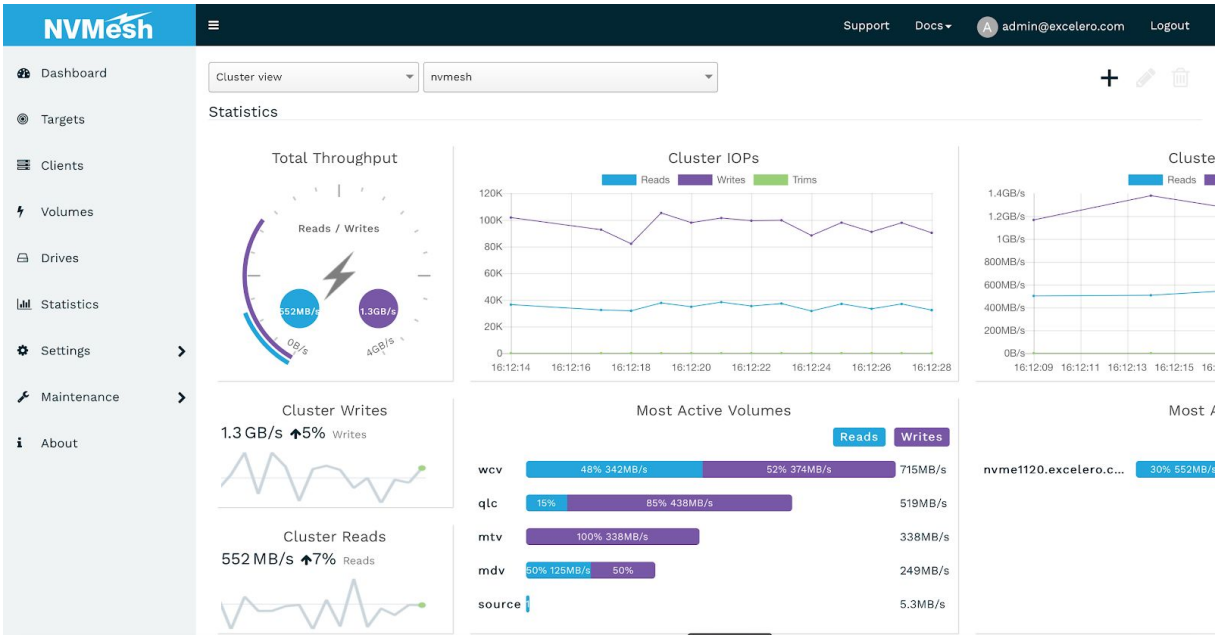


Single-thread read latency is 240 usecs and with a write load of over 70% of maximum, the latency grows to 308 usecs, i.e. an impact of less than 30%. It also remains very stable with minimal jitter.

File System Level Test - Multi-Stream Copying of Many Small Files

We filtered just over a million small files (between 6 and 30 KB each with an average size of 18KB) from a real world Imagenet dataset widely used for Deep Learning benchmarks. All files were put in a single directory. The test consists of 20 simultaneous copy operations of this directory into separate directories on an ELECT 8+2+2 volume. This test is constructed to stress the destage and write aggregation mechanisms. With the PoC code it took 22 minutes and 31 seconds to complete all 20 copy operations which means that from a user perspective it ran with over 266 MB/s average external bandwidth.

As can be seen from the snapshot below, QLC operations were mostly writes which means that the write aggregation to the Write Cache entries worked as designed.



The system also demonstrated consistent bandwidth during the whole test.

4.4 Discussion

The tests demonstrate that the design passes the basic block level tests and provides a solid foundation for the work on a GA version of the product with this functionality. Most of the measured numbers were sufficiently close to the limits calculated from the drive specifications and the test results variability was very low for PoC-level code.

4.4.1 Comparison to Solutions w/o Optane Drives

To put the presented results into perspective, we compare them to best-case baseline scenarios that do not involve any fast and persistent aggregation layer, regardless of whether they are based upon battery-backed DRAM, NVRAM or Optane. The latter provides the best price/performance ratio.

For read operations, it is clear that ideally the performance should be limited only by the drives comprising the EC volume. This is the case with NVMesh EC volumes based on TLC drives with metadata support. Read performance of QLC drives is very similar to TLC drives. In the worst case, for every read operation we need to access the Metadata Volume at least once. This means we need to have drives that satisfy the following inequality, with D = 10 for an 8+2+2 configuration:

$$2 \times [\text{Optane Read IOPS}] > D \times [\text{QLC Read IOPS}]$$

For the current family of Optane drives (48xx), this clearly does not hold true. Optane drive's read IOPS are slightly lower than QLC read IOPS. However, the next generation of Optane drives provides much higher performance, especially for 512B blocks. How much improvement is required to satisfy the equation above? An improvement of 6x for 512B read compared to the current Optane drives will be enough. According to industry reports, the new Optane drives will provide even more than this.

For write operations, aggregation of stripes in the Optane-based Write Cache Volume allows performing full stripe write operations to the underlying EC volume unlike the baseline case. The advantage of this is clearly demonstrated by the file system test above where the performance shown by the ELECT volume is very close to the maximum 300 MB/s limited by the file system capabilities, as observed by us on any volume type, even unprotected striped layouts akin to RAID-0. This is higher than 240 MB/s, which is the maximum observed from an EC volume based on TLC drives without such an aggregation layer.

4.4.2 QLC Endurance Improvement

I've run an intensive testing of Intel's QLC drive's endurance under various regimes using Intel MAS tool and the result is as follows:

1. Our first GA solution essentially transforms any random write mix to a small number (4-8) of streams of 128KB writes per drive. Under such a regime the drive reaches 0.22-0.25 DWPD with 4-4.1 WAF (write amplification factor). This is the expected DWPD of our first GA (up from 0.1 DWPD of the raw drive).
2. It's important to emphasize that sequential write accesses (no matter what size) are transformed to sequential 128KB writes and under this assumption our GA solution will reach 1 DPWD with 1.11 WAF.
3. When we complete the development of NVMesh Thin Provisioning currently planned to 2021 the projected DWPD of the solution should become closer to 0.8-1 DPWD since we'll be able to transform any random write sequence to several sequential write streams of 128KB

5 Summary

NVMesh ELECT volumes provide an important extension to the architecture of the most performant and scalable block storage on the market. Their excellent price/performance characteristics allow expanding the applicability of all-NVMe deployments to the performance hungry use cases of AI, Machine Learning and Data Analytics, even where previously the economics forced customers to accept trade-offs.

We demonstrated the viability of the proposed architecture on a series of block level tests demonstrating excellent performance from a small converged single-server. Even with 64KB reads such a small setup reaches 15 GB/s bandwidth with a millisecond latency. Such a server can be easily deployed using NVMe-over-Fabrics access to support multiple GPU servers without levying a heavy storage tax and allowing easy and seamless expansion of the storage capacity when needed.