Line-Rate Compression on Lustre/ZFS-based Parallel Filesystems using NoLoad® Computational Storage Processor

Eideticom's NoLoad CSP achieves higher Compression Ratio and Throughput than software while using 70% less CPU

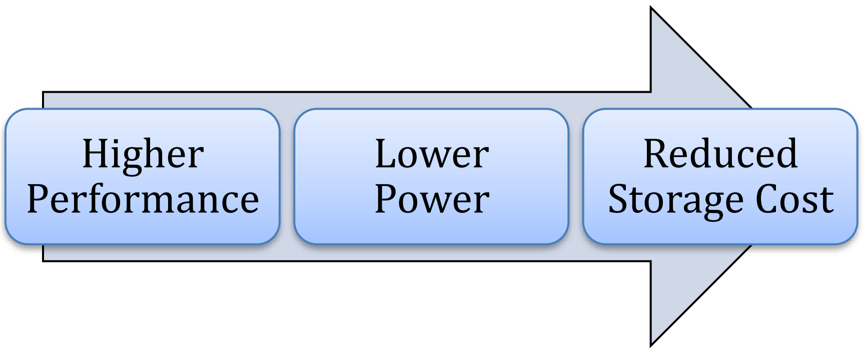
Executive Summary

- Eideticom partnered with Los Alamos National Laboratory (LANL) to explore the benefits of NVMe-based Computational Storage for ZFS-based parallel file systems when applied to High-Performance Computing.
- Eideticom's NoLoad Computational Storage Processor (CSP) integrates directly into ZFS and provides line-rate compression.
- NoLoad enables Reduced Storage Cost (50% lower total cost) by increasing storage capacity, reducing server count, optimizing I/O and maximizing storage lifetime.
- NoLoad provides higher performance by transparently offloading the host CPU (70% less CPU utilized) from compute intensive compression operations, providing vastly improved application performance and superior Quality of Service (QoS).
- NoLoad drives best-in-class power efficiency (60% lower power) as NoLoad compression is dramatically more CPU efficient than software alternatives (gzip-9 and lz4-1).
- **Eideticom's NoLoad CSP provides hardware-based compression that enables increased capacity without sacrificing performance.**

Overview

Eideticom partnered with Los Alamos National Laboratory (LANL) to explore the benefits of NVMe-based Computational Storage Processors (CSPs) and identified parallel filesystems as a place to exploit them. This is because parallel filesystems are very sensitive to performance and have aggressive cost considerations.

Eideticom and LANL collaborated on the integration of the NoLoad® CSP software stack into ZFS and on deploying NoLoad-enabled hardware on servers inside LANL and gathering performance data on those servers. This white paper presents the results.



1 Introduction

In this white paper, we discuss how Eideticom's NoLoad Computational Storage Processor (CSP) can be used to build very performant and cost-effective HPC parallel filesystems. We outline how to tie an NVMe-based CSP into the software of the parallel filesystem and the benefits of such a tie-in by performing product trials on LANL storage servers.

High-Performance Computing (HPC) relies on parallel filesystems to move data off compute nodes and into a storage backend in a performant manner. Since the compute nodes in an HPC system can quickly generate terabytes of data a second, the filesystems used must be performant. Failure to do so can result in bottlenecks that do not allow the HPC system to operate at its full potential.

2 NoLoad CSP: An NVMe-based Computational Storage Processor

NoLoad® is an NVMe-based Computational Storage Processor (NVMe-based CSP) developed and sold by Eideticom. The NoLoad CSP takes the form of an FPGA accelerator card that performs a variety of storage-centric offload functions while presenting an NVMe compliant PCIe interface to the host. As such, we developed a NoLoad software framework to allow applications, such as filesystems, to offload key storage tasks to the NoLoad. This offloading leads to improved performance and efficiency and reduced costs for the storage system. In this section, we provide an overview of the NoLoad CSP and describe the NoLoad hardware and software stack.

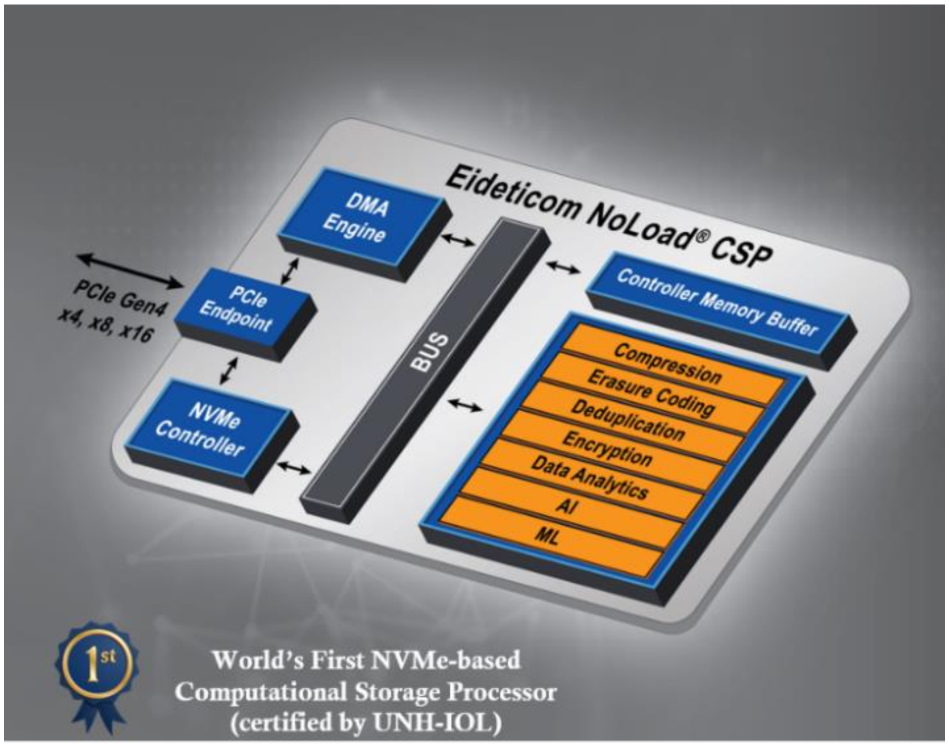


Figure 1: Eideticom's NoLoad Computational Storage Processor (CSP).

3 NoLoad Overview

The NoLoad CSP is purpose-built for the acceleration of storage and compute-intensive workloads**.** NoLoad eliminates performance bottlenecks by embedding processing power directly into the storage fabric. Eideticom's NoLoad supports several acceleration functions, ranging from Compression, Encryption, Erasure Coding, Deduplication and Data Analytics.

NoLoad presents acceleration functions to a host via a standard-compliant NVM Express PCIe interface. There are several reasons Eideticom chose an NVMe interface instead of a proprietary PCIe interface. These include:

- The NVMe driver is inbox in all major operating systems and the Linux NVMe driver is tuned for latency, performance and efficiency. Competing accelerator solutions utilize custom drivers which require porting or may not even work under all operating systems.
- NoLoad's NVMe compliant interface provides seamless integration for all CPU platforms and has been validated on Intel, AMD, ARM and IBM Power8/9 CPUs and at PCIe Gen 3 and Gen 4 rates.
- By choosing NVMe, we can leverage a rich ecosystem of open-source tools. In addition to drivers, this includes user-space management tools (e.g. nvme-cli) and user-space frameworks like SPDK.
- Aligning with NVMe allows us to implement and deploy on any PCIe enabled FPGA card and use a common software stack across all form factors.
- We wanted to align with a performant storage protocol as part of the SNIA efforts around Computational Storage. NVMe is the perfect choice for this.
- NVMe has a rich management specification we leverage to manage NoLoad at scale.

4 NoLoad Hardware

Eideticom NoLoad CSP supports deployment on any PCIe-enabled FPGA card. Two examples are:

- The Xilinx Alveo U50 PCIe-Enabled FPGA card (NoLoad-U50) is a PCIe Gen3x16, or dual Gen4x8 developed by Xilinx and includes a Virtex Ultrascale+ XCU50 FPGA and 8GB of High Bandwidth Memory (HBM). A typical 2 Rack Unit (RU) server accommodates two NoLoad-U50s.
- The BittWare 250-U2 FPGA Accelerator Card (NoLoad-U2) is a PCIe Gen3x4 enabled FPGA card developed by BittWare and includes a Xilinx Kintex Ultrascale+ KU15P FPGA and 16GB of DDR4 DRAM. Most servers today can accommodate a number of these devices, and some storage servers can provide as many as 24 in a 2 Rack Unit (2RU) server

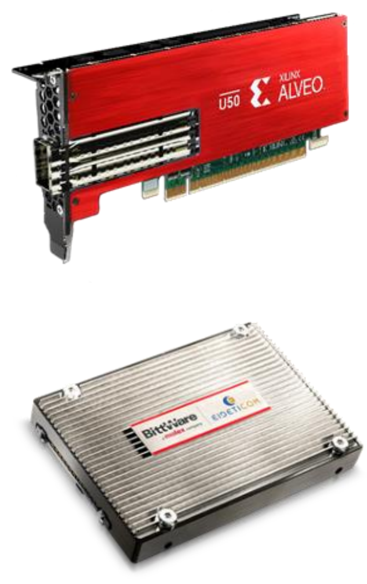


Figure 2: NoLoad enabled Xilinx Alveo U50 and Bittware 250-U2

5 NoLoad Software

Eideticom's rich suite of software enables the connection between acceleration functions on the NoLoad FPGA cards and applications running on a host CPU. In Figure 3, we showcase the different software components Eideticom has developed. We discuss some of these in more detail:

- **libnoload**. A user-space library that allows applications to leverage the computational storage services offered by NoLoad. Note that applications need to be updated to access the libnoload API before being offloaded.
- **ZFS on Linux**. A modified version of the upstream ZFS on Linux that can offload key parts of ZFS onto NoLoad devices. Eideticom software extends the NVM Express driver to comprehend computation as well as storage.

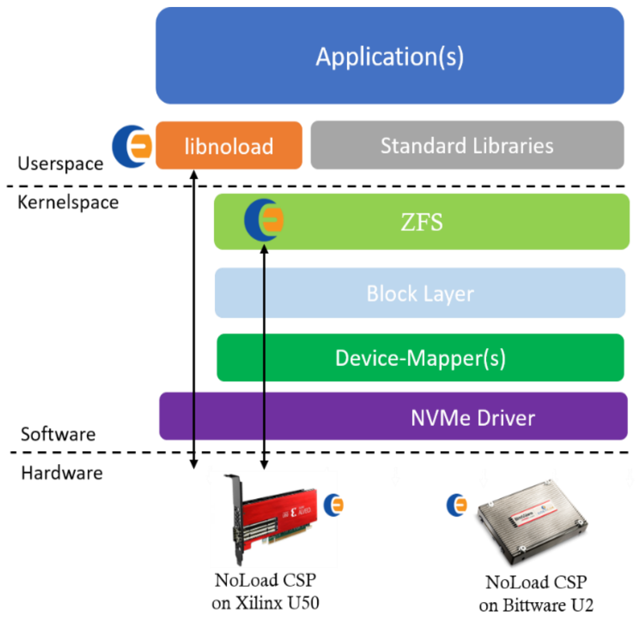


Figure 3: NoLoad Software Stack

6 NoLoad System Configuration

In this white paper, we discuss integrating the Eideticom NoLoad® NVMe-based CSP into a parallel filesystem (namely, Lustre on ZFS on Linux). Figure 4 shows the server configuration for the NoLoad deployment. The server consists of two CPUs connected via their chip-to-chip coherency bus. NVMe SSDs are directly connected to each processor via the PCIe bus, and the NoLoad U50 and NoLoad U.2 is attached to one CPU with a high-performance Network Interface Card (NIC) connected to the other.

NoLoad performance scales linearly with the number of NoLoad devices in the system. Additionally, NoLoad-U2s and NoLoad-U50s can be seamlessly mixed in the same system to achieve the expected combined throughput performance. The Eideticom NoLoad software stack discussed in Section 5 is capable of discovering and utilizing all the NoLoad devices in a system even when the devices are deployed on different hardware FPGA cards.

Regardless of form-factor, the NoLoad NVMe-based CSP always presents an NVMe interface to the host and the software consumption model remains the same.

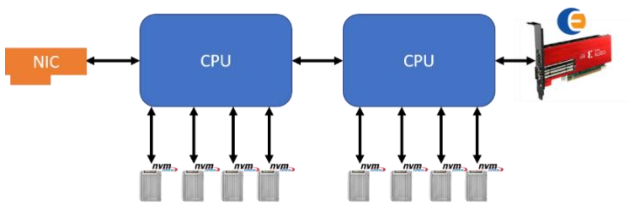


Figure 4: NoLoad System Configuration

7 NoLoad Compression

In this white paper, we are most interested in the offloading of compression to improve the performance and cost-efficiency of a ZFS-based parallel filesystem. As such, it is useful to baseline the NoLoad compression performance outside of ZFS. In Figure 5, we present compression results for several datasets associated with different application spaces. The server configuration was the Intel Xeon Silver 4210 running Ubuntu 18.04.

For each compression algorithm, we present the Compression Ratio (CR), the compression input throughput (in MB/s) and the compression input throughput per CPU core (MB/s/core). We present results for NoLoad compression, a high throughput software compression algorithm (lz4 level 1) and a high-compression software algorithm (zlib level 9).

8 Results Conclusions

The main conclusion from Figure 5 is that NoLoad-based compression can provide significant Compression Ratio (CR) advantages over lz4-1, while achieving 2-6 times higher throughput and 3-6 times better efficiency.

NoLoad's compression ratio is comparable to that of gzip-9 but achieves throughputs and efficiencies that are orders of magnitude better than software gzip-9. It is this combination of improvements in compression ratio, throughput (MB/s) and efficiency (MB/s/core) that makes NoLoad attractive to customers like LANL for their parallel filesystems.

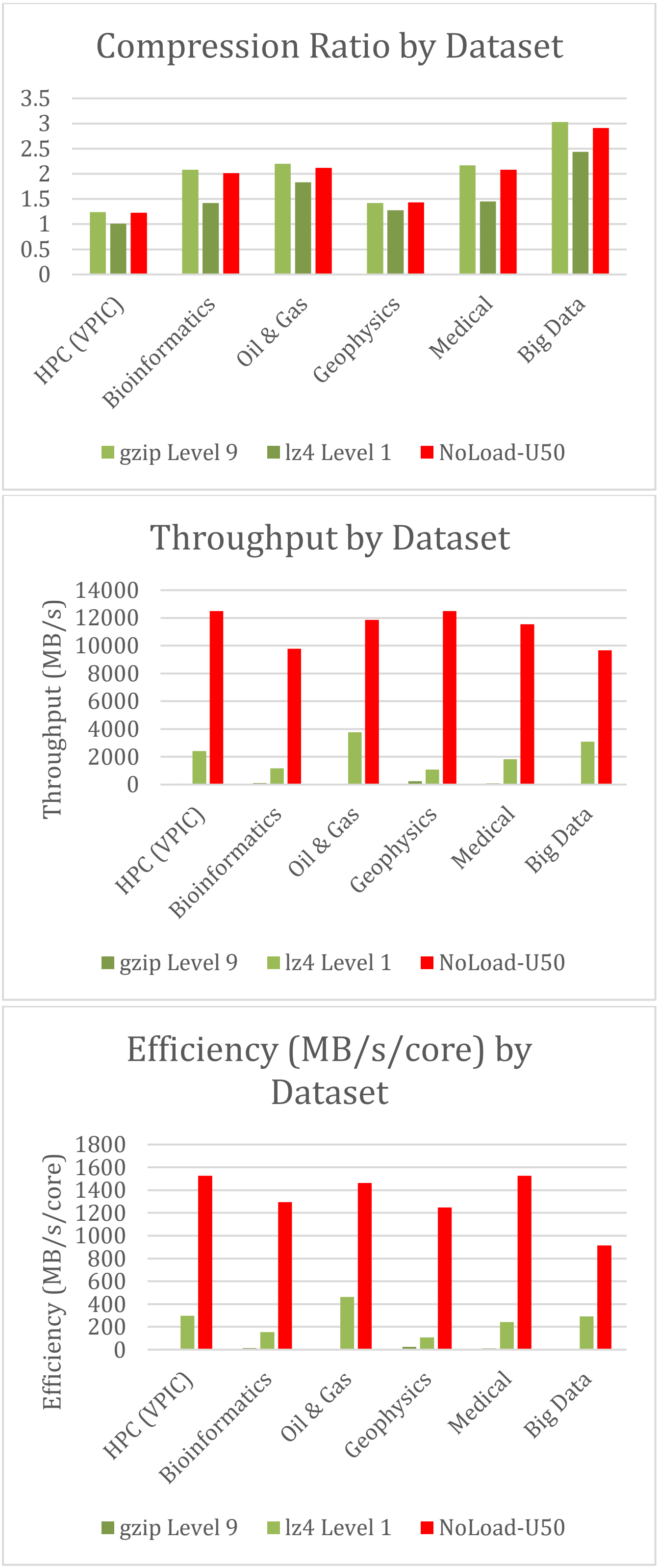


Figure 5: Compression Ratio, Throughput and Efficiency of NoLoad vs. lz4-1 and gzip-9

9 Performance Efficiency

In addition to the categories above, we define Performance Efficiency as a simple equation to compare the throughput and compression ratio of different compression solutions. Performance efficiency is defined as the throughput times the compression ratio for a given algorithm normalized by the throughput times the compression ratio for gzip-9. We define this as follows:

$$performance\ efficiency = \left(\frac{throughput_{algo}}{throughput_{gzip}}\right)\left(\frac{CR_{algo}}{CR_{gzip}}\right).$$

Figure 6 clearly shows the advantage of using Eideticom's NoLoad, our Performance Efficiency is significantly better than both gzip-9 and lz4-1 across all datasets detailed in Figure 5.

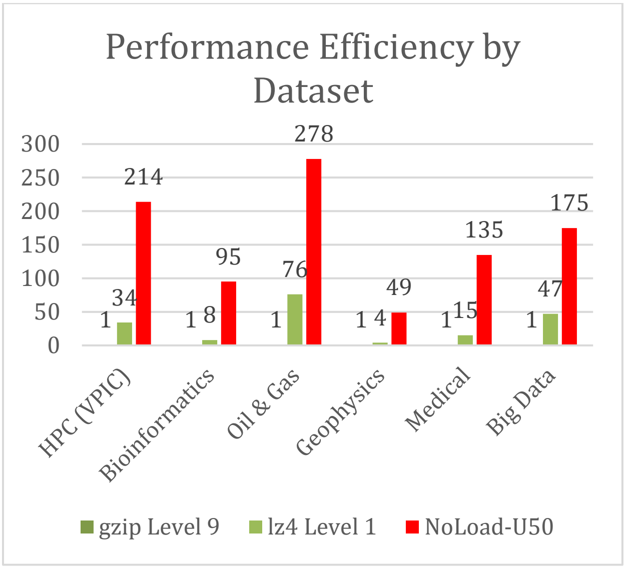


Figure 6: NoLoad Performance Efficiency

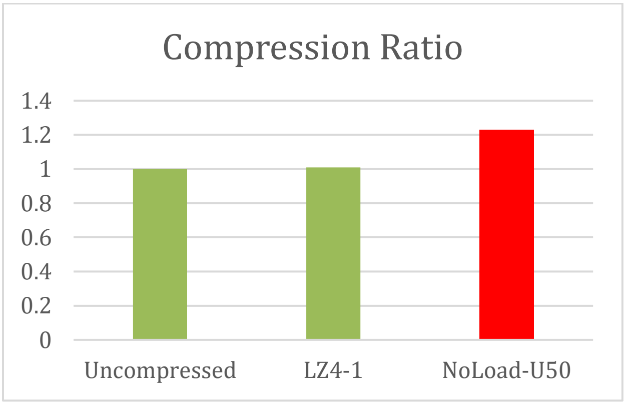
10 NoLoad Compression on ZFS

To test the performance gains of a NoLoad-based parallel filesystem Eideticom and Los Alamos National Laboratory gathered data on NoLoad enabled servers. The NoLoad enabled version of ZFS on Linux discussed in Section 5 was installed on these servers, and HPC (VPIC) data was performed on ZFS with RAIDZ2 enabled. The server configuration was the Dell PowerEdge R740xd using dual Intel Xeon Platinum 8280 2.70GHz.

For the dataset, the following was run:

- Compression: No compression, lz4-1 software and NoLoad compression.
- Data Protection: RAIDZ2 (protects against two disk failures per sever) was turned ON.

The main conclusion from Figure 7 is that NoLoad-based compression can provide **significant Compression Ratio (CR) advantages over lz4-1 in ZFS on Linux with RAIDZ2 enabled, while also achieving significally better than line-rate throughput.**



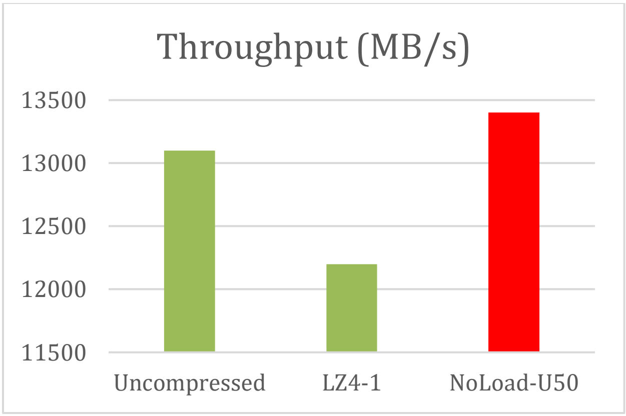


Figure 7: Compression Ratio and Throughput for HPC (VPIC) Data in ZFS with RAIDZ2 Enabled

11 Cost-Benefit Analysis

When performing the cost-benefit analysis of the NoLoad enabled parallel filesystem, we considered four key factors, namely:

1. **Storage Capacity Costs:** NoLoad Compression reduces the effective $/GB of the system by increasing the amount of data stored on SSDs. The performance data in Figure 5 demonstrates that NoLoad Compression Ratio (CR) is always higher than lz4-1, therefore increasing the storage capacity of the SSD.
2. **Storage Lifetime Costs:** NoLoad's higher Compression Ratio extends the lifetime of an SSD, since for a given compression input throughput it reduces the Drive Writes Per Day (DWPD).
3. **Throughput Performance Costs:** The performance data in Figure 5 clearly shows that NoLoad compression is 3-6 more CPU efficient than lz4-1 and over 100 times more CPU efficient than gzip-9. NoLoad permits a smaller CPU with less cores to be used to achieve the same throughput. As a result, this leads to fewer servers in the final system, see Figure 8.
4. **Storage Performance Costs:** As noted in the Storage Lifetime Costs, increased CR means the reduced write throughput to the SSDs. NoLoad allows for the use of cheaper, less performant SSDs and that the SSDs will consume less power.
5. **Power Consumption:** The data in Figure 5 demonstrates that NoLoad compression is dramatically more CPU efficient than both lz4-1 and gzip-9, therefore allowing a given CPU to run cooler and consume less power since less cores are fully loaded. NoLoad results in a power consumption reduction of up to 50% per server.

12 Bottomline

Combining the savings from the NoLoad based-system due to Storage Capacity Cost, Storage Lifetime Costs, Throughput Performance Costs, Storage Performance Costs and Power Consumption leads to combined savings of almost **50% per server!**

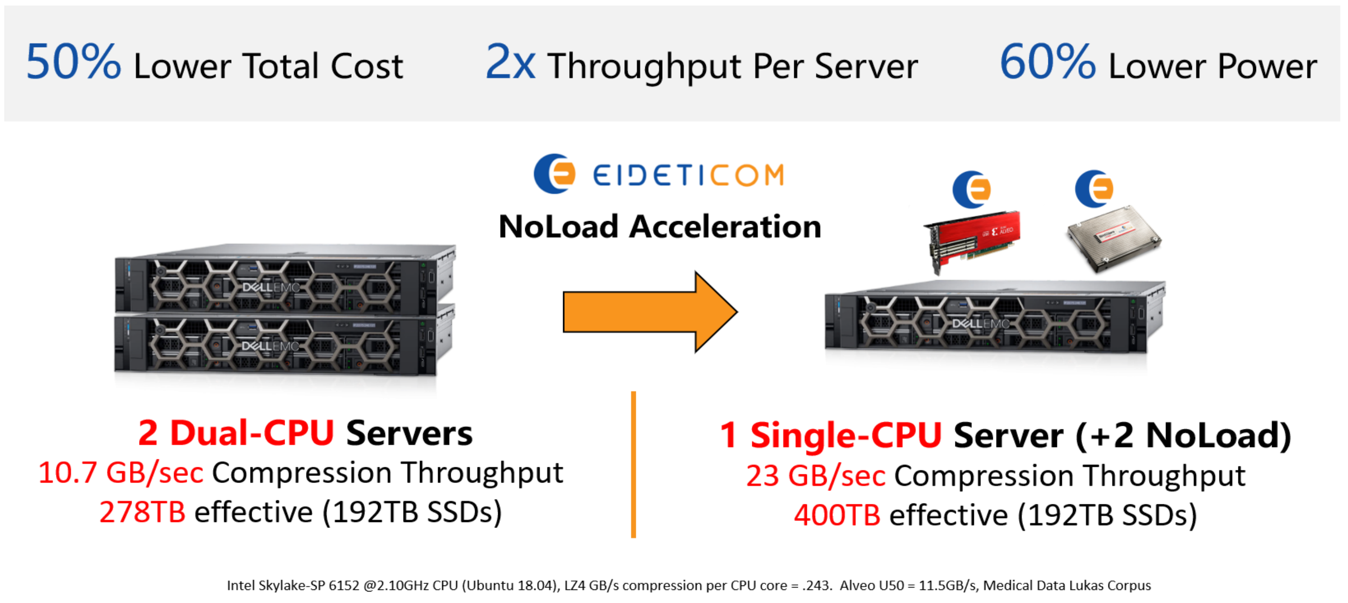

50% Lower Total Cost
2x Throughput Per Server
60% Lower Power
EIDETICOM
NoLoad Acceleration
2 Dual-CPU Servers
10.7 GB/sec Compression Throughput
278TB effective (192TB SSDs)
1 Single-CPU Server (+2 NoLoad)
23 GB/sec Compression Throughput
400TB effective (192TB SSDs)
Intel Skylake-SP 6152 @2.10GHz CPU (Ubuntu 18.04), LZ4 GB/s compression per CPU core = .243. Alveo U50 = 11.5GB/s, Medical Data Lukas Corpus

Figure 8: NoLoad Cost-Benefit

Summary

Computational Storage aims to improve system performance by making compute more efficient and by reducing data movement. In this white paper, we have demonstrated the advantage of NVMe-based Computational Storage in the form of Eideticom's NoLoad® NVMe-based CSP integrated into an HPC parallel filesystem.

Eideticom's NoLoad Computational Storage Processor (CSP) results in accelerating data center infrastructure, ensuring affordable scaling and dramatically lowering cost.

"We are excited to see standards-based computational storage technology being applied to a growing, acute problem in data movement, namely storage server memory bandwidth"

Gary Grider, Deputy Division Leader,
Los Alamos National Laboratory

"The Eideticom NoLoad devices have demonstrated that we can offload storage functions onto accelerators enabling line-rate compression, improving CPU utilization, and reducing memory bandwidth pressure."

Brad Settlemyer, Senior Scientist,
Los Alamos National Laboratory