

# MACHINE LEARNING, AI AND IOT HAVE A FILE SYSTEM PROBLEM

by George Crump, Lead Analyst | Storage Switzerland



Storage Switzerland, LLC

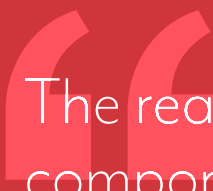
Unstructured data applications are increasingly IO hungry. It's a problem that impacts not only the High-Performance Computing (HPC) market, but also the traditional enterprise as they start initiatives like Machine Learning, Artificial Intelligence (AI) and Internet of Things (IoT). In fact, a recent survey by DataDirect Networks (DDN) shows 56% of respondents indicate that storage IO is the number one bottleneck in scaling analytics workloads.

The problem is the file systems these initiatives count on to store the data. Even object stores, once thought to be the new storage front end for HPC and modern data center applications, now seems to be just another tier for a sophisticated workload that a global file system manages.

The heart of the problem is the design of the file system, which was created for humans with a goal to provide

structure and organization to the way data is stored. These file systems evolved over the years. First, scale, at least regarding capacity, was resolved by the introduction of scale-out file systems. But, these systems bottlenecked because one node alone is responsible for metadata and IO routing. The next step was the addition of parallel file systems, where more or all nodes could manage metadata and IO. These parallel file systems are now common across the HPC landscape.

As initiatives like machine learning, AI, and IoT were introduced, data volumes increased. At the same time, the compute layer became able to process even more data and more complex algorithms, thanks to faster processors, more cores and GPUs (Graphics Processing Unit) to assist in processing. This has led to where we are today, a massive unstructured data IO processing gap.



The reality is that the parallel file system is a necessary component of a storage infrastructure...

Organizations on the front end of these initiatives are quickly learning that upgrading to flash itself is not the answer. The problem is latency and response time. The overhead required to keep all the nodes of a scale-out, parallel file system in-sync adds too much to the IO wait time. If the file-system itself is not replaced or enhanced then even upgrades to faster NVMe-based flash systems and faster networking will not provide much help.

The reality is that the parallel file system is a necessary component of a storage infrastructure that supports these environments. One way to reduce the latency and improve response time would be to create a simpler file system with fewer features. But, the environments that

a parallel file system supports need the capabilities of these file systems. Additionally, at some level latency can only be reduced so far, since at a minimum there will be cluster management and metadata management requirements. The other alternative is to upgrade the processing power and network connection of the parallel file system itself. The problem is that this raises the cost of the storage infrastructure significantly and is not practical for most use cases.

Instead of replacing the parallel file system, a known quantity, a better solution is to provide it with some assistance, similar to how GPUs are helping traditional processors with AI and machine learning; essentially the parallel file system needs an IO co-processor.





## INTRODUCING BURST BUFFERS

Burst buffers are an IO shock absorber similar to the way a flash cache works in a hybrid (flash and HDD) storage system. But, unlike a traditional cache, the burst buffer is connected to the application via a high-speed IO connection, typically PCIe or InfiniBand. The memory storage in the system is typically high-performance flash (NVMe) and will likely soon migrate to 3DXPOINT and eventually may even migrate to a non-volatile memory technology like MRAM.

Although it can be used for read IO, the primary focus of a burst buffer is to improve write performance. One of the more time-consuming tasks of a parallel file system is dealing with writes. That data has to travel down the network connection, be protected via RAID, replication or erasure coding, then metadata needs to be updated with the location of the data and its protected copies, and finally an acknowledgment is sent to the application that originated the write.

With burst buffer in-place, the acknowledgment is sent to the host immediately after the buffer receives it. The burst buffer has no other features to manage, and

protection, while more than adequate for the purpose is relatively simple and most importantly almost latency free. After it sends the acknowledgment to the host, the burst buffer then sends the data to the parallel file system.

Other than accelerated write IO, a use case for burst buffers is to enable a checkpoint restart. In HPC applications as well as AI and machine learning, the algorithms within jobs can take a considerable amount of time to process. If there is a failure, the job typically has to be restarted and re-run. With a burst buffer, the job can be restarted at the point of failure, which can save a tremendous amount of time.

The problem with burst buffers is that for the most part they are do-it-yourself projects and require a lot of manual configuration. The other is that they need specific application customization to make the environment know they are aware and to take advantage of it. Finally, organizations want to use the high-performance storage area for more than just a write cache.

## BEYOND BURST BUFFERS, FLASH NATIVE CACHE

The idea behind a burst buffer is to protect the environment from the latency of the parallel file system and absorb write IO overhead from many simultaneous threads. A flash-native cache is a more generalized platform used to support a variety of file sizes and workload types. Organizations want to use it for such tasks as pre-loading data to be analyzed to make processing faster. They also want the flash-native cache to perform block alignment so that when data is eventually written to the parallel file system it is aligned to the file system, which makes subsequent reads more efficient.

“...Infinite Memory Engine (IME) should save organizations money by allowing its other resource investments to perform at their optimum levels.”

## THE UNIQUE AI CHALLENGE

While ingest performance is critical to AI workflows, they also can be very read intensive. AI workflows are very well served by flash native cache as their IO profiles can be very randomized at time. For example, GPU-enabled in-memory databases gain lower start-up times from the fast population of the AI database while it is feed from a data warehousing environment. GPU-accelerated analytics demand the support of large thread counts each with low-latency access to small data segments. Another example is image-based deep learning for classification, object detection/segmentation which benefit from high streaming bandwidth, random access, and often fast memory mapped

calls. A final example is recurrent networks for text/speech analysis. These also benefit from high performance random small file or small IO access.

The random IO nature of each of these workloads can cause performance challenges with traditional scale-out file systems. In the past the organization might be faced with creating a different storage silo for processing and one for long term data storage. Inserting the flash native cache allows these environments to deliver required performance without replacing the file system.

# INTRODUCTION TO INFINITE MEMORY ENGINE

DDN's Infinite Memory Engine (IME) is an example of a flash native cache that provides complete burst buffer functionality as well as read caching, data pre-staging and block alignment. IME, while available as a software-only solution, is more of a turnkey solution. The customer simply loads the software onto the hardware of their choosing. For organizations that want an even more turnkey experience, IME is available as a complete hardware stack, which only needs to be plugged in.

The IME solution installs transparently between the parallel file system and the application's compute nodes, meaning the application requires no changes. It connects to the host via InfiniBand and is itself an expandable storage cluster. Multiple nodes can be added so IME can meet the capacity requirements of the environment.

IME resolves the limitations of POSIX file system semantics caused by small file IO and file locking,

by dynamically aligning fragmented data into full stripe writes. Write alignment increases the capacity utilization efficiency, improves the performance of parallel file system reads and with flash media, should extend its life.

In the end, IME should save organizations money by allowing its other resource investments to perform at their optimum levels. Most environments lose about 30% of their CPU efficiency due to storage IO latency, even if they invest in a more expensive back-end storage infrastructure like flash. With IME, the back end parallel file system still may be flash-based but it does not need to be as a hard disk-based cluster of nodes may work fine. If the organization prefers an all-flash back end, IME gives them the ability to leverage the very high-density flash drives coming to market, saving them shelf space and data center floor space.







## STORAGESWISS TAKE

HPC, AI and machine learning are quickly becoming environments judged on “time to answer”. How long it takes to answer a query directly impacts user experience and in many cases can make a monetary difference to the organization. Examples of enterprise use include financial institutions, which can leverage solutions like IME to process quickly, ticker data, both historical and real time. Oil and gas companies may use IME to provide in-depth analysis of historical seismic data.

Storage architects may find they have the same problem as compute tier architects who, instead of waiting for faster Intel CPUs, added GPUs to improve their processing capabilities. Storage architects, instead of trying to build a faster parallel file system with all flash, may be better served by adding a flash-native cache that can be used as both an IO shock absorber and a staging area.

# ABOUT US



**Storage Switzerland** is an analyst firm focused on the storage, virtualization and cloud marketplaces. Our goal is to educate IT Professionals on the various technologies and techniques available to help their applications scale further, perform better and be better protected. The results of this research can be found in the articles, videos, webinars, product analysis and case studies on our website [storageswiss.com](http://storageswiss.com)



**DataDirect Networks (DDN)** is the world's leading big data storage supplier to data-intensive, global organizations. For almost 20 years, DDN has designed, developed, deployed and optimized systems, software and storage solutions that enable enterprises, service providers, universities and government agencies to generate more value and to accelerate time to insight from their data and information, on premise and in the cloud. Organizations leverage the power of DDN storage technology and the deep technical expertise of its team to capture, store, process, analyze, collaborate and distribute data, information and content at the largest scale in the most efficient, reliable and cost-effective manner. DDN customers include many of the world's leading financial services firms and banks, healthcare and life science organizations, manufacturing and energy companies, government and research facilities, and web and cloud service providers.



**George Crump** is President and Founder of Storage Switzerland. With over 25 years of experience designing storage solutions for data centers across the US, he has seen the birth of such technologies as RAID, NAS and SAN. Prior to founding Storage Switzerland he was CTO at one the nation's largest storage integrators where he was in charge of technology testing, integration and product selection.