Improve the Performance, Efficiency, and ROI of GPU Computing with Hammerspace Tier 0

TECHNICAL BRIEF

Topics

Maximizing GPU Compute Time with Tier 0 Storage

Servers equipped with GPUs are expensive but vital for modern computing, particularly projects involving AI. Due to huge demand, however, GPU systems can be difficult to obtain, even when budget is available. And then there is the challenge of keeping them fed. GPUs require significant power, which is in short supply in most data centers. They also require lots of data, and to maximize ROI they need it to be supplied as fast as possible to minimize idle time. This often means organizations must purchase new high-performance storage and networking, which consumes additional funding and power, two resources that are chronically in short supply.

With all these constraints, organizations are rightly focused on making the most of the GPUs they have, which means keeping them busy doing productive work as much as possible. Yet there is one resource in every GPU server that is frequently overlooked, unused, or at best, underutilized. That resource is storage, specifically NVMe SSDs housed locally within the GPU server itself.

Beginning with version 5.1, Hammerspace unlocks this GPU-local NVMe storage to create a new Tier 0 of ultra-high performance, persistent, shared storage that can be used to accelerate compute jobs and reduce time spent on overhead tasks such as checkpointing. This efficiency has a snowball effect, not only increasing performance but lowering costs (both acquisition and operational) and accelerating time to value. Spend less, yet get more work done? That's right. Continue reading to learn how.

GPU-Local Storage – A Missed Opportunity

While the price of enterprise NVMe flash storage continues to drop, it's not exactly inexpensive, especially when a lot of it is required. And yet, many organizations have hundreds of Terabytes or even Petabytes of this storage sitting idle. How can this be?

If you've purchased one (or a thousand!) GPU servers, chances are that they each contain at least a few NVMe SSDs. As one example, if you purchase an NVIDIA DGX H200 or B200 today, it comes with eight 3.84TB U.2 NVMe SSDs, whether you want them or not. Other vendors make this storage optional, but support more and larger NVMe SSDs. The Dell PowerEdge XE9680, for example, supports up to 16 of the huge 122.88TB NVMe SSDs that will be available in 2025. This means it will soon be possible to put nearly 2PB of NVMe storage in a server, right next to the GPUs. This is also true if you are renting GPU servers in public cloud providers. The local NVMe is included in cloud-based GPU configurations, whether you can effectively use it or not.

Everyone knows that local storage is much faster than networked storage, so why is such storage within GPU servers going unused? There are three main reasons:

1. **It's a silo:** The full performance of the included NVMe storage can only be exploited by GPUs located in the same server. Data on this storage can be accessed via an NFS mount point, for example, but if there are lots of GPU servers that means lots of individual exports to manage manually.

2. **It's not protected:** Out of the box, data written to local storage isn't protected. That's why vendors position it as cache space. If it's configured at all, the local drives are usually striped together with RAID0. RAID1 or RAID10 can be used, but that eats up capacity and only provides partial protection, since it still doesn't get a copy off the host.

   One could stripe an erasure-coded file system across multiple GPU servers, but then only a fraction of the I/O remains local, squandering most of the performance. GPU servers are also frequently reconfigured and restarted, which makes them a poor choice for this.

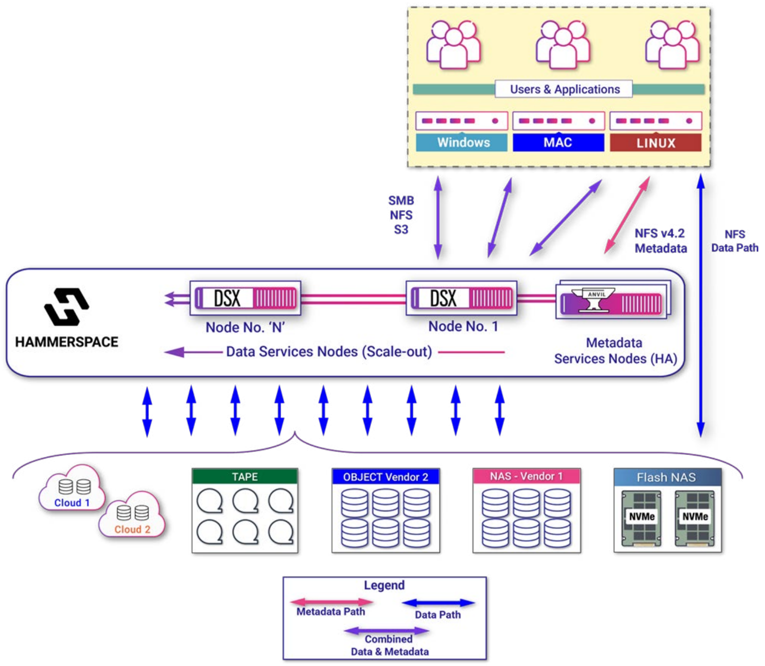
3. **It's difficult to get data in and out:** Manually copying data in and out of individual shares on dozens, hundreds, or thousands of GPU servers isn't practical. Even when many of the steps can be scripted, keeping track of what data is where and monitoring the status of copies simply takes too much time, plus adds operational overhead and risk.

So, while this local storage could be used, it usually isn't because using it effectively gets complicated, fast. That's where Hammerspace comes in.

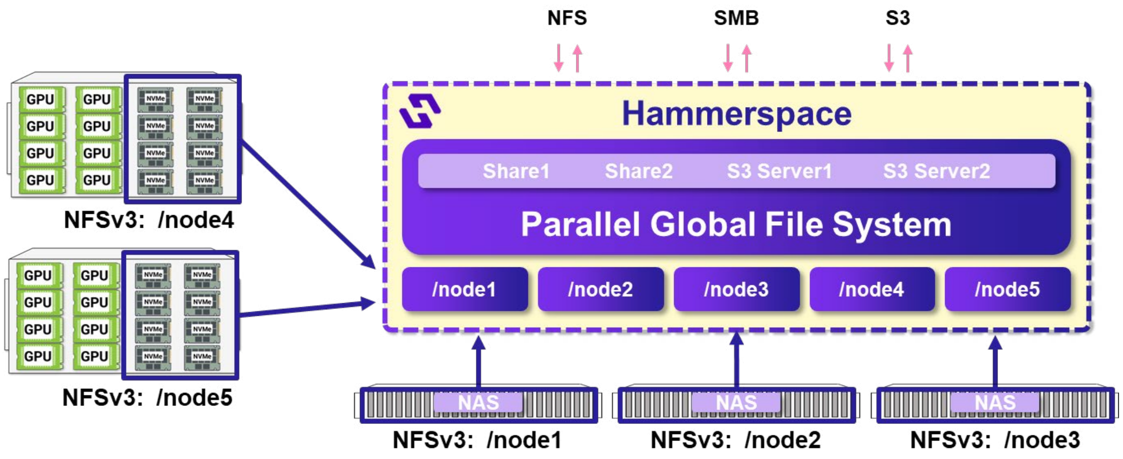
Unlocking Tier 0 with Hammerspace

Hammerspace is a consolidated data platform for distributed file and object data. It uses a standards-based parallel file system architecture and creates a global namespace that can span storage of any type from any vendor, including multiple sites, and clouds. The included orchestration engine automates data protection and other data services, and places data on the proper storage at the proper time, using declarative policies known as Objectives. No matter where data lives within the system, it is always visible and accessible to clients, even when it is in motion. Because Hammerspace is based on open standards, it requires no client software or proprietary kernel modifications, making it simple to deploy.

For a thorough overview of Hammerspace's architecture and capabilities, refer to the Technology White Paper.



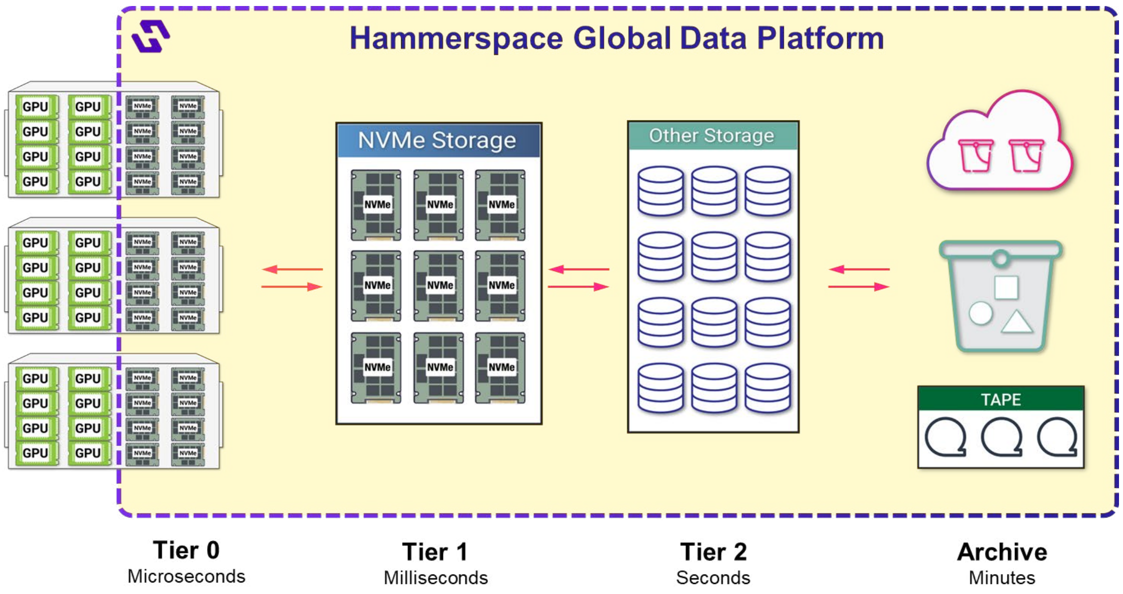
To build a Hammerspace global data environment, new and/or existing storage assets such as NFS exports on NAS filers, Linux storage servers, et cetera are configured into Hammerspace as storage volumes. This is done with data-in-place metadata assimilation, without needing to migrate data off of existing storage. Hammerspace file shares and S3 servers are then created which may be accessed via NFS, SMB, and S3. Shares and S3 servers are logical constructs used by users and applications to access data. The data itself is automatically placed on the appropriate storage volumes using policy-based Objectives that define durability, availability, performance, location, and more. These can also include custom metadata that reflects richer insights into how data should be managed.



To create Tier 0, the storage on each individual GPU server is configured as one or more NFS exports, which become additional storage volumes in Hammerspace. Once this is done, all Hammerspace's features and capabilities may be applied to the Tier 0 storage exactly as is done globally with external storage.

Data on Tier 0 Is Not Siloed

Because the storage on the GPU servers has been brought into the Hammerspace Global Data Platform, it is no longer siloed. Files and objects residing on it are not "stranded" on the GPU server, rather they are part of a cross-platform unified namespace.



Data on Tier 0 is Protected

Once the GPU server storage is configured into Hammerspace, the data on it is protected using standard Hammerspace Objectives. Durability and Availability policies are defined in terms of "number of nines" and can be applied intelligently on a file-granular basis. Based upon these rules, data is asynchronously placed on one or more storage volumes to satisfy the desired Objectives.

Data on Tier 0 is Orchestrated

Objectives are not only used for data protection, but also for other data services including orchestration, which is a fancy way of saying "moving data around." Because the Hammerspace global file system presents a unified namespace, data movement is a background operation that happens non-disruptively behind the scenes. No matter where the data lives or may move to across physical storage volumes both on-premises and in the cloud, files always appear in the global file system with the same file/folder structure users expect to see. Data is persistent and always accessible via the same standard SMB, NFS, and/or S3 protocols, even when it is in motion. Automated data orchestration solves the practical and logistical problems that otherwise make it impossible to use GPU-local storage effectively.

Some examples of different ways data orchestration may be used with Tier 0 include:

- Keeping only the most recent data on Tier 0 storage, and automatically moving older files and objects off to lower cost storage such as Tier 1 or 2 external storage or cloud storage.
- Pinning a copy of files required for startup to every server's local storage, preventing "boot storms," slowdowns that occur on cluster startup when many nodes need to read the same files.
- Moving a data set from lower-performing tiers to Tier 0 storage for processing, and then automatically moving the results off-host to other tiers of local, remote, or cloud storage.
- Shortening checkpointing times by saving checkpoint files to local NVMe storage, then asynchronously migrating them to a lower tier of storage to reduce wait time for GPUs. Objectives may be set to retain the most recent checkpoints locally for fast recovery.

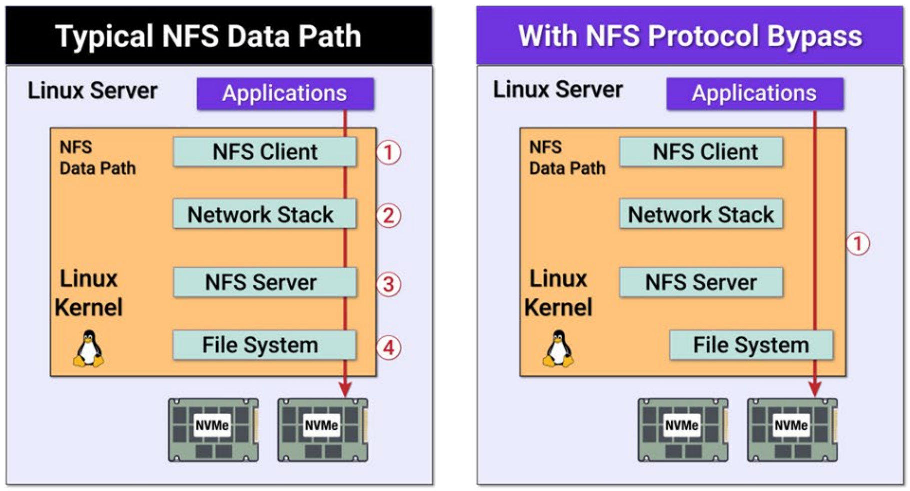
New NFS Protocol Bypass Minimizes GPU Idle Time

Back when spinning hard disks were the most common form of primary storage for active data, storage software had it easy. Efficiency was important, but the amount of time spent waiting for the right sectors to spin under the drive heads allowed for a lot of slack. Now with NVMe SSDs, the software stack is in the hot seat. Speed is now paramount for GPU workflows, so maximizing software performance requires removing bottlenecks and adding shortcuts wherever possible.

Some vendors build custom clients, or require proprietary protocols, and exclusive kernel modifications to improve performance. This can be a shorter path, but it is also good for locking customers into a vendor's solution and making the silo problem even worse. Hammerspace took a different approach, instead choosing to focus on improving and extending the existing Linux standards, which are already in use in every data center on the planet. The Hyperscale NAS architecture using standard pNFSv4.2 with Flexible Files layouts is a prime example.

By using storage that's GPU-local, Tier 0 is extremely fast, but Hammerspace has engineered an optimization into Linux that makes it even faster. This enhancement to the Linux kernel is based on the realization that if an NFS client and server can detect when they are running on the same host, there are a lot of steps, and thus latency, between the application and the storage that can be eliminated.

Prior to this optimization, even when the NFS client and server are local to each other, data must flow from the NFS client, through the network protocol stack, back to the NFS server, and only then to the local file system and storage device.



✓ 12x faster IOPS and bandwidth within the GPU server

✓ Standards-based; no special software required

✓ Engineered by Hammerspace, contributed to Linux

The changes to NFS that Hammerspace has contributed upstream into Linux (available soon starting with kernel 6.12) add intelligence so that the NFS clients and servers know when they are on the same host and can bypass these unneeded steps. This is done in a robust way that works even if the NFS client and server are running in containers. Once locality is confirmed, a path is established essentially directly between the application and the local file system. Because the data path now bypasses nearly all the NFS code in the kernel, we refer to it as the NFS Protocol Bypass, but to Linux kernel developers it's known as the LOCALIO auxiliary protocol.

NFS Protocol Bypass provides a tangible performance benefit, with up to 12x faster reads and 3x faster writes compared to the typical NFS data path for local storage that doesn't use this optimization. It continues Hammerspace's record of contributing significant innovation and performance enhancements to the NFS stack in standard Linux, which are available in all major distributions.

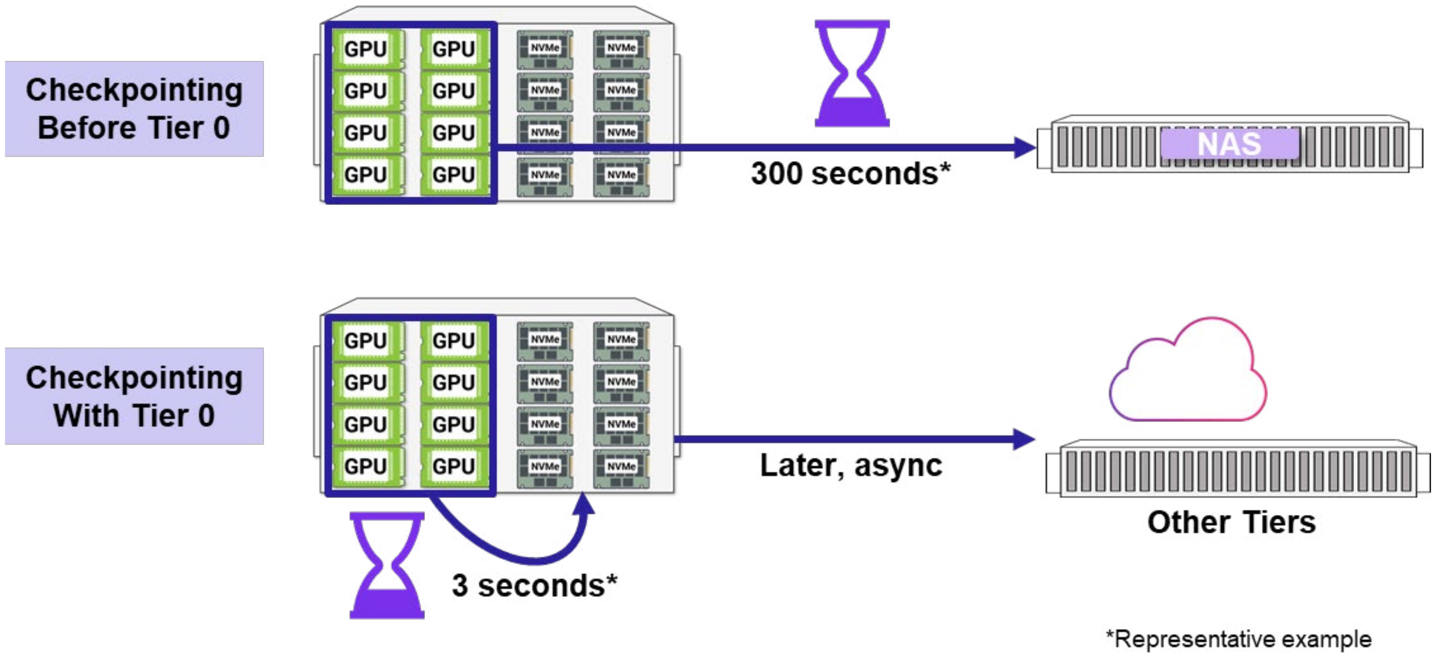
Tier 0 Use Case Example - Checkpointing

AI and other high-performance computing (HPC) applications involve calculations that are performed across large clusters of servers and may run for days, weeks, or even longer. If servers or storage subsystems experience a failure, all work up to that point may be lost. Checkpointing is used to minimize this risk.

Checkpointing is simply saving the current state of an ongoing computation job within the cluster to persistent storage at a point in time. This is done in a way that enables servers within the cluster to be restarted and the compute jobs resumed from the checkpoint instead of starting over from the beginning.

Formulating a checkpointing strategy is like buying insurance – it's important to balance the level of protection with the risk. Over-insuring (checkpointing too frequently) provides more protection but has a cost. Checkpointing takes time, and during this time the GPUs are idle and waiting, so meaningful work is not being performed. This reduces the efficiency of the cluster, increases the time to deliver results, and can significantly hurt the ROI of the GPUs.

Using Tier 0 storage with Hammerspace for checkpointing provides a dramatic reduction in checkpointing time, with a corresponding increase in GPU utilization rates for productive work. In a typical checkpoint workflow without Tier 0, the GPU cluster will write checkpoints to shared storage over a network. These checkpoint files are large, and this can take several minutes. Once the checkpoint files are written, the GPUs resume their computation jobs.



With Hammerspace and Tier 0, checkpoints from the cluster are written to the Tier 0 local NVMe storage, leveraging the NFS Protocol Bypass for the fastest possible speed. Since the local storage of each GPU server is part of the global Tier 0 capacity, this can be done cluster-wide at extreme speeds. Instead of minutes, checkpointing takes just a few seconds, after which the GPUs may resume their jobs. With the checkpoints now successfully written to Tier 0, Hammerspace then automatically protects the files by tiering them asynchronously to any other storage as appropriate.

Because both the Tier 0 storage and the storage used to protect the checkpoint files are part of the Hammerspace unified namespace that spans all storage types including cloud, the checkpoint files are always visible and accessible in the same logical location within the file system regardless of where they may be moved physically. If a copy of the most recent checkpoint file is kept locally on each node, checkpoint recovery times are substantially reduced as well, since only the checkpoint files from failed nodes need to traverse the network.

In addition to providing more time for computation, drastically reduced checkpoint times provide researchers with the option to perform checkpoints more frequently, shrinking the vulnerability window.

The increased cluster efficiency provided by checkpointing to Tier 0 delivers real financial benefits. A detailed ROI analysis by Hammerspace concluded that for a 1,000-node cluster with 8,000 GPUs, the savings can be tens of millions of dollars per year. Even for smaller clusters, the savings can be substantial.

Only Hammerspace Can Deliver Tier 0 Today

As described in this technical brief, Tier 0 is more than just the ability to incorporate local storage into a shared file system. That alone does not eliminate the bottlenecks and inconveniences that prevent practical, effective, and high-performance use of this storage. Only Hammerspace has all the features and characteristics necessary to make full use of the local NVMe storage in GPU servers:

Tier 0 is Native Linux

- No custom software is needed on GPU/compute nodes. No client agents, no proprietary kernel patches, since it is included in all standard Linux distributions.
- Version compatibility issues and software maintenance headaches caused by proprietary offerings are avoided.

Tier 0 is Protected

- Data on Tier 0 is easily protected on the Tier 0 storage that exists on other GPU servers and/or on lower tiers of storage, including object storage and cloud.
- Tier 0 storage doesn't require RAID protection at the server level, preserving that local capacity for more productive uses.

Tier 0 is Orchestrated

- Local storage becomes a seamless part of the global file system and namespace.
- Tier 0 storage augments and is compatible with 3rd-party storage from any vendor or cloud.
- Affinitization and N-way replication enable multiple instantiations of the same file on multiple nodes for concurrent local access.

Tier 0 uses NFS Protocol Bypass to Maximize Performance

- LOCALIO patches in the Linux kernel streamline the data path through the Linux kernel, adding intelligence and eliminating unnecessary steps to reduce latency.
- The streamlined data path better utilizes local NVMe bandwidth within the server.

Tier 0 Benefits Summary

Hammerspace v5.1 with Tier 0 enables organizations to transform their GPU computing infrastructure, maximizing the value of their investment and delivering faster time to results.

The benefits of Tier 0 fall into three categories:

Cost Savings

- Tier 0 reduces external storage costs and can reduce the overall power needed for the infrastructure.
- Tier 0 increases GPU utilization enabling more work to be accomplished with fewer GPUs.
- Hammerspace automated data orchestration reduces administrative time spent managing storage infrastructure and wrangling data copies across incompatible storage silos.
- When used in cloud-based GPU clusters, increasing utilization and reducing the number of GPUs needed has a direct impact on the bottom line and can reduce cloud compute costs.

Performance

- Local storage is always faster than networked storage. Tier 0 leverages this frequently overlooked and unused storage capacity to maximize GPU utilization.
- In addition to speeding up processing by bringing data closer to the GPU, Tier 0 can increase checkpointing speed by 20x, 40x, or even more. This provides many more hours of GPU computing time per year with the same infrastructure.

Time To Value

- Hammerspace is standards-based, requiring no software installation on clients or storage servers. It’s possible to begin using the NVMe capacity in GPU servers on premises and in the cloud in less than 30 minutes.
- Taking advantage of storage that’s already present inside the GPU servers requires much less effort than deploying a new external storage system.