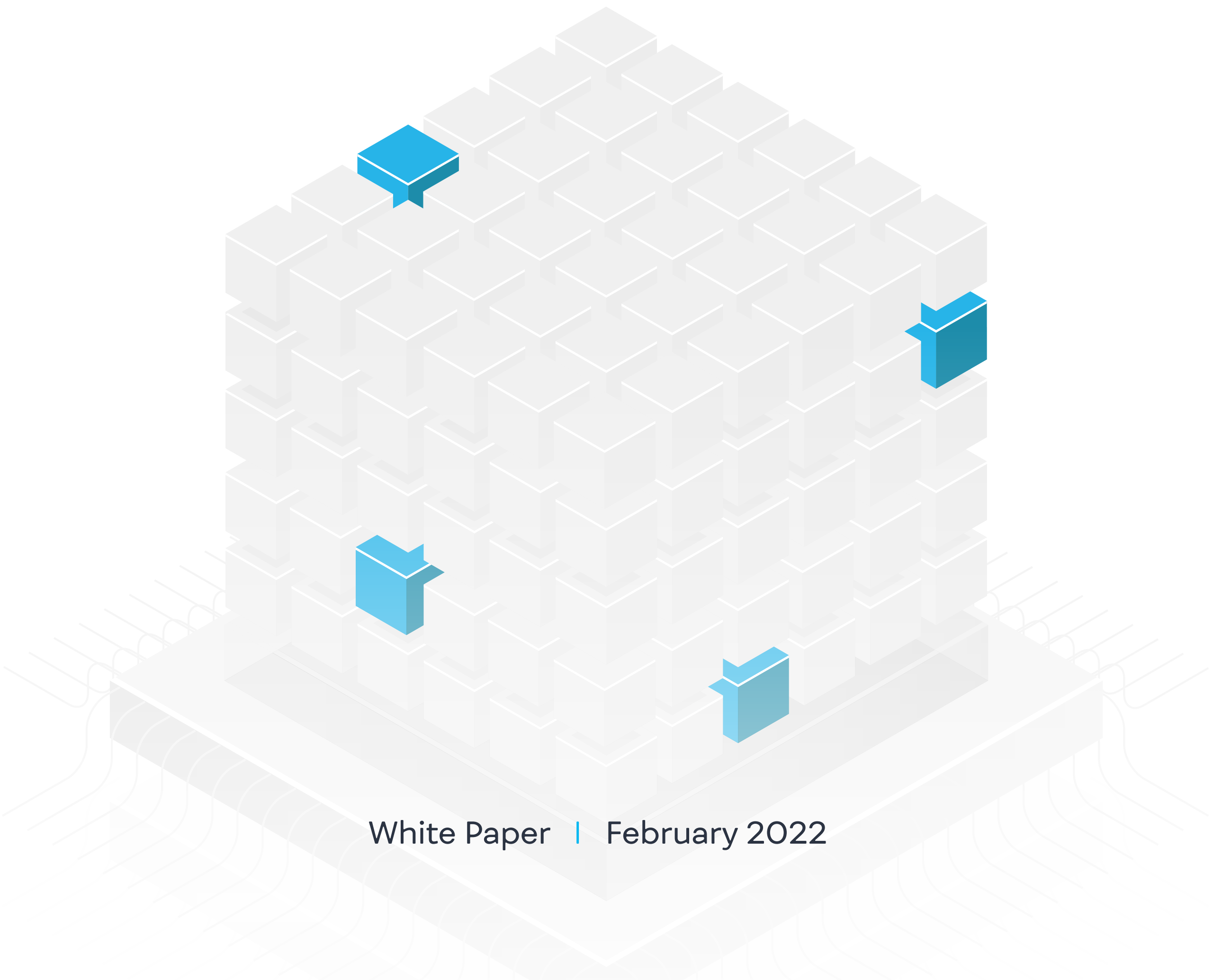
VAST

DATA PROTECTION

ENSURING STORAGE RELIABILITY AT SCALE

As storage systems grow to multi-petabyte, and even exabyte-scale, storage architects face new challenges that require new architectural solutions. This paper examines how the traditional shared-nothing architecture and VAST Data's Disaggregated Shared-Everything architecture react to the multiple failures that large systems encounter and estimates the probabilities of data loss on each.



White Paper | February 2022

TABLE OF CONTENTS

EXECUTIVE SUMMARY

It's a cliché among IT architects that everything's different at scale. That's certainly true when talking about storage. Data protection techniques that work for 50 or 100 TBs aren't practical for a storage system that holds tens or hundreds of petabytes of data. A system of that size will have thousands of devices and more devices mean more failures.

VAST's Disaggregated Shared Everything (DASE) architecture is a radical approach that delivers 2,000 times more resilience compared to a shared-nothing architecture. Its layered approach, described in detail below, ensures a VAST system can not only survive multiple device failures, but fully self-repair without waiting for failed devices to be replaced.

This paper compares the reliability of VAST's DASE architecture to the 20-year-old, shared-nothing architectiure used by the leading scale-out file storage system. We examine the combinations of failures that could lead to data loss in each system and calculate the probability of data loss for clusters using each architecture.

CONCLUSIONS

Our calculations compare two clusters:

- An 8-node cluster of shared-nothing nodes with 24 15.36 TB SSDs per node yielding ~2.5 PB of usable capacity.
- A VAST cluster of four 675 TB VAST Enclosures and 16 VAST protocol servers yielding 2.38 PB of usable capacity.

We calculated the data loss probability of a VAST system at about 5E-7, or a Mean Time to Data Loss (MTDL) of ~2 million years. The probability of the shared-nothing cluster, using the industry leader's default 12D+2P Reed-Solomon erasure coding, worked out to 0.11% (~1000 year MTDL), making the DASE system 2,000 times more reliable.

THE NIGHTMARE

Every storage admin's been there. You wake up in the middle of the night, heart pounding after that nightmare. You know the one: your pager, phone, email, What's App, SMS, and smoke detector (we did say it's a dream, right?) all go off at 3AM. You click the alert on your phone and see "Storage Array BUF887 – Drive 88 failed. You're on call, Sucker." So you brush the snow off your car, pop a capsule in the Nespresso, and drive 45 minutes in the Buffalo snow to the data center.

You drop your parka and touque in your office while you log into BUF887, see the drive is still dead, and that Marvin the storage admin who configured BUF887 didn't believe in hot-spares. You grab a spare drive off the shelf, walk up to BUF887, and pull out the wrong drive. Or you see another drive's LED go from green to red as you walk up the aisle. Or static from your contact rebooted the array which then doesn't come up.

Storage administrators have nightmares because they know that storage failures, like storage itself, are persistent. When the network has a problem, the router can be replaced, or the config file fixed, and as soon as that's done the network is back up and running, (Don't start with how DNS has to propagate. I know. I also know it's always DNS, but I'm not letting that get in the way of a good story.)

When storage fails, the failure almost always has long-term consequences. When system fails to store data reliably and durably, that data has to be restored from another copy made at some point in the past. Any data created since the backup copy was made is lost. What's more, the system is typically unavailable until the restore completes, a period of some hours for substantial datasets.

Battling the source of our nightmare

To avoid the nightmare of data loss, and the slightly less frightening data unavailable event, vendors maximize storage uptime through redundancy. Before the flash era, disk drives and fans – with all their moving parts – were the weakest links in any storage system. System architects addressed this weakness with redundant fans and RAID, or Reed-Solomon erasure coding to protect data against drive failures. They even make the drives and fans hot-swappable to speed up Mean Time To Repair (MTTR) when the system is degraded and vulnerable.

It's comforting to think that the Mean Time Between Failures (MTBF) or Annual Failure Rate (AFR) on a manufacturer's spec sheet is derived from the vendor's field experience. But those spec sheets are printed before the product ships to customers, making them projections, not observations.

The adoption of flash SSD means the storage media is no longer the weakest link in the chain. Server vendors stopped publishing reliability specs long ago, so we can't just compare server and SSD reliability from their respective spec sheets

To calculate the MTBF of electronics like servers and SSDs, vendors follow standards such as MIL-217 that multiply the number of each type of component by that component's reliability into a total. A board can have a lot of reliable resistors and chips, but each solder joint or connector that can vibrate loose reduces the overall reliability. By those standards, a server motherboard with socketed CPUs, DIMMs, and NICs can be expected to be less reliable than an SSD with fewer components and only one connector where it plugs into the system.

All-flash users still see more drive failures than controller failures simply because they have many more SSDs than controllers. If SSDs and the servers that make up the nodes in a shared-nothing system both have an annual failure rate of 1% (876,600-hour MTBF) and your shared-nothing system has 15 or 24 drives per node, you should see 15 or 24 times as many drive failures as node failures.

That 1% AFR also means the average user of a 20-node cluster will see 1 node failure over the cluster's 5-year lifetime (.01*20*5=100%). However, failures aren't evenly distributed across all clusters so only a minority of users will see any failures at all while an unlucky few will have multiple failures. The storage architect's goal is keep users accessing their data, even if they're as unlucky as Joe Btfsplk, Li'l Abner's world's worst jinx. The law of large numbers makes that even more challenging.

	AFR	UBER (bits)	UBER (TB)
Nearline Hard Drive	.44-3%	$1x10^{15}$	125
SSD	.1-0.5%	$1x10^{17}$	12,500
X86 Server	1-8%		

Table 1 – Core Component Reliability

Table 1 above shows the reliability of the core components in a modern storage system. Because our object today is to illustrate relative resiliency, and therefore reliability, more than provide accurate projections of data loss probability, we use a 1% AFR for both drives and the x86 servers.

FAILURES AREN'T THE ONLY PROBLEM

While device failures of one sort or another are an obvious cause of data loss, disk drives and SSDs aren't purely binary devices that are either working perfectly or, like the Wicked Witch of the West, not only merely dead but really most sincerely dead. There are times when a drive that's otherwise working perfectly doesn't return the data it should when asked. Because today's drives, especially SSDs, perform internal error correction we call the errors that still occur uncorrectable read errors.

When a drive has an uncorrectable read error under normal conditions, a modern storage system detects that error, rebuilds the unreadable data from parity, and writes it to a new location. Uncorrectable read errors can cause data loss when they strike a storage system, like a double parity solution with two drive failures, that's already rebuilding from its maximum number of drive failures.

We call this precarious condition an unprotected rebuild because the storage system is already at its limit, with no remaining parity to protect itself from an uncorrectable read error. Once a system is rebuilding from the maximum number of losses it can withstand, then any error, however small, will result in data loss.

Disk drive vendors rate the 7200 RPM nearline disk drives that dominate today's data center as having an uncorrectable read error rate of 1 error for every 1 x 1015 ($1E^{15}$ in geek) bits read. 1E15 bits is 125 TB, which means a system rebuilding a 12+1 single parity array of 12 TB hard drives would have to read 144 TB and be more likely than not to run into a bad block. Storage vendors have increased their recommendations from single to double and more recently triple parity protection because hard drive sizes have increased 10-fold, but their error rates remain at one bad sector in 1E15 bits. Enterprise SSDs have a UBER of 1E17 bits (12,500 TB), reducing this risk from an almost certainty to about a 1% probability.

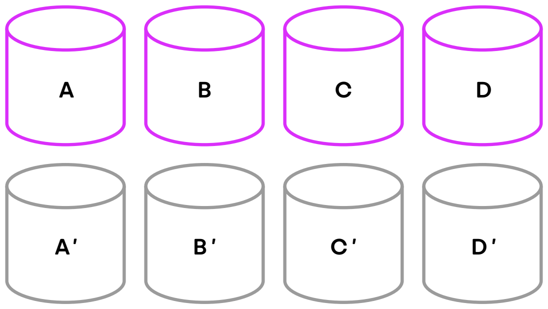
Even that 1% probability is too high for your valuable data, so a storage system should be designed such that it requires multiple failures before data is even put at risk in an unprotected rebuild.

The Path to Data Loss Is Paved with Cascading Failures

Modern storage systems are designed to accommodate, and recover from, the failure of a single device, especially a storage drive. Therefore, actually losing data requires a series of cascading failures that exceeds the system's ability to recover.

The most obvious series of cascading failures is simply more drive failures than the system's protection scheme can handle before it can rebuild. The probability of this sort of data loss is a function of the device failure rate, and the length of time the system runs in a degraded mode where data is vulnerable to additional failures.

When talking about risks and the probability of data loss, it is important to avoid the specious argument of specific failures adding up the number of devices that could fail without data loss, but only if those specific devices fail. This argument is clearest when discussing a set of 8 mirrored drives in some variation of RAID10.



Because a mirrored system keeps 2 copies of any piece of data, it can survive one device failure. Some will incorrectly argue that this setup can survive up to 4 device failures, as long as they were the right devices, so this is better than a double parity scheme that can only survive 2 failures. Einstein once famously said that "God does not play dice with the universe" and you shouldn't rely on being lucky about which devices in your system fail.

Shared-Nothing Storage

A shared-nothing storage system uses software to create a logical storage array and/or file system from storage media contained in a series of interconnected nodes. Each node in a shared-nothing system contains memory and a set of storage drives that are permanently owned and only accessible by that node. Shared-nothing systems protect data by replicating, or erasure coding, the data across multiple nodes.

For more information on shared-nothing storage, visit our blog.

Exploring Shared Nothing Resilience

Early shared-nothing systems protected data by replication. When a user wrote a file, shared-nothing systems stored copies of that file on two or three nodes. Replication is simple, but the replication requires several times as much storage as the original data, which just isn't sustainable.

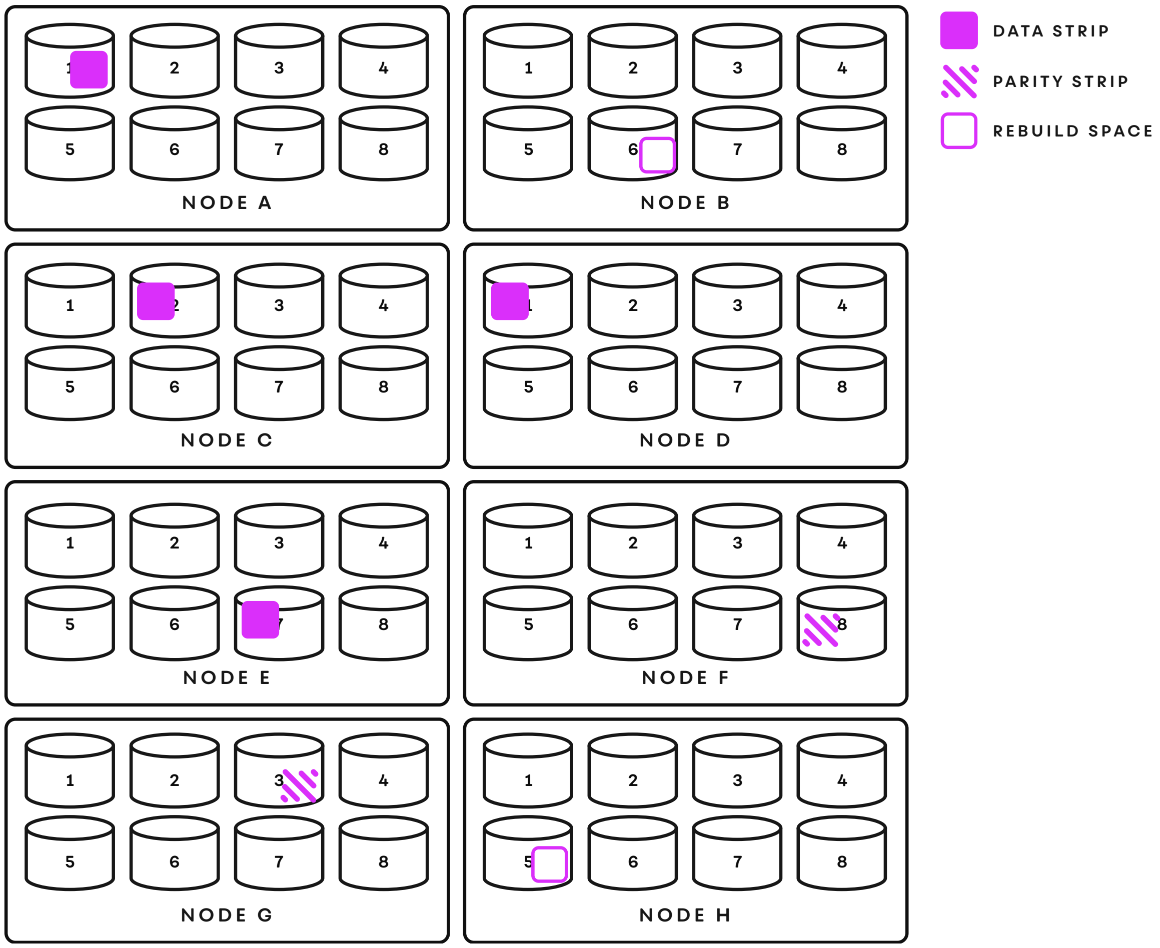
Over time more efficient parity and Reed-Solomon erasure codes replaced replication, allowing shared-nothing systems to shard data and parity strips across the nodes of a cluster. The most obvious, and most common, way to use erasure codes in a shared-nothing cluster places one data, or parity, strip of each erasure coded stripe on each drive. This layout protects equally against node and device failures. Plus, with the standard double parity coding, multiple failures would have to occur before data is at risk from uncorrectable read errors during the rebuild.

Node failures remain much more significant than drive failures for two reasons:

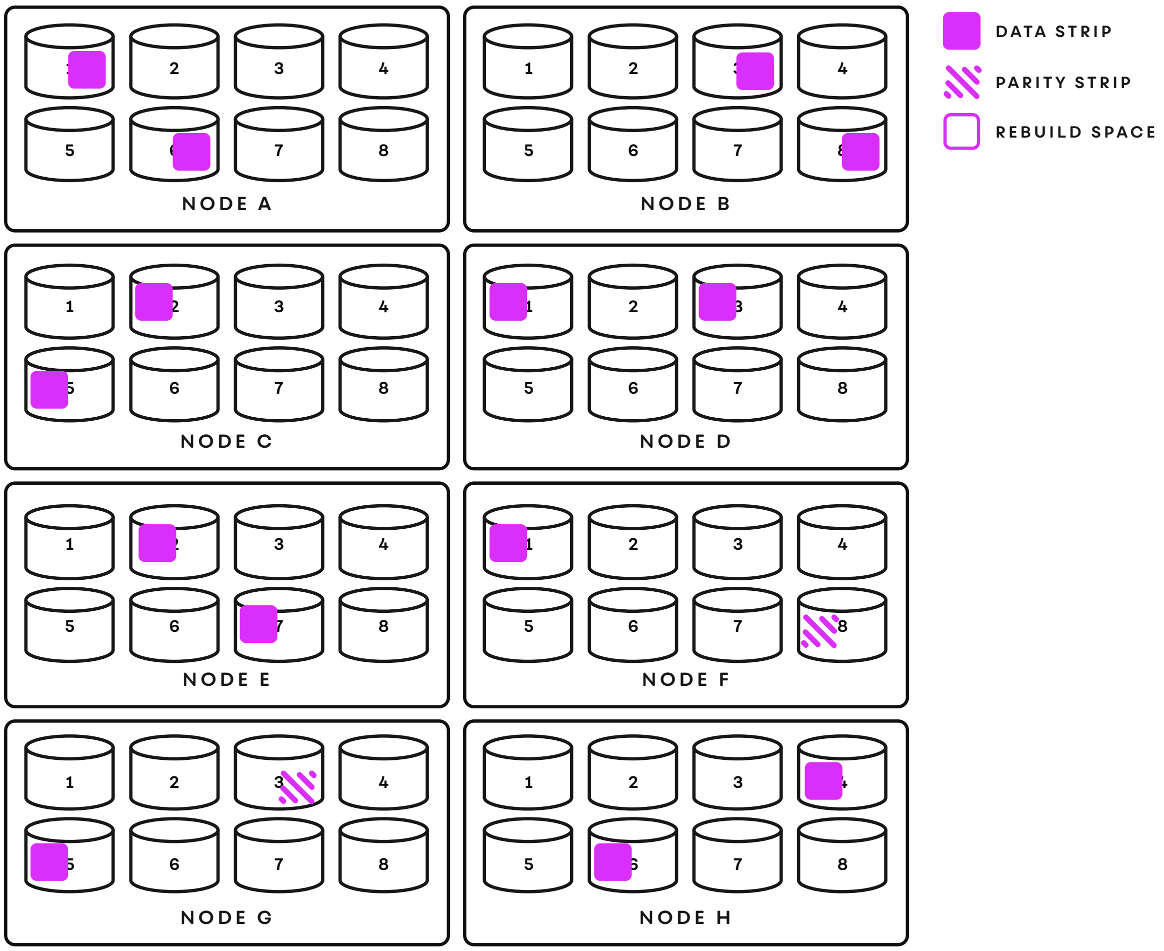
1. The loss of the node's compute power reduces the cluster's performance for servicing users and extends the rebuild time, thereby risking additional failures before completion.

2. Node failures require rebuilding many times as much data from a much larger pool of drives, which expands both the time and the universe of devices whose failure during the rebuild would result in data loss.

Figure 1 shows how a shared-nothing system would lay out a double-parity data stripe.



The problem should be obvious. The system consumes half its capacity with parity and rebuild space. One strip per node is inefficient for clusters under a dozen nodes, so the developers at the leading scale-out system decided to get clever: while they still used double-parity, they placed their data with 2 strips per node, and placed strips on every node, as shown in the figure below.



This striping of 14 data strips and 2 parity strips per stripe, boosted system efficiency from 50% to 87.5% but it created a couple of new problems:

1. Every node failure is an unprotected rebuild of a large amount of data, opening the door to data loss by uncorrectable read error.

2. It doesn't allocate space for the system to rebuild to. When a drive fails, the system must either wait for the failed drive to be replaced before rebuilding, or place three strips of each stripe on one or more nodes. If a node holding 3 strips of a double-parity stripe fails, data will be lost.

The second problem can be addressed by simply adjusting the erasure-code stripe width from 14D+2P to 12D+2P, which only reduces efficiency from 87.5% to 85.7%.

How Shared-Nothing Systems Handle Failure

In this section we will conduct some of what Einstein called Gedankenexperiment (thought experiments) and examine how a shared-nothing cluster will behave when exposed to device failures and errors. Our experiments will be performed on a cluster with:

- 8 nodes
- 24 15.36TB SSDs per node
- Data striped 12+2 with 2 strips/node (1n:2d)

This will allow us to logically determine the combinations of failures and/or errors that could lead to data loss and the probability that such a chain of events might occur.

Let's start off with a disk failure. In a cluster of 8 nodes, each with 24 SSDs (192 total SSDs), each with a 1% annual failure rate, we should expect 8*24*.01 = 1.92 drive failures a year. As long as the system can rebuild before the next failure our data is safe.

On our example system that stripes 12D+2P, the system has to read 12 data and/or parity strips and then write the rebuilt data to a new SSD. This will transfer 13 times the amount of data on the failed SSDs, just under 200 TB for 15.36 TB SSDs. This data volume influences the risk of data loss in two ways: First it determines how long the rebuild will take, and second, the amount of time data is at risk from additional failures. Reading and writing more data also increases the risk of an uncorrectable read error.

If our system can rebuild at 1 GB/s, –not unreasonable for a system still serving users–reading and writing the 200 TB needed to rebuild the data on a full 15.36 TB drive would take 55 ½ hours. During those 55 ½ hours the system could lose data in three ways:

1. **The failure of any of the 7 nodes other than the one with the failed SSD.**
   7 nodes * 1% AFR * (55.5/8760 hours/year) = .044%

2. **The failure of two more SSDs.**
   2nd failure: 191 remaining SSDs * 1% AFR * (55.5/8760 hours/year) = 1.2101%*
   Since the second failure has to occur for a third failure to be possible in the first place we multiply those two probabilities:
   3rd failure: 190 remaining SSDs * 1% AFT * (55.5/8760 hours/year) = 1.2038%
   1.2101% * 1.2038% = .0146%
   Note this doesn't adjust the rebuild time for the additional load of rebuilding 2 lost strips simultaneously and so may understate the risk.

3. **A 2nd drive failure combined with an uncorrectable read error during the rebuild.**
   2nd failure probability 1.2101% * (199.9TB data read/written / 12,500 TB SSD UBER) = .00023%

Add them all up and we get a 0.059% probability of data loss each time an SSD fails. With an expected 1.92 failures a year that's a .11% annual data loss rate.

Now let's see what happens if a node fails. Our example has 8 nodes and a 1% node AFR that has an 8% probability. Because each node holds 2 data strips, data will be lost if anything goes wrong during the rebuild. With 24 SSDs taken offline, that rebuild is going to take a lot longer because the system will need to read and write up to 2952 TB.

If we use the same 1 GB/s rebuild speed, which is generous since the system is now rebuilding with 7 nodes, it will take 820 hours (over 30 days). During that time the system can go offline and/or lose data when any of three things happens:

1. **A second node failure**
   7 nodes * 1% AFR * (802/8760 hours/year) = .64%

2. **An SSD failure**
   168 surviving drives * .01 * (802/8760 hours/year) = 15.4%

3. **An uncorrectable read error**
   2952 TB / 12,500 UBER = 23.62

Add them up .64+23.62+15.4=39.65%, then multiply that by the 8% annual node failure rate and there's a 3.17% risk of data loss by node failure per year. This dwarfs the .11% risk for loss for failure chains that begin with drive failures and bring the overall data loss expectation for our 8-node cluster to 3.28%. Over the 5-year life of a cluster, the risk of data loss is over 16%.*

If the very thought of a 16% data loss expectation makes you shudder, you may be a steely-eyed storage administrator who'll make sure their storage vendor replaces the failed node, and transfers the SSDs to reduce their exposure.

Imagine, however, what happens when we grow the cluster to hundreds of nodes, thus increasing the number of node failures.

Of course, all these calculations only predict data loss from failures within the system. Shared-nothing systems are also vulnerable to failures of the chassis, power distribution, or networking between the nodes. While many of these failures are comparatively transitory, allowing the system to recover when the power or network is restored, shared-nothing systems can do little to keep data available through extrinsic failures.

ENTER DASE – RELIABILITY AT SCALE

Shared-nothing file systems are built around an implicit assumption that failure events are independent. The system can withstand the failure of 2 drives or 1 node, but any failure beyond that before the system is repaired will result in data loss. This is in line with the enterprise storage systems of the day with RAID5 or RAID6 protection.

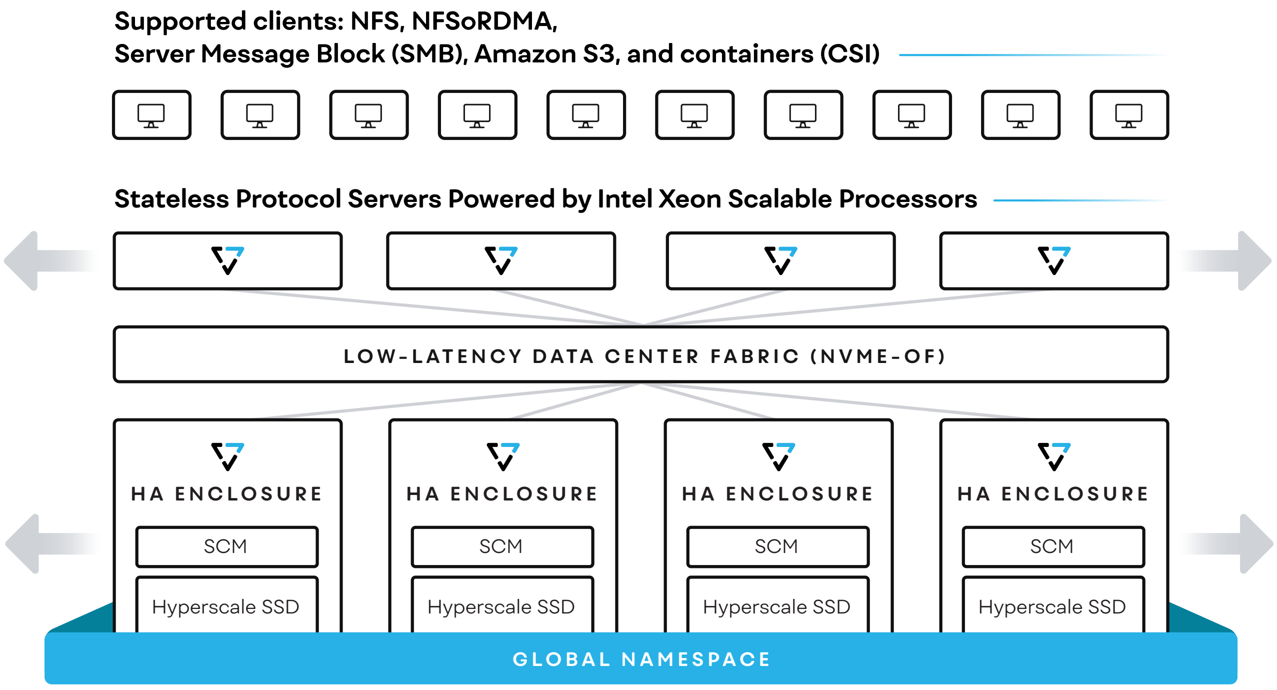
When VAST set out to design Disaggregated Shared-Everything Architecture (DASE), we had exabyte systems with hundreds of servers and thousands of SSDs in mind. Systems that big have to be designed to accommodate device failures not as rare events but as regular occurrences that should be healed in place.

- * Yes the drives from a failed node could be transferred to a replacement node and brought back on line. We've chosen to ignore this kind of heroic effort for two reasons:
- ** The risk of replacing SSDs in the wrong slots or making other mistakes during this heroic repair effort are significant but unquantifiable.
- *** Acquiring and installing a new node will introduce a significant delay (four hours best case to weeks for remote sites) during which data on the effected system will be unavailable. While data unavailable events are less serious than data losses, they're still unacceptable to the steely-eyed-storage-admin.

A Philosophy of Reliability

VAST Universal Storage is designed for cloud scale. A philosophy of reliability is the foundation of that design. This philosophy is based on two key concepts beyond the basics of high media resilience:

- **Simplicity** – Simple things, including software, have fewer points of failure and failure modes, which makes them more reliable and resilient.
- **Fail-in-Place** – Ensure systems self-heal without human interaction.



Supported clients: NFS, NFSoRDMA, Server Message Block (SMB), Amazon S3, and containers (CSI)
Stateless Protocol Servers Powered by Intel Xeon Scalable Processors
LOW-LATENCY DATA CENTER FABRIC (NVME-OF)
HA ENCLOSURE
SCM
Hyperscale SSD
GLOBAL NAMESPACE

The DASE Architecture

In VAST's DASE architecture every protocol server has direct access to, and mounts, every Storage Class Memory (SCM) SSD and QLC hyperscale SSD in the cluster over the NVMe fabric. All media is shared among all the protocol servers. In the DASE architecture servers don't own SSDs, they share them.

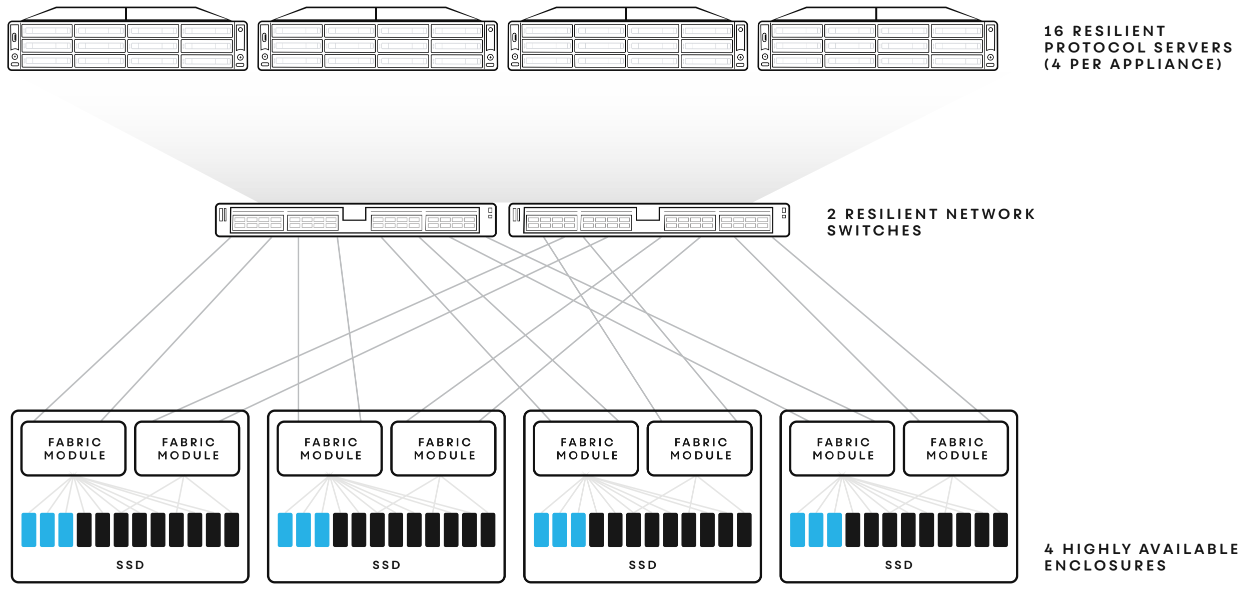
Protocol servers process client requests to completion, directly accessing the metadata from SCM and the data on hyperscale SSDs to fetch or store the data requested. This vastly simplifies the read and write processes as protocol servers access data directly without the need to coordinate with multiple other nodes. VAST protocol servers are stateless. They store all the system's state in the shared enclosures.

LAYERS OF SURVIVABILITY

VAST's DASE architecture creates layers of survivability to provide defense in depth against data loss. Those layers begin at the protocol server.

When a protocol server goes offline, the system assigns the virtual IP addresses that "node" was servicing to other protocol servers in the cluster. All system functions, such as rebuilding after an SSD failure, are performed by any available protocol server. The system will continue to run and provide a full set of services with a single operating protocol server– and the system schedules tasks across the surviving servers at a reduced performance level, of course.

For VAST's reliability calculations we'll use a VAST cluster comparable to the 8-node shared-nothing cluster. This VAST cluster with 4 enclosures and 16 protocol servers has a usable capacity of 2.38 PB vs 2.21 PB for the shared-nothing system. These calculations also assume that failed servers are repaired or replaced (MTTR) within 72 hours to allow for worst-case shipping of replacement parts.


16 RESILIENT PROTOCOL SERVERS (4 PER APPLIANCE)
2 RESILIENT NETWORK SWITCHES
FABRIC MODULE
SSD
4 HIGHLY AVAILABLE ENCLOSURES

The Example VAST Cluster

To calculate the risk of all 16 protocol servers failing within a 72 hour period, we multiply the 16% (16 nodes * 1% AFR = 16%) likelihood of a server failing times the ever-decreasing probability that each server in turn would fail for 15 more iterations. Because 72 hours is just .0082 of a year, any given server has just a .0082% probability of failing in 72 hours. The odds against all 16 servers failing within 72 hours is just 1 in 10E50.

It's much more likely the data center will lose all its redundant power sources, and since the servers are stateless the loss of all of a cluster's protocol servers would be a data unavailability event. This is admittedly a bad thing, but no data would be lost and the system can be quickly brought back online.

PROTECTING DATA FROM LOSS

While ridiculously high reliability at the protocol server layer is great, it only provides protection against the storage administrator's venial sin–allowing data to be unavailable. Let's turn our attention to the more important job of preventing the storage admin's one mortal sin–losing the user's data.

Locally Decodable Codes

The first step in any modern, efficient data protection scheme is protecting the data from device loss using an erasure-code that lets the system recalculate any data that's erased when a device fails. VAST protects all data against as many as four simultaneous device failures with as little as 2.7% overhead.

This combination of high protection and low overhead is enabled by VAST's locally decodable codes, which allow the system to rebuild from an SSD failure by reading from only 1/4th of the surviving SSDs. Our 4 enclosures hold a total of 176 hyperscale SSDs and so the system will write 146 data strips plus 4 parity strips (146D+4P) in each erasure code stripe.

The DASE architecture shares ownership of all the SSDs in a cluster across all the protocol servers in that cluster, and the erasure code stripes are distributed across all the SSDs. In the event of an SSD failure, the rebuild process is parallelized across all the protocol servers and all the SSDs in the cluster to minimize rebuild time. In fact, since data is not at risk, VAST systems throttle repairs of single SSD failures to approximately 30 hours to minimize the performance impact of the rebuild.

There are theoretically two ways for a failure cascade to cause data loss. Both start with four SSDs in a cluster failing before the system completes rebuilding any of the four. It's only after four failures that the system enters an unprotected rebuild and becomes vulnerable to data loss with another failure or error.

We will call it four simultaneous failures for short. That leaves two paths to data loss:

- Four simultaneous failures plus a fifth drive failure before the rebuild of the first failure completes
- Four simultaneous failures plus an unrecoverable read error of some unprotected data

Because both of those paths require an additional failure when the system is recovering from four SSD failures in rapid succession, let's start by calculating the likelihood of the system suffering four drive failures at essentially the same time.

- With 176 SSDs at a 1 % AFR the average system should suffer 1.76 SSD failures per year.

Since we're comparing how different architectures handle failures we'll use the same 55-hour rebuild time we used for the shared nothing math. The probability a cluster will see a second SSD fail before the first is rebuilt is:

- 1.76*(55 hours/8760hours/year) *1% AFR * 175 surviving drives = 1.93%
  Even though that's higher than the 30 hour rebuild time VAST systems target.

The likelihood of 3 failures within 55 hours:

- 1.93% *(55 hours/8760hours/year) *1% AFR* 174 surviving drives = .021%

So the likelihood of the fourth failure that puts data at risk if anything else goes wrong:

- 0.21% *(55 hours/8760hours/year) *1% AFR* 173 surviving drives = 2.29E-6

To put 2.29E-6 in perspective, that's a mean time of 437,544 years to even to put data at any risk of loss. To actually lose data requires either another 5th drive failure:

- 2.29E-6 *(55hours/8760hours/year) *1% AFR * 172 surviving drives = 2.47E-8

or an uncorrectable read error; in the worst case, the system would have to read 100% of its useable capacity (2385 TB) and write 63.60 TB of erasure code (2,448.60 total):

- 2.29E-06* (2448.60 / 12,500 TB UBER) = 4.48E-07

Add it all up, and the total risk of data loss is 4.72E-07—in other words, a worst-case time to data loss of over 2 million years.

In reality, VAST systems throttle the rebuild of a single SSD failure to complete in approximately 30 hours, reducing risk by about a factor of three. In addition, the system rebuilds erasure-code stripes with multiple SSD losses at a higher priority to further minimize the risk of data loss. Since rebuilds are parallelized across all the VAST protocol servers and SSDs in a cluster, rebuild speed is a function of cluster size.

Accepting Failures in Place

Our calculations above are built around the assumption that each incident, or failure chain, is independent. Effectively that means our calculations assume that failed devices are replaced before the next incident begins restoring the system to full redundancy.

Most enterprise storage systems are similarly built around the assumption that failed devices will be replaced before another incident occurs. These systems run a double parity protection scheme and can rebuild from 2 drive failures before replacements are installed, regardless of the timing of those failures. They can't repair a third drive failure, or in some cases node failures, until failed components are replaced.

VAST once again does things a bit differently. Following the cloud operator model of treating the SSDs in a VAST system as cattle not pets, VAST can afford to let SSDs die and leave them in place indefinitely. VAST systems don't simply rebuild so the data is once again protected against up to four (4) drive failures; they also ensure that, should space be available, the system will be not only be capable of reading data after an additional four failures, but that it can fully rebuild to that level of protection.

Fail-in-place is relatively simple on larger clusters, like our example four enclosure cluster, that have significantly more SSDs (176) than the maximum 150 SSD stripe width. As long as there's enough unused space the system can rebuild and maintain the 146D+4P coding through as many as 26 SSD failures without replacements, and still maintain the ability to protect data with up to four additional failures.

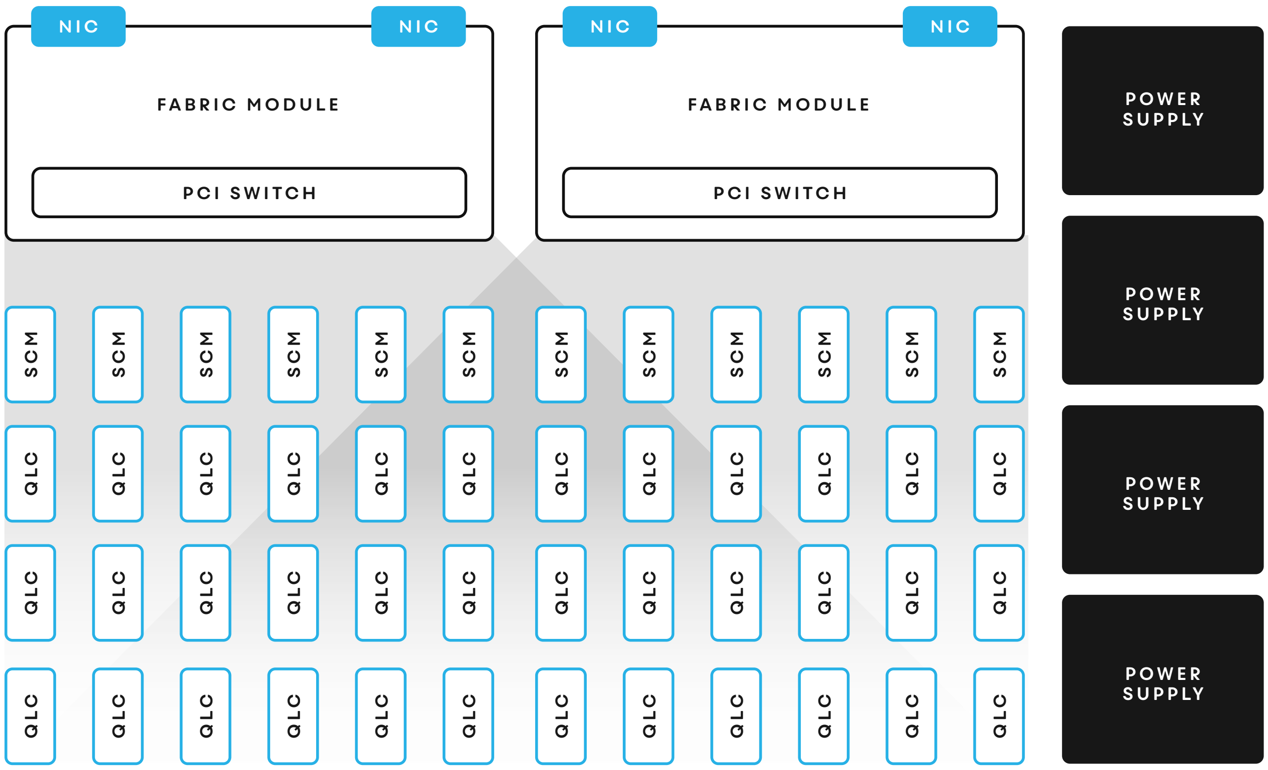
This fail-in-place philosophy doesn't just apply to large systems with SSDs to spare. As the number of SSDs in the system shrinks under 154, the system narrows the erasure-code stripes so there are always four SSDs that don't participate in each new stripe. In a small VAST system with just 1 enclosure there are 44 hyperscale SSDs so the system writes stripes 36D+4P = 40 strips wide, with 4 SSDs excluded from each stripe. This exclusion ensures that stripes can be rebuilt without reserving drives, or even space as spare.

When an SSD fails on that system, the total number of SSDs drops to 43 so the system narrows the new erasure-code stripes it writes to 35D+4P until the failed SSD is replaced. Should the system reach the point where the widest stripes are at-risk, the system will restripe the at-risk data, excluding four SSDs from each new stripe.

Regardless of the system size, as long as there are sufficient SSDs to store the user's data a VAST system will continue to protect that data.

Highly-Available Enclosure

The SSDs the system erasure-codes across are held in highly available enclosures to ensure connectivity between the protocol servers and the SSDs. As you can see in the block diagram below, each VAST enclosure has redundant network interface cards plugged into redundant fabric modules, all powered by four redundant power supplies.



While the fabric modules are servers, their primary function in DASE is to route NVMe messages between the NICs, and thereby the fabric and the SSDs. Under normal conditions each fabric module provides connectivity between half the SDDs, including the SCM SSDs, and the NVMe fabric.

Should a fabric module fail, the remaining fabric module will reprogram its PCIe switch complex to take over all the SSDs. Because all the data, including the all-important metadata and meta-metadata that make up the system's state, are directly written to the persistent SSDs, even if both fabric modules fail all the data will be available when either fabric module is replaced.

Calculating the probability of both fabric modules failing is simple. Our cluster has eight fabric modules at a 1% AFR that makes the probability of a fabric module failure 8% per year.

If we use a worst-case 72-hour replacement time, the likelihood of the second module failing is:

- (72 hours MTTR /8760 hours/year)*1% AFR = 0.0082%

Note that this calculation doesn't include the number of surviving Fabric Modules in the cluster because the only failure that will effect this system's availability is the failure of the one remaining fabric module in the same enclosure. Failure of the fabric modules in any other enclosure will simply degrade that enclosure's network bandwidth but not take the system offline.

That .0082% probability, or roughly 1 in 12,000 odds, is obviously based on the 72-hour worst case spare parts delivery over a weekend. With on-site spares and VAST's Co-Pilot support, customers can bring that probability down to .0005 (1 in 200,000).

Customers can completely eliminate the risk of full enclosure failures with the Enclosure HA feature described below.

Rack-Scale Resilience the VAST Way

To provide rack-scale resilience, VAST systems adjust their resiliency model to treat each VAST enclosure, as well as each SSD, as an independent failure domain. This allows VAST systems with Enclosure HA (High Availability) enabled to recover from enclosures going off-line because of issues such as a rack losing power.

Systems that enable Enclosure HA provide this additional resilience by limiting to two the number of data and/or parity strips of each erasure-coded stripe that can be stored on each enclosure. While this reduces the width of the erasure code stripes from the 146+4 coding normally used in large VAST clusters to 26+4 in a 15-enclosure system, that system will have 13.3% erasure-code overhead. That compares favorably to the 30% or greater overhead of a dispersal coding system like those erasure-coded object stores that promise 10 or more 9s of durability.

CONCLUSIONS

The very scale of multi-petabyte storage systems creates challenges for conventional shared-nothing storage systems. As systems scale to hundreds or thousands of drives, more drives are going to fail. This increases the likelihood of multiple failure scenarios that cause data loss in conventional storage systems.

VAST's Universal Storage systems are designed specifically for large-scale environments with layers of resilaince. This allows a VAST system to survive, and continue servicing users, after multiple failures at each layer. Multi-petabyte data stores demand unprecedented reliability. VAST DASE delivers with a failure-in-place philosophy and a formula of stateless servers, plus more reliable SSDs, plus a highly available enclosure and erasure codes that can survive up to four simultaneous drive failures, or a failure of one of those highly available enclosures.

When we compared VAST to the leading shared-nothing storage system, our example 2.5 PB VAST system had a Mean Time to Data Loss (MTDL) of over 2 million years where the shared-nothing system's MTDL was only 1,000 years. While the risk of data loss on either system is small, the VAST system is over 2000 times more reliable.

Notes on the Math in This Paper

The calculations in this paper are intended to illustrate the reliability differences between storage systems based on the shared-nothing and DASE scale-out architectures. These calculations are purposely abstracted from the implementation particulars of these architectures and the actual components used. We've used an AFR of 1% and a MTTR of 55 hours for all the examples to illustrate differences that are architecturally significant, eliminating the implementation specifics.

This of course means that the absolute probability projections of our calculations are not intended to be accurate, but to illustrate the difference. So 1% was chosen as a round number, and 72 hours was chosen because it fit a support contract example for drive replacements. SSD failure rates range between almost 2% (based on published field results that include early SSDs) to .3% on some manufacturer spec sheets. There are no publicly available reliability figures for servers so 1% is a simplification.

When it comes to the mean time to repair, or rebuild time, it too is implementation specific. Yes, some shared-nothing systems have rebuild jobs that run for weeks, but others, with smaller SSDs and a better rebuild method, can recover from a failed SSD in roughly the same tens of hours as a VAST system. Using 55 hours in all the math reveals the differences that are architecturally significant and hides the implementation differences.

To learn more about how **Universal Storage** can help accelerate your A.I. initiatives,
reach out to us at hello@vastdata.com.

GLOSSARY

MTBF: Mean Time Between Failures, also called MTTF for Mean Time To Failure, measured in hours. MTBF is an average, so a million-hour MTBF means that in a population of a million drives the failure rate will average out to one per hour. MTBF should not be confused with device longevity; no drive hard or Solid-State will operate for over 114 years.

AFR: Annual Failure Rate, expressed as a percentage, is the average percentage of devices that fail per year. To convert between MTBF and AFR divide the 8,766 hours in a year by the MBTF.

- AFR=8766/MBTF
- MTBF=8766/AFR

MTTR: Mean Time To Repair is the time between a device's failure and its repair or replacement. MTTR of failed components is an important factor in calculating the probability of system failure and/or data loss.

Storage Class Memory: Storage Class Memory (SCM) defines a class of devices that deliver the persistence of conventional flash with performance and endurance profiles between conventional flash and DRAM. Examples include phase change memory, high-performance flash, 3D XPoint, and magnetic memories (MRAM).

Hyperscale SSDs: Hyperscale SSDs are designed to deliver maximum capacity for hyperscale data centers. They lack the features enterprise SSDs use to enhance endurance such as DRAM cahces, dual-port controllers, and all the power failure protection circuitry required to protect the data in the volatile DRAM cache.