Accelerating Instant Recovery:

Cohesity Outperforms Veeam-Based System

A DeepStorage Technology Validation Report

Prepared for
Cohesity

About DeepStorage

DeepStorage, LLC. is dedicated to revealing the deeper truth about storage, networking and related data center technologies to help information technology professionals deliver superior services to their users and still get home at a reasonable hour.

DeepStorage Reports are based on our hands-on testing and over 30 years of experience making technology work in the real world.

Our philosophy of real world testing means we configure systems as we expect most customers will use them thereby avoiding "Lab Queen" configurations designed to maximize benchmark performance.

This report was sponsored by our client. However, DeepStorage always retains final editorial control over our publications.

Contents

Introduction

As the building blocks of IT infrastructure have standardized into fungible, if not fully commoditized, building blocks of x86 virtualization hosts, storage and Ethernet many organizations now acquire these building blocks in pre-integrated solutions. These organizations have outsourced the expertise needed to architect a system that properly balances network bandwidth, CPU capacity, storage requirements, and the like to vendors providing converged and hyperconverged solutions.

While converged and hyperconverged systems liberate IT departments from the heavy lifting of designing their primary IT infrastructure, protecting the virtual machines that run on that infrastructure still requires them to architect a secondary storage and backup infrastructure. Users must evaluate multiple devices and software for their backup solution, including deduplicating backup appliances, master control and management servers, data mover/proxies/media servers, and of course, the network impacts of the backup process.

A new generation of data protection solutions, led by Cohesity, are bringing the sort of integration to backup that HCI vendors brought to hosting VMs. Hyperconverged secondary storage solutions such as the Cohesity DataPlatform deliver a single integrated system that provides both a scale-out storage repository and a complete data protection system from policy engine to data mover.

The Bottom Line

The recent success of converged and hyperconverged infrastructure solutions clearly demonstrates that many IT buyers would rather purchase pre-engineered solutions than spend the time, effort, and expertise to build a system in-house. Cohesity's DataPlatform brings hyperconvergence to the secondary storage market, combining a scale-out file system with all the management and data movement of a backup solution.

This Technology Validation Report compares Cohesity DataPlatform with a fairly typical scale-out solution built around Veeam Backup and Recovery, the leading backup software in the VMware era. We focused on how well each platform performed when using direct recovery; that is, the ability to run virtual machines directly from the backup repository.

The Cohesity system delivered over three times the performance, in transactions per minute, than the Veeam-based system using recovered VMs running the HammerDB TPC-C "like" benchmark. This performance difference was even more significant when migrating the VM back to primary storage via Storage vMotion. The VM migrated off the Cohesity in 1:17 while maintaining over 45,000 TPM. Migrating from the Veeam based system backup took 8:19 with extended periods where it only processed 7,000 TPM.

With this level of performance the Cohesity DataPlatform is not just a backup appliance, but a platform fully capable of running your recovered apps before motioning them back to primary storage.

This Technology Validation Report examines the Cohesity DataPlatform and compares it to an internally engineered system using Veeam Backup and Replication and Veeam's scale-out data repository. The comparison is based on direct recovery, a hallmark feature of modern data protection systems.

To see just how well the two solutions support running VMs, we performed direct recovery of a VM running the OLTP application benchmark HammerDB. We measured VM performance on each system when recovered directly on the backup repository, and also as it was returned to the primary storage array via Storage vMotion.

In all cases Cohesity provided several times the performance of the Veeam-based solution. While Veeam's performance could almost certainly be improved by optimizing the storage layer, pre-engineered systems like Cohesity's provide complete, optimized solutions out of the box. This eliminates the need for in-house expertise on backup repository engineering that's typically required with traditional backup software. Since many organizations only update their backup/recovery infrastructure every 3-5 years, the expertise they develop each time is obsolete by the time the next upgrade comes along, and may even prevent the organization from considering new approaches.

Direct Recovery

Whether you call it instant recovery (Veeam), cloning (Cohesity), or just black magic, direct recovery is the ability to bring virtual machines back into action in minutes.

Direct recovery has revolutionized data protection by freeing us from the interminable wait as data was copied from a tape, or Data Domain, backup repository to a new home where the VM could actually run.

When direct recovery was flashy new technology, we were so pleased that the domain controller was up in just a few minutes that we didn't pay much attention to how quickly the freshly recovered VMs ran. After all, a slow database server restored in10 minutes is a lot better than a fast database server restored in 10 hours.

Now that direct recovery is more widely available, vendors have to improve their acts. We've gone from "Spot The Talking Dog", where the novelty of a dog that speaks was enough to fill the seats; to Triumph the Insult Comic Dog , who actually has to say something funny.

Cohesity takes direct recovery from a checkmark feature to a business-critical, high-performance application. Direct recovery becomes a realistic way to bring your systems back online and running during an emergency, and to verify that systems restore properly.

Once application VMs can run with reasonable performance from the secondary storage system, organizations can leverage the copies of the VMs they create for protection for other uses. For example, developers can test updates on full copies of the VMs, and because those copies are running from the secondary storage system, the tests won't affect the production system's performance.

The Cohesity DataPlatform

The Cohesity DataPlatform integrates a scale-out file system, SpanFS™, with Cohesity's DataProtect, which provides the management and data transport engines traditionally provided by backup applications like NetBackup or Veeam Backup and Replication.

Because the DataPlatform is software-defined, Cohesity offers it in hyperconverged appliances, like the one we tested. It's also available on HPE or on Cisco UCS servers, as a vSphere virtual machine for remote and branch offices, and even as VMs in public cloud environments.

Cohesity's distributed file system SpanFS™ forms the foundation of the DataPlatform. It provides storage not only for DataProtect backup jobs but also via NFS, SMB, and S3. This lets organizations use the DataPlatform as a target for applications that perform their own backups, email and other archiving, media repositories, and other secondary storage applications.

The file system provides those applications with an SSD-accelerated file system with global data deduplication, compression, erasure coding, and encryption that can scale across 100s of nodes with petabytes of capacity. The file system supports essentially unlimited snapshots and clones so each developer can have their own copy of the full production database at minimal cost in system resources. Data is always synchronously replicated and stored on two nodes, ensuring data is consistent when the system acknowledges the write.

Cohesity's DataProtect provides the management, scheduling, and data movement functions to back up physical Windows and Linux servers, SQL Servers, file shares, and Pure FlashArrays like the one we used for testing, as well as virtual machines to snapshots in the Cohesity file system.

Users define snapshot frequency, snapshot retention, and whether snapshots should be replicated to another Cohesity cluster, to a cloud storage repository, or to tape. Users can also define extended retention policies to store snapshots with reduced granularity over longer retention periods.

Cohesity's SpanFS file system organizes data into storage domains: logical containers or volumes, where data reduction and cloud tiering policies are applied. Each domain contains one or more "views" that store cloned VMs and present storage via standard SMB/NFS/S3 protocols to allow applications beyond Cohesity's built-in backup application to use the Cohesity datastore and present directly recovered VMs.

Users can also define multiple views to limit access to data or to assign a QoS level to the view. Cohesity has predefined several QoS policies, including some specifically designed for applications like Oracle RMAN that include their own data mover and use the Cohesity system as an NFS datastore. In our testing we assigned the TestAndDevHigh QoS policy to the primary HammerDB and the TestAndDev policy to the load generator VMs.

In addition to replicating snapshots to a public cloud provider, which has become a common feature for backup applications, Cohesity can also use a cloud provider, or on-premises object store, as a storage tier for migrating infrequently accessed data blocks to cloud storage if the local HDDs fill beyond a user-definable threshold.

Cohesity System Configuration

We used a Cohesity C2105 block to test the Cohesity DataPlatform. The C2105 is based on Intel's OEM version of the now-familiar 2U, four node high-density server appliance.

Hardware Specs

- 2 Xeon E5-2603 v4 processors (6 1.7Ghz cores)
- 64GB RAM
- 800GB NVMe SSD
- 6TB HDD (3x2TB)
- 2x10Gbps Ethernet

The Scenario – Build Vs Buy

The new CIO is all gung-ho to use a hyperconverged solution for this year's backup infrastructure refresh. He thinks merging the current mashup of NetBackup, Veeam, TSM (it came in an acquisition), Data Domain, and tape with a modern solution that exports to Amazon S3 lets him check off several of his digital reformation MBO items.

The problem is that his head storage guy Jim, and his "genius" backup guy Malvin (who've been at the company since 1983) say that they can build a scale-out backup infrastructure with Veeam Backup and Recovery. They also say that because Veeam has data reduction, cloud storage, and a scale-out data repository it will solve the company's backup problems—and they can build it all from the last generation servers they have in the boneyard.

Since all they have to buy are disk drives and maybe a few SSDs, they could build a new backup system and the department would still have enough money in the budget for Malvin and Jim's trip to DockerCon and an air hockey table.

Our Internally Engineered Veeam System

We set out to build the kind of Veeam backup system a pair of basically competent server and storage admins, who don't have deep expertise in designing a Veeam system for maximum performance, would build. Then we would test it to see how it compared to the Cohesity DataPlatform.

Since most storage admins think of backup data as being primarily sequential, and hard drives, even multi-terabyte hard drives, are pretty good at handling sequential data, we used 6TB 7200 RPM hard drives. Since we're the paranoid type, we used RAID 6 across the six drives in each server.

We're making no claims about our Veeam storage repository being optimized in any way. We'll simply argue that having four or more servers available using local RAID and Veeam's scale-out repository is an obvious solution. Could the same set of hardware deliver better performance if we used Storage Spaces Direct, vSAN, or Gluster as a scale-out file system underneath the Veeam repository? Possibly.

The very question “Could this system have been optimized by using file system X, SSD Y, or some other technique?” brings us back to the subject of expertise. Our admin friends could find a top notch Veeam VAR, ask questions on social media about configuring a Veeam repository, or acquire the expertise to optimize the Veeam repository themselves through experimentation.

While they’ll enjoy the process it will divert their valuable talents from the other 5,000 Post-Its on their Kanban board, many of which require institutional knowledge only they have.

By choosing a hyperconverged solution like Cohesity’s, the organization can outsource the expertise of building a balanced data repository to Cohesity, knowing they’ve designed the system to accommodate the 90th or 99th percentile of possible demand and therefore most users, without tweaking.

Comparing The Scale-Out Architectures

While both the Cohesity DataPlatform and the Veeam scale-out repository provide the elastic capacity promised by a scale-out architecture, they do so in very different ways. As a result, they perform differently.

Cohesity builds the DataPlatform on a scale-out file system that provides data protection by writing each block of data to drives on at least two separate DataPlatform nodes (using erasure coding or replication) as written to the file system. Because data is distributed across all the drives in all the nodes in small blocks, I/O requests are pretty evenly distributed across all the drives in the cluster, whether for a single backup or, more significantly, for a directly restored VM trying to do its job.

Veeam’s scale-out backup repository acts more like a backup job load-balancer than a scale-out file system. In the Veeam architecture, a scale-out repository is a collection of standard Veeam repositories. Backup jobs are allocated to repositories by policy, but any given backup job, or running VM, will use one of the constituent repositories (in our case, Windows servers with six 6TB hard drives each). Since each job or recovered VM will really only be hitting one of the repositories, each job is limited to the performance of that one repository’s storage.

Veeam’s design scales out capacity, and aggregates performance when you’re trying to get all 734 VMs in your data center backed up in your measly 11PM-6AM window, but adding more Windows servers to the repository won’t make HammerDB run any faster when directly recovered.

The Tests

We set out to compare the Cohesity C2105 appliance with the sort of Veeam backup infrastructure a couple of sysadmins without deep experience building backup architectures would come up with. Our tests focused on the direct recovery feature of both systems, which Veeam calls Instant Recovery and Cohesity calls Clones.

Because our admins assume backup jobs create primarily sequential I/O they selected 7200RPM disk drives, and because the Cohesity was scale-out they chose Veeam’s scale-out repository. Therefore, the Veeam repository consists of four Windows 2016 servers each equipped with six 6TB HDDs in a RAID 6 array managed by an LSI RAID controller.

We installed Veeam proxies on each of the four ESXi hosts that housed our test VMs to move the data, and installed a Veeam master VM on a fifth ESXi host. For more detail on the system configurations see The Test Environment section towards the end of this report.

The Workloads

Once we racked up the Cohesity appliance and built out the Veeam infrastructure, the first step was to create a group of VMs to back up, restore and recover.

Over the years we've seen some secondary storage systems restore, or in this case recover, from full backups significantly faster than from an incremental backup. Other storage systems are much faster delivering the last backup's data than they are at accessing data from older backups further down the pile, so to speak.

To test how well the systems performed during recovery, we first had to create a chain of full and CBT incremental backups with enough data to expose big differences between performance from the full backup and one of the latest incrementals.

We set up four load generator VMs configured to emulate the block changes of typical applications. We tuned these applications to create the IOPS rates below using a Pure FlashArray //m10 for storage:

- IOmeter (File Server workload) 600 IOPS
- Jeststress 2010 400 IOPS
- JetStress 2013 1800 IOPS
- VDBench (Our OLTP workload) 1450 IOPS

Since we tuned the VMs to generate specific workloads by tweaking queue depths, I/O rates, and thread counts, the performance these VMs report from another storage system will not be representative of the performance of a real, unthrottled application.

We used HammerDB to measure how well the systems performed as direct recovery platforms.

HammerDB TPC-C "like" workload

HammerDB is an open source application benchmark that runs workloads modeled on the Transaction Performance Council's TPC-C, an OLTP benchmark, or TPC-H OLAP benchmark. Since the TPC only certifies results for benchmarks run by TPC members using TPC testing tools, HammerDB is generally referred to as a TPC-C "like" or TPC-H "like" benchmark. It performs the same database operations but since it's not the official tool there may be variations between HammerDB results and official TPC results.

HammerDB is an application-layer benchmark that generates requests to a database server; in our case Microsoft SQL Server 2012. The TPC-C like workload emulates a distribution company with virtual users entering orders and shipping documents for shipping products from a specified number of warehouses. The number of warehouses determines the size of the database; in our case there are1,520 warehouses for a database size of 200GB. Demand is managed by the number of virtual users.

During the data creation and backup period, HammerDB ran with 13 users creating about 8,000 IOPS.

In the performance testing phase we stepped up the number of users via HammerDB's autopilot function until the systems performance in transactions per minute plateaued.

The Backup Phase

We ran our 5 VMs continuously to generate traffic and create block changes. We set up the appropriate policies and backup jobs on each platform to create a full backup and an incremental backup every 2 hours. Using a high transaction rate on our load generator VMs meant that our incremental backups averaged around 375GB of changed blocks.

We ran first the Cohesity system until we had a total of 24 incremental snapshots stored. We then suspended the Cohesity backup job and ran a backup job on the Veeam infrastructure with the same two hour interval until we had 24 incremental backups on that platform as well. Twenty four incremental backups of our high data rate VMs could then stand in for a month of daily incremental backups on more typically laconic systems.

In both cases we configured the backup jobs on both platforms by accepting the defaults, except for the schedule. Those defaults did not include quiesing the applications and creating application-consistent snapshots.

The Results

Basic Backup/ Restore Performance

Our first test was to simply to measure how long it took to back up and restore our HammerDB VM. Since both Veeam and Cohesity leverage VMware's changed block tracking (CBT) and they weren't copying exactly the same data the variations between the solutions were smaller than the variations between backup jobs on the same platform. We therfore decided there was no real value in reporting the incremental backup speed.

	Cohesity	Veeam Based System
Full Backup Time	1:25	1:28

Table 1: Backup and Restore Performance

Both systems took about an hour and half to make an initial full backup of the HammerDB server, which has 633GB of data across 3 virtual disks that total just over 1.6TB of space. With such similar results, we assume the bottleneck was at the data source, vSphere and/or our all flash array.

Direct Recovery Performance

To test each system's performance during a direct recovery, we recovered all the virtual machines that we had previously backed up. Our plan was to emulate the panicked recovery after a primary storage failure, a scenario we'd run all too many times in our consulting days.

The HammerDB system represented the ERP system that senior management insists be up and running as soon, and as fast, as possible. The other load generator VMs represented the other applications that the ERP system, or the rest of our recovery relied on, such as email, Active Directory and the file server that holds the disaster recovery planning files. Since the ERP system was the highest priority, we used whatever QoS or other features that were available to give the HammerDB VM priority over the others.

On the Cohesity platform we used the cloning feature to create new VMs with a –Co suffix. We restored the HammerDB VM to a view configured with Cohesity's TestAndDevHigh QoS policy, which has the highest priority, while the miscellaneous load generator VMs were cloned to another view using the TestAndDevLow QoS policy, the next step down on Cohesity's policy ladder. The Cohesity system mounted the views as NFS datastores and instantiated the VMs into the resource pool we specified.

For the internally engineered system we used Veeam's instant recovery feature to create new VMs with a –ve suffix. Instant Recovery uses the Veeam's vPower NFS server we installed on the Widows repository server to, like the Cohesity system, mount an NFS datastore and instantiate the VMs into the vSphere environment.

Unlike the Cohesity system, Veeam didn't reset the MAC addresses on the VMs, so vCenter reported MAC address conflicts, but that's a minor issue to fix.While both Cohesity and Veeam use NFS to present the directly recovered VMs to vSphere, they do it in very different ways. Cohesity uses a distributed file system, and like many modern file systems, it creates a read-write clone of the VM. New writes are simply written to file system as part of the clone.

Veeam's repositories, by contrast, are primarily designed to store backup data and lack the sophisticated metadata of a modern file system to manage read-write clones. Veeam's solution uses a journal to hold new writes, much the way vSphere uses a journal for snapshots.

This journal is by default stored to the C: drive of the repository server. For our recoveries we redirected the journal to the VMware datastore on our Pure FlashArray to take advantage of its performance

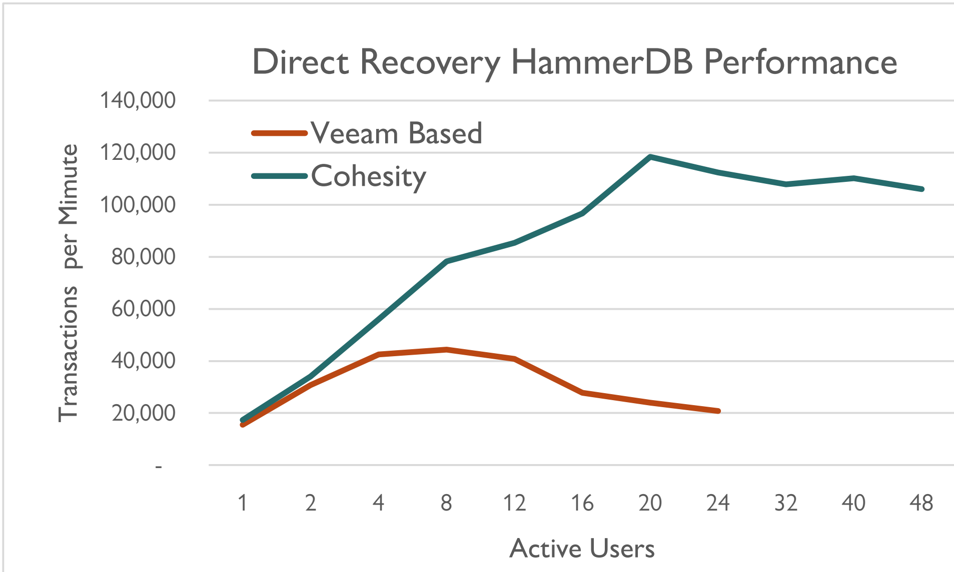

Figure 1: HammerDB Performance During Instant Recovery

To generate the data for Figure 1 above we first fired up the IOmeter and VDBench VMs with exactly the same configurations we used during the backup phase. These VMs were intended to simply generate load, as real world recoveries almost always involve more than one server.

We then ran the HammerDB VM for the same 2 minute warm-up and 13 minute test run cycle we used in the backup phase, but with an increasing number of users. We ran each platform until we found the point where performance plateaued and the storage system was the bottleneck.

For the Cohesity system the plateau was at 21 users at 118,417 TPM. The Veeam system topped out with 9 users at 44,821 TPM.

Storage vMotion Test

This test measures how the system performs while the HammerDB is moved from the direct recovery platform back to primary storage. We created a fresh clone of the HammerDB VM and started it running the TPC-C "like" workload with 13 users in 15 minute cycle (2 min warmup, 13 minute measured) with HammerDB reporting the TPM rate for each cycle.

Using Storage vMotion meant we had to redirect writes to a local Intel DC S3700 SSD on the Veeam repository server, where in the previous testing we redirected writes to the VMware datastore on the Pure FlashArray. Changing from one high performance all flash datastore to another didn't have a significant impact on the absolute performance of the VM.

We started the storage vMotion during the warmup period of the second cycle when migrating from the Cohesity, and during the warmup period of the third cycle when migrating from the Veeam vPowerNFS datastore. The start times are marked with a green dot on figure 2 below.

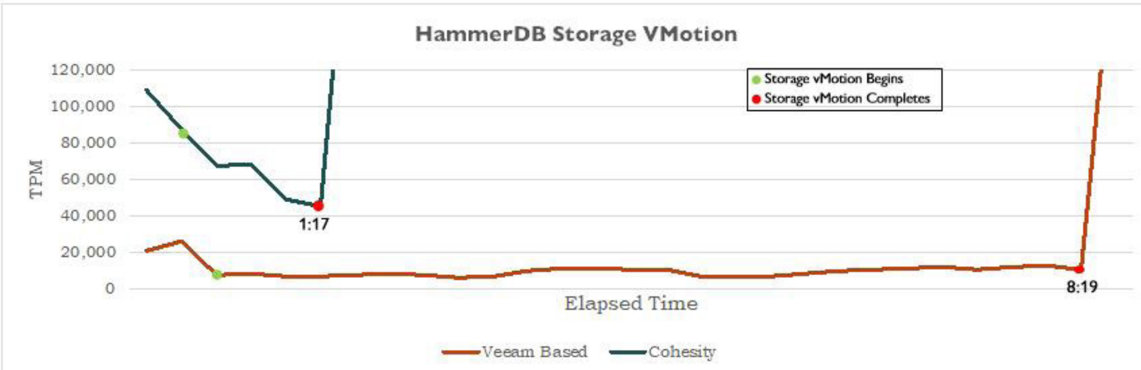

Figure 2: HammerDB Performance During Storage vMotion

The Cohesity system completed the Storage vMotion in 1:17 with a minimum TPM of 45,000, a little more than half the 85,000 TPM that system achieved with 13 workers without the vMotion. The Veeam system took 8:19 to complete the vMotion and for much of that time delivered under 7,000 TPM, less than a third the 26,000 TPM it delivered in the cycle before the Storage vMotion began.

Conclusions

At first glance, on paper, our internally engineered Veeam solution and the Cohesity DataPlatform have a lot in common. They both use modern data movers to protect our primary data and both save that data to a scale-out repository.

Once we got both solutions in the lab, big differences appeared, especially for use cases like the direct recovery case we tested that requires random I/O performance. The internally engineered solution was designed for backup performance, and delivered that performance, matching the Cohesity appliance.

The real test, however, was how well each solution supported VMs via direct recovery. Here the Cohesity platform's design as a more general-purpose file system delivered 2.6 times the transactions per minute, and migrated the VM back to primary storage over six times faster.

This higher VM performance makes Cohesity's clones useful for more than backup verification and emergencies. It opens to the door for test/development and other use cases.

Appendix I - The Test Environment

Our testing used eight servers: four as vSphere hosts for the VMs we were going to protect and recover, and four Windows servers that housed the Veeam scale-out backup repository.

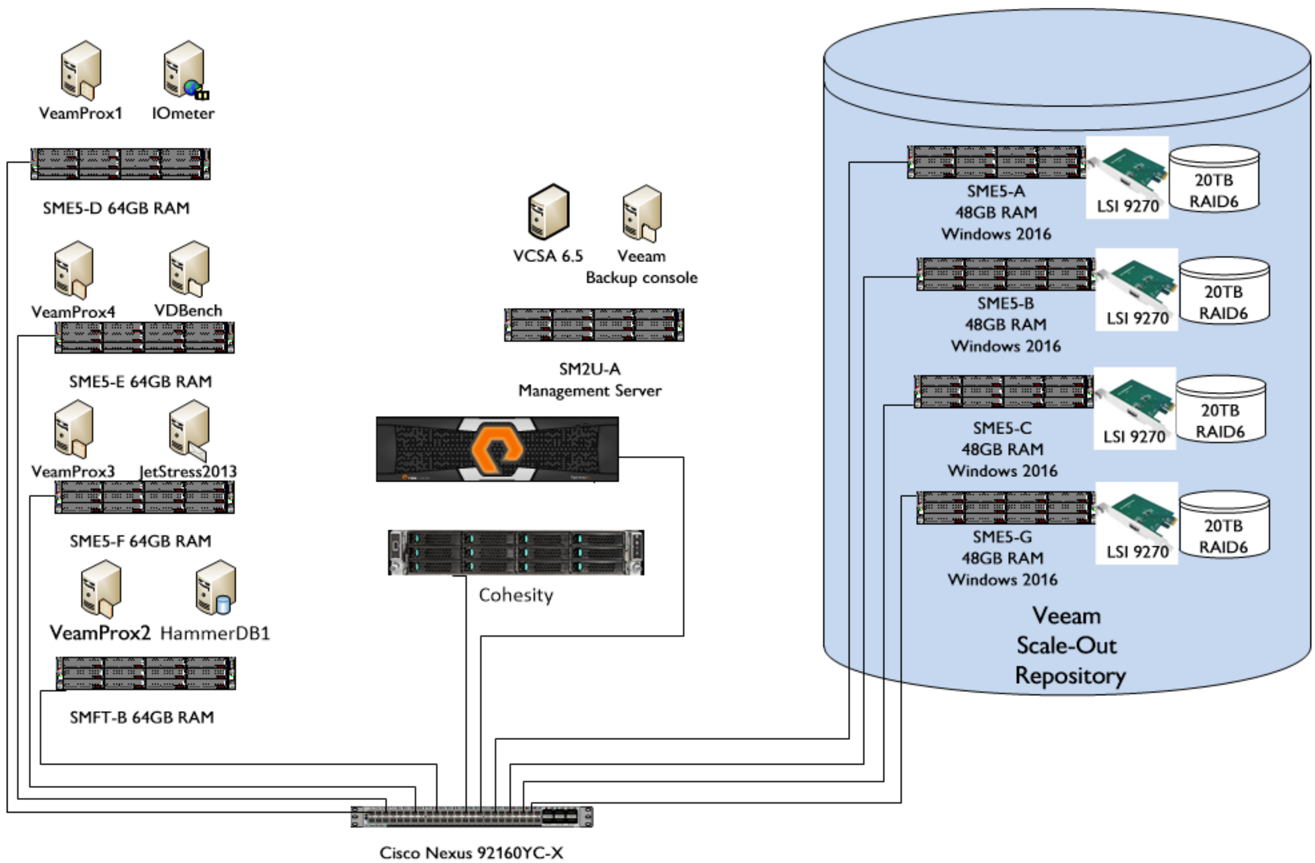

Figure 3: The Test Envionment

All eight servers connected to a pair of Cisco Nexus 92160 10/25gbps Ethernet switches, as was the Pure Storage Flash Array used to host the VMs and the Cohesity appliance.

Server Specifications:

vSphere Hosts

- Supermicro Superserver 6027R-3RF4+
- Xeon E5-2640 v1 processors (6 cores, 2.5Ghz)
- 64GB RAM
- Mellanox ConnectX-4 LX 25 Gbps NIC

Windows Servers

- Supermicro Superserver 6027R-3RF4+
- Xeon E5-2640 v1 processors (6 cores, 2.5Ghz)
- 48GB RAM
- Emulex OCe14102 10Gbps CNA
- LSI 9270-8i RAID controller
- 6 WD RE GOLD 6TB HDDs
- 200GB Intel DC S3700 SSD

The Veeam Environment

Our Veeam repository is based on the admittedly naïve assumption that all implementations of any given feature are created equal. We used the RAID controller on each server to build a RAID 6 array from the six 6TB hard drives with a net capacity of 21 TB, formatted that volume in NTFS, and installed a Veeam repository on it.

While Microsoft's newer ReFS does have some advantages over NTFS as a Veeam repository, those advantages are related to synthetic full backups and other features we weren't testing. We assume Windows SysAdmins like Malvin are more comfortable with NTFS.

We then built a Veeam scale-out repository out of the four individual repositories.

We installed a Veeam Proxy VM (4vcpu, 12GB RAM Windows 2016) VM on each host that would hold our test VMs. The proxy VMs have direct access to the 10/25Gbps network the Pure FlashArray is connected to.

An organization repurposing servers being decommissioned for newer, denser, ones could equip those servers with the same drives Western Digital was generous enough to send us and a SATA SSD from NewEgg for about $2,000[1]. A comparable R730 prices out at just under $9,000 on the Dell site.

1 WD RE Gold 6TB disk $250, Samsung PM863a 480GB SSD $350

Appendix II – Workload Definitions

HammerDB TPC-C "like" workload

HammerDB is an open source application benchmark that runs workloads modeled on the Transaction Performance Council's TPC-C, an OLTP benchmark, or TPC-H OLAP benchmark. Because the TPC only certifies results for benchmarks run by TPC members using TPC testing tools, HammerDB is generally referred to as a TPC-C "like" or TPC-H "like" benchmark. It performs the same database operations but since it's not the official tool, there may be variations between HammerDB results and official TPC results.

For this report we used our integrated HammerDB virtual machine that uses version 2.20 of HammerDB to generate a TPC-C "like" workload and runs the SQL Server 2012 instance for the database. This VM has 10 vCPUs and 6GB of memory. The memory allocation is small at 6GB to prevent SQL Server from caching a substantial percentage of the data being accessed.

IOmeter File Server Access Specification

The IOmeter file server access specification was originally released by Intel in the mid 1990s. It has been widely distributed and used for many product reviews and other published test sets.

I/O size (bytes)	Percentage of total I/Os	Read Percentage	Random Percentage	Alignment
512	10	80	100	Sector (512)
1024	5	80	100	Sector (512)
2048	5	80	100	Sector (512)
4096	60	80	100	Sector (512)
8192	2	80	100	Sector (512)
16384	4	80	100	Sector (512)
32768	4	80	100	Sector (512)
65536	10	80	100	Sector (512)

All tests were performed using the Full Random data pattern to minimize the impact of data reduction.

DeepStorage VDbench Pseudo OLTP Workload

The DeepStorage Pseudo OLTP workload is designed to emulate a relational data base server running an online transaction processing system like ERP. Where most "OLTP" tests simply perform 4KB or 8KB reads and writes randomly across the entire device under test, our goal was to build something better than the 8K standard in four ways:

- Use data that's compressible and dedupeable like real data rather than the repeating or random data usually used
- A mix of I/O sizes more representative of a real SQL Server than the 8KB commonly used.
- Distribute the I/Os unevenly to create hotspots
- Include sequential writes to the transaction logs as well as the random I/O to the database

Data Reducibility

While we await real data, we've used our best guess about the OLTP database server we're emulating. The conventional wisdom is that relational databases compress relatively well but only dedupe a bit. We quantified this as:

- Dedupe ratio 1.2:1
- Deduplication block size 4KB
- Compression ratio 2:1

We choose a 4KB deduplication block size as this is the most common size for the primary storage systems we've been testing.

I/O Size Distribution

We based our I/O size distribution on data from a Nimble Storage blog post -*Storage Performance Benchmarks Are Useful If You Read Them Carefully*

I/O size	Percentage of total I/Os
512B	3.5%
1KB	2.5%
2KB	3.5%
4KB	16%
8KB	34.5%
16KB	16%
32KB	8%
64KB	5.5%
128KB	7.5%
256KB	3%

The POLTP.CFG I/O Size Distribution

I/O Distribution For The Device Under Test

The database workload configuration uses several VDbench configuration options to control where I/Os are sent:

- In the storage definition the **size=** parameter should be set to the total size of the device under test
- The **wd_SQL_DB1** workload limits all I/Os to the first 80% of the **size** specified above
- Thirty hotspots totaling 15% of the device under test's address space and receiving 83% of all I/Os. The remaining I/Os are distributed across the remaining 85% of the disk.
- I/Os are 90% random
- 65% of all I/Os are reads

The hotspots were created using VDBsetup, a Java application we whipped up to create hotspots for VDBench.

Emulating Transaction Logs

It's always annoyed us that benchmarks trying to emulate an OLTP workload were 100% random when relational database servers don't just write randomly to their database files; they also write sequentially to their transaction logs on another logical disk.

The LOG.CFG workload performs 64KB sequential writes to emulate SQL Server. In our testing the amount of data written to the transaction logs roughly equals the amount of data written to the database. Since the database writes have an average I/O size of 30KB with 35% writes we perform about 15% as many 64KB I/Os to the emulated logs as total I/Os to the emulated database.