Technical White Paper

Dell EMC PowerStore: MongoDB Solution Guide

Abstract

This document provides a solution overview for MongoDB running on a Dell EMC™ PowerStore™ appliance.

May 2021

Revisions

Date	Description
February 2020	Initial release
August 2020	Solution guide with data reduction comparison
May 2021	PowerStoreOS 2.0 updates

Acknowledgments

Author: Henry Wong

Table of contents

Executive summary

As data becomes abundantly available and inexpensive to obtain in this data era, new opportunities emerge. Businesses and organizations make new discoveries and create new business models that are based on these valuable data. New analytic applications like MongoDB® are designed to be flexible and scalable, and with the help of Dell EMC™ PowerStore™, can harness this data growth and unlock the power of data.

This paper offers a high-level overview of the PowerStore appliance and MongoDB. The document provides insights about using various MongoDB compression libraries and the integrated PowerStore advanced data reduction feature. This paper is not a performance-focused study.

Audience

This document is intended for IT administrators, storage architects, partners, and Dell Technologies™ employees. This audience also includes individuals who may evaluate, acquire, manage, operate, or design a Dell EMC networked storage environment using PowerStore systems.

1 Introduction

This document was developed using the PowerStore X model appliance, MongoDB Enterprise Edition, and CentOS 7.x Linux. This section provides an overview for PowerStore and MongoDB, and discusses their combined benefits.

1.1 PowerStore overview

PowerStore achieves new levels of operational simplicity and agility. It uses a container-based microservices architecture, advanced storage technologies, and integrated machine learning to unlock the power of your data. PowerStore is a versatile platform with a performance-centric design that delivers multidimensional scale, always-on data reduction, and support for next-generation media.

PowerStore brings the simplicity of public cloud to on-premises infrastructure, streamlining operations with an integrated machine-learning engine and seamless automation. It also offers predictive analytics to easily monitor, analyze, and troubleshoot the environment. PowerStore is highly adaptable, providing the flexibility to host specialized workloads directly on the appliance and modernize infrastructure without disruption. It also offers investment protection through flexible payment solutions and data-in-place upgrades.

The PowerStore platform is available in two different product models: PowerStore T models and PowerStore X models. PowerStore T models are bare-metal, unified storage arrays which can service block, file, and VMware® vSphere® Virtual Volumes™ (vVols) resources along with numerous data services and efficiencies. PowerStore X model appliances enable running applications directly on the appliance through the AppsON capability. A native VMware ESXi™ layer runs embedded applications alongside the PowerStore operating system, all in the form of virtual machines. This feature adds to the traditional storage functionality of PowerStore X model appliances, and supports serving block and vVol storage to external servers.

For more information about PowerStore T models and PowerStore X models, see the documents Dell EMC PowerStore: Introduction to the Platform and Dell EMC PowerStore Virtualization Guide.

1.2 MongoDB overview

MongoDB is a modern NoSQL database that uses a document-based data model to store both structured and unstructured data. It is highly scalable and can process massive amounts of data efficiently. A MongoDB database can scale up to hundreds of systems with petabytes of data distributed across them. With a modern database architecture comes the need for modern storage and application-driven infrastructure that is engineered to optimize and consolidate existing and new business use cases. The PowerStore storage platform, together with the latest capabilities of MongoDB, introduces AppsON and a new era of onboard application support.

MongoDB is engineered with replica sets to increase data availability and fault tolerance of the MongoDB servers. Full copies of the data are replicated to multiple secondary members. A single replica set supports up to 50 members. Using a larger number of replicas increases data availability and protection. It also provides automatic failover of the primary member during planned or unplanned events such as server updates, server failures, rack failures, data-center failures, or network partitions. Replicating the data to a different server in a different data center further increases data availability and data locality for distributed clients. However, having too many replica members can lead to lower storage efficiencies, higher network-bandwidth usage, and increased management complexities. PowerStore can alleviate these challenges with its integrated data reduction feature, AppsON capability to bring applications closer to storage, and tight integration of the storage platform and VMware virtualization environment.

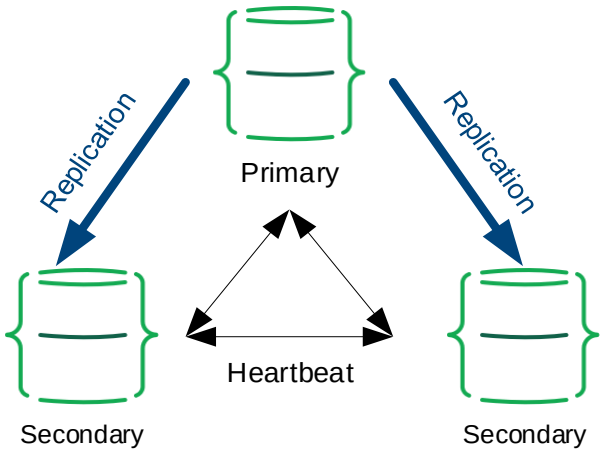

Figure 1 A three-member MongoDB replica set

By default, the primary member is responsible for the write and read operations for the replica set. The clients can specify a read preference to send read operations to the closest secondaries also. You can also configure a data-bearing member (a primary or secondary member but not an arbiter) to be hidden and serve as a backup copy if needed.

As the workload grows, the primary or secondary members must be able to scale their processing capacity by adding CPU, memory, or storage. In a read-oriented workload, more secondaries, each with a full copy of data, might be required. To increase the data durability and to avoid data from being rolled back when a primary member fails over, you can specify a write concern with a value of majority and enable journaling on all voting members. The write concern specifies how many members must acknowledge the write operations before it is considered to be successful. The default write concern for replica sets is 1, which means that it only requires the primary member to return a success acknowledgment after the member applies the write successfully. When the write concern is set to majority, MongoDB calculates the required number of received acknowledgments from the members. WiredTiger journal is a write-ahead transaction log. The journal logs preserve all data modifications between checkpoints. When MongoDB fails between checkpoints, it uses the journal logs to replay the changes since the last checkpoint.

1.2.1 Building a flexible scale-out distributed database architecture

With large datasets and high-throughput environments, MongoDB uses the sharding process to distribute data across multiple systems to increase storage capacity, throughput, and performance. A sharded cluster consists of three components:

Shards hold a subset of the data and are deployed as a replica set.

Mongos process communications with the config servers and route the client requests to the appropriate shards.

Config servers store the metadata for the cluster configuration settings.

The config servers are deployed as a replica set. In a non-sharded database, there is only one primary member in a replica set that is responsible for write operations. However, in a sharded cluster, each shard can perform write operations respective to its dataset.

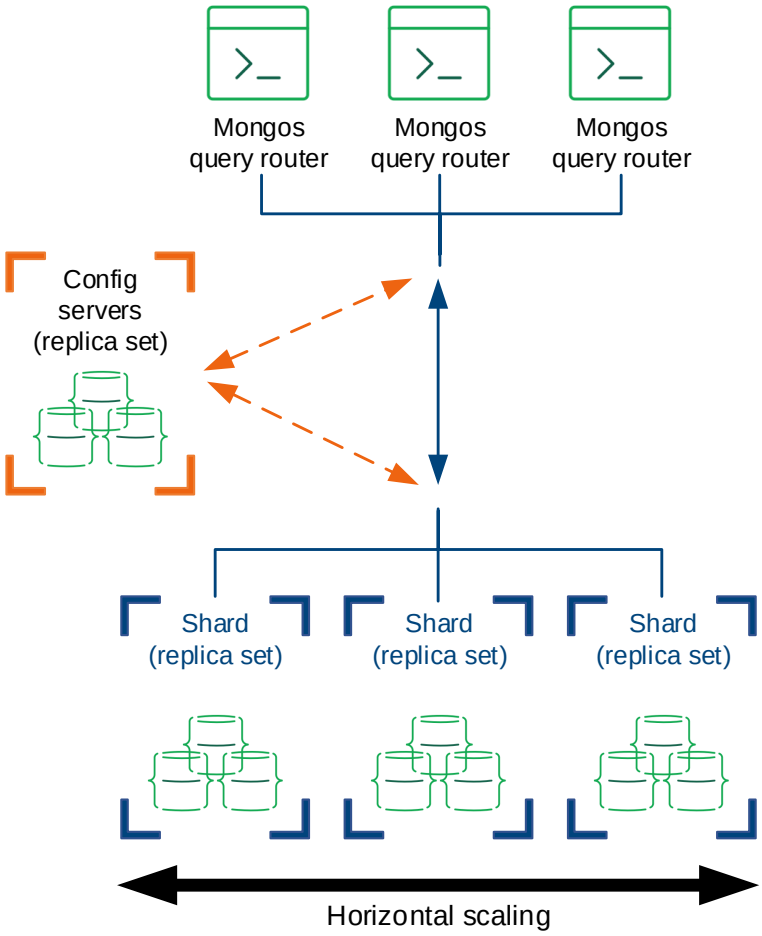

Figure 2 A MongoDB sharded cluster

1.2.2 Modern pluggable storage platform engines

MongoDB supports a wide variety of traditional and business-critical workloads including both operational and real-time analytics workloads. The MongoDB pluggable storage architecture extends new capabilities to the storage platform depending on the different workloads. These storage engines are responsible for storing the data and specify how the data is stored. Starting with version 4.2, MongoDB supports various storage engines including the WiredTiger storage engine, the in-memory engine, and the encrypted storage engine. The MMAPv1 storage engine was deprecated in version 4.2.

The **WiredTiger storage engine** is the default and preferred storage engine for most workloads. It persists data on disk and provides features such as a document-level concurrent model, journaling, checkpoints, and compression.

The **in-memory storage engine** stores the dataset in the memory to reduce data-access latency but does not persist data on disk. It is available only in the MongoDB Enterprise Edition.

The **encrypted storage engine** is the native encryption option for the WiredTiger storage engine. It provides encryption at rest and is only available in the MongoDB Enterprise Edition.

It is possible to mix the different engines based on the use case in the same replica set. This capability allows you to optimize and meet the needs of specific application requirements in a way that benefits the specific

engines. For example, you can combine the in-memory engine for ultralow latency operations with the WiredTiger engine for on-disk persistence.

1.3 The advantages of MongoDB on PowerStore

MongoDB is a modern distributed database that requires a powerful, highly scalable, and flexible infrastructure. PowerStore is performance-optimized for any workload, and its adaptable platform complements modern distributed databases such as MongoDB. This section highlights the PowerStore features that benefit and extend the MongoDB environment.

1.3.1 AppsON brings MongoDB closer to the infrastructure and storage

Bringing applications closer to the data increases density and simplifies infrastructure operations. The PowerStore AppsON capability integrates with VMware vSphere®, resulting in streamlined management in which storage resources plug directly into the virtualization layer. Using VMware as the onboard application environment results in unmatched simplicity, since support is inherently available for any standard VM-based applications. When a new PowerStore X model is deployed, the VASA provider is automatically registered, and the datastore is created, eliminating manual steps and saving time. PowerStore seamlessly integrates the VMware ESXi software into the same hardware. Two ESXi nodes are embedded inside the appliance which has direct access to the same storage resources. This close integration allows applications like MongoDB to take full advantage of server and storage virtualization with simplified deployment and management. AppsON is available on the PowerStore X model exclusively.

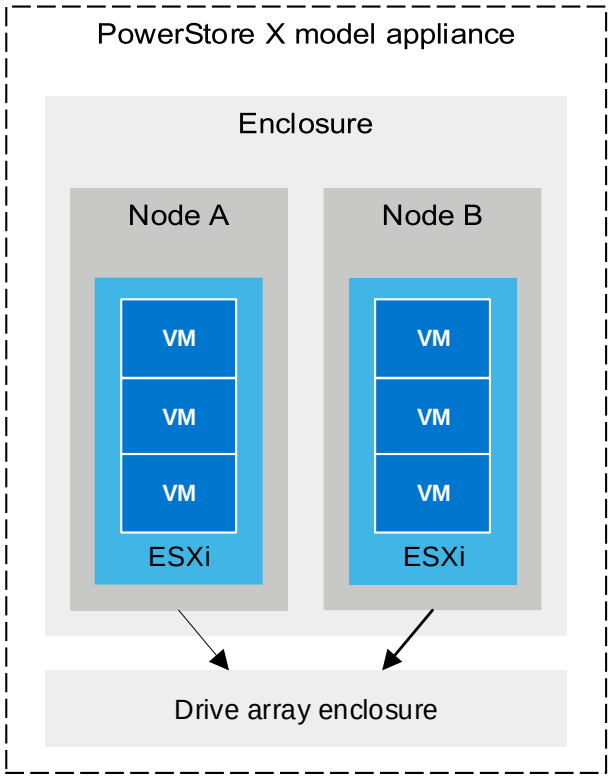

Figure 3 AppsON with embedded ESXi in the PowerStore X model appliance

1.3.2 Agile infrastructure, flexible scaling on a high-performing storage and compute platform

PowerStore provides flexible scaling with ease of management that compliments the MongoDB scale-up and scale-out distribution model. The integrated hypervisor dynamically scales up the replica set members when the workload requires it, while new replica sets, or shards, can be provisioned rapidly on the same or on additional appliances in a different location.

When the application grows and requires more storage from a PowerStore appliance, administrators can scale up the storage capacity by adding disk expansion enclosures without service interruption at any time. Multiple PowerStore appliances can also be configured into a cluster to increase CPUs, memory, storage capacity, and front-end connectivity. Clustering simplifies and centralized the management of multiple appliances from a single HTML5-based management interface. A cluster can comprise up to four PowerStore T appliances or four PowerStore X appliances. The support of clustering multiple PowerStore X appliances is introduced in PowerStoreOS 2.0. Each appliance within the cluster can have different configurations of CPUs, memory, NVMe drives, and expansion enclosures. For more information about PowerStore cluster, see the document Dell EMC PowerStore: Clustering and High Availability

A single PowerStore appliance can scale up to 112 vCPUs, 2.5 TB of memory, and 3.59 PB raw storage capacity. The NVMe architecture is designed for the next-generation NVMe-based storage and takes advantage of low-overhead NVRAM cache. PowerStore is engineered to handle the most demanding MongoDB mixed workloads.

1.3.3 Mission-critical high availability and fault-tolerant MongoDB platform

At the hardware level, PowerStore is designed to be highly available and fault tolerant. It monitors the storage devices continuously and automatically relocates data from failing devices to avoid data loss. The PowerStore X model appliance includes two ESXi nodes and redundant hardware components. The non-disruptive upgrade (NDU) feature further increases overall PowerStore availability. The updates are performed on the nodes in a rolling fashion. NDU supports PowerStore software releases, hotfixes, and hardware and disk firmware.

To support high-value business workloads and service requirements on the application level, it is essential to protect and ensure the availability of the primary member of a replica set. When the primary member of a replica set becomes inaccessible, the replica set cannot process any write operations until the primary member recovers or a new primary is elected. Furthermore, the election requires most of the members to be available.

With standard VMware vSphere High Availability (HA) integrated into PowerStore, the embedded VMware ESXi hypervisor automatically restarts or migrates failed MongoDB servers to a different ESXi node to resume operations. This helps to restore MongoDB to its full operation capacity and minimizes the chance of the database going offline or read-only.

To achieve an even higher level of redundancy and application availability, you can deploy the MongoDB replica set and sharded cluster across multiple PowerStore appliances in different data centers. PowerStore improves MongoDB availability and provides unparalleled flexibility and mobility to relocate and move across data centers and appliances.

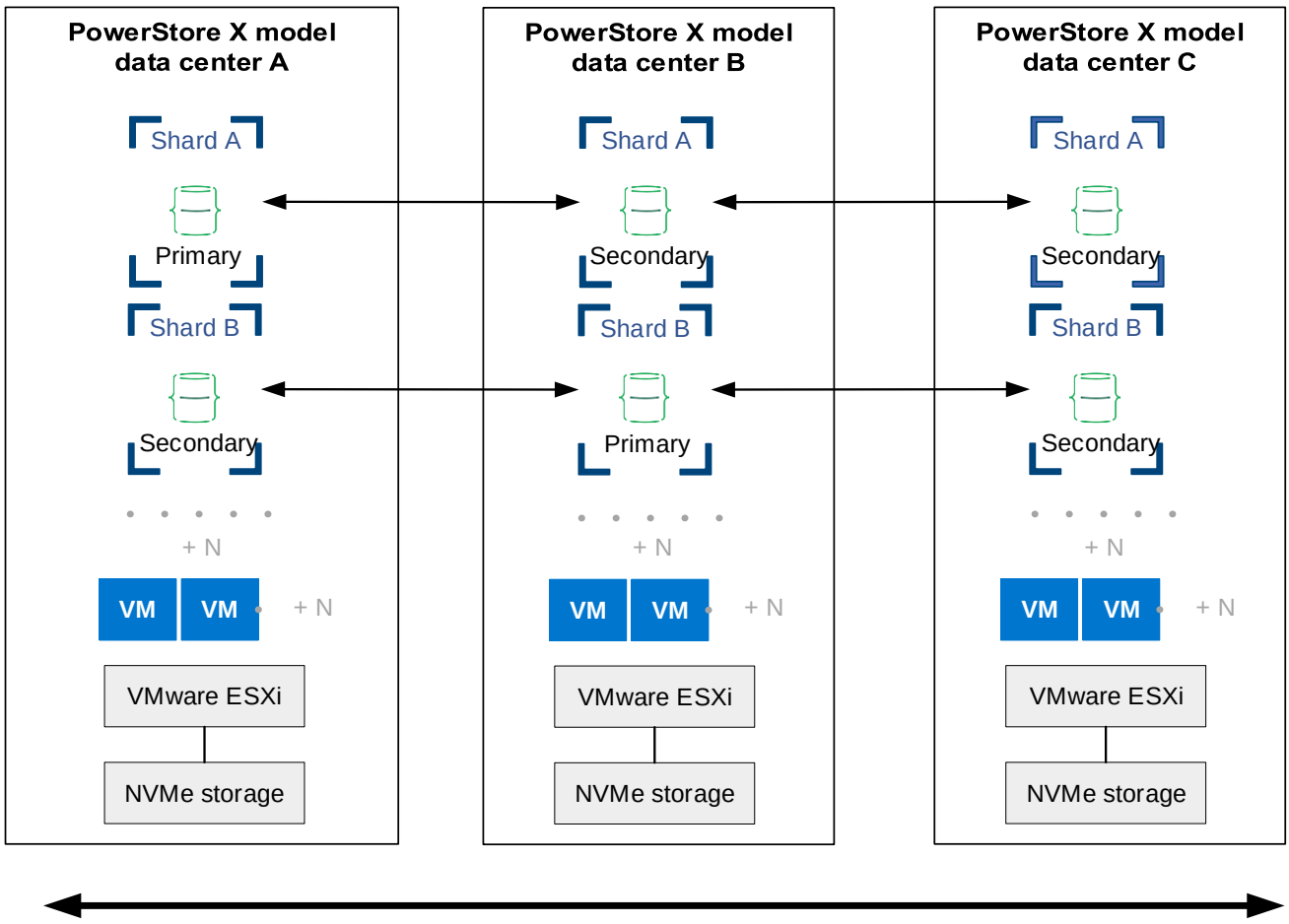

Figure 4 Geographically distributed MongoDB sharded cluster on multiple PowerStore X model appliances

1.3.4 PowerStore inline data reduction reduces storage consumption and cost

As business data continues to grow, big data has become a critical component in the business analytics world. A tremendous amount of data is pulled from all kinds of sources continuously and run through cloud-scale applications like MongoDB to gain insights into customers and businesses. When putting MongoDB replica sets on PowerStore, the always-on inline data reduction feature greatly reduces the actual storage used but still maintains the application data availability and protection that is expected from MongoDB.

1.3.5 Efficient and convenient snapshot data backup

PowerStore provides MongoDB with additional data protection through array-based snapshots. A PowerStore snapshot is a point-in-time copy of the data. The snapshots are space efficient and require seconds to create. Snapshot data are exact copies of the target data and can be used for application testing, backup, or DevOps. Because of the tight integration with VMware vSphere, PowerStore can take vVol-based VM snapshots directly from PowerStore Manager using a protection policy schedule or on demand. When taking the VM snapshots from vSphere, it passes the request to PowerStore to create the vVol-based VM snapshots which have no performance impact on the VMs. You can view the VM snapshot information in PowerStore and vCenter.

1.3.6 Secure data protection with ease of mind

With high-value data driving business applications, data security is a top concern for all organizations. Lost or stolen data can seriously damage the reputation of an organization and result in huge financial costs and loss of customer trust. Dell Technologies™ engineered PowerStore with Data at Rest Encryption (D@RE) which uses self-encrypting drives and supports array-based, self-managed keys. Once activated, data is encrypted as it is written to disk using the 256-bit Advanced Encryption Standard (AES). PowerStore D@RE provides this data security benefit to MongoDB while eliminating application overhead, performance penalties, and administrative overhead that is typically associated with software-based solutions.

1.3.7 Unified infrastructure and services management

PowerStore provides deep integration with VMware management tools and services with Dell EMC Virtual Storage Integrator (VSI), VMware vRealize® Operations Manager (vROps), VMware vRealize Orchestrator (vRO), and VMware Storage Replication Adapter (SRA). You can easily incorporate ESXi on PowerStore X models into your existing vCenter and manage all VMware infrastructure and services from a unified management platform.

1.3.8 MongoDB value and future expansion

New business analytics applications like MongoDB are fundamentally changing the way data is used to support the business. Massive amounts of data and technology innovation together provide the opportunity for organizations to transform. As the value and scale of this data grows, it is critical to have a future-proof platform that is easy to manage, provides technical innovation for future growth, and can support the application architecture. MongoDB on PowerStore brings IT organizations the ability to be agile, efficient, and responsive to business demands.

1.4 Terminology

The following terms are used with PowerStore.

Appliance: Solution containing a base enclosure and attached expansion enclosures. The size of an appliance could be only the base enclosure or the base enclosure plus expansion enclosures.

PowerStore node: Storage controller that provides the processing resources for performing storage operations and servicing I/O between storage and hosts. Each PowerStore appliance contains two nodes.

Base enclosure: Enclosure containing both nodes (node A and node B) and 25 NVMe drive slots

Expansion enclosure: Enclosures that can be attached to a base enclosure to provide additional storage.

Fibre Channel (FC) protocol: Protocol used to perform SCSI commands over a Fibre Channel network.

iSCSI: Provides a mechanism for accessing block-level data storage over network connections.

NDU: A non-disruptive upgrade (NDU) updates PowerStore and maximizes its availability by performing rolling updates. This includes updates for PowerStore software releases, hotfixes, and hardware and disk firmware.

NVMe: Non-Volatile Memory Express is a communication interface and driver for accessing non-volatile storage media such as solid-state drives (SSD) and SCM drives through the PCIe bus.

NVRAM: Non-volatile random-access memory is persistent random-access memory that retains data without an electrical charge. NVRAM drives are used in a PowerStore appliance as additional system write caching.

Volume: A block-level storage device that can be shared out using a protocol such as iSCSI or Fibre Channel.

Snapshot: A point-in-time view of data that is stored on a storage resource. You can recover files from a snapshot, restore a storage resource from a snapshot, or provide access to a host.

Storage container: A VMware term for a logical entity that consists of one or more capability profiles and their storage limits. This entity is known as a vVol datastore when it is mounted in vSphere.

PCIe: Peripheral Component Interconnect Express is a high-speed serial computer expansion bus standard.

PowerStore Manager: An HTML5 management interface for creating storage resources and configuring and scheduling protection of stored data on PowerStore. PowerStore Manager can be used for all management of PowerStore native replication.

PowerStore T model: Container-based storage system that is running on purpose-built hardware. This storage system supports unified (block and file) workloads, or block-optimized workloads.

PowerStore X model: Container-based storage system that runs inside a virtual machine that is deployed on a VMware hypervisor. Besides offering block-optimized workloads, PowerStore also allows you to deploy applications directly on the array.

RecoverPoint for Virtual Machines: Protects VMs in a VMware environment with VM-level granularity and provides local or remote replication for any point-in-time recovery. This feature is integrated with VMware vCenter and has integrated orchestration and automation capabilities.

SCM: Storage-class memory, also known as persistent memory, is an extremely fast storage technology supported by the PowerStore appliance.

Snapshot: A point-in-time view of data stored on a storage resource. You can recover files from a snapshot, restore a storage resource from a snapshot, or provide access to a host.

Storage Policy Based Management (SPBM): Using policies to control storage-related capabilities for a VM and ensure compliance throughout its life cycle.

Thin clone: A read/write copy of a thin block storage resource (volume, volume group, or vSphere VMFS datastore) that shares blocks with the parent resource.

User snapshot: Snapshot that is created manually by the user or by a protection policy with an associated snapshot rule. This snapshot type is different than an internal snapshot, which is taken automatically by the system with asynchronous replication.

Virtual machine (VM): An operating system running on a hypervisor, which is used to emulate physical hardware.

vCenter: VMware vCenter server provides a centralized management platform for VMware vSphere environments.

VMware vSphere Virtual Volumes (vVols): A VMware storage framework which allows VM data to be stored on individual vVols. This ability allows for data services to be applied at a VM-level of granularity and

according to SPBM. vVols can also refer to the individual storage objects that are used to enable this functionality.

vSphere API for Array Integration (VAAI): A VMware API that improves ESXi host utilization by offloading storage-related tasks to the storage system.

vSphere API for Storage Awareness (VASA): A VMware vendor-neutral API that enables vSphere to determine the capabilities of a storage system. This feature requires a VASA provider on the storage system for communication.

2 Sizing considerations

Before you select the PowerStore model, storage media and capacity, and connectivity options, you must first understand the target MongoDB environment. There are many factors to consider that are not limited to the following:

- Consider the amount of data to keep and future data growth. After you perform a careful analysis against the dataset, you may factor the data-reduction savings into the calculated storage capacity requirements. Be aware that the actual savings might vary because the type of data stored can dramatically affect the effectiveness of the data-reduction ratio. For example, the binary datatype is less compressible than the text datatype.
- Account for the number of database servers in a replica set. MongoDB uses replica sets to provide data redundancy and availability. Having multiple copies of data also increases the read capacity to clients in some cases.
- Plan to have extra copies of data for reporting, backup, or disaster recovery.
- Understand the performance that is required by the database servers in terms of IOPS, latencies, and bandwidth.
- Know the expected workload types, such as a read-mostly workload, update-intensive workload, or a combination of several workload types.

Also, plan for the following resources for the database servers on PowerStore X appliances.

- Plan for the number of MongoDB database servers that will be hosted on a single appliance and on each ESXi host. When multiple PowerStore appliances are configured in a PowerStore cluster, all appliances should be in the same data center. Spread the database serves as evenly as possible on the appliances.
- Know the CPU and memory requirements of the database servers.
- Prepare for the event in which one of the ESXi nodes fails, and decide whether CPU and memory resources should be reserved on an ESXi node to accommodate full performance for all database hosts.
- Do not overcommit CPU and memory resources on PowerStore ESXi nodes in any production or mission-critical environments. However, this practice might be acceptable in test or development environments where the guaranteed performance level is not a concern.

Review the MongoDB documentation to learn about other software and hardware requirements.

The Dell Technologies account team has access to a suite of tools, such as LiveOptics and CloudIQ, that are designed to help gather and analyze workload and performance data in an existing environment. The account team can then use the PowerStore sizer to estimate the storage need.

3 Data reduction comparison

The PowerStore inline data reduction feature provides maximum space savings by combining both software data deduplication and hardware compression. MongoDB also includes several compression libraries to help reduce the amount of space that is consumed by the operating system.

This section reviews the compression options that MongoDB offers and explains the effects of combining PowerStore data reduction with MongoDB compression. A test environment was set up on a PowerStore X model to run a series of tests with the different MongoDB libraries. Both logical and physical space consumptions were recorded and compared.

3.1 Test environment topology

The test environment consists of a MongoDB replica set that consists of three database-host virtual machines. The VMs are installed on the PowerStore X appliance. Another VM is configured to host the YCSB benchmark tool. The YCSB VM resides on a different VMware cluster that uses a different storage appliance.

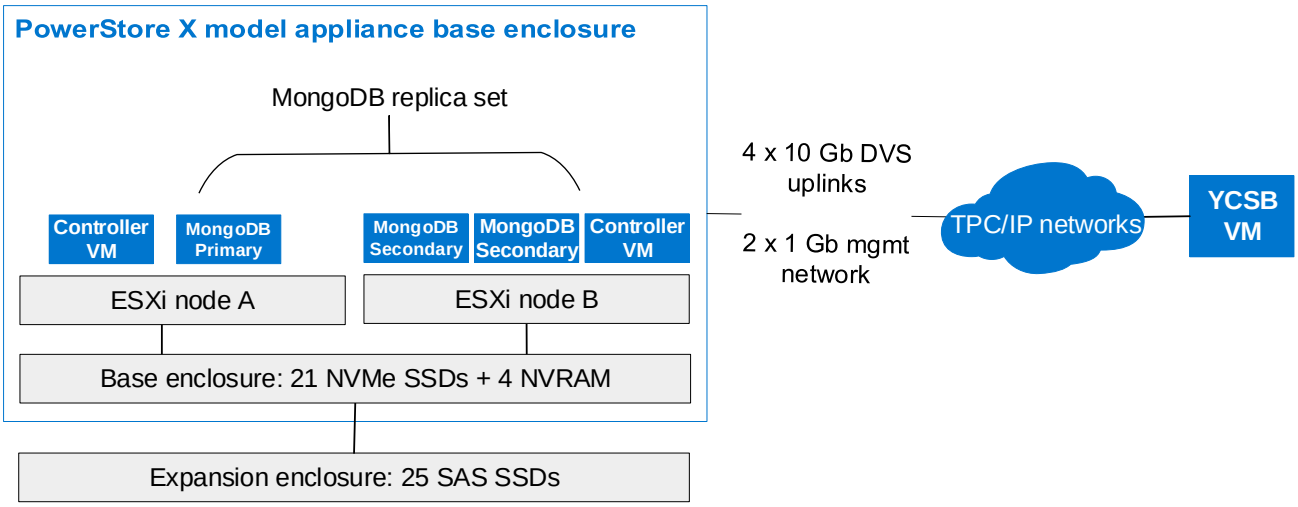

Figure 5 Test environment topology

3.2 PowerStore X appliance

Table 1 summarizes the PowerStore X appliance hardware specification that is configured in the test.

Dell Technologies offers various media options and hardware specifications to choose from. To learn more about the full PowerStore family, go to the PowerStore product page.

Table 1 PowerStore X specification used in the test

Specification	Detail
Model	PowerStore X model
NVMe SSD configured	21 x 3.5 TB drives in the base enclosure
SAS SSD configured	25 x 1.7 TB drives in an expansion enclosure
System cache (memory)	1.5 TB

Specification	Detail
Mezzanine cards	2
I/O modules	4
Embedded SAS I/O ports	4 x 4 lane 12 Gb/s SAS ports for back-end connection
32 Gb FC ports	8 total; 4 active
10 Gb iSCSI ports	8 total; 4 active

3.2.1 MongoDB database hosts

For our tests, the database host VMs that run CentOS Linux® are installed directly on the PowerStore X model appliance. The unique AppsON feature enables you to consolidate infrastructure and bring the applications closer to the storage resources. Table 2 summarizes the database host VM specification.

Table 2 Database host VM specification

Specs	Value
vCPUs	16
Memory	128 GB
Virtual NICs	1 x 10 Gb
Virtual SCSI controllers	2
Virtual disks	• 1 x 50 GB for the operating system drive • 1 x 1 TB for the database
Guest operating system	CentOS Linux release 7.8.2003 (Core)
Kernel version	Linux 3.10.0-1127.13.1.el7.x86_64

3.2.2 PowerStore storage containers and virtual volumes

Three MongoDB database host VMs are provisioned with vVols on the default storage container. On PowerStore X models, the VASA provider is automatically registered with vSphere, and the default storage container is mounted automatically on the internal ESXi nodes through the iSCSI protocol. PowerStore can also present the storage containers to external ESXi hosts using FC, iSCSI, or both. However, you must manually register the VASA provider, and you must mount the storage containers manually on the external ESXi hosts. For our tests, all VMs are internal. No external hosts are configured.

PowerStore automatically tracks the vVols that belong to each VM. The PowerStore Manager UI shows these vVols objects under the Virtual Machines view. See Figure 6, Figure 7, and Figure 8.

For more information about vVols, storage containers, and vSphere VASA, see the Dell EMC PowerStore Virtualization Guide.

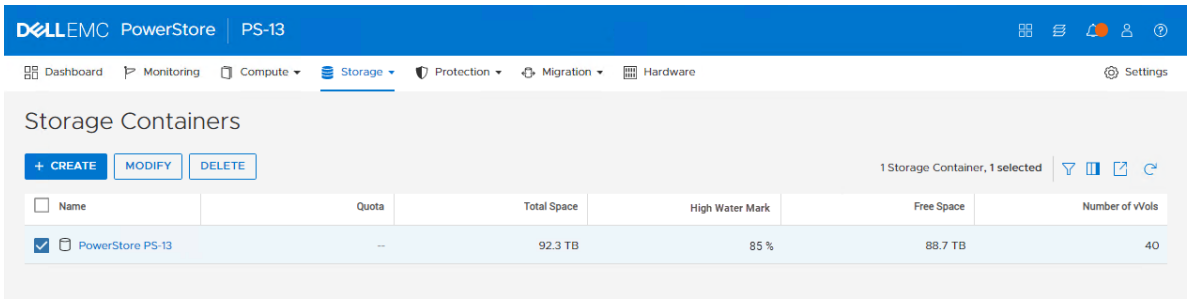

Figure 6 Default storage container on the PowerStore X model appliance

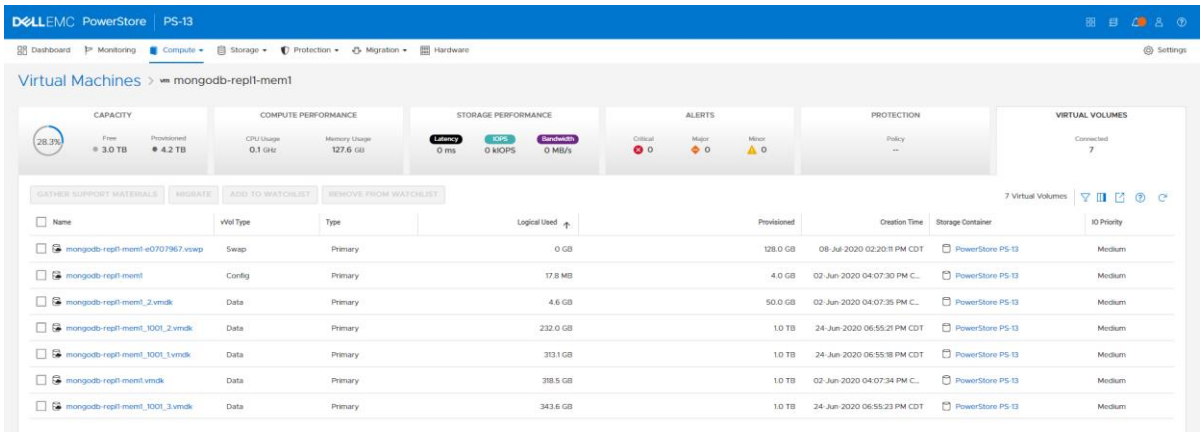

Figure 7 Listing vVols objects that are associated with a VM

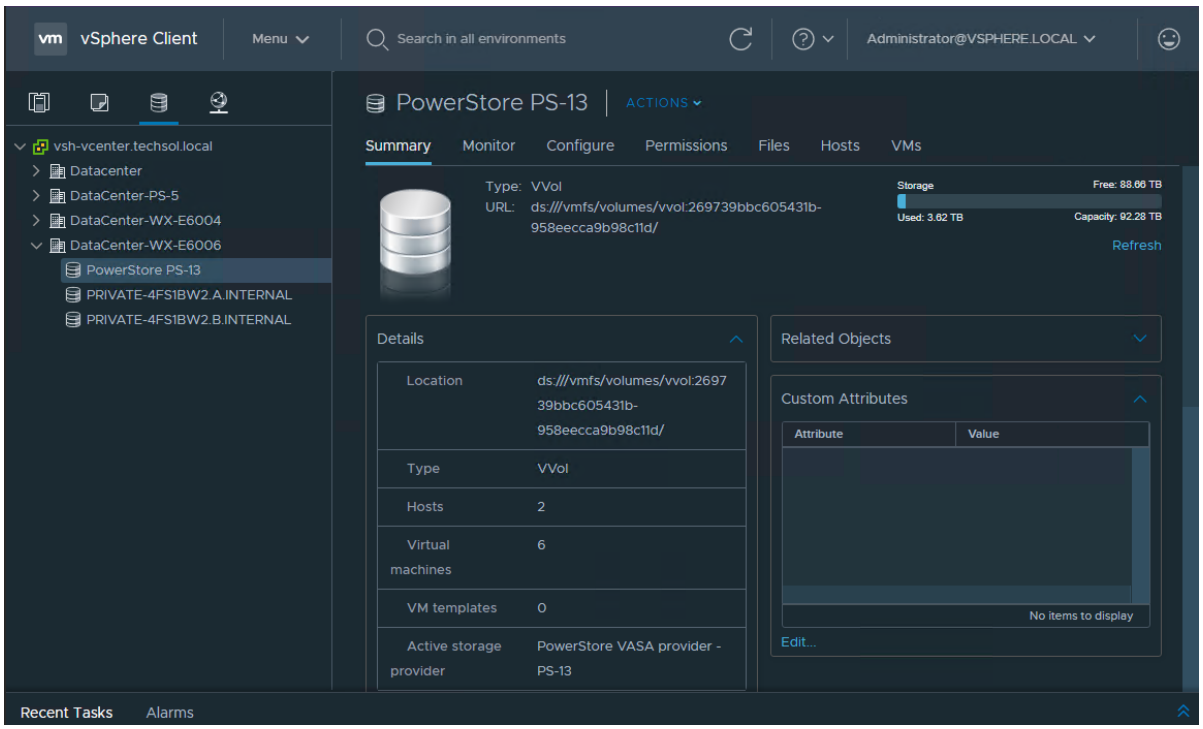

Figure 8 vVol-based storage container automatically mounted in vSphere on the PowerStore X model

3.2.3 PowerStore X virtualization and performance best practices

The PowerStore X appliance is tuned according to the best practices in the following documents. These changes include increasing the ESXi queue depth and enabling Jumbo Frames for the iSCSI storage network. For details about the performance best practices, see the following documents.

- PowerStore: PowerStore X Performance Best Practice Tuning
- Dell EMC PowerStore Virtualization Guide

3.3 YCSB host

Another centOS VM is configured to host the YCSB software. This VM resides on a vSphere infrastructure cluster that is separate from the PowerStore X model appliance. Table 3 shows the YCSB host VM specification.

Table 3 YCSB host VM specification

Specification	Value
vCPUs	40
Memory	128 GB
Virtual NICs	1 x 10 Gb
Virtual SCSI controllers	2
Virtual disks	1 x 50 GB for the operating-system drive
Guest operating system	CentOS Linux release 7.7.1908 (Core)
Kernel version	3.10.0-1062.12.1.el7.x86_64

3.4 Networking

The PowerStore X model creates a vSphere distributed switch (vDS) and a set of preconfigured distributed port groups for internal communications during the initial configuration process. On the PowerStore X model in the test, each node has two 10 Gb connections for vDS uplinks and one 1 Gb connection for the management network. A new distributed port group is created on the vDS and assigned a different VLAN to allow guest VMs network communication. All MongoDB database host VMs are assigned to the same distributed port group in the test. The PowerStore X appliance is connected to a Dell EMC Networking 5148F-ON 48-port 25 GbE switch.

3.5 MongoDB installation and configuration

This section describes the deployment and configuration of the MongoDB server software.

3.5.1 Operating system tuning

MongoDB recommends adjusting the following configuration on the Linux operating system.

1. Modify the operating system user limits for the mongod user. Create the /etc/security/limits.d/99-mongod.conf file with the following lines.

```
mongod  soft    nofile  64000
mongod  hard    nofile  64000
mongod  soft    nproc   64000
mongod  hard    nproc   64000
mongod  soft    memlock unlimited
mongod  hard    memlock unlimited

# chmod mongod:mongod /etc/security/limits
```

2. To disable transparent hugepages, append the following to the GRUB_CMDLINE_LINUX line in the /etc/default/grub file.

```
transparent_hugepage=never
```

3. Reboot the system.
4. To disable NUMA, use one of the following methods.

- Disable NUMA on the BIOS on a physical server. This method does not apply to the PowerStore X model internal ESXi nodes.
- Disable vNUMA on a VM.
- If hot-add CPU is enabled on a VM, it automatically disables the vNUMA feature for that VM.
- If NUMA is not disabled, start the **mongod** process with the following **numactl** command that would achieve similar effect of turning off numa.

```
# numactl --interleave=all mongod -f /etc/mongod.conf
```

5. To review and adjust Linux kernel parameters, run sysctl -p to list all current values. To override any parameter, define the new value in /etc/sysctl.conf or /etc/sysctl.d/${parameter file}. Reload the sysctl file or reboot the system. See Table 4 for kernel parameter values.

Table 4 Kernel parameter values

Parameters	Values used in test
fs.file-max	13071518
kernel.pid_max	131072
kernel.threads-max	1029862
vm.max_map_count	64000
net.ipv4.tcp_keepalive_time	7200*
vm.zone_reclaim_mode	0

* mongod and mongos sockets override keepalive values and set to 300 if the value is larger than 300.

3.5.2 File system

A separate file system is created for MongoDB database files to isolate the database from the operating system. MongoDB recommends using the XFS file system because it generally performs better than ext4 file system. Also, consider using the **noatime** mount option with the file system. It might improve overall performance in some situations.

3.5.3 MongoDB installation

The installation steps are well documented on the MongoDB portal. There are several ways to install the MongoDB on a self-managed platform. The following summarizes the installation method chosen for the test environment. MongoDB Enterprise server edition version 4.2.8 is used in our test.

1. Configure the MongoDB YUM repository.

```
[mongodb-enterprise-4.2]
name=MongoDB Enterprise Repository
baseurl=https://repo.mongodb.com/yum/redhat/$releasever/mongodb-
enterprise/4.2/$basearch/
gpgcheck=1
enabled=1
gpgkey=https://www.mongodb.org/static/pgp/server-4.2.asc
```

2. Install the MongoDB packages.

```
# yum install mongodb-enterprise-shell mongodb-enterprise-tools mongodb-
enterprise-server mongodb-enterprise-cryptd mongodb-enterprise mongodb-
enterprise-mongos
```

3. Exclude MongoDB packages in **/etc/yum.conf – optional.**

   Besides disabling the MongoDB YUM repository, you can explicitly exclude the packages from future yum updates by adding the following line in the **/etc/yum.conf** file.

```
exclude=mongodb-enterprise,mongodb-enterprise-server,mongodb-enterprise-
shell,mongodb-enterprise-mongos,mongodb-enterprise-tools,mongodb-
enterprise-cryptd
```

3.5.4 MongoDB configuration

The following mongodb configuration template is used for the test. The appropriate compression library value is set for each compression test. See section 4.2 for the compression library options.

```
storage:
  dbPath: /${fs_mountpoint}/mongodb
  journal:
    enabled: true
  directoryPerDB: true
  wiredTiger:
    engineConfig:
        journalCompressor: ${compression_library}
        directoryForIndexes: true
    collectionConfig:
        blockCompressor: ${compression_library}
    indexConfig:
        prefixCompression: false
systemLog:
  destination: file
  logAppend: true
```

```
  path: /${fs_mountpoint}/mongodb/mongod.log
net:
  port: 27017
  bindIp: "${server_hostname},127.0.0.1"
processManagement:
  timeZoneInfo: /usr/share/zoneinfo
  fork: true
replication:
  replSetName: replica_set10
```

3.6 YCSB installation

YCSB is an open-source benchmark tool used for testing the performance of various databases, including MongoDB, Cassandra, HBase, and many more. The tool is used to generate the test data in the MongoDB database and to run various workloads against the database. You can download the software from https://github.com/brianfrankcooper/YCSB.

Also, Apache Maven and Java are required for YCSB. You can download Apache Maven software from https://maven.apache.org/. You can install Java jdk directly from the public YUM repository.

The following steps outline the process to install the YCSB software on the YCSB host VM. Perform the following steps as root user on the YCSB VM.

1. Install Java JDK.

   ```
   # yum install java-1.8.0-openjdk*
   ```

2. Install Apache Maven software which is required by YCSB.

   ```
   # cd /usr/local
   # curl -O --location
   http://mirrors.advancedhosters.com/apache/maven/maven-
   3/3.6.3/binaries/apache-maven-3.6.3-bin.tar.gz
   # tar xfz apache-maven-3.6.3-bin.tar.gz
   # ln -s /usr/local/apache-maven-3.6.3 /usr/local/maven
   ```

3. Create a Maven profile with the following lines in **/etc/profile.d/maven.sh**.

   ```
   export M2_HOME=/usr/local/maven
   export PATH=${M2_HOME}/bin:${PATH}
   ```

4. Log off and log in to the system.
5. Verify that Maven is installed properly.

   ```
   # mvn -v
   Apache Maven 3.6.3 (cecedd343002696d0abb50b32b541b8a6ba2883f)
   Maven home: /usr/local/maven
   Java version: 1.8.0_242, vendor: Oracle Corporation, runtime:
   /usr/lib/jvm/java-1.8.0-openjdk-1.8.0.242.b08-0.el7_7.x86_64/jre
   Default locale: en_US, platform encoding: UTF-8
   OS name: "linux", version: "3.10.0-1062.12.1.el7.x86_64", arch: "amd64",
   family: "unix"
   ```

6. Install the YCSB software.

```
# cd /usr/local
# curl -O --location
https://github.com/brianfrankcooper/YCSB/releases/download/0.17.0/ycsb-0.17.0.tar.gz
# tar xfz ycsb-0.17.0.tar.gz
# ln -s /usr/local/ycsb-0.17.0 /usr/local/ycsb
```

For more information about YCSB, read the wiki articles on https://github.com/brianfrankcooper/YCSB/wiki.

Note: The mongodb binding files are already in this archive. There is no need to install the binding files separately.

4 Test methodology and results

This section describes the test methodology that is used in our test, and the final results are shown in a series of charts.

Several tests are performed to invoke YCSB processes to insert a fixed number of records into the database. Each test cycle consists of the following steps:

- Reboot the database VMs to clear out memory cache.
- Delete the data previous loaded to clear out the operating system and PowerStore data reduction saving information from the previous test.
- Start YCSB processes to insert 200,000,000 records into the replica set.
- After each test, the following information is collected:
  - Guest operating system statistics: CPU and memory utilization
  - Guest operating system disk space: File system usage
  - PowerStore data reduction statistics: Physical and logical used
  - YCSB results: Operations per second, average latencies, and thread counts

After completing all the tests using different compression libraries, the results collected are analyzed and compared based on the following metrics:

- Runtime of the data load reported by YCSB
- Total disk consumption reported by MongoDB and the operating system
- Data reduction saving reported by PowerStore X appliance
- Guest OS CPU and memory utilization reported by atop

4.1 Loading data with YCSB

YCSB provides six predefined core workloads. You can also add custom workloads if the predefined workloads do not meet the testing criteria. All six workloads have a similar dataset. For our compression tests, the **workloada** dataset is used to populate the data in the database.

The following command shows an example of a YCSB process that ingests the data.

```
./bin/ycsb load mongodb-async -s -P workloads/workloada -p
mongodb.url=mongodb://localhost:27017/ycsb?w=0
```

For more information about YCSB and the core workloads, see the GitHub article Core Workloads.

4.2 WiredTiger storage engine and compression

The WiredTiger storage engine is chosen for the tests because it is the default storage engine since MongoDB version 3.2. The MMAPv1 storage engine is obsolete and is no longer available in MongoDB. Among other features such as journaling, document-level concurrent model, checkpointing, WiredTiger also provides several compression libraries to use with document collections, journals, and prefix compression for indices. The benefit of using MongoDB compression is to save both memory and disk consumption on the database hosts.

MongoDB provides four different compressor settings:

- none – turn off compression
- snappy – this is the default compression library
- zlib
- zstd – new in version 4.2

To change the compression library, edit and set the storage.wiredTiger options in /etc/mongod.conf. If the options are not explicitly set, the snappy library is used implicitly.

```
storage:
   wiredTiger:
      engineConfig:
         journalCompressor: <snappy(default)|zlib|zstd|none>
      collectionConfig:
         blockCompressor: <snappy(default)|zlib|zstd|none>
      indexConfig:
         prefixCompression: <true(default)|false>
```

4.3 Data ingestion

Figure 9 summarizes the results for insert operations per second when inserting the data into the database. The numbers indicate that the compression libraries may affect how fast YCSB inserts data into the database. For best performance, turn off compression completely in MongoDB. The zstd or zlib, which are considered to be better compression algorithms than the snappy library, have lower values for insert operations per second.

Note: The insert ops/sec numbers provided are intended to show the impact of using different compression libraries in the context of data reduction. This allows us to compare the relative effects to each library. They are not meant to be the indicator of the absolute best performance numbers.

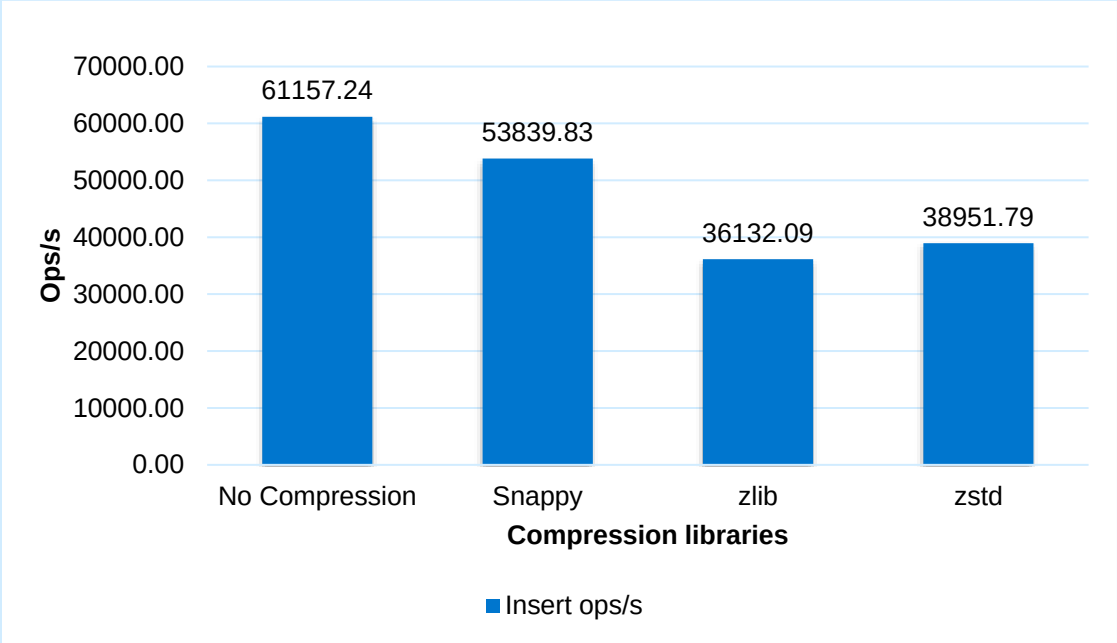

Figure 9 Insert performance comparison on different compression libraries

4.4 Data reduction savings

Figure 10 shows the space savings on the operating system and the extra savings on the PowerStore appliance.

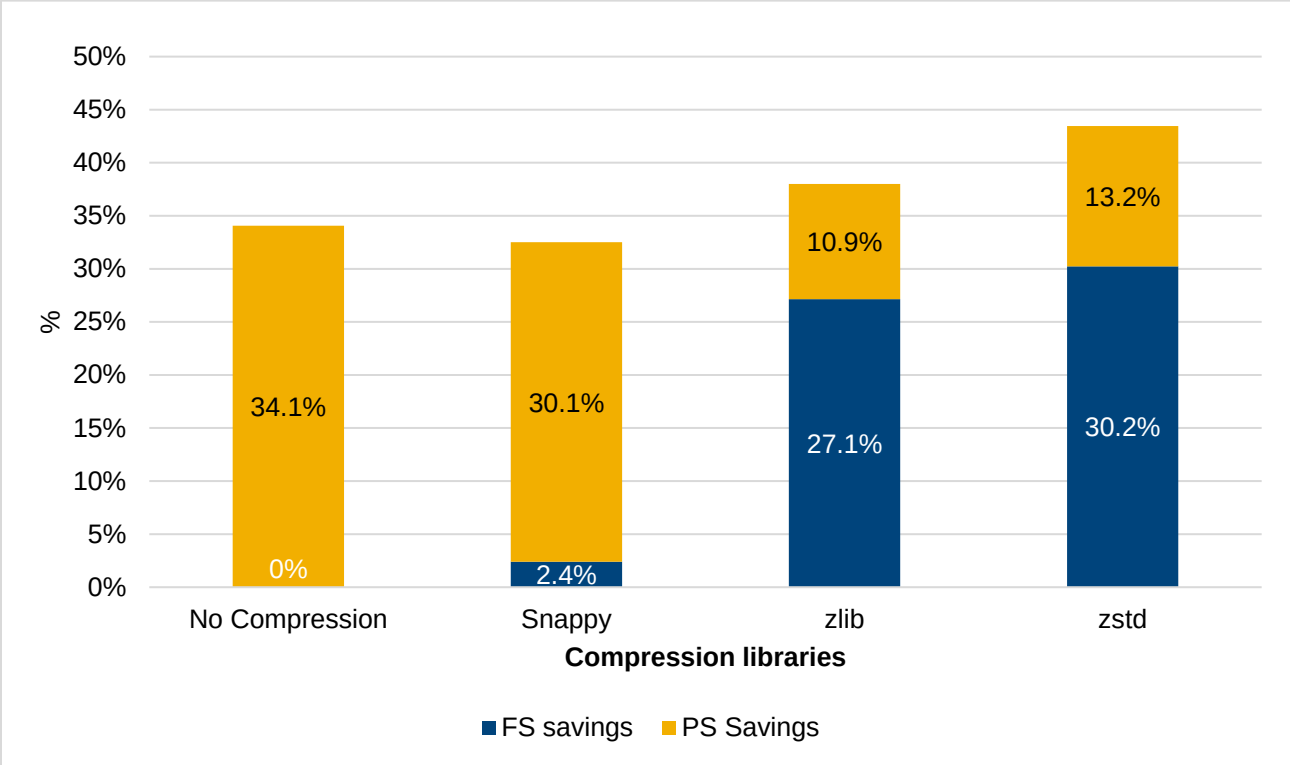

Figure 10 MongoDB compression and PowerStore data reduction savings different compression libraries

Blue column: Represents the space reduction on the operating system level

Yellow column: Represents the additional reduction saving on the PowerStore appliance.

For example, without MongoDB compression enabled, the total file system space consumed by all three replica set members would be around 1 TB including the operating system. It is represented as 0% savings on the operating system. However, the actual physical space consumed on the PowerStore appliance is further reduced by 34%.

Without MongoDB compression, PowerStore alone reduced the total physical space consumption by 34.1%.

With the snappy compression, which is the default in MongoDB, only 2% reduction is realized on the file system. However, PowerStore provided an additional 30.1% reduction savings.

Both zlib and zstd compression do a much better job on space savings on the file system, 27.1% and 30.2%, respectively. It is worth to note that even with the more efficient compression libraries like zstd and zlib, PowerStore can further reduce the total physical space consumption by 10.9% and 13.2% respectively.

4.5 Data ingestion time

Using YCSB, 200 million documents are inserted into the database using different compression libraries. Figure 11 compares the total load time of these compression libraries.

With zlib and zstd compressions, the times spent to write the dataset to the database are longer than using the default snappy compression or no compression. The zstd library appears to perform better than the zlib library as it achieved better reduction saving and shorter load time.

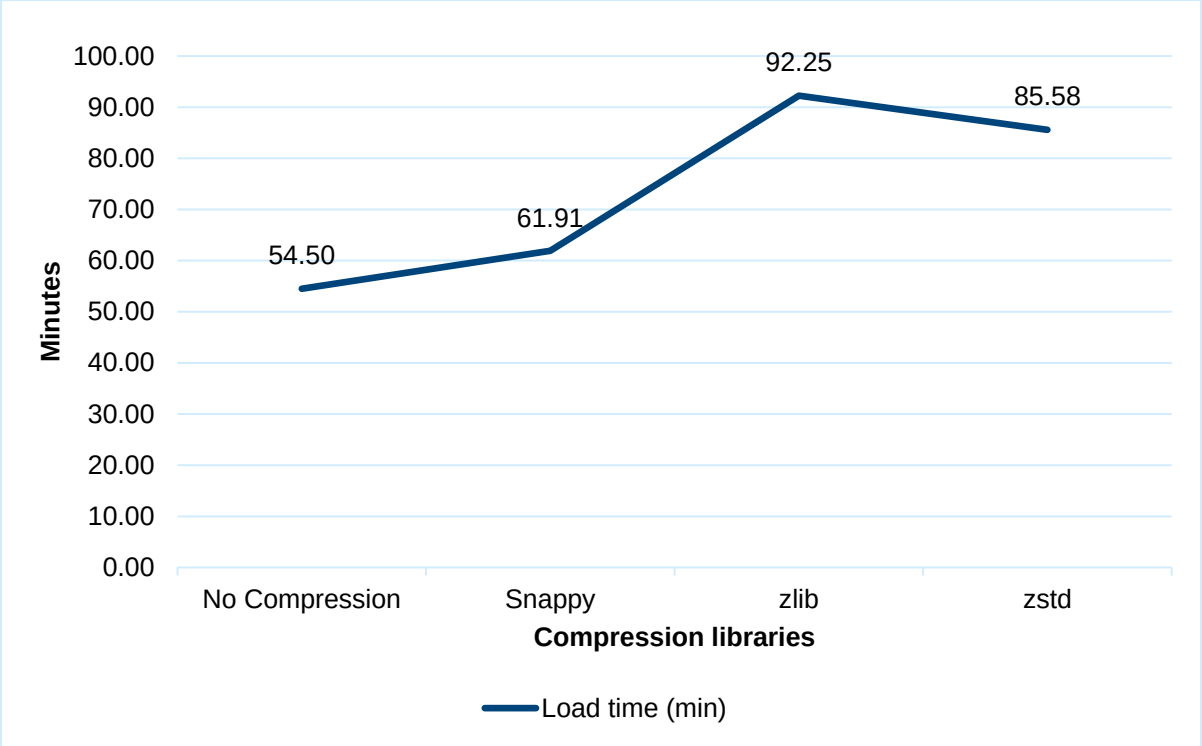

Figure 11 Data load time comparison

4.6 MongoDB database VM CPU comparison

Figure 12 examines the database VMs CPU utilization of different compression libraries.

The best compression zstd also has a higher CPU utilization compared to other compression libraries. Disabling compression consumes the least amount of CPU. The CPU utilization of the snappy library is similar to the CPU consumption of having the compression turned off. The zstd library, which has the best compression saving, also consumes more CPU resources than the rest.

With the zstd library, the CPU consumptions of all the replica set members are a lot more even than the other libraries. The primary member of a replica set is the only host that accepts write requests from the client.

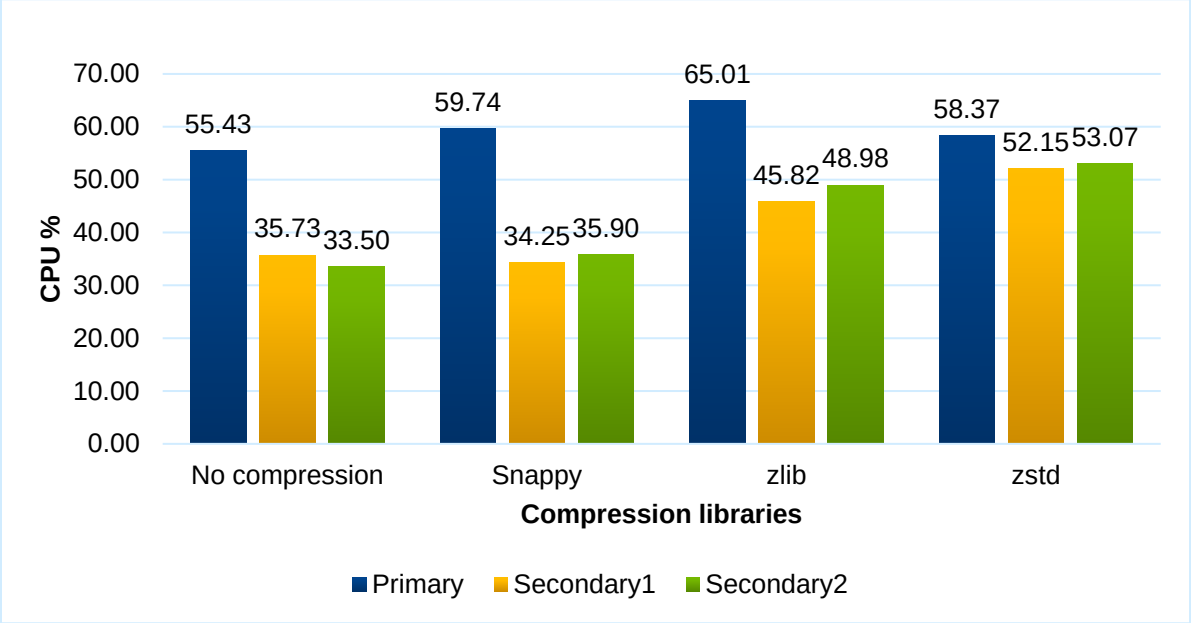

Figure 12 CPU utilization on MongoDB replica set members

4.7 MongoDB database VM memory comparison

Figure 13 examines the database VMs memory utilization of different compression libraries.

MongoDB uses both the internal WiredTiger cache and the file system cache when WiredTiger storage engine is used. The default internal cache takes up to a minimal of 256 MB and caps at 50% of (MEM – 1 GB). With file system cache, MongoDB automatically uses up the rest of free memory available on the system.

WiredTiger compression can reduce the amount of memory consumed by keeping the data in the file system cache the same format as the compressed data on disk. The indexes in the internal cache can also benefit from the index prefix compression. However, the data in the internal cache is uncompressed.

For our tests, the internal cache is kept at the default. The average memory consumption of the replica set is lowest when using the zlib or zstd library. Conversely, the average memory consumption is higher with the snappy library or no compression is used.

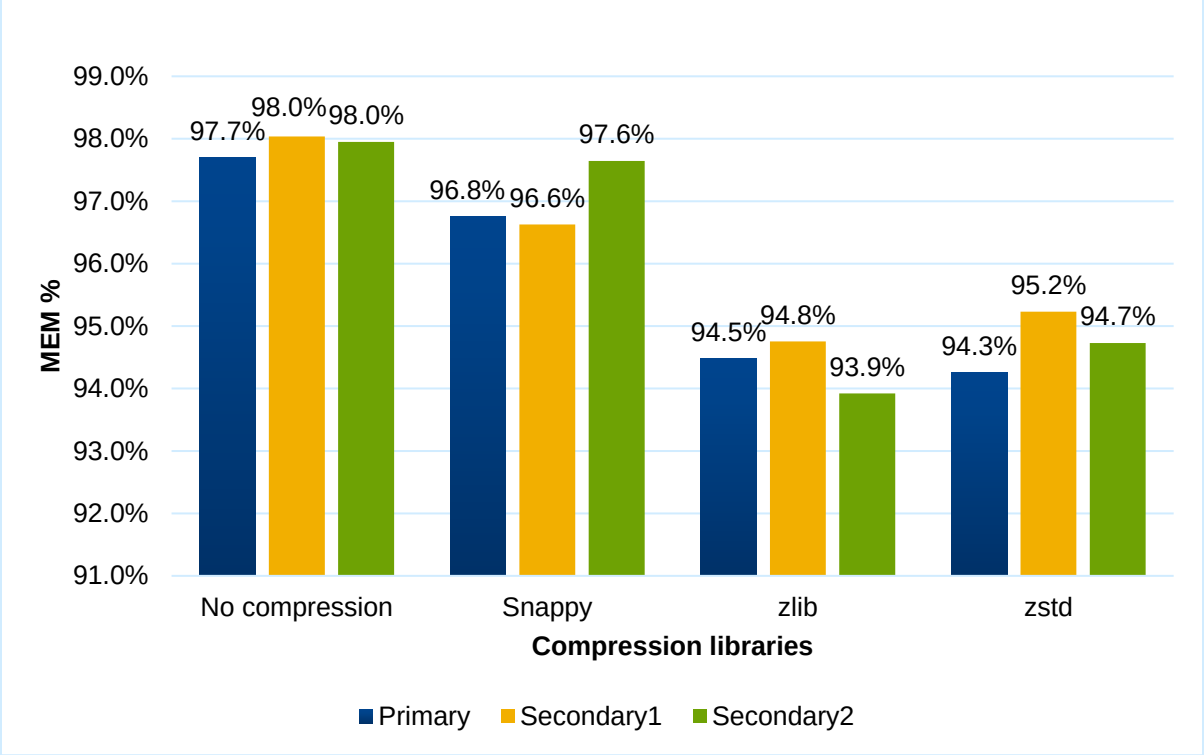

Figure 13 Memory utilization on MongoDB replica set members

4.8 Test conclusion

While the zstd library offers the best software-based compression savings and lower memory utilization, it also has a higher cost of CPU utilization on the operating system.

MongoDB chooses snappy compression as the default because it offers a good balance between compression savings and CPU utilization. As our tests confirm, snappy compression offers some amount of compression savings but requires fewer CPU resources on the operating system.

Combining the MongoDB snappy compression and the PowerStore advanced hardware-based data reduction, the total space-saving is significantly increased and bring it closer to the level of the zstd compression without the downside of higher CPU utilization on the operating system.

5 Data protection

PowerStore includes native data protection features with the snapshot and thin clone technologies. Also, Dell Technologies has a wide range of products to enhance data protection and enable disaster recovery beyond a local storage system.

5.1 Snapshots and thin clones

PowerStore snapshot is a point-in-time copy of data of a storage resource such as a volume, volume group, virtual machine, or file system. You can take manual snapshots in the PowerStore Manager UI or create snapshot policies within protection policies to automatically take snapshots on a predefined schedule or frequency. Snapshots are inaccessible to the hosts directly.

To access the data in a snapshot for a storage resource, except for VM, you can create a read-writable thin clone from the snapshot and map it to the same host or different host. A thin clone is a space-efficient copy that shares the data blocks with its parent object. Multiple thin clones are allowed from a snapshot. Changes to one thin clone do not affect the parent object or other associated thin clones and conversely.

For VMs, PowerStore creates snapshots at the VM level, and the snapshot information reflects in the vCenter automatically. Use PowerStore snapshots as short-term temporary protection for MongoDB. It is not intended to replace the official MongoDB backup solution. To ensure the snapshot is application-consistent, we recommend quiescing any running MongoDB instance before taking a snapshot.

Some of the snapshot and thin clone use cases are as follows:

- Provide quick and easy rollback on operating system or application updates
- Clone the production environment to development or test environments with ease and without a full-size copy of the data.
- Reduce the complexity and time to refresh the data in a clone environment from the latest snapshot or thin clone

To perform snapshot operations in the PowerStore Manager, you take the following actions. The clone, refresh, and restore functions do not apply to VMs.

- Take a snapshot from the Protection card on the Overview page of the storage resource.
- Clone a snapshot from the Protection card on the Overview page of the storage resource.
- Delete a snapshot from the Protection card on the Overview page of the storage resource.
- Refresh the data of a storage resource from the More Actions menu of the storage resource or snapshot.
- Restore a storage resource from the More Actions menu of the storage resource or snapshot.
- Configure, view, and manage snapshot rules from the Policies page.

To recover data from a snapshot of a storage resource, except for VM, perform the following actions in PowerStore Manager:

- Create a thin clone from a snapshot and map it to a host
- Use the refresh operation to replace existing data in the volume with the data from a snapshot or thin clone related to the parent storage resource.

- Use the restore operation to replace the data of a parent storage resource with data from an associated snapshot. The restore operation resets the data in the parent storage resource to the point in time the snapshot was taken.

PowerStore also offers PowerStore command-line interface (CLI) and REST APIs that can be used to automate operations in scripts and other programming languages.

To recover a VM from a VM snapshot in vCenter, revert the VM in vCenter using the VM **Manage Snapshots** operation.

For more information about PowerStore data protection, see the document Dell EMC PowerStore Protecting Your Data.

5.2 AppSync

Dell EMC AppSync™ is an optional software that enhances the overall application protection supported by the software. With its deep integration of PowerStore and applications such as Oracle® databases and Microsoft® SQL Server®, AppSync uses the native PowerStore asynchronous replication, snapshot, and thin clone technologies to create and manage local and remote copies of applications. However, AppSync supports only block storage resources on PowerStore and does not have application integration with MongoDB. For vVols storage resources, see Dell EMC RecoverPoint™ for Virtual Machine in section 5.3.

For more information about PowerStore AppSync integration, see the document *Dell EMC PowerStore: AppSync* on Dell Support.

5.3 RecoverPoint for VMs

Dell EMC RecoverPoint for Virtual Machines is an optional software that extends data protection and enables disaster recovery for VMware virtualized environment to on-premises or cloud environment. RecoverPoint for Virtual Machines is a software-only solution that protects VMs with local and remote replication. It is storage and application agnostic and supports both synchronous and asynchronous replication on all storage types supported by VMware. It also allows replicating multiple VMs in a consistency group.

PowerStore does not support the vVols replication and VM consistency group. Therefore, RecoverPoint for Virtual Machines is a great addition if these features are required. Using RecoverPoint, all MongoDB replica set members can be protected and replicated collectively in a consistency group.

For more information about RecoverPoint for VMs, see the document *RecoverPoint for Virtual Machines Administrator's Guide* on Dell Support.

A Additional resources

A.1 Technical support and resources

Dell.com/support is focused on meeting customer needs with proven services and support.

Storage technical documents and videos provide expertise that helps to ensure customer success on Dell EMC storage platforms.

The PowerStore Info Hub provides detailed documentation on how to install, configure, and manage PowerStore systems.

A.2 MongoDB resources

For MongoDB documentation and support forum, see the following resources:

- www.mongodb.com
- MongoDB Getting Started Guides
- docs.mongodb.com