Best Practices

Dell EMC PowerStore: Microsoft SQL Server Best Practices

Abstract

This document includes architecture, deployment, and performance guidance for Microsoft® SQL Server® with Dell EMC™ PowerStore™.

February 2021

H18250.2

Revisions

Date	Description
April 2020	Initial release: PowerStoreOS 1.0
July 2020	Updated AppsON host configuration and sizing guidance
Feb 2021	Legal disclaimer update

Acknowledgments

Author: Doug Bernhardt

Table of contents

Executive summary

This paper provides best practices for using Dell EMC™ PowerStore™ in a Microsoft® SQL Server® environment. SQL Server is a robust product that can be used in various solutions. The relative priorities of critical design goals such as performance, manageability, and flexibility depend on the environment. This paper provides considerations and recommendations to help meet design goals. For general best practices for using PowerStore systems, see the *Dell EMC PowerStore: Best Practices Guide* on Dell.com/StorageResources.

These guidelines are intended to cover most use cases. They are recommended by Dell Technologies™, but they are not strictly required. For questions about the applicability of these guidelines in your environment, contact your Dell Technologies representative to discuss the appropriateness of the recommendations.

Audience

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

1 Introduction

Dell EMC PowerStore is a robust and flexible storage and compute option that is well suited to SQL Server. The capabilities of the PowerStore platform provide some unique benefits and exciting architecture options for SQL Server workloads.

1.1 PowerStore product 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.

1.2 Microsoft SQL Server product overview

Microsoft SQL Server is the industry-leading data platform. A product which once started as a database engine on Microsoft Windows has evolved into an entire data platform. This platform enables cloud, hybrid, and on-premises deployments in several different environments including Windows, Linux, containers, and Kubernetes. In addition to the wide number of environments that SQL Server can be deployed in, the use cases have expanded. These uses now range from typical online transactional processing (OLTP) to various analytics options such as data warehousing, artificial intelligence (AI), machine learning (ML), and big data. The adaptable and flexible nature of SQL Server makes PowerStore a perfect fit for SQL Server.

1.3 Prerequisite reading

The best practices explained in this document require knowledge from the following resources:

- PowerStore documentation on the PowerStore Info Hub
- *Dell EMC PowerStore Host Configuration Guide* on the PowerStore Info Hub
- Microsoft SQL Server documentation library

1.4 Terminology

The following terms are used with PowerStore.

PowerStore Manager: The web-based user interface (UI) for storage management.

Appliance: Term used for solution containing a base enclosure and any attached expansion enclosures. The size of an appliance could be only the base enclosure or the base enclosure plus expansion enclosures.

Node: The component within the base enclosure that contains processors and memory. Each appliance consists of two nodes.

Cluster: Multiple PowerStore appliances in a single grouping. Clusters can consist of one appliance or more. Clusters are expandable by adding more appliances. A cluster supports up to four appliances.

Base enclosure: Used to reference the enclosure containing both nodes (node A and node B) and the NVMe drive slots.

Expansion enclosure: Enclosures that can be attached to a base enclosure to provide 25 additional SAS-based drive slots.

Virtual Volumes (vVols): PowerStore X model volumes support VMware® vSphere® Virtual Volumes™ (vVols). These volumes are displayed in the Virtualization area of PowerStore Manager.

Storage volumes: PowerStore volumes using block storage. These volumes are displayed in the Block area of the PowerStore dashboard.

2 Scale-Up and Scale-Out Architecture

Scale-up and scale-out are common architecture models used when accommodating a growing data footprint. Microsoft SQL Server has supported both scale-up and scale-out architecture models for several years. Dell EMC PowerStore aligns with the SQL Server by also providing both scale-up and scale-out architecture options. When designing data architectures, there are tradeoffs in design, functionality and scalability that must be considered.

2.1 SQL Server Scale-up

When you scale up an instance of SQL Server, the amount of memory, compute, and storage can all be increased by upgrading to a higher software version or increasing the hardware capabilities. You get the same or more functionality and retain the same logical data design and therefore the data architecture is unaffected. This is a common approach because it is low impact.

2.2 SQL Server Scale-out

Scale-out architectures on SQL Server enable a data footprint to scale past what can be achieved by a single instance. The trade-off is that these architectures require considerably more planning and consideration and often require applications to be designed accordingly. Concepts such as data placement, load balancing, and data sharding may need to be addressed as well as the impact on features such as replication, availability groups, etc.

2.3 PowerStore Scale-up

PowerStore models can scale up with several different upgrade paths, by adding additional drives or drive expansion shelves to an existing appliance, adding additional storage network paths, upgrading to the next generation hardware as it becomes available, or upgrading to more powerful models within the same family. Similar to SQL Server, this is also a low impact approach. PowerStore redundancy features and the Anytime Upgrades program offered by Dell Technologies ensure a frictionless upgrade experience.

2.4 PowerStore Scale-out

PowerStore can also scale-out by adding additional nodes to the PowerStore cluster. PowerStore can add up to 4 nodes in a cluster to provide additional capacity and performance. Much like scale-up vs. scale-out architecture decisions for SQL Server, there are similar considerations for PowerStore as well. Therefore, consult the Dell EMC PowerStore: Clustering and High Availability Guide on dell.com/PowerStoreDocs for complete details. There are a couple of differences worth mentioning:

Volume groups must reside on a single appliance

vVol based virtual machines must be powered off prior to vVol migration to another appliance in the cluster

3 Host configuration

PowerStore provides the option of running workloads directly on the PowerStore Appliance as virtual machines or running workloads on external hosts and presenting storage through block or file interfaces.

Prior to creating objects or deploying workloads on PowerStore it is recommended to review the following guides for best practices and tuning recommendations. It is recommended to apply these changes prior to deploying workloads since some change require a reboot of the PowerStore appliance nodes. For resiliency, it is recommended that external hosts are connected via multiple paths, or multipath, to the PowerStore appliance.

3.1 External SQL Server hosts

PowerStore presents storage to external hosts through either block or file interfaces. Block storage is commonly used for SQL Server workloads due to the various speeds and protocols that are offered which make it ideal for performance. PowerStore supports 25 GbE and 32 Gb Fibre Channel (FC) which are recommended for SQL Server high-bandwidth storage workloads such as analytics that can generate a large amount of large block I/O. Instructions and best practices for configuring hosts can be found in the *Dell EMC PowerStore Host Configuration Guide* on the PowerStore Info Hub. File storage (SMB) can also be used in environments running SQL Server on Windows.

For PowerStore T model appliances follow recommendations for configuring external hosts in the Dell EMC PowerStore Host Configuration Guide. In addition, when running SQL Server on virtual machines with VMware ESXi™ hosts, review the additional guidance in the Dell EMC PowerStore Virtualization Guide and Dell EMC PowerStore: VMware vSphere Best Practices papers.

3.1.1 Boot from SAN

PowerStore supports boot-from-SAN for environments that want to further virtualize storage resources. In addition to the storage virtualization benefits, Boot-from-SAN can also be used to protect the operating-system configuration and allow for quick recovery. This configuration can be beneficial in environments where the recovery time objective (RTO) is strict. This scenario is detailed in section 6.

3.2 AppsON

Integration of the PowerStore container-based architecture with integrated VMware ESXi results in a new level of consolidation for enterprise storage. This integration combines the benefits of a local on-array application environment with unmatched integration with the vSphere management environment and server resources. This ability allows users to bring applications closer to storage, by running applications as virtual machines directly on PowerStore.

Benefits of the AppsON capability include a new level of agility for application deployments, with seamless movement between the PowerStore appliances and VMware ESXi servers. AppsON also provides the ability to shrink the stack by eliminating server and networking footprint for space-efficient edge and remote deployments. AppsON is ideal for data-intensive applications that require low latency or a storage-heavy imbalance of compute and storage.

AppsON can be an attractive option whether the goal is to optimize I/O performance by reducing components in the data path, add performance to an existing vSphere cluster, offload a noisy neighbor, or various other reasons.

For PowerStore X model appliances the PowerStore X Performance Best Practice Tuning paper provides several tips that increase performance for SQL Server workloads, especially when running workloads that use block sizes greater than 128k such as analytic workloads and backup/restore operations. In addition to the recommendations in this paper, when running workloads larger than 128k, additional performance gains may be found by increasing the iSCSI maximum IO size, which is 128k by default. This can be achieved by running the following command on each host:

```
esxcli system settings advanced set -o /ISCSI/MaxIoSizeKB -i 512
```

Instructions for running esxcli commands on PowerStore X nodes are found in the PowerStore X Performance Best Practice Tuning paper.

4 Sizing and performance optimization

There are several best practices for provisioning compute and storage to SQL Server. The size of power and flexibility of SQL Server allow it to be used in several different deployment scenarios. This section covers some considerations for optimizing PowerStore.

4.1 SQL Server design considerations

The I/O storage system is a critical component of any SQL Server environment. Sizing and configuring a storage system without understanding the I/O requirements can have unfavorable results. Analyzing performance in an existing environment using a tool like Live Optics can help define the I/O requirements. For best results, capture performance statistics for a period of at least 24 hours that includes the system peak workload.

Demanding SQL Server workloads that require low latency, high bandwidth, or both can benefit from moving the compute to PowerStore X model arrays. Moving these intensive workloads onto the storage platform where compute and storage are located together can deliver higher IO performance for performance sensitive applications. Also, virtual machines running other application components can be on PowerStore X models for further performance gains. Using PowerStore X models in a larger vSphere cluster and taking advantage of VMware vSphere vMotion® capabilities can be a powerful tool in balancing not only SQL Server workloads but also application workloads across the entire environment.

4.1.1 OLTP workloads

While every environment is unique, an online transaction processing (OLTP) workload typically consists of small, random reads and writes. A storage system that services this type of workload is primarily sized based on capacity and the number of IOPS required. PowerStore models allow the flexibility to be configured for block-optimized, unified (block and file), or AppsON to meet unique OLTP requirements.

4.1.2 Analytic workloads

An online analytic processing (OLAP), decision support system (DSS), or big data workload is typically dominated by large, sequential reads. A storage system services this type of workload is primarily sized based on throughput. When designing for throughput, the performance of the entire path between the server and the drives in PowerStore needs consideration. This is handled automatically when using virtual volumes on PowerStore X models. When mapping volumes to hosts, consider using 32 Gb Fibre Channel (FC) or 25 Gbps iSCSI connectivity to the array. For high-throughput requirements to be met, multiple HBAs may also be used in the server, the array, or both.

SQL Server 2019 introduced Big Data Clusters as a brand-new environment for analytics. Combining the power of SQL Server along with Apache® Hadoop® Distributed File System (HDFS), Spark, and other technologies to provide a data analytics or data lake platform and easily deploy that platform on Kubernetes is revolutionary. The flexibility and compute capabilities of PowerStore X models make a great deployment platform for SQL Server Big Data Clusters. For more information, see the document *Dell EMC PowerStore: SQL Server 2019 Big Data Clusters* on Dell.com/StorageResources.

4.1.3 Mixed workloads

The most common scenario is a mixed workload environment. Typically, SQL Server I/O patterns do not strictly fall into an OLTP or analytics pattern. This is what can make SQL Server workloads challenging because no two workloads behave the same. In addition, the same SQL Server host or instance may be

servicing multiple applications or transaction workloads. Mixed workload can also imply that multiple applications (in addition to SQL Server) are residing on the same host or accessing the same storage. The combined workload of these applications invalidates any typical application I/O usage pattern. For these reasons, it is important to gather performance metrics for best sizing results.

4.2 SQL Server deployment platforms

SQL Server is now available and supported on several different platforms. PowerStore presents several flexible options for various SQL Server deployments.

4.2.1 Windows

When deploying SQL Server on Windows on PowerStore, virtual volumes and storage (block) volumes are available on PowerStore X models, while storage (block) volumes and SMB are available on PowerStore T models. All of these options are acceptable for use with SQL Server and rely on the configuration of the volumes through Windows. When using SMB storage for SQL Server, ensure the Sync Writes Enabled and Oplocks Enabled are turned on for the file system. See the Windows section of the *PowerStore Host Configuration Guide* on the PowerStore Info Hub for instructions about configuring host access on Windows.

4.2.1.1 SQL Server on Windows deployment steps

SQL Server on Windows is deployed with an executable (exe) file which can be found on the Microsoft SQL Server website. Deployment is done by downloading the SQL Server exe file for the intended version and running the executable. The SQL Server deployment wizard will walk through several available options. Storage layout can either be customized during the install or after install once the SQL Server instance is running. When performing large enterprise deployments there is an unintended install option available to automate the installation.

4.2.2 Linux

When deploying SQL Server on Linux on PowerStore, Virtual Volumes and storage (block) volumes are available on PowerStore X models, while only storage (block) volumes are available for use with SQL Server on Linux on PowerStore T models. Although PowerStore T models contain file capabilities, they do not support NFS v 4.2 which is the minimum that is required for SQL Server on Linux. Therefore, NFS is not supported.

Be sure to follow Microsoft Installation guidance for deploying SQL Server on Linux for supported Linux versions, minimum requirements, and general recommendations. Currently RHEL, SUSE, and Ubuntu Linux distributions are supported. The partnership of Dell Technologies, Microsoft, and RedHat provides a platform will full end-to-end enterprise support and therefore RHEL has quickly become a platform of choice for deploying SQL Server on Linux.

4.2.2.1 SQL Server on Linux deployment steps

Once the Linux host is presented to the array, storage volumes can be presented to the host. Since Linux is a relatively new platform for SQL Server, an example is provided below for configuring a RHEL host to consume PowerStore block volumes. This example uses multipath, which is a best practice for resiliency. For completed instructions on configuring Linux hosts and multipath consult the *PowerStore Host Configuration Guide* on the PowerStore Info Hub.

After the storage volume have been provisioned to the host, run the following command to make the volumes visible to the host:

```
#/usr/bin/rescan-scsi-bus.sh –a
```

Identify the volume name in the mapper folder by listing the multipath volumes and searching for the volume WWN from PowerStore. The volume WWN is visible in the PowerStore volumes list under Storage -> Volumes. For example, if the WWN for the new volume is “68ccf0980018e6c92364df1970e08d33” then the following command will identify the mapper folder to use:

```
# multipath -ll | grep 68ccf0980018e6c92364df1970e08d33
```

The volume will need to be formatted prior to use. Consult the Microsoft SQL Server documentation for supported file systems. Currently XFS and EXT4 are supported. In this example, XFS is being used. For XFS the mkfs.xfs command is used to format the volume passing the correct mapper folder as the parameter.

```
# mkfs.xfs /dev/mapper/mpathdb
```

A mount folder needs to be created to represent the new volume. Create a folder to represent the mount:

```
# mkdir /var/opt/mssql/data5
```

To make the mount persistent, the mapping needs to be stored in `/etc/fstab` so it will be created at boot time. The entry in the file will consist of six fields contained on a single line separated by a space: device, mount point, file system type, options, backup operation, and file system check order. When mounting volumes for SQL Server it is a Microsoft best practice to add the `noatime` attribute to the list of options to disable updates to the volume timestamp for better performance. The fields *backup operation* and *file system check order* will be left at the default of 0 and can be changed as needed. Based on our example, the new entry in the `/etc/fstab` file would look like this:

```
/dev/mapper/mpathdb /var/opt/mssql/data5 xfs defaults,noatime 0 0
```

Once the entry for the new mount is created, running the mount command will read the file and refresh the mounted devices. Even if the mount was manually created, it is a good idea to run the mount command after updating /etc/fstab to ensure there are no errors. If there are errors in the file, it could prevent Linux from booting properly.

```
# mount -a
```

Finally, before SQL Server can access the new mount, permissions need to be set. In a production environment, follow the security best practices for your organization when assigning permissions and ownerships to mount points. This example uses the same default permissions and ownership from the SQL Server installation by copying the permissions from the /var/opt/mssql/data folder and applying them to the new mount point:

```
# chmod --reference /var/opt/mssql/data /var/opt/mssql/data5
# chown --reference /var/opt/mssql/data /var/opt/mssql/data5
```

At this point the new volume is ready for SQL Server to use. See the Linux section of the *PowerStore Host Configuration Guide* on the PowerStore Info Hub for further instructions on configuring storage volume access on Linux.

4.2.3 Containers

Running SQL Server inside Docker® containers is a deployment option that was introduced with SQL Server 2017. By default, the database storage is inside of the container and is ephemeral in nature. If databases

need to be persisted beyond the lifetime of the container, external storage must be presented. External storage is presented to a container through a bind mount which is a local file system folder or mount from the operating system that is presented to the container at startup. Follow the storage guidance above for either Windows or Linux to create file system folders or mount points to present as a volume bind mounts to containers.

4.2.3.1 Deploying SQL Server in Containers

Deploying SQL Server inside Docker containers is extremely simple. Below is an example of running SQL Server inside a container followed by a brief explanation of the parameters:

```
# sudo docker run -e "ACCEPT_EULA=Y" -e "SA_PASSWORD=P@ssword1" -p 1401:1433  -v
/root/sql1/tpcc_data:/var/opt/mssql/data2 -v
/root/sql1/tpcc_log:/var/opt/mssql/log2 --name sql1 -d
mcr.microsoft.com/mssql/server:2019-latest
```

Table 1 Docker run parameters

Parameter	Description
-e	Sets environment variables, in this case options for licensing and the SA password are set
-p	Publishes the SQL Server port 1433 inside the container as 1401 on the host
-v	Bind mount for a volume. In this example three are specified. The format is `local mount:container mount`
--name	Name of the container, otherwise it will be assigned a random identifier
-d	Container image to run. If it does not exist, it will try to locate it from any configured repositories and download it

4.2.4 Kubernetes

Due to the clustered architecture of Kubernetes, there are special storage considerations when running SQL Server instances or SQL Server Big Data Clusters on Kubernetes. Kubernetes has created its own method of dealing with persistent storage in a Kubernetes cluster using persistent volumes. PowerStore storage integrates with Kubernetes through the PowerStore CSI driver. The PowerStore CSI driver supports storage volumes only. Kubernetes clusters running on PowerStore X models can either use the vSphere Cloud Provider for virtual volumes or the PowerStore CSI driver for storage volumes. Detailed information about running SQL Server Big Data Clusters on PowerStore is available in the document *Dell EMC PowerStore: SQL Server 2019 Big Data Clusters* on Dell.com/StorageResources. The PowerStore CSI driver and CSI drivers for other Dell Technologies products can be found on GitHub.

Whenever possible make sure to perform storage operations through Kubernetes. In some cases, modifying CSI volumes directly on the PowerStore appliance can cause CSI operations to fail or volumes to be orphaned. For example, CSI volumes added to a volume group need to be removed from the volume group before the CSI driver can delete them.

The capabilities of the Kubernetes CSI specification are constantly evolving to support additional enterprise storage features, so make sure to use the latest Dell Technologies PowerStore CSI driver and keep up to date on new releases. The latest documentation for Dell Technologies CSI drivers for all Dell EMC storage products can be found at dell.github.io/storage-plugin-docs.

4.2.4.1 Deploying SQL Server on Kubernetes

Dell EMC PowerStore storage for SQL Server running on Kubernetes is deployed and managed almost entirely through Kubernetes. The PowerStore CSI driver translates Kubernetes storage provisioning commands into PowerStore storage provisioning commands that are orchestrated on the storage appliance. Since common storage provisioning tasks such as creating, deleting, and expanding volumes are done through Kubernetes, the DBA or Kubernetes administrator can provision storage without knowing the details of PowerStore or requiring direct access. This enables self-service of common storage provisioning tasks.

Once the Dell EMC PowerStore CSI driver is deployed, one or more Kubernetes storage classes will be created. These storage classes describe the details of the CSI driver, encapsulate the connectivity and protocol options of the storage, and contain defaults for common storage provisioning values. Below is an example of powerstore-xfs storage class definition:

```
allowVolumeExpansion: true
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  annotations:
    meta.helm.sh/release-name: powerstore
    meta.helm.sh/release-namespace: csi-powerstore
  creationTimestamp: "2020-10-07T13:33:14Z"
  labels:
    app.kubernetes.io/managed-by: Helm
  name: powerstore-xfs
  resourceVersion: "221004"
  selfLink: /apis/storage.k8s.io/v1/storageclasses/powerstore-xfs
  uid: cd013881-522e-4e16-bbd0-1f3c31315c94
parameters:
  FsType: xfs
provisioner: csi-powerstore.dellemc.com
reclaimPolicy: Delete
volumeBindingMode: Immediate
```

Kubernetes storage is provisioned as Persistent Volumes and Persistent Volume Claims. More information on Persistent Volumes and Persistent Volume Claims can be found at Kubernetes.io. For example, to create three volumes for a SQL Server pod running on Kubernetes, the definitions are stored in a YAML file:

```
# cat pvc.yaml
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: mssql-data
  annotations:
    volume.beta.kubernetes.io/storage-class: powerstore-xfs
spec:
  accessModes:
  - ReadWriteOnce
```

```
  resources:
    requests:
      storage: 8Gi
---
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: mssql-data2
  annotations:
    volume.beta.kubernetes.io/storage-class: powerstore-xfs
spec:
  accessModes:
  - ReadWriteOnce
  resources:
    requests:
      storage: 5Gi
---
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: mssql-log2
  annotations:
    volume.beta.kubernetes.io/storage-class: powerstore-xfs
spec:
  accessModes:
  - ReadWriteOnce
  resources:
    requests:
      storage: 2Gi
```

To create the persistent volume claims, the YAML file above is used as input. Running the following command will create the volumes on PowerStore and create a volume alias that can be used when creating the SQL Server pod.

```
#kubectl create -f pvc.yaml
```

Next, to deploy SQL Server a definition file is also used. The `persistentVolumeClaim` attributes in the definition file refer to the Persistent Volume Claims created in the previous step.

```
#cat sql.yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: mssql-deployment
spec:
  replicas: 1
```

```
    selector:
       matchLabels:
         app: mssql
    template:
      metadata:
        labels:
          app: mssql
      spec:
        terminationGracePeriodSeconds: 30
        hostname: mssqlinst
        securityContext:
          fsGroup: 10001
        containers:
        - name: mssql
          image: mcr.microsoft.com/mssql/server:2019-latest
          ports:
          - containerPort: 1433
          env:
          - name: MSSQL_PID
            value: "Developer"
          - name: ACCEPT_EULA
            value: "Y"
          - name: SA_PASSWORD
            valueFrom:
              secretKeyRef:
                name: mssql
                key: SA_PASSWORD
          volumeMounts:
          - name: mssqldb
            mountPath: /var/opt/mssql
          - name: mssqldata
            mountPath: /var/opt/mssql/data2
          - name: mssqllog
            mountPath: /var/opt/mssql/log2
        volumes:
        - name: mssqldb
          persistentVolumeClaim:
            claimName: mssql-data
        - name: mssqldata
          persistentVolumeClaim:
            claimName: mssql-data2
        - name: mssqllog
          persistentVolumeClaim:
            claimName: mssql-log2
```

```
---
apiVersion: v1
kind: Service
metadata:
  name: mssql-deployment
spec:
  selector:
    app: mssql
  ports:
    - protocol: TCP
      port: 1433
      targetPort: 1433
  type: NodePort
```

To deploy SQL Server inside of a Kubernetes pod, first a secret is created to store the password:

```
#kubectl create secret generic mssql --from-literal=SA_PASSWORD="P@ssword1"
```

Next, using the YAML file above as input, kubectl create will deploy the SQL Server instance on Kubernetes:

```
# kubectl create -f sql.yaml
```

4.3 RAID configurations and storage pools

Since the inception of storage devices that used multiple drives for a single volume, the topic of RAID configuration and storage pools has been highly debated. Designing and maintaining the proper design could be time consuming and the wrong decisions could have a negative impact on performance.

PowerStore automatically manages the underlying storage for maximum performance and capacity, eliminating the need for administrators to configure RAID or storage pools. Manually setting or configuring these options is unnecessary in PowerStore.

4.4 Validating the I/O path

Before deploying workloads on PowerStore, it is a good idea to validate the I/O path between the server and the array. Running a large block sequential read test using small files should saturate the path between the server and the array. This test verifies that all paths are fully functional and can be used for I/O traffic. Run this test on an isolated server and PowerStore. Running this test in a production environment could cause significant performance issues for active workloads. To validate the I/O path, run a large block sequential read test using the following guidelines:

- Create one volume per node.
- Format the volumes using a 64 KB allocation unit.
- Use a block size of 512 KB for the test.
- Configure the test for 32 outstanding I/Os.
- Use multiple threads; eight is the recommended starting point.

If the displayed throughput matches the expected throughput for the number of HBA ports in the server, the paths between the server and PowerStore are set up correctly.

4.5 MPIO

A best practice for SQL Server hosts is to provide multiple paths to the storage for resiliency and performance. This protects against possible data loss or downtime, should a physical component fail. In scenarios where multiple paths to the storage are used, MPIO must be enabled and configured. See the document *Dell EMC PowerStore Host Configuration Guide* on the PowerStore Info Hub for recommendations.

4.6 Performance

Managing storage performance with PowerStore is simple and intuitive. When using virtual volumes, there is nothing to change to adjust storage performance. When block volumes are used, the performance policy can be adjusted at the volume level by selecting high, medium, or low. Performance policies are relative in PowerStore, so leaving all volumes at the default of medium provides equal performance to all volumes. The volumes all perform at the highest level until a PowerStore volume is set to a different performance policy.

When running SQL Server inside virtual machines (VMs), it is recommended to follow the VMware document SQL Server on VMware Best Practices Guide for performance and scalability. Also, it is recommended to follow the performance recommendations in document *Dell EMC PowerStore: Virtualization Integration* on Dell.com/StorageResources. The recommendations in this guide provide substantial improvements in I/O performance.

4.7 Reducing SQL Server I/O

Many common techniques can help reduce SQL disk I/O from the host.

4.7.1 Memory

Allocating the proper amount of memory to SQL Server can increase performance and avoid unnecessary I/O. SQL Server performs all I/O through the buffer pool (cache) and uses a large portion of its memory allocation for the buffer pool. Ideally, when SQL Server performs I/O, the data is already in the buffer pool and it does not need to go to disk. This type of I/O is referred to as logical I/O and is the most desirable because it results in the best performance. If the SQL Server data does not need to reside in the buffer pool, it needs to access disk resulting in physical I/O. Proper memory allocation is critical to SQL Server performance and can improve storage performance as well. Often, SQL Server and storage performance can be further improved by adding additional memory to the SQL Server instance. Generally, allocating more memory is better, but there is a point of diminishing returns that is unique to each environment.

4.7.2 Buffer pool extension

Starting with SQL Server 2014, the buffer pool can be extended to a file on the file system to provide additional space to cache data or index pages. Using this feature can provide significant performance benefits without adding memory to the database server in some cases. With more pages cached on the server, the I/O load on the array is reduced. When placing the buffer pool extension on the array, create a separate volume for the buffer pool extension and do not configure snapshots for the buffer pool extension volume. The buffer pool data is repopulated by SQL Server when the instance is restarted. Therefore, data recovery does not apply.

4.7.3 Persistent memory

SQL Server 2016 introduced support for persistent memory (PMEM). The capabilities of PMEM are being expanded with SQL Server 2019 to cover more scenarios as well as the Linux operating system. Often, PMEM can be used to accelerate challenging I/O workloads and make I/O patterns more efficient with SQL Server PMEM features such as tail-of-log-cache and Hybrid Buffer Pool. Virtualized environments running Hyper-V or VMware can also take advantage of PMEM making it a wise investment. Dell EMC servers support PMEM starting with Dell EMC PowerEdge™ 14th-generation servers. For more information, see Storage Class Memory in SQL Server 2016 SP1 and the Dell EMC NVDIMM-N Persistent Memory User Guide.

4.7.4 Instant File Initialization

By default, SQL Server writes zeros to the data file during the allocation process. The process of zeroing out the data files consumes I/O and acquires locks as the SQL Server data pages are written. This activity can occur for minutes or even hours depending on the file size. While this may seem minor, writing zeros to these files can occur at disruptive periods when time and performance are critical. Example scenarios include database auto growth, expanding a full data file, replication, or restoring a database as part of a disaster-recovery event. When Instant File Initialization is enabled, SQL Server will skip the process of zeroing out its data files when allocating space. Dell Technologies recommends enabling Instant File Initialization.

4.7.5 Resource Governor

The Resource Governor was added in SQL Server 2008 to allow database administrators to limit the CPU and memory resources a SQL query can consume. This feature was enhanced in SQL Server 2014 to allow I/O resources to be limited as well. For example, the Resource Governor can be used to reduce the impact of a user running an I/O intensive report by limiting the maximum number of IOPS that user can perform. While a query throttled by the Resource Governor takes more time to complete, overall database performance is better.

4.7.6 Database design considerations

Reducing SQL Server I/O requires a holistic approach. Many of the items in this section require involvement from the whole team that is responsible for the SQL Server applications. This team could include the business owner, architect, developer, database administrator, and system administrator. Decisions at the design level have a multiplied downstream impact as data is written and read multiple times and duplicated in various types of database copies. These include databases that are copied for other uses such as testing and reporting, replicated databases, replicated storage, and backups. One of the most challenging aspects of SQL Server is that the I/O pattern and the amount of I/O that is generated can vary greatly depending on the application, even if those applications have databases of the same size. This is because the design of both the database and the data access code control SQL Server I/O.

Database tuning can be one of the most cost-effective ways to reduce I/O and improve scalability. At a high level, consider the following areas when tuning a database to reduce I/O.

4.7.7 Database design

The foundation of the entire database and the schema for how data will be stored and ultimately accessed is determined by the database design. The database design should support both usability and efficient data access. This includes efficient table design and data types as well as indexes, partitioning, and other features that can improve efficiency. It is common for database design to only be focused on usability while performance and scale are overlooked.

4.7.8 Query design

How a query is written can greatly affect the amount of I/O SQL Server must perform when running the query. Queries should return only the required amount of data in the most efficient manner possible. Tune the queries responsible for consuming a relatively large number of resources as well as possible for best performance and scale.

4.7.9 Application design

Consider how applications are using the data and the way it is requested. Sometimes code and component reuse can result in the same data being unnecessarily retrieved repeatedly. All data access should be purposeful.

4.7.10 Maintenance

SQL Server uses a cost-based optimizer to generate query plans for data access. These plans are based on the statistics regarding how data is distributed in the tables. If the statistics are inaccurate, bad query plans may result and unnecessary I/O will be performed. Proper database maintenance includes ensuring that statistics are up to date. Frequent data modifications can also lead to fragmentation within SQL Server data files, producing unnecessary I/O. Fragmentation can be addressed through index reorganization or rebuilds as part of regular database maintenance. The database maintenance process itself can also have a large I/O impact. Typically, every table and index does not need to be rebuilt or reorganized every time maintenance is run. In addition, table partitioning strategies can also be leveraged to make the maintenance process more selective. Consider implementing intelligent maintenance scripts that utilize usage statistics to perform maintenance on an as-needed basis. For mission-critical databases, maintenance activities need to be considered as part of the overall design. If maintenance is not considered as part of the overall process, issues can arise, such as unmanageable sizes and feature incompatibilities that limit available options and strategies.

4.8 AppsON sizing guidance

When running workloads on virtual machines on the PowerStore X appliance, or AppsON mode there are some important considerations when sizing workloads. In general, it's best to prioritize the most storage demanding workloads for AppsON mode and run general workloads on other ESXi hosts in the vSphere cluster.

General VMware tuning and resource allocation principles apply, consult the VMware article About vSphere Resource Management for general tuning guidelines. If resources in the vSphere cluster become a bottleneck, guest storage performance will suffer, reducing the effectiveness of running workloads in AppsON mode.

4.8.1 Resource reservation

PowerStore X appliances reserve approximately half the CPU and memory resources for running PowerStore appliance tasks. When sizing workloads to run on the appliance, this needs to be considered. This reservation is visible in vSphere as the CPU and memory will be displayed as consumed by the PowerStore X controller virtual machines. In addition, the ESXi cluster can't be run at 100% capacity without incurring degraded performance. A minimal amount of resources are required for ESXi overhead. Consult the VMware articles Configuring Resource Allocation Settings and Administering Memory Resources for resource reservation and sizing guidance

For production applications where guaranteed performance is necessary, make sure to size for possible node failover in the PowerStore cluster.

4.8.2 SQL Server virtual machines

General best practices apply when running SQL Server virtual machines in AppsOn mode with a couple of items of note. First, VMware vVols will be used as storage for VMDKs. While other storage options are still available, PowerStore X is optimized for vVols storage. Also, use the VMware Paravirtualized SCSI (PVSCSI) Controller as the virtual SCSI Controller. The PVSCSI controller improves performance and reduces CPU overhead.

For additional best practices on running SQL Server virtual machines consult the VMware guide Architecting Microsoft SQL Server on VMware vSphere.

4.8.3 SQL Server Big Data Clusters

The Kubernetes platform performs resource balancing and scheduling within the Big Data Cluster and is not aware of the virtualization layer. Therefore, it is recommended to balance the workload across both nodes in the AppsON environment. In a simple scenario this means an even number of worker nodes in the Big Data Cluster. In more complex scenarios, this may mean placing certain nodes in the AppsON environment and leveraging Kubernetes scheduling features to place targeted workloads on AppsON nodes. These concepts are covered in the Kubernetes documentation Assign Pods to Nodes.

Kubernetes nodes in the master role consume relatively little resources. However, they need to be responsive and highly available. Master node VMs could be run on other available nodes in the vSphere cluster when applicable to reserve resources for the AppsON environment.

Regardless of where master role VMs run, based on internal testing, it is recommended to reserve CPU and memory resources for these K8s masters. If resource contention is detected within K8s masters and can't immediately get the resources they require, this can panic the K8s resource monitor and workload pods may be restarted.

5 Management and monitoring

PowerStore offers multiple features for managing and monitoring storage.

5.1 Mapping or unmapping best practices

After creating a volume, mapping presents storage from the PowerStore array to a host or a cluster of hosts that are defined in a host group.

5.1.1 SQL Server failover cluster

SQL Server Failover Cluster Instances rely on Windows Server Failover Clustering. Dell Technologies recommends using host groups to uniformly present storage to all the initiators for reduced management complexity. This allows a volume or set of volumes to be mapped to multiple hosts at a time and maintain the same Logical Unit Number (LUN) across all hosts. If volumes in the failover cluster do not have the same LUN number across Windows hosts in the cluster, failover does not work properly.

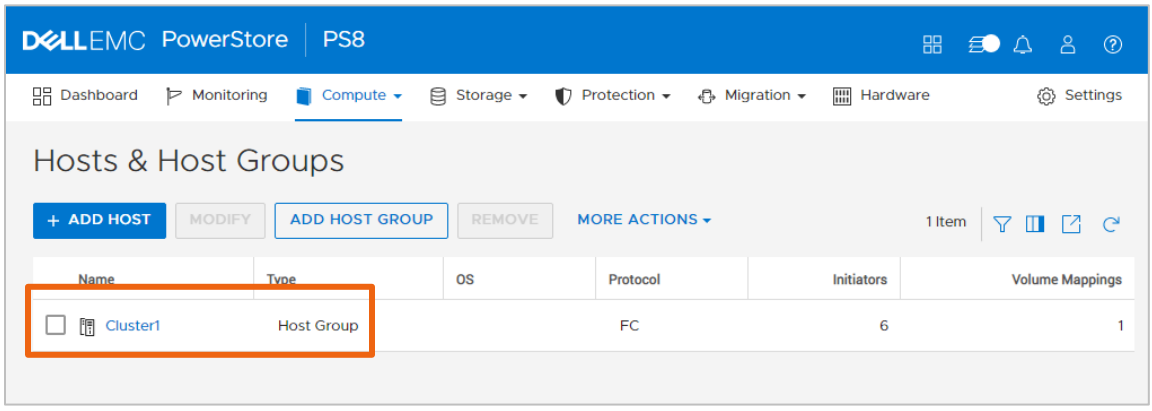

Figure 1 Example of failover cluster containing multiple Windows hosts

5.2 Creating volumes

PowerStore is virtualized to take advantage of all the drives in the storage pool, and in addition there are many types of files that are part of a SQL Server instance. These types of data often have different performance and snapshot requirements. For performance-sensitive applications, Dell Technologies recommends creating at least five volumes for an instance of SQL Server as shown in the following table.

Table 2 Volume provisioning recommendations

File type	Number of volumes per instance	Typical performance requirements	Typical snapshot requirements
User DB data	1+	Lower performance may be acceptable	Frequent snapshots, same volume group as log volumes
User DB transaction log	1+	High performance required	Frequent snapshots, same volume group as data volumes
Data root directory (includes system DBs)	1	Lower performance may be acceptable	Infrequent snapshots, independent schedule

File type	Number of volumes per instance	Typical performance requirements	Typical snapshot requirements
Tempdb data and transaction log	1	High performance required	No snapshots
Native SQL Server backup	1	Lower performance may be acceptable	Snapshots optional, independent schedule
Memory-optimized filegroup (if used)	1+	High performance required	Frequent snapshots, same schedule as log volume

Associate databases that span multiple LUNs within a volume group. This helps ensure that other features such as snapshots, replication, and thin clones will be configured properly and maintain data consistency.

5.3 Allocation unit size

When formatting volumes, choose a 64 KB allocation unit size. This aligns with the 64 KB database extent size, which is the allocation unit that is used by SQL Server. The allocation unit size should be the same when formatting volumes on Windows or Linux platforms.

5.4 Disk-format wait time

With trim/unmap enabled (default setting in Windows Server), significant wait time occurs when formatting a PowerStore volume that is more than a few hundred GB. The larger the volume is, the longer the format wait time is, which is a common situation with external storage. This case applies to volumes formatted as NTFS or ReFS.

To avoid a long format wait time when mapping and formatting a large volume, temporarily disable trim/unmap. This setting is disabled using the **fsutil** command from a command prompt with administrator privileges.

Figure 2 shows the commands to query the state and to disable trim/unmap for NTFS and ReFS volumes on a host. A **DisableDeleteNotify** value of **1** means that trim/unmap is disabled, and long format wait times are avoided when performing a quick format.

Changing the state of DisableDeleteNotify does not require a host reboot to take effect.

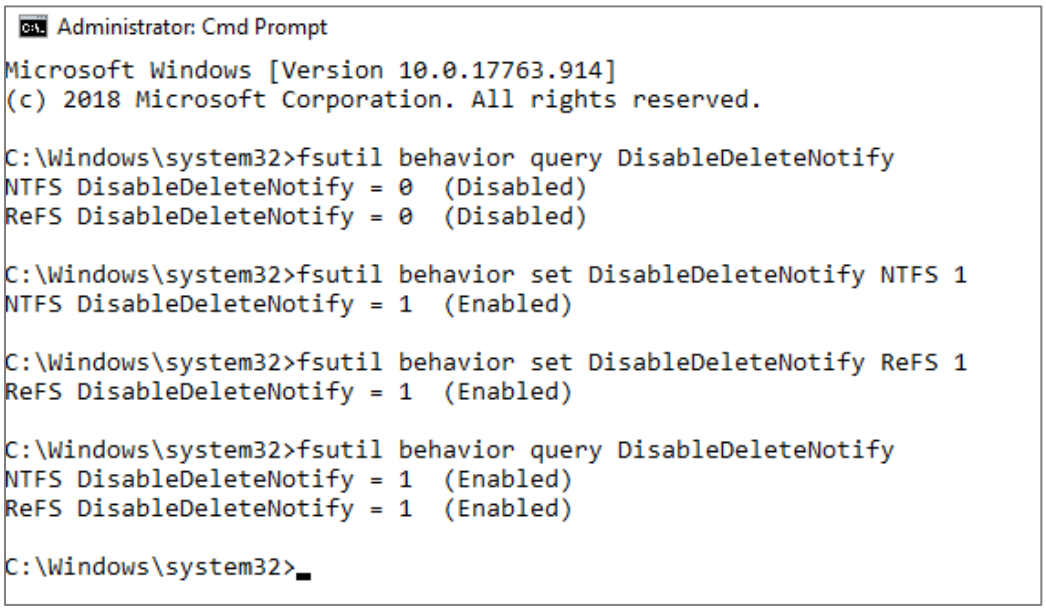

Figure 2 Use the fsutil command to query or change the state of trim/unmap

Once the volume is formatted, re-enable trim/unmap so the host can take advantage of deleted space reclamation for NTFS volumes.

For more background information about trim/unmap with PowerStore, see the document *Dell EMC PowerStore: Microsoft Hyper-V Best Practices* on Dell.com/StorageResources.

5.5 Thin clones

Thin clone uses a pointer-based technology to create a read/write copy of a volume, volume group, or file system. For data in a snapshot to be accessed, a thin clone must be created. Because the thin clone volume shares data blocks with the parent, the physical used space of the child volume is essentially just the delta changes from after it was created. Thin clones are advantageous in a SQL Server environment because a SQL Server database can be duplicated for testing purposes, all while consuming less storage. For example, if a SQL Server admin needs to clone a multi-terabyte database server for a developer to run some tests, the database can be isolated, tested, and only consume blocks that changed.

Within the PowerStore architecture, thin clones have several advantages for storage administrators.

- The thin clone can have a different data protection policy from the parent volume.
- The parent volume can be deleted, and the thin clones become their own resource.
- Clone databases for testing monthly patches or development.
- It is treated as a regular storage resource and therefore supports many data services such as protection policies, snapshot rules, or replication.
- Can be refreshed quickly to bring latest production changes to lower environment.
- Can be restored quickly to a baseline after testing.

When cloning databases for SQL Server, ensure all data and log volumes for the database are on the same volume or are part of the same volume group. See best practices for creating snapshots of SQL Server databases below. For more information, see the document *Dell EMC PowerStore: Snapshots and Thin Clones* on Dell.com/StorageResources.

5.6 Data encryption

Many SQL Server applications have data encryption requirements, specifically on data at rest. Data at Rest Encryption (D@RE) can be used as an encryption solution for SQL Server without requiring any database or application changes. This also avoids any potential performance impact to the database server or the applications and has no performance impact on the array.

D@RE is enabled by default on the PowerStore array, so no configuration steps are necessary to protect the drives.

5.7 Data reduction

PowerStore includes zero detection, compression, and deduplication integrated in the core product as part of the data reduction feature. While the amount of reduction varies depending on the dataset, the overall savings is similar to or better than SQL Server database compression. Since data reduction is always enabled on the array, SQL Server database compression and backup compression can typically be disabled, saving CPU resources on SQL Server.

When running SQL Server compute workloads on PowerStore X in AppsON mode, it is recommended to disable SQL Server compression to allow CPU resources to be allocated to other activities on the appliance.

For more information about the PowerStore data reduction benefits, see the document *PowerStore Data Efficiencies* on Dell.com/StorageResources

5.8 Scripting and automation

Scripting and automation are often used to improve quality of repetitive tasks and increase speed of deployment. The PowerStore platform supports several different tools for administrators who wish to automate management tasks or implementing Infrastructure as Code (IaC).

5.8.1 REST API

The PowerStore REST API provides a complete interface for those wanting to perform automation or custom applications for the PowerStore platform. Going to **https://<PowerStore Mgmt IP>/swaggerui** in a browser provides an interactive API reference for discovering available commands and viewing sample usage.

5.8.2 PowerStore CLI

The PowerStore CLI or PSTCLI is a command-line interface that is provided for managing PowerStore systems. The PSTCLI provides a convenient way to manage PowerStore without the complexity of using a programming interface. For more information, see the document *Dell EMC PowerStore Command Line Reference* on the PowerStore Info Hub.

5.8.3 Ansible

Ansible is a popular tool for automating IT infrastructure. Since Ansible is widely supported by various hardware and software platforms, it makes it ideal for deployment automation and IaC scenarios. The Ansible plug in for PowerStore as well as complete documentation can be found at GitHub.com/Dell/Ansible-PowerStore

6 Data protection and disaster recovery

PowerStore has integrated snapshot and replication capabilities to protect data, and is all policy driven for ease of administration.

6.1 Snapshots and recoveries

To automate and simplify protecting data, PowerStore uses protection policies. These protection policies help to protect data, set retention policies, and help guarantee recovery point objectives for an organization. Protection policies are a set of snapshot and replication rules that are applied to a volume or group of volumes.

In addition, protection policies can be applied to volume groups. By applying a protection policy to a volume group, it allows snapshots, replication, and recovery of the entire group. This allows the protection of a database that spans multiple volumes. To ensure successful recovery of SQL Server databases, ensure all database files, including the transaction log file, are in the same volume group and the write-order consistency option is enabled on the volume group.

6.2 Snapshots and application backup and restore

Using array-based snapshots is an effective way to protect virtual machine data and establish a recovery point objective (RPO). In the PowerStore architecture, the snapshot schedule is created using protection policies. Each protection policy can define snapshot rules to establish a schedule and retention, and replication rules to specify a destination array and RPO.

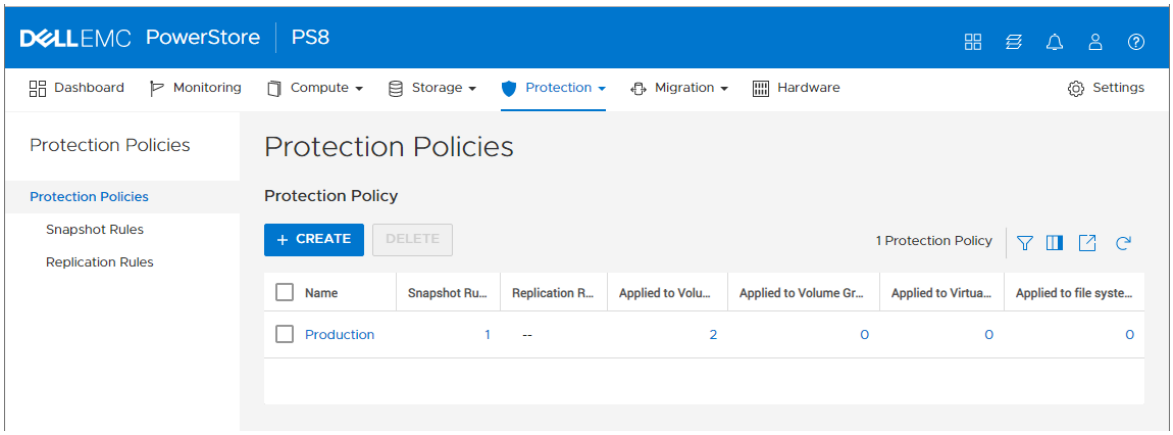

Figure 3 Protection policies screen in PowerStore Manager

PowerStore has three data recovery mechanisms that all behave differently depending on the usage scenario.

- **Thin Clone**: This takes an existing snapshot from a parent volume or volume group and creates a child volume or volume group from that point in time.
- **Refresh**: Using the refresh operation, snapshot data can replace existing data in the volume group. The existing data is removed, and snapshot data from the new source is copied to it in-place. A parent volume or volume group can refresh a child, and a child can refresh a parent.
- **Restore**: The restore operation replaces the contents of a parent storage resource with data from an associated snapshot. Restoring resets the data in the parent storage resource to the point in time the snapshot was taken.

Caution: Use of the refresh and restore operations on active database volumes may cause unexpected results and behaviors. All host access to the volume must be ceased before attempting these operations by either shutting down the SQL Server instance or taking the SQL Server databases offline.

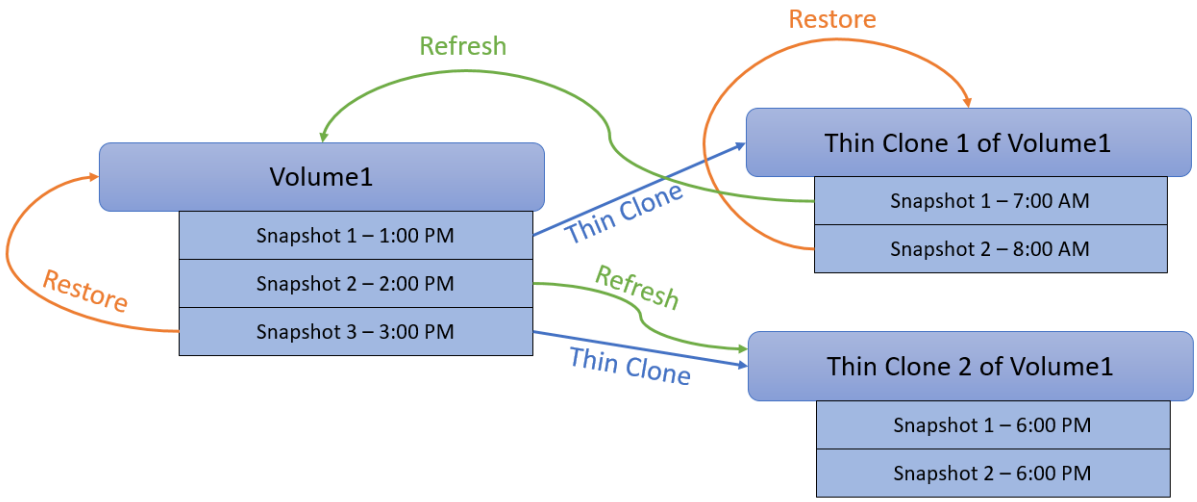

Figure 4 PowerStore snapshot and recovery anatomy

In addition to using snapshots to protect databases, if SQL Server hosts are leveraging boot-from-SAN, the boot volume can be added to the same volume group as the database volumes and the entire host can be protected.

6.3 Crash consistency and application consistency

When taking array-based snapshots of virtual machines, it is important to remember that snapshots that are taken without application coordination are considered crash consistent. Crash consistency is the storage term for data that has a snapshot taken in-flight without application awareness. While most modern applications can recover from crash-consistent data, their recovery can yield varying levels of success. For example, when recovering a Microsoft Windows virtual machine, as the operating system boots, it behaves like it experienced unexpected power loss, and potentially invoke check disk (chkdsk) on startup.

Dell EMC AppSync™ enables taking application-consistent snapshots. When using products such as AppSync, coordination between the array and the application can help to assure that the data is quiesced, the caches are flushed, and the data is preserved in a known-good state. Application consistent snapshots offer a higher probability of recovery success.

6.4 Replication and remote recovery

PowerStore offers asynchronous replication to transfer data to another PowerStore array. When the secondary array is at a different location, this can help to protect data from localized geographical disasters. Replication RPOs and options are set within protection policies.

6.5 Integrated copy data management

AppSync is software that enables integrated Copy Data Management (iCDM) with the Dell EMC primary storage systems, including PowerStore. AppSync, integrated with PowerStore snapshots, simplifies and automates the process of generating, consuming, and managing copies of production data. This solution

addresses copy management use cases in a consolidated SQL database environment for database recovery and database repurposing. AppSync automatically discovers application databases, learns the database structure, and maps it through the virtualization layer to the underlying PowerStore storage. Orchestration of all the activities required from copy creation and validation through mounting the snapshots at the target host and launching or recovering the application. This solution supports and simplifies Microsoft SQL Server workflows that include refreshing, expiring, and restoring the production database. Additional information about AppSync can be found in the documents AppSync Integration with Microsoft SQL Server and AppSync Performance and Scalability Guidelines.

A 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 Dell EMC PowerStore systems.

A.1 Related resources

- Storage Class Memory in SQL Server 2016 SP1
- Dell EMC NVDIMM-N Persistent Memory User Guide
- Dell EMC SQL Server Solutions
- Microsoft SQL Server
- GitHub.com/Dell/Ansible-PowerStore
- AppSync Integration with Microsoft SQL Server
- AppSync Performance and Scalability Guidelines
- VMware Documentation
- SQL Server on VMware Best Practices Guide