Table of Contents

1. Introduction
   1.1. Software Defined Storage
   1.2. Introducing Virtual Volumes
   1.3. Key Benefits

2. What's New in Virtual Volumes 2.0
   2.1. Virtual Volumes 2.0
   2.2. Array-Based Replication
   2.3. Introducing Lines of Service
   2.4. Oracle RAC Support

3. VVol Architecture
   3.1. VVol Components
   3.2. Architecture Comparison
   3.3. Ecosystem Support

4. Policy-Based Management and Automation
   4.1. Storage Policy-Based Management
   4.2. ESXCLI
   4.3. PowerCLI

5. Deployment and Operations
   5.1. vSphere Requirements
   5.2. Storage Requirements
   5.3. Interoperability

6. Summary
   6.1. Summary
   6.2. Reference
   6.3. Acknowledgements
   6.4. About the Author
1. Introduction

Introducing vSphere Virtual Volumes
1.1 Software Defined Storage

Software Defined Storage

VMware's Software-Defined Storage vision and strategy is to drive transformation through the hypervisor, bringing to storage the same operational efficiency that server virtualization brought to compute.

As the abstraction between applications and available resources, the hypervisor can balance all IT resources – compute, memory, storage and networking – needed by an application. With server virtualization as the de-facto platform to run enterprise applications, VMware is uniquely positioned to deliver Software-Defined Storage utilizing the pervasiveness of this software tier.

By transitioning from the legacy storage model to Software-Defined Storage with Virtual Volumes, customers will gain the following benefits:

- Automation of storage “class of service” at scale: Provision virtual machines quickly across data center using a common control plane (SPBM) for automation.
- Self-Service capabilities: Empower application administrators with cloud automation tool integration (vRealize Automation, PowerCLI, OpenStack).
- Simple change management using policies: Eliminate change management overhead and use policies to drive infrastructure changes.
- Finer control of storage class of service: Match VM storage requirements exactly as needed with class of service delivered per virtual disk.
- Effective monitoring/troubleshooting with per VM visibility: Gain visibility on individual VM performance and storage consumption.
- Non-disruptive transition: Use existing protocols (Fiber channel, ISCSI, NFS) across heterogeneous storage devices.
- Safeguard existing investment: Use existing resources more efficiently with an operational model that eliminates inefficient static and rigid storage constructs.

The goal of Software-Defined Storage is to introduce a new approach that enables a more efficient and flexible operational model for storage in virtual environments. This is accomplished in two ways:

1. The abstraction of the Virtual Data Plane enables additional functions that an array may provide to be offered as data services for consumption on a per-VM (or individual disk) basis. Current implementation, data services are bound to the array for the most part. Data Services can provide functionality such as compression, replication, caching, snapshots, de-duplication, availability, migration and data mobility, performance capabilities, disaster recovery, and other capabilities. While the data services may be instantiated at any level of the infrastructure, the virtualized data plane allows for these services to be offered via policy on a per-VM (or individual disk within a VM) basis.

2. Implementing an automation layer that enables dynamic control and monitoring of storage services levels to individual virtual machines across heterogeneous devices – VMware refers to this as the Policy-Driven Control Plane

Virtual Data Plane

The virtual data plane is responsible both for storing data and applying data services (compression, replication, caching, snapshots, de-duplication, availability, etc). While data services may be provided by a physical array or implemented in software, the virtual data plane abstracts the services and will present them to the policy-driven control plane for consumption and applies the resultant policy to the objects in the virtual datastore.

In today’s model, the data plane operates on rigid infrastructure-centric constructs (LUNs or storage volumes) that are typically static allocations of storage service levels (capacity, performance and data services), independently defined from applications.

In the VMware Software-Defined Storage model, the data plane is virtualized by abstracting physical
hardware resources and aggregating them into logical pools of capacity (virtual datastores) that can be more flexibly consumed and managed.

Additionally, to simplify the delivery of storage service levels for individual applications, the virtual data plane makes the virtual disk the fundamental unit of management around which all storage operations are controlled. As a result, exact combinations of data services can be instantiated and controlled independently for each VM. For each virtual machine that is deployed, the data services offered can be applied individually depending on the vendor implementation: Each application can have its own unique storage service level and capabilities assigned to it at its time of creation.

This allows for per-application storage policies, ensuring both simpler yet individualized management of applications without the requirement of mapping applications to broad infrastructure concepts like a physical datastore.

In the Software-Defined Storage environment, the storage infrastructure expresses the available data services and capabilities (compression, replication, caching, snapshots, de-duplication, availability, etc) to the control plane to enable automated provisioning and dynamic control of storage services levels through programmatic APIs. These storage services may come from many different locations: Directly from a storage array, from a software solution within vSphere itself, or from a third party location via API.

These capabilities are given to the control plane for consumption and expression by policies. The ability to pull in multiple sources of data services and abstract them to a policy engine gives the administrator the ability to create unique policies for each VM in accordance with their business requirements, consuming data services from different providers in each.

VMware’s implementation of the virtual data plane is delivered through Virtual Volumes for external SAN/NAS arrays and vSAN for x86 hypervisor-converged storage.

Policy-Driven Control Plane

In the VMware Software-Defined Storage model, the control plane acts as the bridge between applications and storage infrastructure. The control plane provides a standardized management framework for provisioning and consuming storage across all tiers, whether on external arrays, x86 server storage or cloud storage.

The policy-driven control plane is the management layer responsible for controlling and monitoring storage operations. In today’s model, the control plane is typically, tied to each storage device. Each array is operated in a different way and implements a “bottom-up” array-centric approach in which storage service levels are aggregated into physical tiers or “classes of services”, which are static pre-allocations of resources and data services tied to the infrastructure.

Upon provisioning, an application is rigidly mapped to these pre-configured storage containers. These storage containers are rarely aligned to precise application boundaries, and their capabilities need to be defined broadly to encompass the requirements of a broad set of applications.

This restricts the ability of a storage container to be focused specifically on the business requirements of an individual application. To circumvent this problem, storage administrators may be asked to create numerous purpose-built datastores, increasing the management overhead and complexity associated with storage.

Through Software-Defined Storage, the storage classes of service become logical entities controlled entirely by software and interpreted through policies. Defining and making adjustments to these policies enables automating the provisioning process at scale, while dynamically controlling individual service levels over individual virtual machines at any point in time.

This makes the Software-Defined Storage model able to more flexibly adapt to ongoing changes on specific application requirements. Policies become the control mechanism to automate the monitoring process and to ensure compliance of storage service levels throughout the lifecycle of the application.
The control plane is programmable via public APIs, used to consume and control policies via scripting and cloud automation tools, which in turn enable self-service consumption of storage to application tenants as well as a variety of external management frameworks.

VMware’s implementation of the policy-driven control plane is delivered through Storage Policy-Based Management (SPBM).

1.2 Introducing Virtual Volumes

**vSphere Virtual Volumes**

Virtual Volumes is a new virtual machine disk management and integration framework that exposes virtual disks as the primary unit of data management for storage arrays. This new framework implements the core tenants of the VMware Software-Defined Storage vision to enable a fundamentally more efficient operational model for external storage in virtualized environments, centering it on the application instead of the physical infrastructure.

Flexible consumption at the logical level

Virtual Volumes virtualizes SAN and NAS devices by abstracting physical hardware resources into logical pools of capacity (represented as Virtual Datastore in vSphere) that can be more flexibly consumed and configured to span a portion of, one or several storage arrays.

The Virtual Datastore defines capacity boundaries, access logic, and exposes a set of data services accessible to the virtual machines provisioned in the pool. Virtual Datastores are purely logical.
constructs that can be configured on the fly, when needed, without disruption and don’t require formatting with a file system.

**Finer control at the Virtual Disk level**

Virtual Volumes defines a new virtual disk container (the Virtual Volume) that is independent of the underlying physical storage representation (LUN, file system, object, etc.). In other terms, with Virtual Volumes the virtual disk becomes the primary unit of data management at the array level. This turns the Virtual Datastore into a VM-centric pool of capacity.

It becomes possible to execute storage operations with virtual disk granularity and to provision native array-based data services such as compression, snapshots, de-duplication, encryption, etc. to individual virtual machines. This allows admins to provide the right storage service levels to each individual virtual disk within a virtual machine.

To enable efficient storage operations at scale, even when managing thousands of virtual machines, Virtual Volumes uses vSphere Storage Policy-Based Management (SPBM). SPBM is the implementation of the policy-driven control plane in the VMware Software-Defined Storage model.

**Efficient operations through automation**

SPBM allows capturing storage service levels requirements (capacity, performance, availability, etc.) in the form of logical templates (policies) to which virtual machines are associated. SPBM automates virtual machine placement by identifying available datastores that meet policy requirements and coupled with Virtual Volumes, it dynamically instantiates necessary data services. Through policy enforcement, SPBM also automates service level monitoring and compliance throughout the lifecycle of the virtual machine.

The goal of Virtual Volumes is to provide a simpler operational model for managing virtual machines in external storage while leveraging the rich set of capabilities available in storage arrays. With Virtual Volumes, VMware seeks to transform a “bottom-up” operational model in which hardware is provisioned and then virtual machines are fit as best as possible, to a “top-down” operational model in which virtual machines requirements drive storage provisioning.

Virtual Volumes transform the data plane and control plane of supported SAN/NAS storage systems by aligning storage consumptions and operations with virtual machines. Virtual Volumes make supported SAN/NAS storage systems VM-aware and unlocks the ability to leverage array based data services and storage array capabilities with a VM-centric approach at the granularity of a single virtual disk.

With Virtual Volumes most of the data operations such as snapshot, cloning, and migrations are offloaded to the storage arrays. New data management and monitoring operations, as well as communication mechanism have been implemented to manage the required communications between vSphere, storage arrays and the Virtual Volumes.

This model eliminates the complexity of managing the storage infrastructure and its services independently from the virtual machine consumers, and introduces a new control plane (SPBM) that centers on the virtual machine. Each virtual disk within a virtual machine receives what it needs—no more no less—with a management framework that is common across heterogeneous storage systems.

**Traditional architecture**

In contrast, in the legacy model, each virtual machine needs to be mapped to a pre-defined storage tier and pre-configured storage container (LUN/Volume) and any virtual machine placed inside these structures will inherit its class of service. In the legacy model, it is common to manage different tiers of service using LUN/volumes.

For instance a “Gold LUN” would denote the highest level of service and any VMs placed inside this LUN would inherit the class of service of this LUN. In many cases, a LUN/volume with the right
combination of capacity and storage class of service may not be available. So, a virtual machine would be placed in a Gold LUN without requiring all the storage services. This leads to complexity in managing data services that are mandatorily inherited by all VMs in the Gold LUN: Perhaps not all systems need to be replicated but are nonetheless, as the Gold LUN is not only replicated but is also the highest performing, an attribute that is required by those systems. There is an operational cost associated with managing these interdependencies, as well as financial costs associated with the byproducts of over provisioning, or overconsumption of bandwidth, etc. Thus, transitioning from legacy storage model to Virtual Volumes is similar to moving from a fixed menu consumption model to an “à la carte” consumption model.

Systems often have changing requirements after being provisioned. Perhaps a given workload has end-of-month or seasonal performance spikes, or a less critical system has become mission-critical over time. In a traditional storage model it would require careful analysis of available capacity, LUN management, storage vMotion and possibly even re-provisioning of the application to meet the new requirements.

Policy-Based Management with Virtual Volumes

With Virtual Volumes and SPBM one can simply assign a new policy to the already deployed system, and the array will automatically move the system to a location where its new policy can be met or perhaps change the policy in-place, depending on the implementation. This allows for easy adaptation of even individual applications to meet changing business requirements simply by altering the policy and having the Virtual Volume respond.

Virtual Volumes enable rapid and dynamic provisioning of storage services to individual virtual machines at scale with a policy based approach that drives high levels of automation and simpler control.

In comparison, with the legacy model, hardware dictated virtual machine class of service and provisioning required manual operations causing slow deployment of virtual machines and a model that was not scalable.

LUNs and volumes, moreover, are pre-allocated storage containers whereas with Virtual Volumes, storage capacity and class of service are delivered dynamically on demand in real-time.

1.3 Key Benefits

Key Benefits

Virtual Volumes (VVols) is a new integration and management framework that virtualizes SAN/NAS arrays, enabling a more efficient operational model that is optimized for virtualized environments and centered on the application instead of the infrastructure. Virtual Volumes simplifies operations through policy-driven automation that enables more agile storage consumption for virtual machines and dynamic adjustments in real time, when they are needed. It simplifies the delivery of storage service levels to individual applications by providing finer control of hardware resources and native array-based data services that can be instantiated with virtual machine granularity.

With Virtual Volumes (VVols), VMware offers a new paradigm in which an individual virtual machine and its disks, rather than a LUN, becomes a unit of storage management for a storage system. Virtual volumes encapsulate virtual disks and other virtual machine files, and natively store the files on the storage system.
Faster Virtual Volumes map virtual disks and their respective components directly to objects, called virtual volumes, on a storage system. This mapping allows vSphere to offload intensive storage operations such as snapshot, cloning, and replication to the storage system.

In addition to faster storage operations, Virtual Volumes simplifies storage operations by automating manual tasks and eliminating operational dependencies between the vSphere Admin and the Storage Admin. Provisioning is faster, and change management is simpler as the new operational model is built upon policy driven automation.

Smarter In a traditional architecture the vSphere admin works with the Storage Admin to manually define the characteristics of each LUN/Volume to group VMs with similar requirements. This is a manual process with room for human error.

Using Storage Policy-Based Management with Virtual Volumes, vSphere admins predefine VM Storage Policies to meet the various needs of each application. Then each virtual machine created with a VM Storage Policy gets provisioned on the appropriate storage with the desired class of service described in the VM Storage Policy. If the needs of the VM change, the vSphere Admin simply changes the changes the policy assigned to the VM.

Virtual Volumes also improves resource utilization by enabling more flexible consumption of storage resources, when needed and with greater granularity. The precise consumption of storage resources eliminates over provisioning. The Virtual Datastore defines capacity boundaries, access logic, and exposes a set of data services accessible to the virtual machines provisioned in the pool.

Simpler For both the vSphere Admin and Storage Admin, Virtual Volumes greatly simplifies management over the existing operational model. Virtual Volumes allows the separation of provisioning and consumption of storage for VMs.

In a traditional architecture many standard size LUNs are preallocated and tied to individual VMFS datastores. Capacity and class of service is tied to the individual LUNS.

In the VMware Software-Defined Storage model with Virtual Volumes, instead of many pre-allocated LUNs, the Storage Admin sets up an entity called the storage container. The capacity and data services published from the storage array by the Storage Admin in the storage container become similar to menu items from which the vSphere Administrator can consume on demand. Because Storage Policy-Based Management can be assigned at the individual disk, many Virtual Machines with varying classes of service can reside in the same VVol Datastore (Storage Container).
The Storage Admin retains control of the storage resources, as the vSphere Administrator can only consume the published storage array capabilities. However, the Storage Admin no longer needs to determine which data services should be assigned to an application. Thus, the Storage Admin is responsible for up front setup, but the vSphere Admin is self-sufficient afterwards.

<table>
<thead>
<tr>
<th>Storage Admins</th>
<th>Current</th>
<th>With Virtual Volumes</th>
<th>Benefit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Datastore provisioning</td>
<td>Many standard size LUNs based policies – one for each application Capacity mgmt may be tied to LUNs Needs to format VMFS</td>
<td>Small number of datastores; but will need to be shared Capacity planning on a pool</td>
<td>Define capabilities to be consumed without static limitations Vendor mgmt. tools many need to be updated</td>
</tr>
<tr>
<td>Troubleshooting and Monitoring</td>
<td>Keyed off from datastores</td>
<td>Based on VM/VMDK</td>
<td>VM-awareness on storage systems Vendor mgmt tools many need to be updated</td>
</tr>
<tr>
<td>Provisioning</td>
<td>To datastores Using tag based policies</td>
<td>To datastore With granular policy and ability to dynamically change the policies</td>
<td>Storage consumption has been automated. No need to map VMs to LUNs</td>
</tr>
<tr>
<td>Troubleshooting and Monitoring</td>
<td>Per VM Datastore driven Manual process to ensure policy compliance</td>
<td>Per-VM Vendor container Automated monitoring of policy compliance</td>
<td>VM-awareness at storage Full visibility of policy compliance VMware tools could be made better with Virtual Volumes</td>
</tr>
</tbody>
</table>

Policy-driven automation enables more agile storage consumption for virtual machines, which ultimately delivers faster provisioning for new applications with different requirements and simplifies change management, as the vSphere Administrator no longer depends on the Storage Admin to fulfill infrastructure change requests. The vSphere Administrator can make changes to policies at any time, and the necessary infrastructure changes are configured through automation. This allows for quicker adjustment to business changes.
2. What’s New in Virtual Volumes 2.0

vSphere 6.5 introduces Virtual Volumes 2.0
2.1 Virtual Volumes 2.0

Whats New in Virtual Volumes

vSphere 6.0 (Virtual Volumes 1.0) introduced several key features to provide VMDK granular operations. It introduced a new storage paradigm through:

- New APIs that map VMDK objects and their derivatives (such as clones, snapshots, replicas) directly to the storage subsystem. This enabled vSphere to offload storage services such as snapshots, clones, backup to the storage
- New storage management entities called "Storage Containers" which allows for a flexible and easy way to manage storage capacity
- New profile-based storage management, wherein, admins can express the application needs in a granular fashion.

Since its release as part of vSphere 6.0 Virtual Volumes (VVols) momentum has been quietly building with more and more customer interest and partner activity. The adoption of VVols has steadily increased as more and more customers upgrade to vSphere 6, but one of the key inhibitors to widespread adoption had been its lack of support for array-based replication (ABR).

vSphere 6.5 Virtual Volumes aka VVol 2.0

The primary focus of VVol 2.0 is to extend the fine granular storage approach to provide data protection and recovery. Providing per VM/VMDK disaster recovery allows enterprise customers to deploy Virtual Volumes beyond test/dev workloads to all use cases. That said, Virtual Volumes 2.0 introduces support for array-based replication (ABR), granular lines of service, automated disaster recovery using public APIs and PowerCLI Cmdlets, and support for Oracle RAC.

Whats New in Virtual Volumes

vSphere 6.0 (Virtual Volumes 1.0) introduced several key features to provide VMDK granular operations. It introduced a new storage paradigm through:

- New APIs that map VMDK objects and their derivatives (such as clones, snapshots, replicas) directly to the storage subsystem. This enabled vSphere to offload storage services such as snapshots, clones, backup to the storage
- New storage management entities called "Storage Containers" which allows for a flexible and easy way to manage storage capacity
- New profile-based storage management, wherein, admins can express the application needs in a granular fashion.

Since its release as part of vSphere 6.0 Virtual Volumes (VVols) momentum has been quietly building with more and more customer interest and partner activity. The adoption of VVols has steadily increased as more and more customers upgrade to vSphere 6, but one of the key inhibitors to widespread adoption had been its lack of support for array-based replication (ABR). Enter vSphere 6.5.

vSphere 6.5 Virtual Volumes VVol 2.0

The primary focus of

2.2 Array-Based Replication
Array-Based Replication

vSphere 6.5 introduces VVols support of array-based replication, but unlike legacy array-based replication that required explicit placement on specific datastores to ensure a VM was replicated, VVol replication provides fine-grained control over VM replication. This means you have the flexibility to replicate a group of virtual machines (Replication Group) or you can replicate at an individual VM level.

VM Centric Replication

In a traditional storage environment, replication and subsequently failover happens at the datastore level, complicating VM placement when designing a DR strategy. This leads to significant operational and organizational challenges.

vSphere Virtual Volumes introduces the concept of Replication Groups, bringing more efficient, accurate, and responsive recovery of your virtual machines. A Replication Group is a group of replicated storage devices as defined by a storage administrator (perhaps acting on behalf of an application owner) to provide atomic failover for an application. In other words, a Replication Group is the minimum unit of failover. In vSphere 6.5, Replication Groups are created and managed by a storage administrator using the storage vendor’s tools.

How it works:

1. Vendor-specific replication capabilities are advertised up to vSphere via VASA.
2. VI administrators create VM storage policies containing replication capabilities from the storage system.
3. When VMs are being provisioned, the user:
   - Selects a policy containing the replication capabilities.
   - Chooses the compatible datastore.
   - Chooses a replication group in which to place the VM (to support multi-VM consistency).
   - Completes the provisioning.

The Replication Group that the user selects allows them to place multiple VMs intentionally in the same consistency group. The Replication Groups are advertised by the VASA provider. The storage system...
can also advertise a special “Automatic” replication group that will place the VM into an empty replication group by itself to support per-VM replication.

For discovery, configuration, failover and failback, SPBM provides public APIs for triggering these DR operations, there are also PowerCLI Cmdlets provided for administrator level orchestration.

VMware is working with our broad range of storage partners to introduce support for VVol replication.

2.3 Introducing Lines of Service

Introducing Lines of Service

Building upon the success of SPBM, VASA 3.0 introduces a new concept called “lines of service.” A line of service is a group of related capabilities with a specific purpose, such as inspection, compression, encryption, replication, caching, or persistence (storage). Now in addition to configuring replication at the individual SPBM policy level, I can create a line of service for replication and assign it to multiple policies. As an example, imagine I have 3 storage policies: Gold, Silver and Bronze. While these three categories have very different storage capabilities assigned, I can manage the replication once with a replication line of service instead of 3 times by configuring replication at the VM Storage policy level.

Lines of Service introduces a new UI to allow easier construction of reusable SPBM components i.e. define a “replication” component once and reuse in multiple policies. In the example below I have a VM Storage Policy that contains some QoS settings. I now can add additional components (or lines of service). As you can see there are currently two preconfigured replication policies (one to Anaheim and one to Boulder) These components can be applied to one or more VM Storage Policies.

2.4 Oracle RAC Support
Oracle RAC Support

In addition to all the array-based replication functionality, Virtual Volumes 2.0 is also now validated to support Oracle RAC workloads (similar to validation for VMFS) delivering policy-based, VM-centric storage for Business Critical Applications.

### Supported Versions

<table>
<thead>
<tr>
<th>vSphere Versions</th>
<th>Oracle RAC Versions</th>
</tr>
</thead>
<tbody>
<tr>
<td>vSphere 6.0</td>
<td>Not Supported</td>
</tr>
<tr>
<td>vSphere 6.5</td>
<td>Oracle RAC 11gR2 and 12cR1</td>
</tr>
</tbody>
</table>
To learn more about Virtual Volumes and how to plan, architect and administer a business-critical Oracle environment, see the *VMware vSphere Virtual Volumes: A Game Changer for Business-Critical Oracle Databases* white paper.

Also check out the following blog series covering Oracle on Virtual Volumes

- Part 1: [Virtual Volumes: A game changer for operations of virtualized business critical databases](#)
- Part 2: [Virtual Volumes for Database Backup and Recovery](#)
- Part 3: [Crash consistent backups and database cloning with Virtual Volumes](#)
- Part 4: [Virtual Volumes and Storage Policy-Based Management for Databases](#)
3. VVol Architecture

Architecture and VVol components
3.1 VVol Components

vSphere Virtual Volume Components

The following is a summarized description and definition of the key components of vSphere Virtual Volumes:

**Virtual Volumes (VVols)**

Virtual Volumes are a new type of virtual machine objects, which are created and stored natively on the storage array. VVs are stored in storage containers and mapped to virtual machine files/objects such as VM swap, VMDKs and their derivatives.

There are five different types of Virtual Volumes object types and each of them map to a different and specific virtual machine file.

- **Config-VVols**
  - Config – VM Home, Configuration files, logs
- **Data-VVols**
  - Data – Equivalent to a VMDK
- **Memory-VVols**
  - Memory – Snapshots
- **Swap-VVols**
  - SWAP – Virtual machine memory swap
- **Other-VVols**
  - Other – vSphere solution specific object

**VASA Provider (VP)**

VASA has introduced new APIs to support virtual volumes (VVols) starting with vSphere 6.0. These updated VASA APIs enhance storage system awareness of individual VM disks. This enables the storage system to perform operations on individual VM disks such as snapshots and clones.

The VASA provider (also known as the Vendor provider) is a storage side software component that acts as a storage awareness service for vSphere and mediates out of band communication between vCenter Server and ESXi hosts on one side and a storage system on the other. Storage vendors exclusively develop VASA providers.

ESXi hosts and vCenter Servers connect to the VASA Provider and obtain information about available storage topology, capabilities, and status. Subsequently vCenter Server provides this information to vSphere clients, exposing the capabilities around which the administrator might craft storage policies in SPBM.

VASA Providers are typically setup and configured by the vSphere administrator in one of two ways:

- Automatically via the array vendors plug in
- Manually through the vCenter Server
Storage Container (SC)

Unlike traditional LUN and NFS based vSphere storage, the Virtual Volumes functionality does not require preconfigured volumes on the storage side.

Instead, Virtual Volumes uses a storage container, which is a pool of raw storage capacity and/or an aggregation of storage capabilities that a storage system can provide to virtual volumes.

Depending on the storage array implementation, a single array may support multiple storage containers. Storage Containers are typically setup and configured by storage administrators.

Containers are used to define:

- Storage capacity allocations and restrictions
- Storage policy settings based on data service capabilities on a per virtual machine basis
Virtual Datastore

A Virtual Datastore (VVol Datastore) represents a storage container in a vCenter Server instance and the vSphere Web Client. A VVol Datastore represents a one-to-one mapping to the storage system’s storage container. The storage container represents a logical pool where individual Virtual Volumes and VMDKs are created. Virtual Datastores are typically setup and configured by vSphere administrators.
Protocol Endpoints (PE)

Although storage systems manage all aspects of virtual volumes, ESXi hosts have no direct access to virtual volumes on the storage side. Instead, ESXi hosts use a logical I/O proxy, called the protocol endpoint, to communicate with virtual volumes and virtual disk files that virtual volumes encapsulate. ESXi uses protocol endpoints to establish a data path on demand from virtual machines to their respective virtual volumes.

Each virtual volume is bound to a specific protocol endpoint. When a virtual machine on the host performs an I/O operation, the protocol endpoint directs the I/O to the appropriate virtual volume. Typically, a storage system requires a very small number of protocol endpoints. A single protocol endpoint can connect to hundreds or thousands of virtual volumes.

Protocol Endpoints are compatible with all SAN/NAS industry standard protocols:

- iSCSI
- NFS v3
- Fiber Channel (FC)
- Fiber Channel over Ethernet (FCoE)

ESXi uses protocol endpoints to establish a data path on demand from virtual machines to their respective virtual volumes. This binding operation happens when a VM is powered on and unbinds when powered off.
Protocol Endpoints are setup and configured by Storage administrators. To manage the properties, paths and datastores associated with a PE from the vSphere Web Client navigate to Manage > Storage > Protocol Endpoints, and select the endpoint to manage. For example, the multipathing policy of a PE could be modified from this location.

### 3.2 Architecture Comparison

**Architecture Comparison**

In traditional vSphere and storage virtualization architectures for virtual machines, the datastore is either a LUN formatted as VMware Virtual Machine File System (VMFS) or an NFS mount point. The datastore serves two purposes:

1. The endpoint to receive SCSI or NFS read and write commands
2. A storage container for a number of virtual machines metadata and data files.

In a Virtual Volumes architecture, the Protocol Endpoints act as the network transport mechanism, and the logically defined Storage Containers acts as virtual datastores. A default storage policy can be attached to a storage container and passed to each virtual machine located within.
LUN Centric Vs. Application-Centric

In a standard storage operating model, one may deploy a large number of virtual machines onto a single datastore to reduce the number of LUN or NFS entities on the storage fabric, and minimize the configuration and scalability overhead.

Yet this approach limits the granularity with which data services can be delivered to virtual machines and requires broadly similar profiles across all applications in order to be successful. This is commonly referred to as a LUN or volume centric approach to storage management.

Virtual Volumes is more application centric. Virtual machines within a storage container do not have to share a set of global properties defined by the membership in that storage container. Rather, each virtual machine can have an individual policy attached that can be persistent across storage containers. Because they are logically defined and not tied to a static LUN or disk group, they can instantiate a virtual machine anywhere with a container that will satisfy the policy attached to it.

This application-centric approach allows virtual machines (or individual disks within a VM) to have their own storage policy, and be placed appropriately within the container alongside other virtual machines, each with their own policy that is not mandatorily shared by all VMDKs stored within the container.
<table>
<thead>
<tr>
<th>vSphere feature</th>
<th>Non-VVol SAN/NAS</th>
<th>Virtual Volumes enabled SAN/NAS</th>
</tr>
</thead>
<tbody>
<tr>
<td>VMFS</td>
<td>Clustered file system used to format LUN</td>
<td>Not needed</td>
</tr>
<tr>
<td>vSphere API for Storage Awareness (VASA)</td>
<td>Storage capabilities are exposed to the virtualization layer in the form of read-only profiles associated with datastores. This capability is currently available in vSphere through a feature called–Profile-driven storage. In the current implementation profiles are used as tags of virtual disks and datastores for simplified provisioning. They are not used to control the instantiation of individual data services. For instance, if replication is set to “on” at the LUN, this capability cannot be modified from the VM policy.</td>
<td>Storage Capabilities can be controlled and instantiated through active policy profiles associated with individual VM objects without the need of LUNs or volumes. For instance, array replication can be set to “on” or “off” as required by the virtual machine. VVol uses VASA 2.0 and VASA 3.0</td>
</tr>
<tr>
<td>vSphere API for Array Integration (VAAI)</td>
<td>Limited set of Block, NAS, and thin provisioning primitives.</td>
<td>Integrated with VVols.</td>
</tr>
<tr>
<td>vSphere Profile-Driven Storage</td>
<td>Allows tagging of virtual disks and based on read-only static profiles.</td>
<td>Evolves into vSphere Storage Policy-Based Management. SPBM will continue to support tagging for traditional storage as well as policy-based automation for Software-Defined Storage.</td>
</tr>
<tr>
<td>Storage DRS</td>
<td>Performed at LUN/volume level.</td>
<td>When supported by storage vendors, it will be performed at the Virtual Volumes level. With Storage DRS, an application or VM is moved to meet requirements.</td>
</tr>
<tr>
<td>Tiering</td>
<td>Tiers determined by the LUN level (e.g.gold, silver, bronze) and bound by the LUN size limit.</td>
<td>Determined by datastore capabilities provided by vendors. Datastores offer a wide range of capabilities and from this universe of capabilities, many combinations can be derived to create a policy. Capabilities are specific to vendor implementations.</td>
</tr>
</tbody>
</table>
Storage IO Control
Adjusted by administrators to control VM network loads. When configured, it will allow end users to establish performance thresholds at the Virtual Volume level.

Virtual Volumes does not replace VMFS or other existing file system and storage access protocols that are supported by vSphere. The use of VMFS is still available and supported whenever needed in vSphere. The vSphere platform will be able to simultaneously connect both to Virtual Volumes and to traditional LUN based SAN/NAS arrays.

Depending on vendor implementation, a single array could be capable of supporting both VVols and LUN based operations. VVols present several advantages over traditional storage solutions.

3.3 Ecosystem Support

vSphere Virtual Volumes Ecosystem Support

Virtual Volumes is an industry-wide initiative that will allow customers to leverage the unique capabilities of their current storage investments and transition without disruption to a simpler and more efficient operational model optimized for virtual environments that works across all storage types.

Check the VMware Compatibility Guide for up to date information on VVol supported storage arrays. There are currently over 30 certified storage partners and 170 arrays models listed.

Certified Partners with Solutions
4. Policy-Based Management and Automation

Storage Policy-Based Management, ESXCLI and PowerCLI
4.1 Storage Policy-Based Management

Traditional Storage Management

Traditional storage models utilize LUNs or volumes. A LUN or a volume is commonly configured with a specific disk configuration such as RAID to provide a specific level of performance and availability. The challenge with this model is each LUN or volume is confined to providing only one level of service regardless of the workloads that it contains. This leads to the provisioning of numerous LUNs or volumes in an attempt to provide the right levels or storage services to each workload. Maintaining a large number of LUNs or volumes leads to management complexity. Deployment and management of workloads and storage in traditional storage environments can be time consuming and error prone.

Storage Policy-Based Management

VMware’s implementation of the policy-driven control plane is delivered through Storage Policy-Based Management (SPBM). SPBM enables precise control of the storage services. This management framework is used by VMware vSAN (vSAN), vSphere Virtual Volumes (VVols) and vSphere APIs for IO Filtering (VAIo).

Storage Policy-Based Management (SPBM) allows capturing storage service levels requirements (capacity, performance, availability, etc.) in the form of logical templates (policies) to which VMs are associated. SPBM automates VM placement by identifying available datastores that meet policy requirements and coupled with Virtual Volumes, it dynamically instantiates necessary data services.

VM Storage Policies can easily be changed and/or reassigned if application requirements change. These changes are performed with no downtime and without the need to migrate (Storage vMotion) virtual machines from one LUN or volume to another. This approach makes it possible to assign and modify service levels based on specific application needs even though the virtual machines reside on the same datastore.
The image below displays a VM Storage Policy for Virtual Volumes. The storage characteristics presented in the VM Storage Policies vary depending on the Storage Vendor’s VASA Provider. In the example below, Nimble is offering data encryption, deduplication, and VVol replication.

<table>
<thead>
<tr>
<th>Virtual Machine Storage policy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reserve Capacity</td>
</tr>
<tr>
<td>Deduplication</td>
</tr>
<tr>
<td>Disk Type</td>
</tr>
<tr>
<td>Disk Encryption</td>
</tr>
</tbody>
</table>
The image below displays a VM Storage Policy for vSAN.
Introducing SPBM Components

Building upon the success of SPBM, VMware delivers additional data services via SPBM and the VAIO framework. These features include VMcrypt, a per-VM encryption capability provided by vSphere, replication groups, and re-implementing Storage I/O control to provide the ability to limit and prioritize I/O traffic from VMs.
4.2 ESXCLI

vSphere Virtual Volumes CLI Commands

The esxcli command line framework has been updated to include a Virtual Volume module. All of the new Virtual Volumes esxcli commands are grouped under the storage, VVol namespace.

VVols ESXCLI Namespaces

The VVols namespace contains multiple sets of commands. Each namespace is focused on a different operating function of the Virtual Volumes related components. There are five available namespaces and their respective description are listed below:

- **storagecontainer** – Operations to create, manage, remove VVol Storage Containers
- **deamon** – Operations pertaining to VVol deamon
- **protocolendpoint** – Operations on VVol Protocol Endpoints.
- **vasacontext** – Operations on the VVol and VASA context.
- **vasaprovider** – Manage VVol VASA Provider Operations

Some esxcli storage VVol command line syntax samples:

- esxcli storage vvol –h
- Usage: esxcli storage vvol {cmd} [cmd options]

Available Namespaces:

abandonedvvol Operations on Abandoned Virtual Volumes.

Available Commands:

list List the VVol StorageContainers currently known to the ESX host.
abandonedvvol – is a state in which VVols are placed whenever a failure to delete event happens. i.e. failure to delete swap vvol during a VM power-off operation thought a particular path. This behavior typically happens when there are communication issues with the Vendor/VASA Provider.

In this scenario instead of failing the VM power-off operation, the system makes note of that VVol on a per-VM-namespace basis onto an abandon VVols tracking file so that it could be deleted later when the Vendor/VASA Provider is back online. A periodic thread tries to delete such abandoned VVols.

can option – this option allows the initiation of a background scan of a respective VVol datastore, searching for abandoned VVols. The operation goes over all the Config-VVols, looking for the abandoned VVols tracking files and tries to delete them.

The successful initiation of the scan doesn’t indicate that the operation succeeded or failed. This is a long running operation that might take long time to complete, as we don’t scan all the config-VVols at once to avoid putting load on the Vendor/VASA Provider for a non-important operation like garbage collecting old VVols.

esxcli storage vvol storagecontainer abandonedvvol syntax samples:

```
esxcli storage vvol storagecontainer abandonedvvol scan -p eqlDatastore true
```

List – provides the ability to display or list the number of virtual datastores and details for VVols and that are know to a particular ESX host.

esxcli storage vvol storagecontainer list syntax sample:

```
esxcli storage vvol storagecontainer list eqlDatastore
```

- StorageContainer Name: eqlDatastore
- UUID: vvol:6090a0681067ae78-2e48c5020000a0f6
- Array: com.dell.storageprofile.equallogic.std:eqlgrp1
- Size(MB): 1048590
- Free(MB): 972540
- Accessible: true
- Default Policy:

- StorageContainer Name: engDatastore
- UUID: vvol:6090a06810770d5b-cd4ad5d7a1042074
- Array: com.dell.storageprofile.equallogic.std:eqlgrp1
- Size(MB): 4194315
- Free(MB): 4173930
- Accessible: true
- Default Policy:

- StorageContainer Name: dbDatastore
- UUID: vvol:6090a0681077bdce-8b4b1515a2049013
- Array: com.dell.storageprofile.equallogic.std:eqlgrp1
- Size(MB): 1024005
- Free(MB): 1009635
- Accessible: true
- Default Policy:

The daemon in the namespace – Is utilized to perform unbind virtual volume operations from all Vendor/VASA Provider that are known to a particular ESX host.
unbindall – this option is utilized to unbind all VVols from all the Vendor/VASA Provider known to a particular ESXi Host. This operation is performed for testing purposes or to force the cleanup of all VVols data path.

**esxcli storage vvol storagecontainer daemon unbindall syntax sample:**

Usage: esxcli storage vvol daemon unbindall [cmd options]

Description:

unbindall          Unbind all virtual Volumes from all VPs known to the ESX host.

Cmd options:

esxcli storage vvol daemon unbindall

The **protocolendpoint** namespace commands provide the ability to list the all the information with regards to the Protocol Endpoints configuration to an ESXi host.

**List** – provides the ability to display or list the number of protocol endpoints and their configuration details to a particular ESX host.

**esxcli storage vvol protocolendpoint**

Usage: esxcli storage vvol protocolendpoint {cmd} [cmd options]

Available Commands:

list              List the VVol Protocol EndPoints currently known to the ESX host.

esxcli storage vvol protocolendpoint list

naa.6090a0681077ad11863e05020000a061

- Host Id: naa.6090a0681077ad11863e05020000a061
- Array Id: com.dell.storageprofile.equallogic.std:eqlgrp1
- Type: SCSI
- Accessible: true
- Configured: true
- Lun Id: naa.6090a0681077ad11863e05020000a061
- Remote Host:
- Remote Share:
- Storage Containers: 6090a068-1067-ae78-2e48-c5020000a0f6

The **vasacontext** namespace command provide the ability to get the vCenter Server UUID for which the Vendor/VASA Provider is currently registered.

**get** - this option is utilized to get the VVOL VASA context or vCenter Server UUID’s.

**esxcli storage vvol vasacontext –h**

Usage: esxcli storage vvol vasacontext {cmd} [cmd options]

Available Commands:

get                Get the VVol VASA Context (VC UUID).

esxcli storage vvol vasacontext get

5742ead8-0695-48bd-9ae4-7416164423ef

The **vasaprovider** namespace command provides the ability to list the Vendor/VASA Providers that are currently registered onto a particular ESXi host.
**list** - this option is utilized to list all the Vendor/VASA Providers and their information details that are registered to a particular ESX host.

```bash
esxcli storage vvol vasaprovider -h
```

Usage: `esxcli storage vvol vasaprovider {cmd} [cmd options]`

Available Commands:

- list  List the VASA Providers registered on the host.

```bash
esxcli storage vvol vasaprovider list
```

Dell Equallogic VASA Provider

- VP Name: Dell Equallogic VASA Provider
- URL: https://10.144.106.39:8443/vasa-version.xml
- Status: online
- Arrays:
  - Array Id: com.dell.storageprofile.equallogic.std:eqlgrp1
  - Is Active: true
  - Priority: 0

### 4.3 PowerCLI

**PowerCLI 6.5 for Virtual Volumes**

PowerCLI Cmdlets are broadly used across a range of customers and is the predominant administration automation interface used by vSphere administrators. Automation via administrative interfaces like PowerCLI reduces the operational overhead of managing storage in a vSphere environment and enables agility in operating the infrastructure.

VMware vSphere and PowerCLI 6.5 are the first versions to support array-based replication (ABR) for Virtual Volumes replicated by a VASA 3 compliant VASA Providers. Now using the public API and PowerCLI, administrators can discover and configure policy driven replication and trigger recovery workflows using SPBM and VASA.

#### Replication Discovery

<table>
<thead>
<tr>
<th>Cmdlet Name</th>
<th>Use Case</th>
<th>Parameter Info</th>
<th>Example</th>
</tr>
</thead>
</table>

Copyright © 2017 VMware, Inc. All rights reserved.
<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
<th>Parameters</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>Get-SpbmFaultDomain</td>
<td>Get all fault domain (info objects) available at the current site</td>
<td></td>
<td>• Get-SpbmFaultDomain</td>
</tr>
<tr>
<td>Get-SpbmFaultDomain</td>
<td>Get fault domain (info objects) by specifying the name. This parameter should support wildcard.</td>
<td>[-Name]</td>
<td>• Get-SpbmFaultDomain &quot;FD-Anaheim&quot;</td>
</tr>
<tr>
<td>Get-SpbmReplicationGroup</td>
<td>Get replication group (info objects - same for all use cases below) by specifying the name. This parameter should support wildcard.</td>
<td>[-Name]</td>
<td>• Get-SpbmReplicationGroup &quot;RG-123456&quot;</td>
</tr>
<tr>
<td>Get-SpbmReplicationGroup</td>
<td>Get replication group (info objects - same for all use cases below) by specifying the name. This parameter should support wildcard.</td>
<td>[-Name]</td>
<td>• Get-SpbmReplicationGroup &quot;RG-*&quot;</td>
</tr>
<tr>
<td>Retrieve all replication groups in a given fault domain</td>
<td></td>
<td>-FaultDomain</td>
<td>• $faultDomain = Get-SpbmFaultDomain &quot;FD-Anaheim&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• Get-SpbmReplicationGroup -FaultDomain $faultDomain</td>
</tr>
</tbody>
</table>
### Retrieve all replication groups replicated between specific fault domains

| Retrieve all replication groups replicated between specific fault domains | -FaultDomain - ReplicationPeerFaultDomain [-IsReplicationSource | -IsReplicationTarget] | -FaultDomain1 = Get-SpbmFaultDomain “FD-Anaheim”
-FaultDomain2 = Get-SpbmFaultDomain “FD-Boulder”
Get-SpbmReplicationGroup -FaultDomain $faultDomain1 - ReplicationPeerFaultDomain $faultDomain2 -IsReplicationSource |

### Retrieve peer replication group(s) for a given replication group

| Retrieve peer replication group(s) for a given replication group | -ReplicationPeerGroup | Get-SpbmReplicationGroup -ReplicationPeerGroup $replicationGroup |

### Retrieve replication groups that support a given policy and datastore (VVol use case)

| Retrieve replication groups that support a given policy and datastore (VVol use case) | -Policy -Datastore | $policy = Get-SpbmStoragePolicy –Name "Policy With VVol Replication"
$datastore = Get-SpbmCompatibleStorage -StoragePolicy $policy | select –first 1
Get-SpbmReplicationGroup -Policy $policy -Datastore $datastore |

### Retrieve replication groups that support a given policy and compute resource (VAIO use case)

| Retrieve replication groups that support a given policy and compute resource (VAIO use case) | -Policy -Cluster | $policy = Get-SpbmStoragePolicy –Name "Policy With VAIO Replication"
$cluster = Get-Cluster "Cluster-B" Get-SpbmReplicationGroup -Policy $policy -Cluster $cluster |

### Retrieve replication groups containing given Virtual Machine home object(s)

| Retrieve replication groups containing given Virtual Machine home object(s) | -VM | $vms = Get-VM "u12-*" Get-SpbmReplicationGroup -VM $vms |
Retrieve replication groups containing given Virtual Disk(s)

- Disk

-IsFailedOver

Get-VM

Retrieve Virtual Machine(s) associated with the given replication group

- ReplicationGroup

Get-HardDisk

Retrieve Virtual Disk(s) associated with the given replication group

- ReplicationGroup

Disaster Recovery Workflows

<table>
<thead>
<tr>
<th>Cmdlet Name</th>
<th>Use Case</th>
<th>Parameter Info</th>
<th>Example</th>
</tr>
</thead>
</table>
| Start-SpbmReplicationPrepareFailover | Prepare to fail over the specified replication group. This command should be executed on source site only | - ReplicationGroup                            | • $replGroup = Get-SpbmReplicationGroup "RG-123"
  • $prepareTask = Start-SpbmReplicationPrepareFailover –ReplicationGroup $replGroup
  • Wait-Task -Task $prepareTask |
### Start-SpbmReplicationSync

Synchronize the replication group. This command should be executed on target site only.

- $replGroup = Get-SpbmReplicationGroup "RG-456"
- $syncTask = Start-SpbmReplicationSync – ReplicationGroup $replGroup
- Wait-Task -Task $syncTask

### Start-SpbmReplicationFailover

Fail over the specified replication group. This command should be executed on target site only.

- $replGroup = Get-SpbmReplicationGroup "RG-456"
- $failoverTask = Start-SpbmReplicationFailover – ReplicationGroup $replGroup – Planned $true
- $failoverResult = Wait-Task -Task $failoverTask

### Start-SpbmUpdateVVolVirtualMachineFiles

Patch VVol based virtual machine files for the specified replication group. This cmdlet returns VMX files, which can be used to register the VMs.

- $updateTask = Start-UpdateVVolVirtualMachineFiles –failoverResult $failoverResult
- $vmxFiles = Wait-Task $updateTask

  $EsxiCluster = Get-Cluster "Cluster-A"
  $VMFolder = Get-Folder "DR Test"
  foreach ($vmxFile in $vmxFiles) {
    New-VM -VMFilePath $vmxFile -ResourcePool $EsxiCluster -Location $VMFolder -RunAsync
  }

---

### Failback

<table>
<thead>
<tr>
<th>Cmdlet Name</th>
<th>Use Case</th>
<th>Parameter Info</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Reverse replication direction for a replication group and its associated Virtual Machines - making the currently FAILEDOVER devices as sources.

Optionally, after the reverse operation completes, user can perform a full synchronization to sync the data back to source.

- ReplicationGroup

$replGroup = Get-SpbmReplicationGroup "RG-456" -IsFailedOver
$reverseTask = Start-SpbmReplicationReverse -ReplicationGroup $replGroup
Wait-Task -Task $reverseTask
$targetFaultDomain = Get-SpbmFaultDomain "FD-123456"
-syncTask = Start-SpbmReplicationSync -ReplicationGroup $targetGroup
Wait-Task -Task $syncTask

New Data Object Types

Virtual Volume array-based replication introduces some new data objects to support above replication cmdlets.

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Properties</th>
</tr>
</thead>
<tbody>
<tr>
<td>FaultDomainId</td>
<td>Unique identifier of a replication fault domain</td>
<td>• String id: fault domain identifier</td>
</tr>
</tbody>
</table>

Copyright © 2017 VMware, Inc. All rights reserved.
<table>
<thead>
<tr>
<th>Table Title</th>
<th>Description</th>
<th>Details</th>
</tr>
</thead>
</table>
| **FaultDomainInfo**    | Detailed information about a fault domain                                                               | • String name: fault domain name  
• String description: fault domain description  
• String storageArrayId: identifier of the storage array that this fault domain belongs to  
• FaultDomainId[] children: children of the fault domain  
• Provider provider: VASA provider that is actively managing the fault domain |
| **DeviceGroupId**      | Identifier of a replication device group                                                                 | • String id: device group identifier                                                                                                                                                                  |
| **ReplicationGroupId** | Identifier of a replication group                                                                       | • FaultDomainId faultDomainId: identifier of the fault domain to which the replication group belongs  
• DeviceGroupId deviceGroupId: identifier of the replication device group                                                                                                                                 |
| **SourceGroupInfo**    | Information about the source replication group                                                          | • ReplicationGroupId groupId: identifier of the replication group  
• String name: name of replication group  
• String description: description of the replication group  
• String state: state of the source replication group  
• ReplicationTargetInfo[] replica: information about the target replication groups  
• SourceGroupMemberInfo [] memberInfo: information about the member virtual volumes and their replicas |
<table>
<thead>
<tr>
<th>Class Name</th>
<th>Description</th>
<th>Properties</th>
</tr>
</thead>
<tbody>
<tr>
<td>TargetGroupInfo</td>
<td>Information about the target replication group</td>
<td>• ReplicationGroupId groupId: identifier of the replication group</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• TargetToSourceInfo sourceInfo: replication source information</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• String state: state of the target replication group</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• TargetGroupMemberInfo [] devices: member device information</td>
</tr>
<tr>
<td>GroupOperationResult</td>
<td>Base class for the result of any operation on a replication group</td>
<td>• ReplicationGroupId groupId: identifier of the replication group</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• MethodFault[] warning: warnings related to this replication group</td>
</tr>
<tr>
<td>GroupErrorResult</td>
<td>Error result for any operation on a replication group</td>
<td>• MethodFault[] warning: errors related to this replication group</td>
</tr>
<tr>
<td>SyncReplicationGroupSuccessResult</td>
<td>Result of a successful sync operation</td>
<td>• DateTime timeStamp: Creation time of the PIT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• PointInTimeReplicaId pitId: identifier of the PIT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• String description: description of the PIT</td>
</tr>
<tr>
<td>FailoverSuccessResult</td>
<td>Result of a successful failover operation</td>
<td>• String newState: new state of the replication after failover</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• PointInTimeReplicaId pitId: identifier of the PIT snapshot used during failover</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• PointInTimeReplicaId pitIdBeforeFailover: identifier of the PIT snapshot that was</td>
</tr>
<tr>
<td></td>
<td></td>
<td>automatically created before failover</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• RecoveredDevice[] recoveredDeviceInfo: recovered devices</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• DateTime timeStamp: time stamp of the failover</td>
</tr>
</tbody>
</table>
| ReverseReplicationSuccessResult | Result of a successful reverse replication operation | • DeviceGroupId
newGroupId: identifier of the newly created source replication group |
5. Deployment and Operations

Requirements for vSphere and storage and interoperability
5.1 vSphere Requirements

vSphere Virtual Volumes Requirements

Software
The use of Virtual Volumes requires the following software components:

- vCenter Server Appliance (VCSA) 6.0, 6.5 or vCenter Server for Windows 6.0, 6.5
- ESXi 6.0, 6.5
- vSphere Web Client

Hardware
The use of vSphere Virtual Volumes requires the following hardware components:

- Any Server that is certified for vSphere 6.0 or 6.5 that is listed on the VMware compatibility guide.
- A third party storage array system that supports vSphere Virtual Volumes and able to integrate with vSphere through the VMware APIs for Storage Awareness.
- Depending on the vendor specific implementation, storage array system may or may not require a firmware upgrade in order to support vSphere Virtual Volumes. Check with your storage vendor for detailed information and configuration procedures.

License
The use of vSphere Virtual Volumes requires the following license:

- Standard
- Enterprise Plus

5.2 Storage Requirements

Storage Arrays
Virtual Volumes requires a compatible storage array system. In most cases, a software solution such as a virtual storage appliance from one of the supporting vendors is supported for testing management workflows, operations, and functionalities.

Depending on the vendor specific implementation, storage array system may or may not require a firmware upgrade in order to support vSphere Virtual Volumes. Check with your storage vendor for detailed information and configuration procedures.

Check the VMware Compatibility Guide for up to date information on VVol supported storage arrays. There are currently over 30 certified storage partners and 170 arrays models listed.
5.3 Interoperability

**vSphere Virtual Volumes Interoperability**

**vSphere Enterprise Features**

The following table highlights the vSphere enterprise features that are available in the vSphere platform which are supported with Virtual Volumes.

<table>
<thead>
<tr>
<th>vSphere Features</th>
<th>vSphere 6.0</th>
<th>vSphere 6.5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Storage Policy Based Management</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Thin Provisioning</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Linked Clones</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Snapshots</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>NFSv3.x</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>View Storage Accelerator (CBRC)</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>vMotion</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>
### VMware Products and Solutions

The following table highlights the products and solutions which provide support for Virtual Volumes.

<table>
<thead>
<tr>
<th>VMware Products and Solutions</th>
<th>vSphere 6.0</th>
<th>vSphere 6.5</th>
</tr>
</thead>
<tbody>
<tr>
<td>vCloud Director (vCD)</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>vSphere Data Recovery (VDR)</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>vSphere Data Protection (VDP)</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>vSphere Data Protection Advanced (VDPA)</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>vSphere Data Development Kit (VDPA)</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>View Storage Accelerator 6.x (CBRC)</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>vRealize Operations 6.x (vROPS)</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>vRealize Automation 6.x (vRA)</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>VMware vSAN 6.x (vSAN)</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>VMware vCenter Site Recovery Manager</td>
<td>Only using vSphere Replication</td>
<td>Only using vSphere Replication</td>
</tr>
<tr>
<td>-------------------------------------</td>
<td>-------------------------------</td>
<td>-------------------------------</td>
</tr>
<tr>
<td>vSphere Metro Storage Cluster (vMSC)</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>VMware PowerCLI</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>
6. Summary

Summary and VVol reference guide
6.1 Summary

Summary

Virtual Volumes and SPBM bring Software-Defined Storage benefits to SAN/NAS arrays by enabling policy-driven provisioning and control of storage services to individual VMs leveraging native array capabilities.

Virtual Volumes and SPBM expose virtual disks as native storage containers and enable granular array-based operations at the virtual disk level. Through automation and by establishing the VM as the unit of data, storage workflows are streamlined and underlying arrays are utilized more efficiently.

Virtual Volumes 2.0 in vSphere 6.5 places a strong emphasis on disaster recovery by offering support for array-based replication, and automated policy-driven DR using public APIs and PowerCLI 6.5 Cmdlets.

Virtual Volumes is an industry wide initiative, supported by all major vendors and will make provisioning of SAN/NAS arrays in vSphere environments more agile, simpler, and will reduce storage costs.

6.2 Reference

vSphere Virtual Volumes Resources

vSphere Virtual Volumes Product Page

Product Documentation

vSphere Virtual Volumes Solution Overview

vSphere Virtual Volumes FAQ

VMware vSphere Virtual Volumes: A Game Changer or Business-Critical Oracle Databases

VMware Blogs

vSphere Virtual Volumes
http://blogs.vmware.com/vsphere/2015/02/vsphere-virtual-volumes.html

What's New in Virtual Volumes 2.0
Backing up Virtual Volumes

https://blogs.vmware.com/virtualblocks/2016/04/19/backing-up-virtual-volumes/

vSphere Virtual Volumes Interoperability: VAAI APIs vs VVOLs


Also check out the following blog series covering Oracle on Virtual Volumes

- Part 1: Virtual Volumes: A game changer for operations of virtualized business critical databases
- Part 2: Virtual Volumes for Database Backup and Recovery
- Part 3: Crash consistent backups and database cloning with Virtual Volumes
- Part 4: Virtual Volumes and Storage Policy-Based Management for Databases

VMworld Presentations

vSphere Virtual Volumes Technical Deep Dive – STO7645

https://www.youtube.com/watch?v=iqkQXaFr-to

The Virtually Speaking Podcast

The Virtually Speaking Podcast is a weekly technical podcast dedicated to discussing VMware topics related to storage and availability. Each week Pete Flecha (@vPedroArrow) and John Nicholson (@Lost_Signal) bring in various subject matter experts from VMware and within the industry to discuss their respective areas of expertise. If you’re new to the Virtually Speaking Podcast check out all episodes on vSpeakingPodcast.com. Be sure to subscribe to the podcast on iTunes and Soundcloud.

6.3 Acknowledgements

Acknowledgements

This document is a collection of information from various sources within VMware. I would like to thank Ben Meadowcroft Product Manager for Virtual Volumes, Patrick Dirks, Virtual Volumes Sr. Dev Manager, and his team of rockstar developers. Virtual Volumes is vital part of the Software Defined Datacenter story at VMware thanks to each of you.

6.4 About the Author

About the Author

Pete Flecha is a Senior Technical Marketing Architect in the Storage and Availability Business Unit at VMware, Inc. focused on Software-Defined Storage technologies such as VMware vSAN, vSphere Virtual Volumes and other storage related products. Pete is also the host of the Virtually Speaking Podcast.

The Virtually Speaking Podcast is a weekly technical podcast dedicated to discussing VMware topics related to storage and availability. Each week Pete Flecha (@vPedroArrow) and John Nicholson (@Lost_Signal) bring in various subject matter experts from VMware and within the industry to discuss...
their respective areas of expertise. If you're new to the Virtually Speaking Podcast check out all episodes on vSpeakingPodcast.com. Be sure to subscribe to the podcast on iTunes and Soundcloud.