Infinite Volumes Management Guide

September 2016 | 215-11160-B0
doccomments@netapp.com

Updated for ONTAP 9.1
# Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Introduction to Infinite Volumes</td>
<td>11</td>
</tr>
<tr>
<td>What an Infinite Volume is</td>
<td>11</td>
</tr>
<tr>
<td>When to use an Infinite Volume</td>
<td>11</td>
</tr>
<tr>
<td>Capabilities that Infinite Volumes provide</td>
<td>11</td>
</tr>
<tr>
<td>Comparison of FlexVol volumes and Infinite Volumes</td>
<td>12</td>
</tr>
<tr>
<td>How Infinite Volumes are composed of constituents</td>
<td>13</td>
</tr>
<tr>
<td>What a namespace constituent is</td>
<td>14</td>
</tr>
<tr>
<td>What data constituents are</td>
<td>14</td>
</tr>
<tr>
<td>What a namespace mirror constituent is</td>
<td>14</td>
</tr>
<tr>
<td>How Infinite Volumes use storage classes and data policies</td>
<td>15</td>
</tr>
<tr>
<td>What a storage class is</td>
<td>15</td>
</tr>
<tr>
<td>What rules and data policies are</td>
<td>16</td>
</tr>
<tr>
<td>How a data policy filters data written to an Infinite Volume</td>
<td>16</td>
</tr>
<tr>
<td>When a data policy is applied</td>
<td>17</td>
</tr>
<tr>
<td>Unsupported Data ONTAP features in Infinite Volumes</td>
<td>17</td>
</tr>
<tr>
<td>Where to find information about Data ONTAP features</td>
<td>19</td>
</tr>
<tr>
<td>Requirements for Infinite Volumes</td>
<td>20</td>
</tr>
<tr>
<td>Platforms supported by Infinite Volumes</td>
<td>20</td>
</tr>
<tr>
<td>Node requirements for Infinite Volumes</td>
<td>20</td>
</tr>
<tr>
<td>SVM requirements for Infinite Volumes</td>
<td>21</td>
</tr>
<tr>
<td>Aggregate requirements for Infinite Volumes</td>
<td>22</td>
</tr>
<tr>
<td>Aggregate requirements for destination Infinite Volumes</td>
<td>22</td>
</tr>
<tr>
<td>Junction naming requirements for Infinite Volumes</td>
<td>23</td>
</tr>
<tr>
<td>Tool requirements for Infinite Volumes with storage classes</td>
<td>23</td>
</tr>
<tr>
<td>Workflows for setting up and managing Infinite Volumes</td>
<td>24</td>
</tr>
<tr>
<td>Deciding whether to use storage classes</td>
<td>24</td>
</tr>
<tr>
<td>When to use an Infinite Volume with storage classes</td>
<td>24</td>
</tr>
<tr>
<td>How storage classes affect Infinite Volumes</td>
<td>24</td>
</tr>
<tr>
<td>Considerations for using storage classes</td>
<td>25</td>
</tr>
<tr>
<td>Setting up and managing an Infinite Volume</td>
<td>25</td>
</tr>
<tr>
<td>Setting up and managing an Infinite Volume with storage classes</td>
<td>27</td>
</tr>
<tr>
<td>Planning to create Infinite Volumes</td>
<td>30</td>
</tr>
<tr>
<td>Considerations when planning aggregate space</td>
<td>30</td>
</tr>
<tr>
<td>Considerations when using thin provisioning with Infinite Volumes</td>
<td>30</td>
</tr>
<tr>
<td>Space considerations for using incremental tape backup</td>
<td>32</td>
</tr>
<tr>
<td>Constituent size considerations for data protection mirror relationships for Infinite Volumes</td>
<td>32</td>
</tr>
<tr>
<td>Planning aggregates for an Infinite Volume</td>
<td>33</td>
</tr>
<tr>
<td>How FlexVol volumes and Infinite Volumes share aggregates</td>
<td>33</td>
</tr>
<tr>
<td>How aggregates and nodes are associated with Infinite Volumes</td>
<td>34</td>
</tr>
<tr>
<td>How much space namespace-related constituents require</td>
<td>35</td>
</tr>
</tbody>
</table>
Identifying the size of a source Infinite Volume and aggregate details ...... 105
Creating dedicated aggregates for a destination cluster ......................... 106
Creating an SVM with Infinite Volume in a destination cluster ............... 107
Creating cluster and SVM peer relationships ........................................... 107
Creating a destination Infinite Volume .................................................... 108
Creating and initializing a data protection mirror relationship .................. 109
Creating a data protection mirror copy for an Infinite Volume with shared
aggregates ........................................................................................................ 111
Identifying the size of a source Infinite Volume and constituent details .... 112
Verifying aggregate space in a destination cluster ...................................... 113
Creating an SVM with Infinite Volume in a destination cluster ............... 115
Creating cluster and SVM peer relationships ........................................... 115
Creating a destination Infinite Volume .................................................... 116
Creating and initializing a data protection mirror relationship .................. 117
Viewing information about destination Infinite Volumes .......................... 119
When information is available about a destination Infinite Volume .......... 119
Protocols supported on destination Infinite Volumes ............................... 120
When clients can access the active file system on the destination Infinite
Volume ........................................................................................................... 120
Managing disaster recovery for Infinite Volumes ........................................ 120
How the snapmirror resync command works for Infinite Volumes and
constituents ....................................................................................................... 120
Reversing the data protection mirror relationship for Infinite Volumes
when disaster occurs .................................................................................. 121
Creating a namespace mirror constituent on a destination Infinite
Volume ........................................................................................................... 125
Recovering individual failed constituents .................................................. 127
How namespace mirror constituents affect source and destination Infinite
Volumes .......................................................................................................... 127
How Data ONTAP calculates the Infinite Volume size before the transfer of a
mirror copy ................................................................................................... 127
How Data ONTAP selects aggregates for data protection of Infinite Volumes .... 129
When Data ONTAP selects aggregates for destination Infinite Volumes .. 129
How Data ONTAP automatically selects aggregates for destination
Infinite Volumes .............................................................................................. 129
Examples of automatic aggregate selection for destination Infinite
Volumes .......................................................................................................... 132
Error messages and solutions for failed aggregate selection for destination
Infinite Volumes ............................................................................................... 135
Insufficient space on destination for SnapMirror source Infinite Volume
namespace constituent .................................................................................. 135
Size exceeds maximum (namespace constituent) ....................................... 136
Insufficient space for the following source data constituents on
destination Vserver .................................................................................. 136
Expanding Infinite Volumes ................................................................. 168
  How space is allocated when you expand an Infinite Volume .......... 168
  How data protection mirror updates affect aggregate selection .... 169
Determining which aggregates an Infinite Volume uses .................. 169
Preparing to resize an Infinite Volume .............................................. 170
  Determining space availability in an Infinite Volume’s aggregates .... 170
  Adding disks to the aggregates that an Infinite Volume uses ........... 171
  Associating more aggregates with an Infinite Volume ................. 172
  Associating more nodes with an Infinite Volume ......................... 174
Resizing an Infinite Volume ............................................................... 175
Resizing an Infinite Volume by using OnCommand Workflow Automation .......... 176
Resizing source and destination Infinite Volumes ............................ 176
Reenabling SnapDiff after expanding a destination Infinite Volume onto new nodes ................................................................. 178
Reenabling SnapDiff after expanding a destination Infinite Volume with storage classes onto new nodes ................................................................. 178
Managing Infinite Volumes ............................................................... 180
  Monitoring operations and events on Infinite Volumes ................. 180
  How operations run on Infinite Volumes ........................................ 180
  Viewing the status of efficiency operations on an Infinite Volume ... 181
  Restrictions on the private namespace of an SVM with Infinite Volume ... 182
  Viewing information about events on Infinite Volume constituents ...... 182
Rebalancing the used capacity of data constituents ............................ 183
  Commands for rebalancing used capacity of Infinite Volumes ........ 183
  Examples of how used capacity is rebalanced for Infinite Volumes .. 184
  How offline data constituents affect the status of Auto Balance Volume operations ................................................................. 186
  Viewing the used capacity of data constituents in Infinite Volumes ... 187
  Rebalancing the used capacity in an Infinite Volume ..................... 187
Moving constituents of an Infinite Volume from an aggregate ............. 189
How SnapDiff affects moving constituents from an aggregate .......... 190
How the mixed state of an Infinite Volume affects its availability ........ 190
Forcing constituents of an Infinite Volume into the same state .......... 191
Creating a new root volume on an SVM with Infinite Volume .......... 194
Viewing constituents of an Infinite Volume ....................................... 195
Viewing constituents of a specific role in an Infinite Volume .......... 196
Naming convention for constituents .................................................. 197
Using Infinite Volumes with storage classes ..................................... 198
  Planning to create Infinite Volumes with storage classes .......... 198
  Definitions for the default storage classes .................................... 198
  How storage classes affect which aggregates can be associated with
    Infinite Volumes ........................................................................... 199
  How efficiency works on Infinite Volumes with storage classes ..... 199
  Commands disabled for Infinite Volumes with storage classes .... 200
Support for efficiency commands on Infinite Volumes with storage classes ................................................................. 200
This command is not supported for Infinite Volumes that are managed by storage services .................................................. 201
How storage classes relate to storage services ............................................. 201
Tasks and tools for managing Infinite Volumes with storage classes ....... 202
Tools for modifying Infinite Volumes with storage classes ......................... 203
How custom storage classes affect predefined workflows ......................... 203
How customized predefined workflows support array LUNs ....................... 204
Customizing predefined workflows for storage classes (basic) ...................... 204
Customizing predefined workflows for storage classes (advanced) ............. 205
Creating aggregates for Infinite Volumes with storage classes .................. 207
Converting to an Infinite Volume with storage classes .............................. 208
Creating an Infinite Volume with storage classes ...................................... 209
  Creating an SVM and Infinite Volume with storage classes .................... 209
  Adding a storage class to an Infinite Volume ........................................... 210
Configuring a Snapshot policy for an Infinite Volume with storage classes ........................................................................ 211
Configuring file access support for data policies ........................................... 211
  How storage classes affect file access to Infinite Volumes ....................... 211
Configuring users or directories for rules in a data policy .......................... 212
Modifying the data policy for an SVM with Infinite Volume ........................ 212
  Editing rules in a data policy for an SVM with Infinite Volume .................. 212
  Managing JSON data policies for SVMs with Infinite Volume ................. 213
Providing backups of Infinite Volumes with storage classes using Snapshot copies ........................................................................................................ 219
  How storage classes affect Snapshot copies of Infinite Volumes ............... 219
Providing disaster recovery on Infinite Volumes with storage classes using mirroring technology .................................................................................... 219
  Tasks and tools for managing data protection mirror copies for Infinite Volumes with storage classes ............................... 219
  Creating a data protection mirror copy for an Infinite Volume with storage classes ................................................................. 220
  Reversing the data protection mirror relationship for Infinite Volumes with storage classes when disaster occurs .................. 222
Processes used to select aggregates for destination Infinite Volumes with and without storage classes ......................................................... 226
Providing tape backup and restore of Infinite Volumes with storage classes .... 227
  How storage classes affect tape backup and restore of Infinite Volumes ... 227
  Protocol considerations for restoring files to Infinite Volumes with storage classes ................................................................................. 227
  Enabling SnapDiff on an Infinite Volume with storage classes ............... 227
Monitoring space usage in Infinite Volumes with storage classes ................ 228
Expanding Infinite Volumes with storage classes ........................................ 228
Introduction to Infinite Volumes

An Infinite Volume is a logical storage unit that you can use to provide a large, scalable data container with a single namespace and a single junction path. You can choose whether to use storage classes with Infinite Volumes. Snapshot copies, SnapMirror relationships, and tape backup are supported.

What an Infinite Volume is

An Infinite Volume is a single, scalable volume that can store up to 2 billion files and tens of petabytes of data.

With an Infinite Volume, you can manage multiple petabytes of data in one large logical entity and clients can retrieve multiple petabytes of data from a single junction path for the entire volume.

An Infinite Volume uses storage from multiple aggregates on multiple nodes. You can start with a small Infinite Volume and expand it nondisruptively by adding more disks to its aggregates or by providing it with more aggregates to use.

When to use an Infinite Volume

Infinite Volumes are designed to address the needs of large unstructured repositories of data, which are also known as enterprise content repositories.

Infinite Volumes are best suited to non-latency-sensitive applications and environments that are characterized by input/output (I/O) patterns in which data is written once and seldom changed. Because the data is used for normal business operations, the data is kept online for fast retrieval and not moved to secondary storage.

Capabilities that Infinite Volumes provide

Infinite Volumes enable you to store multiple petabytes of data in a single volume that supports multiprotocol access, storage efficiency technologies, and data protection capabilities.

With Infinite Volumes, you can perform the following tasks:

• Manage multiple petabytes of data in a single logical entity with a single junction path and a single namespace.

• Provide multiprotocol access to that data using NFSv3, NFSv4.1, pNFS, and CIFS (SMB 1.0).

• Offer secure multi-tenancy by creating multiple SVMs with FlexVol volumes and multiple SVMs with Infinite Volume in a single cluster.

• Create an Infinite Volume that is larger than the available physical storage by using thin provisioning.

• Maximize storage efficiency by using deduplication and compression technologies.

• Optimize storage by grouping it into storage classes that correspond to specific goals.

• Automatically place incoming files into the appropriate storage class according to rules based on file name, file path, or file owner.

• Protect data by creating Snapshot copies of the volume.
• Create a data protection mirror relationship between two Infinite Volumes on different clusters, and restore data when necessary.

• Back up data with CIFS or NFS over a mounted volume to tape, and restore data when necessary.

• Expand the Infinite Volume by adding more disks to the aggregates used by the Infinite Volume or by assigning more aggregates to the SVM containing the Infinite Volume and then resizing the Infinite Volume.

**Comparison of FlexVol volumes and Infinite Volumes**

Both FlexVol volumes and Infinite Volumes are data containers. However, they have significant differences that you should consider before deciding which type of volume to include in your storage architecture.

The following table summarizes the differences and similarities between FlexVol volumes and Infinite Volumes:

<table>
<thead>
<tr>
<th>Volume capability or feature</th>
<th>FlexVol volumes</th>
<th>Infinite Volumes</th>
<th>For more information</th>
</tr>
</thead>
<tbody>
<tr>
<td>Containing entity</td>
<td>SVM; single node</td>
<td>SVM; can span nodes</td>
<td></td>
</tr>
<tr>
<td>Number of associated aggregates</td>
<td>One</td>
<td>Multiple</td>
<td></td>
</tr>
<tr>
<td>Maximum size</td>
<td>Model-dependent</td>
<td>Up to 20 PB</td>
<td>NetApp Hardware Universe</td>
</tr>
<tr>
<td>Minimum size</td>
<td>20 MB</td>
<td>Approximately 1.33 TB for each node used</td>
<td></td>
</tr>
<tr>
<td>Type of Storage Virtual Machine (SVM)</td>
<td>SVM with FlexVol volumes</td>
<td>SVM with Infinite Volume</td>
<td></td>
</tr>
<tr>
<td>Maximum number per SVM</td>
<td>Model- and protocol-dependent</td>
<td>One</td>
<td>NetApp Hardware Universe</td>
</tr>
<tr>
<td>Maximum number per node</td>
<td>Model-dependent</td>
<td>Model-dependent</td>
<td>NetApp Hardware Universe</td>
</tr>
<tr>
<td>SAN protocols supported</td>
<td>Yes</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>File access protocols supported</td>
<td>NFS, CIFS</td>
<td>NFS, CIFS</td>
<td></td>
</tr>
<tr>
<td>Deduplication</td>
<td>Yes</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>Compression</td>
<td>Yes</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>FlexClone volumes</td>
<td>Yes</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>SnapLock volumes</td>
<td>Yes</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>Quotas</td>
<td>Yes</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>Qtrees</td>
<td>Yes</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>Thin provisioning</td>
<td>Yes</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>Snapshot copies</td>
<td>Yes</td>
<td>Yes</td>
<td></td>
</tr>
</tbody>
</table>
### How Infinite Volumes are composed of constituents

Each Infinite Volume consists of several—typically dozens of—separate components called **constituents**. Understanding how these constituents are created and the roles they play can help you plan aggregates for an Infinite Volume, understand its capacity, and interpret the results of operations performed on the Infinite Volume.

Constituents are internal to the Infinite Volume and are not visible to clients as they access the volume. You can display the constituents from the CLI if you substitute the `-is-constituent true` parameter for the `-volume` parameter in certain `show` commands. When you use the `-is-constituent true` parameter, the output displays the same information for constituents that is displayed for volumes.

**Roles of constituents in an Infinite Volume**

Constituents play one of the following roles:

- **Data constituents**, which store data
- **Namespace constituent**, which tracks file names, directories, and the file's physical data location
- **Namespace mirror constituents**, which are data protection mirror copies of the namespace constituent

The following diagram illustrates the roles of the namespace and data constituents:

### Related information

*Logical storage management*
What a namespace constituent is

Each Infinite Volume has a single namespace constituent that maps directory information and file names to the file's physical data location within the Infinite Volume.

Clients are not aware of the namespace constituent and do not interact directly with it. The namespace constituent is an internal component of the Infinite Volume.

What data constituents are

In an Infinite Volume, data is stored in multiple separate data constituents. Data constituents store only the data from a file, not the file's name.

Clients are not aware of data constituents. When a client requests a file from an Infinite Volume, the node retrieves the file's data from a data constituent and returns the file to the client.

Each Infinite Volume typically has dozens of data constituents. For example, a 6 PB Infinite Volume that contains 1 billion files might have 60 data constituents located on aggregates from 6 nodes.

What a namespace mirror constituent is

A namespace mirror constituent is an intracluster data protection mirror copy of the namespace constituent in an Infinite Volume. The namespace mirror constituent performs two roles: It provides data protection of the namespace constituent, and it supports SnapDiff for incremental tape backup of Infinite Volumes.

Related concepts

*When namespace mirror constituents are created* on page 142

*How incremental tape backup uses namespace-related constituents* on page 142
How Infinite Volumes use storage classes and data policies

You can create an Infinite Volume with one or more storage classes and use a data policy with rules to automatically filter incoming data into different storage classes. Understanding what storage classes and data policies are can help you decide when they are useful for Infinite Volumes.

What a storage class is

A storage class is a definition of aggregate characteristics and volume settings. You can define different storage classes and associate one or more storage classes with an Infinite Volume. You must use OnCommand Workflow Automation to define workflows for your storage class requirements and to assign storage classes to Infinite Volumes.

You can define the following characteristics for a storage class:

- Aggregate characteristics, such as the type of disks to use
- Volume settings, such as compression, deduplication, and volume guarantee

For example, you can define a storage class that uses only aggregates with SAS disks and the following volume settings: thin provisioning with compression and deduplication enabled.

The following diagram illustrates an Infinite Volume that spans multiple nodes and uses the following storage classes: gold, silver, and bronze. Each storage class can span two or more nodes within an Infinite Volume. The diagram also illustrates the placement of data constituents in each storage class.

Related concepts

How storage classes affect which aggregates can be associated with Infinite Volumes on page 199
What rules and data policies are

A rule determines the placement of files (data) in a Storage Virtual Machine (SVM) with Infinite Volume. A collection of such rules is known as a data policy.

Rule

Rules mainly consist of a set of predefined conditions and information that determine where to place files in the Infinite Volume. When a file is placed in the Infinite Volume, the attributes of that file are matched with the list of rules. If attributes match the rules, then that rule's placement information determines the storage class where the file is placed. A default rule in the data policy is used to determine the placement of files if the attributes do not match any of the rules in the rule list.

For example, if you have a rule, “Place all files of type .mp3 in the bronze storage class.”, all .mp3 files that are written to the Infinite Volume would be placed in the bronze storage class.

Data policy

A data policy is a list of rules. Each SVM with Infinite Volume has its own data policy. Each file that is added to the Infinite Volume is compared to its data policy's rules to determine where to place that file. The data policy enables you to filter incoming files based on the file attributes and place these files in the appropriate storage classes.

How a data policy filters data written to an Infinite Volume

A data policy automatically filters data written to the Infinite Volume into different storage classes. All files are written to the single file system in the Infinite Volume's namespace, and rules in the data policy determine which storage class stores the data for the files.

A default data policy is automatically created for a Storage Virtual Machine (SVM) with Infinite Volume when you create the Infinite Volume. The data policy is active and contains a default rule. The default rule stores incoming data for files as follows for Infinite Volumes with and without storage classes:

<table>
<thead>
<tr>
<th>For an Infinite Volume...</th>
<th>The default data policy does this...</th>
</tr>
</thead>
<tbody>
<tr>
<td>Without storage classes</td>
<td>Places all incoming data for files in the Infinite Volume</td>
</tr>
<tr>
<td>With one storage class</td>
<td>Places all incoming data for files into the storage class</td>
</tr>
<tr>
<td>With one or more storage classes</td>
<td>Places all incoming data for files into the first storage class that is created</td>
</tr>
</tbody>
</table>

Important: For an Infinite Volume with two or more storage classes, you should modify the data policy as soon as possible to create rules that filter data for different types of files into the different storage classes. You should modify the data policy by using OnCommand Unified Manager.

The data policy does not affect the location of the files in the file system in the Infinite Volume's namespace, and storage classes are transparent to client applications. The file system in the namespace contains the file names. The data policy affects only which storage class is used to store the data for the files. Data policies are useful when you assign two or more storage classes to an Infinite Volume.

You can modify the data policy to create additional rules, but you cannot delete the data policy or its default rule.

The following diagram illustrates how a data policy filters data for an Infinite Volume. The file name is stored in the namespace constituent, and rules in the data policy specify that data for this particular file is stored in the silver storage class.
When a data policy is applied

The data policy for each Storage Virtual Machine (SVM) with Infinite Volume analyzes files when files are created in the Infinite Volume. If you modify the file, the data policy does not analyze the file again.

For example, the data policy analyzes a file named statistics when the file is created in the Infinite Volume, and the data for the file is stored in a storage class named bronze, according to the instructions in the data policy. If you later modify the file named statistics, the data policy does not analyze the file again. The file name remains in the file system in the Infinite Volume’s namespace, and the data for the file is saved in the storage class named bronze. Similarly, if you later modify the data policy, changes to the data policy affect only incoming files, not data for files already stored in the Infinite Volume’s storage classes. As a result, after data for files is stored in a storage class in an Infinite Volume, you cannot move the data to another storage class by modifying the file or by modifying the data policy.

Unsupported Data ONTAP features in Infinite Volumes

A Storage Virtual Machine (SVM) with Infinite Volume and its Infinite Volume do not support all the Data ONTAP features. Awareness of the unsupported features helps you understand how you can use an SVM with Infinite Volume and its Infinite Volume.

The following Data ONTAP features are not supported by Infinite Volumes:

- LUNs
- SAN access
- Copying the Infinite Volume
• Moving the Infinite Volume
• Shrinking the Infinite Volume
• Recovering a deleted Infinite Volume
• Qtrees
• Quotas
• FlexClone volumes and files
• Storage quality of service (Storage QoS)
• NFSv4.0
• SMB 2.0 or later
• FPolicy
• Auditing
• Load-sharing mirror copies on the Infinite Volume or on the root volume of an SVM with Infinite Volume
• SnapVault relationships
• NDMP and NDMP-based tape backup
  You cannot use NDMP-based backup for an Infinite Volume. You can use other non-NDMP methods to back up and restore data in an Infinite Volume to tape.
• NAS protocols other than NFSv3 for the active file system on the destination Infinite Volume
  You can use supported NFS and CIFS protocols to access Snapshot copies on destination Infinite Volumes.
• VMware vStorage APIs for Array Integration (VAAI)
• Antivirus
• Caching policy for Flash Pool aggregates
• Auto Balance Aggregate
  You can use Auto Balance Volume for an Infinite Volume.
• MetroCluster configuration
• Data ONTAP Edge

Any of these features can exist in the cluster that contains the Infinite Volume, but they cannot be used on the Infinite Volume.

Related concepts

Support for NFS on Infinite Volumes on page 71
Support for CIFS on Infinite Volumes on page 72
Unsupported space management capabilities for Infinite Volumes on page 164
## Where to find information about Data ONTAP features

The *Clustered Data ONTAP Infinite Volumes Management Guide* includes information about how to use Data ONTAP features with Infinite Volumes. For information about how specific Data ONTAP features work, see the appropriate Data ONTAP guide.

<table>
<thead>
<tr>
<th>For information about...</th>
<th>Such as...</th>
<th>See...</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aggregates</td>
<td>How to create, manage, and relocate ownership of aggregates</td>
<td><em>Disk and aggregate management</em></td>
</tr>
<tr>
<td>Storage Virtual Machine (SVM,</td>
<td>How to create and manage SVMs</td>
<td><em>System administration</em></td>
</tr>
<tr>
<td>formerly known as Vserver)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FlexVol volumes</td>
<td>How to create and manage FlexVol volumes</td>
<td><em>Logical storage management</em></td>
</tr>
<tr>
<td>Deduplication and compression</td>
<td>What deduplication and compression are and how they work</td>
<td><em>Logical storage management</em></td>
</tr>
<tr>
<td>NFS</td>
<td>How to set up and manage NFS</td>
<td><em>NFS management</em></td>
</tr>
<tr>
<td>CIFS</td>
<td>How to set up and manage CIFS</td>
<td><em>SMB/CIFS management</em></td>
</tr>
<tr>
<td>Tape backup and restore</td>
<td>How to set up and manage tape backup and restore using NDMP</td>
<td><em>Data protection using tape backup</em></td>
</tr>
</tbody>
</table>

**Related information**

*NetApp Documentation: Data ONTAP 8 (current releases)*
Requirements for Infinite Volumes

Infinite Volumes require specific platforms, nodes, Storage Virtual Machine (SVM) and aggregate configurations, and junction names. Infinite Volumes with storage classes also require specific tools. You should review these requirements before you create or expand an Infinite Volume.

Platforms supported by Infinite Volumes

Infinite Volumes are supported only on certain platforms. If you try to create or expand an Infinite Volume that uses aggregates from an unsupported platform, an error message appears and you cannot proceed.

Infinite Volumes support heterogeneous configuration where you mix different platforms in the same cluster. The nodes that an Infinite Volume spans do not all have to be the same platform.

The platform that you use affects the size of the Infinite Volume's data constituents.

Related concepts

- How aggregates and nodes are associated with Infinite Volumes on page 34
- How data constituents use aggregate space on page 35
- Constituent size considerations for data protection mirror relationships for Infinite Volumes on page 32

Related information

- NetApp Hardware Universe

Node requirements for Infinite Volumes

Unlike FlexVol volumes, Infinite Volumes use multiple nodes in a cluster. Each Infinite Volume can use from 2 through 10 nodes. You should be aware of node requirements and how to meet them before you create or expand an Infinite Volume.

Node requirements when configuring the SVM aggregate list

The nodes that an Infinite Volume uses are determined by the aggregate list of its containing Storage Virtual Machine (SVM) with Infinite Volume.

The aggregate list of the SVM with Infinite Volume must be configured in the following way related to nodes:

- The list must contain an aggregate from each node that you want the Infinite Volume to use. If the aggregate list does not contain an aggregate from a node, the node will not be used by the Infinite Volume.
- It must contain aggregates from at least 2 nodes. An Infinite Volume is not supported on a single-node cluster or on a single node of a multi-node cluster.
- It must contain aggregates from no more than 10 nodes. The cluster that contains Infinite Volumes can have more than 10 nodes, but the aggregate list for each individual SVM with Infinite Volume cannot contain aggregates from more than 10 nodes.
- It can contain one node of an HA pair without containing the partner node. An Infinite Volume does not have to use both nodes of an HA pair.
**Best practice to balance capacity across nodes**

It is a best practice to ensure that the amount of total available space on each node is roughly the same.

For example, if one node has 112 TB, the second node can have 120 TB, but it should not have a significantly different amount of available space, such as 50 TB.

This best practice also applies when you expand the Infinite Volume. If you are adding capacity to existing nodes, you should add equal capacity to each node. If you are adding a node, the new node should have available space that is at least equal to the space that the Infinite Volume uses on its existing nodes. When you increase the size of an Infinite Volume, the space is divided equally across all nodes that the Infinite Volume spans. If the space cannot be equally divided across the nodes, you cannot increase the size of the Infinite Volume.

**Related concepts**

- [Aggregate requirements for Infinite Volumes](#) on page 22
- [SVM requirements for Infinite Volumes](#) on page 21

**SVM requirements for Infinite Volumes**

An Infinite Volume requires a specifically configured Storage Virtual Machine (SVM, formerly known as Vserver) that does not contain any other volumes, and the SVM should have a defined list of associated aggregates. You should be aware of these SVM requirements before you create an Infinite Volume.

An Infinite Volume requires an SVM with the following configuration:

- An SVM that has been configured specifically to make it an SVM with Infinite Volume
  An Infinite Volume cannot be created inside an SVM with the default configuration, which is called an SVM with FlexVol volumes.
  The `-is-repository` parameter is what differentiates an SVM with Infinite Volume from an SVM with FlexVol volumes. The parameter is `true` for SVMs with Infinite Volume and `false` for SVMs with FlexVol volumes.

- An SVM that contains no volumes
  An SVM with Infinite Volume cannot contain more than one Infinite Volume, and it cannot contain any FlexVol volumes. The SVM root volume is an exception; every SVM has a root volume for internal management.

- An SVM with a defined list of aggregates (recommended)
  After creating an SVM with Infinite Volume, you should define its associated aggregates. Without additional configuration, every SVM is initially associated with all the aggregates in the cluster. A cluster administrator can associate aggregates with an SVM by using the `vserver add-aggregates` command.

Unlike in earlier versions of Data ONTAP, you are not required to dedicate an entire cluster to the SVM with Infinite Volume. Multiple SVMs with Infinite Volume and multiple SVMs with FlexVol volumes can coexist in the same cluster.

**Related concepts**

- [Aggregate requirements for Infinite Volumes](#) on page 22
- [Node requirements for Infinite Volumes](#) on page 20
Aggregate requirements for Infinite Volumes

The aggregates that are used by an Infinite Volume should be larger than 100 TB with a minimum of 1.1 TB of available space. If the Infinite Volume uses storage classes, the aggregates must also meet the requirements of the storage class.

If an aggregate has less than 1.1 TB of available space, it is not used by the Storage Virtual Machine (SVM) with Infinite Volume.

If the Infinite Volume uses storage classes, aggregates must meet the requirements of the storage class to be used. For example, if the storage class is designated to use aggregates of type SAS, aggregates created for that storage class must consist entirely of SAS disks.

Related concepts

Node requirements for Infinite Volumes on page 20
SVM requirements for Infinite Volumes on page 21

Aggregate requirements for destination Infinite Volumes

Before you create a destination Infinite Volume for a data protection mirror relationship with an Infinite Volume, you must create enough aggregate space in the destination cluster for the destination Infinite Volume to use.

An Infinite Volume spans several aggregates, and aggregates are automatically selected for a destination Infinite Volume when you initialize a data protection mirror relationship. If the data protection mirror relationship cannot be initialized because of insufficient aggregate space, you receive an error message that informs you how to adjust aggregate space before trying the operation again.

You should use the following guidelines to create aggregates for destination Infinite Volumes:

• The destination Infinite Volume and source Infinite Volume should have the same number of aggregates.
  For example, if the source Infinite Volume uses four aggregates, you should create four aggregates for the destination Infinite Volume. The same number of aggregates for the source and destination Infinite Volumes is recommended, but not required.

• The aggregates for the destination Infinite Volume must have enough space to contain a mirror copy of the source Infinite Volume.
  For the most detailed aggregate sizes, you can compare the sizes in KB.

• The aggregates must meet the requirements of the storage classes used by the source Infinite Volume, if the source Infinite Volume uses storage classes.

  Note: The size of the destination Infinite Volume must be equal to or larger than the size of the source Infinite Volume to successfully create a data protection mirror relationship.

Related concepts

How Data ONTAP selects aggregates for data protection of Infinite Volumes on page 129
Error messages and solutions for failed aggregate selection for destination Infinite Volumes on page 135
Junction naming requirements for Infinite Volumes

When you create an Infinite Volume, you must define its junction path according to certain requirements. The junction path becomes the client-accessible namespace for the entire Infinite Volume and its containing Storage Virtual Machine (SVM).

By default, an Infinite Volume created by using the command-line interface has a default junction path of /NS, which is an abbreviation for “namespace”.

If you define a custom junction path, it must meet the following requirements:

- It must be one directory level below the root of the SVM, which means it must start with “/”. 
- It must be a single element, meaning it cannot contain backslashes, although it can contain spaces.
  
  For example, it can be /NS or /financials, but cannot be /NS/financials.
- It is case insensitive; /NS is the same as /ns.
  
  If you create a CIFS share, Windows treats the junction path as if it is case sensitive. For example, if the junction is /NS, the path of a CIFS share must start with /NS, not /ns.

You can display the junction path by using the `volume show` command with the `-junction` parameter.

You can change a junction path by unmounting and remounting the Infinite Volume. Before changing a junction path, you should consider the impact on client access.

Tool requirements for Infinite Volumes with storage classes

You must use OnCommand Workflow Automation and OnCommand Unified Manager to create and manage Infinite Volumes with storage classes. After creating the volume, you can use the command-line interface for some operations.

Related references

Tasks and tools for managing Infinite Volumes with storage classes on page 202
Tasks and tools for managing data protection mirror copies for Infinite Volumes with storage classes on page 219
Workflows for setting up and managing Infinite Volumes

The workflow for setting up and managing an Infinite Volume differs depending on whether you want to use storage classes with the Infinite Volume. Understanding these workflows enables you to manage the Infinite Volume effectively.

Deciding whether to use storage classes

You can create an Infinite Volume with or without storage classes. Storage classes affect how you plan, create, manage, and use Infinite Volumes. You should decide whether you want to use storage classes before you plan and create the Infinite Volume.

When to use an Infinite Volume with storage classes

Storage classes allow you to provide multiple tiers of storage in one Infinite Volume that can store petabytes of data and billions of files in a single file system. All data is written to the single file system, and a data policy automatically filters data for the files into different storage classes.

You can use storage classes and a data policy to organize data in an Infinite Volume by disk type and volume settings. For example, you can create an Infinite Volume with two storage classes. All files are written to the single file system in the Infinite Volume's namespace, and a data policy automatically filters the data for the files into different storage classes based on file name, file owner, or file location.

Related concepts

What a storage class is on page 15

How a data policy filters data written to an Infinite Volume on page 16

How storage classes affect Infinite Volumes

If you want to use storage classes with an Infinite Volume, you should be aware that storage classes can affect the type of aggregates you can use and what tools you can use to create and manage the Infinite Volume. Some command-line interface commands are disabled for Infinite Volumes with storage classes.

Storage class definitions specify what type of aggregate to use. When you create aggregates for an Infinite Volume with storage classes, you must be aware of the definitions of the storage classes that you want to use so that you can create the appropriate type of aggregates. OnCommand Workflow Automation validates that aggregates comply with the restrictions of the storage class before allowing the storage class to use the aggregates. You must use OnCommand Workflow Automation to define workflows for your storage class requirements and to assign storage classes to Infinite Volumes.

Storage class definitions can also include some volume settings, such as thin provisioning, compression, and deduplication. When you create an Infinite Volume with storage classes, some settings no longer apply to the entire Infinite Volume. Instead some settings apply to individual storage classes in the Infinite Volume.

When you create an Infinite Volume with two or more storage classes, you should use a data policy to automatically filter incoming data into different storage classes. Otherwise all data written to the Infinite Volume is stored in the first storage class created. You should use OnCommand Unified Manager to modify and manage the data policy for an Infinite Volume.
**Related concepts**

*What rules and data policies are* on page 16

**Considerations for using storage classes**

You should understand how storage classes affect Infinite Volumes to help you decide whether to create an Infinite Volume with storage classes.

<table>
<thead>
<tr>
<th>Consideration</th>
<th>For more information...</th>
</tr>
</thead>
<tbody>
<tr>
<td>Storage class definitions specify aggregate type</td>
<td><em>How storage classes affect which aggregates can be associated with Infinite Volumes</em> on page 199</td>
</tr>
</tbody>
</table>
| Each storage class used by the Infinite Volume can have unique efficiency and volume settings | *Considerations when using thin provisioning with Infinite Volumes* on page 30  
*How efficiency works on Infinite Volumes with storage classes* on page 199  
*Support for efficiency commands on Infinite Volumes with storage classes* on page 200  
*Commands disabled for Infinite Volumes with storage classes* on page 200 |
| Rules in a data policy can automatically place data written to the Infinite Volume in a specific storage class | *How a data policy is automatically created and starts filtering data* on page 16 |
| Specific tools are required to work with storage classes                      | *Tool requirements for Infinite Volumes with storage classes* on page 23                |

**Note:** You can convert an Infinite Volume without storage classes to an Infinite Volume with storage classes.

**Related tasks**

*Converting to an Infinite Volume with storage classes* on page 208

**Setting up and managing an Infinite Volume**

Setting up Infinite Volumes includes planning the configuration; creating aggregates; creating the Storage Virtual Machine (SVM); creating the Infinite Volume; setting up file access; and optionally configuring storage efficiency, data protection, and tape backup. After an Infinite Volume is set up, you should monitor its capacity and expand it when necessary.

**About this task**

This flowchart and procedure provide an overview of the tasks involved in the setup and management of an Infinite Volume.
Steps

1. Plan the Infinite Volume:
   a. Understand the requirements for Infinite Volumes.
   b. Identify the size required.
   c. Identify the provisioning approach—thick or thin provisioning.
   d. Determine if the Infinite Volume will be backed up to tape.
   e. Identify the number of nodes that the Infinite Volume will use.
   f. Identify the aggregates that the Infinite Volume will use.

2. If necessary, create aggregates for the Infinite Volume.

3. Create the SVM with Infinite Volume, and assign aggregates to the SVM.

4. Create the Infinite Volume.

5. Optional: Enable deduplication and compression.
6. Set up file access (NFS access, CIFS access, or both):
   a. Configure the data LIFs of the SVM with Infinite Volume.
   b. Set up NFS access to the Infinite Volume if required.
   c. Set up CIFS access to the Infinite Volume if required.

7. Optional: Configure data protection for the Infinite Volume:
   a. Modify the default Snapshot copy schedule if required.
   b. Create a data protection mirror relationship for the Infinite Volume.


9. Monitor space usage, and expand the Infinite Volume if required:
   a. Add or expand aggregates to meet the space guarantee as required.
   b. Scale the Infinite Volume by expanding it to meet your data requirements.

Related information

*Disk and aggregate management*

---

**Setting up and managing an Infinite Volume with storage classes**

Setting up Infinite Volumes with storage classes includes planning the configuration; creating aggregates; creating the Infinite Volume with storage classes; configuring the data policy; and optionally configuring, data protection, and tape backup. After an Infinite Volume is set up, you should monitor its capacity and expand it when necessary.

**Before you begin**

- You must have installed OnCommand Workflow Automation and OnCommand Unified Manager.

**About this task**

While performing this task, you are required to switch between two applications: Workflow Automation and Unified Manager. You can also use the command-line interface (CLI) for some operations.

This flowchart and procedure provide an overview of the tasks involved in the setup and management of an Infinite Volume with storage classes.
**Steps**

1. Plan the Infinite Volume with storage classes:
   a. Understand the requirements for Infinite Volumes.
   b. Understand the storage class definitions in Workflow Automation and their impact on aggregate type, provisioning (thick or thin provisioning), and efficiency (deduplication and compression).
   c. Identify the size required for each storage class.
   d. Identify the number of nodes that the Infinite Volume will use.
   e. Identify the aggregates that the Infinite Volume will use.

2. Optional: (Workflow Automation) Customize predefined workflows for storage class definitions by using Workflow Automation.
3. If necessary, create aggregates for the Infinite Volume that meet the requirements of the storage classes that you want to use for the Infinite Volume.

4. Create an Infinite Volume with storage classes by performing one of the following actions:
   • (Workflow Automation) Use a workflow to convert an Infinite Volume without storage classes to an Infinite Volume with storage classes. The Infinite Volume contains two storage classes: all existing data is grouped into a legacy storage class, and a new storage class of your choice is added.
   • (Workflow Automation) Use workflows to create an Infinite Volume with storage classes. The workflow allows you to specify an aggregate list. You can use the aggregate list to specify the aggregates and nodes that the storage classes should span. The SVM with Infinite Volume that contains an Infinite Volume with storage classes and a data policy is created. Data LIFs are configured for the SVM with Infinite Volume, and file access is configured.

5. Configure file access to support the infrastructure for new rules in the data policy:
   • If you want to filter data based on its location in a directory, use a client to create directories in the Infinite Volume's namespace.
   • If you want to filter data based on who owns the data, configure user names for Data ONTAP to use.

6. Modify the data policy when the Infinite Volume uses two or more storage classes:
   a. (Unified Manager) Modify the data policy for the Storage Virtual Machine (SVM) with Infinite Volume to add rules for filtering data into all storage classes. See the OnCommand Unified Manager Online Help.
   b. (Unified Manager) Activate the data policy for the SVM with Infinite Volume. See the OnCommand Unified Manager Online Help. The data policy uses its rules to filter data written to the Infinite Volume into the appropriate storage classes.

7. Optional: (Workflow Automation) Use a workflow to configure data protection for the Infinite Volume with storage classes.

8. Optional: Set up tape backup of the Infinite Volume with storage classes.

9. Monitor space usage for the storage classes, and expand the Infinite Volume if required:
   a. (Unified Manager) Monitor space usage for storage classes. See the OnCommand Unified Manager Online Help.
   b. (CLI or Workflow Automation) Add or expand aggregates to meet the space guarantee as required.
   c. (Workflow Automation) Add more storage classes or expand existing storage classes to increase the size of the Infinite Volume.

Related concepts
   - What a storage class is on page 15
   - Definitions for the default storage classes on page 198
   - What rules and data policies are on page 16
   - Using Infinite Volumes with storage classes on page 198
Planning to create Infinite Volumes

Before you create an Infinite Volume, you should decide what features to enable and understand how the enabled features affect the Infinite Volume. You should also understand how Infinite Volumes use aggregates to better plan aggregates.

Related concepts

Planning to create Infinite Volumes with storage classes on page 198

Considerations when planning aggregate space

Before you create an Infinite Volume, you might want to consider how the following features affect aggregate space: thin provisioning, incremental tape backup, or a data protection mirror relationship with an Infinite Volume that uses a platform with a smaller maximum data constituent size.

Each of the features affects the creation of an Infinite Volume in the following ways:

- With thin provisioning, the size of the Infinite Volume is not tied directly to the available physical space.
- With incremental tape backup, some space in the Infinite Volume is dedicated to additional namespace mirror constituents.
- If the Infinite Volume will be in a data protection mirror relationship with an Infinite Volume that uses a smaller maximum data constituent size, you must restrict the size of the data constituents in the source Infinite Volume when you create the source Infinite Volume.

Considerations when using thin provisioning with Infinite Volumes

You can use thin provisioning with an Infinite Volume, enabling you to allocate more storage to users than is physically available. Before using thin provisioning, you should understand what it is, where and when it is configured, and what its advantages and disadvantages are.

What thin provisioning is

With thin provisioning, the size of an Infinite Volume is not limited by the size of its associated aggregates. You can create a large volume on a small amount of storage, adding disks only as they are required. For example, you can create a 500 TB volume using aggregates that only have 250 TB of available space. The storage provided by the aggregates is used only as data is written. Thin provisioning is also called aggregate overcommitment.

The alternative of thin provisioning is thick provisioning, which allocates physical space immediately, regardless of whether that space is used for data yet. The allocated space cannot be used by any other volumes. When you use thick provisioning, all of the space required for the volume is allocated from the aggregate at the time of creating the volume.

Thin provisioning affects only the data constituents of an Infinite Volume. The namespace constituent and namespace mirror constituents of an Infinite Volume always use thick provisioning. For example, if you create a new Infinite Volume with a 10 TB namespace constituent and use thin provisioning, the namespace constituent will consume 10 TB of space even if the Infinite Volume does not contain any data.
When and where thin provisioning is configured

The way that you configure thin provisioning on an Infinite Volume depends on whether the Infinite Volume uses storage classes.

For an Infinite Volume without storage classes, thick and thin provisioning are configured at the volume level in the following way:

• If you want to use thin provisioning, you typically specify it when you first create the Infinite Volume.
  By default, an Infinite Volume created through the command line uses thick provisioning.

• You can switch between thick and thin provisioning after the Infinite Volume is created.
  If you change the setting later, you cannot change the Infinite Volume's size at the same time; you must change the size and guarantee of an Infinite Volume in separate operations.
  Before changing a volume from thin provisioning to thick provisioning, you must ensure that the physical storage can support the provisioned size.

• You can configure thick and thin provisioning in the command line by using the -space-guarantee parameter of the volume create or the volume modify command.
  The value none represents thin provisioning and the value volume represents thick provisioning.

For an Infinite Volume with storage classes, thick and thin provisioning are configured at the storage-class level in the following way:

• You can choose to use thick or thin provisioning for each storage class independent of other storage classes.
  For example, one storage class of an Infinite Volume can use thin provisioning while another storage class of the same Infinite Volume uses thick provisioning. When the guarantee differs across storage classes, the guarantee for the entire Infinite Volume is displayed as a dash (—).

• All configuration of thick and thin provisioning is performed by using workflows OnCommand Workflow Automation.

• If an Infinite Volume uses storage classes, it is not possible to configure thick or thin provisioning at the Infinite Volume level.
  The -space-guarantee parameter of the volume create and the volume modify commands is disabled for an Infinite Volume with storage classes.

Advantages of thin provisioning

Using thin provisioning with Infinite Volumes provides the following advantages:

• Defers physical storage costs until the storage is actually required
  Users receive the space allocation that they expect, and valuable resources do not remain unused.

• Facilitates monitoring of aggregate usage
  When you use thin provisioning, information about aggregate usage—for example, the Used Size, Used Percentage, and Available Size—reflects the actual space used to store data.
  When you use thick provisioning, aggregate usage information reflects the allocated space, which typically differs from the space that is actually used to store data.

• In some cases, eliminates the need to change the volume size after adding disks
  If you add more disks to existing aggregates, you do not have to resize the Infinite Volume to make use of the added capacity as long as the total size of the Infinite Volume's associated aggregates is less than the Infinite Volume's size.

Disadvantages of thin provisioning

Thin provisioning includes the following disadvantages:
• If you have overcommitted your aggregate, you must monitor your available space and add storage to the aggregate as needed to avoid write errors due to insufficient space.

• In a multi-tenancy environment, if you share aggregates among volumes that use thin provisioning, be aware that one tenant's aggregate space availability can be adversely affected by the growth of another tenant's volumes.

• The process of balancing incoming files across data constituents is less effective when an Infinite Volume uses thin provisioning because the reported percentage of used space does not always represent the physical used space.

**More information about thin provisioning**

For more information about thin provisioning, see *Thin Provisioning Deployment and Implementation Guide (TR-3965)*.

**Related concepts**

- *How deduplication works on Infinite Volumes* on page 66
- *How compression works on Infinite Volumes* on page 67
- *How efficiency works on Infinite Volumes with storage classes* on page 199

**Related information**


**Space considerations for using incremental tape backup**

If you want to use incremental tape backup with an Infinite Volume, you should be aware that up to 10 TB of space per node is required starting with the third node that an Infinite Volume uses.

Incremental tape backup requires you to enable SnapDiff, which automatically creates a namespace mirror constituent on each node that the Infinite Volume uses that does not already have either a namespace constituent or a namespace mirror constituent. Because every Infinite Volume already has one namespace constituent and one namespace mirror constituent, SnapDiff requires new namespace mirror constituents on every node beyond two.

Each namespace mirror constituent is the same size as the namespace constituent, which can require up to 10 TB of space.

If you plan to enable SnapDiff eventually, the best practice is to enable it when you first create the Infinite Volume.

You can also enable SnapDiff after the Infinite Volume is created. If you enable SnapDiff on an existing Infinite Volume, you must do so in an operation that is separate from any resize operation.

**Related concepts**

- *Providing tape backup and restore of Infinite Volumes* on page 139
- *Providing tape backup and restore of Infinite Volumes with storage classes* on page 227

**Constituent size considerations for data protection mirror relationships for Infinite Volumes**

When the source and destination Infinite Volumes are on platforms that support different maximum data constituent sizes, you must know the maximum data constituent sizes for the different platforms to successfully set up a data protection mirror relationship for Infinite Volumes.

The maximum data constituent size for an Infinite Volume corresponds to the maximum FlexVol volume size for the platform. You must correctly set the maximum data constituent size when you
create the source and the destination Infinite Volumes. For example, if you want to create a data protection mirror relationship between a source Infinite Volume on a platform with a large data constituent size and a destination Infinite Volume on a platform with a small data constituent size, the size of the constituents in the source and destination Infinite Volumes is restricted to the smaller maximum data constituent size. You must set the maximum data constituent size for the source and destination Infinite Volumes to be the size of the smaller maximum data constituent size for the two platforms.

You should set the maximum data constituent size when you create the source and destination Infinite Volumes. You can modify the maximum data constituent size for an Infinite Volume. However, you cannot use the setting to shrink the size of existing data constituents. If the current data constituent size is larger than the size that you want to specify, the setting cannot shrink the existing data constituents to the smaller size.

**Related tasks**

*Creating an Infinite Volume that supports data protection on a different platform* on page 59

**Related references**

*Size exceeds maximum (namespace constituent)* on page 136

*Size exceeds maximum (data constituents)* on page 137

**Related information**

*NetApp Hardware Universe*

---

**Planning aggregates for an Infinite Volume**

By learning how an Infinite Volume uses aggregates, you can better prepare aggregates and select them specifically for use by an Infinite Volume.

When you create an Infinite Volume, Data ONTAP automatically selects the best aggregates for each Infinite Volume constituent. While you can control many aspects of how an Infinite Volume uses aggregates, you can also allow Data ONTAP to determine aggregate use.

**How FlexVol volumes and Infinite Volumes share aggregates**

Aggregates can be shared among the volumes in a cluster. Each aggregate can contain multiple FlexVol volumes alongside multiple constituents of Infinite Volumes.

When you create an Infinite Volume, constituents of the Infinite Volume are placed on aggregates that are assigned to its containing SVM. If the SVM with Infinite Volume includes aggregates that contain FlexVol volumes, one or more of the Infinite Volume's constituents might be placed on aggregates that already include FlexVol volumes, if those aggregates meet the requirements for hosting Infinite Volumes.

Similarly, when you create a FlexVol volume, you can associate that FlexVol volume with an aggregate that is already being used by an Infinite Volume.

The following diagram illustrates aggregate sharing in a four-node cluster that includes both FlexVol volumes and an Infinite Volume. The Infinite Volume uses the aggregates aggrA, aggrB, aggrC, aggrD, aggrE, and aggrG even though the aggregates aggrB, aggrC, and aggrG already provide storage to FlexVol volumes. (For clarity, the individual constituents that make up the Infinite Volume are not shown.)
How aggregates and nodes are associated with Infinite Volumes

The aggregate list of the containing Storage Virtual Machine (SVM) with Infinite Volume determines which aggregates the Infinite Volume uses, as well as who can create an Infinite Volume and which nodes the Infinite Volume uses.

The aggregate list can be specified or unspecified, which is represented as a dash ("-"). By default, when a cluster administrator creates any SVM, its aggregate list is unspecified. After the SVM is created, the cluster administrator can specify the aggregate list by using the `vserver add-aggregates` command.

Considerations when choosing to specify the aggregate list or leave it unspecified

If you are dedicating an entire cluster to the SVM with Infinite Volume, you can leave the aggregate list of an SVM with Infinite Volume unspecified. In most other situations, you should specify the aggregate list of an SVM with Infinite Volume.

Leaving the aggregate list of an SVM with Infinite Volume unspecified has the following outcomes:

- Only a cluster administrator can create the Infinite Volume, not an SVM administrator.
- When the Infinite Volume is created, it uses all nodes in the cluster.
- When the Infinite Volume is created, it can potentially use all of the aggregates in the cluster.

How the aggregate list contains candidate aggregates

The aggregate list of an SVM with Infinite Volume acts only as a candidate aggregate list for an Infinite Volume. An Infinite Volume uses aggregates according to various factors, including the following requirements:

- When an Infinite Volume is created, at least one data constituent is created on at least one aggregate from each node in the aggregate list.
- An Infinite Volume uses only the aggregates that it requires to meet the capacity requirements for its specified size.
  
  If the assigned aggregates have far greater capacity than the Infinite Volume requires when it is first created, some aggregates in the aggregate list might not contain any Infinite Volume constituents.

How the aggregate list determines the nodes

An Infinite Volume uses every node that has an aggregate in the aggregate list of an SVM with Infinite Volume.
When changes to the aggregate list take effect

Changes to the aggregate list do not have any immediate effect. The aggregate list is used only when the size of an Infinite Volume changes. For example, if you add an aggregate to the aggregate list of an SVM with Infinite Volume, that aggregate is not used until you modify the size of the Infinite Volume.

If you add aggregates from a new node to the aggregate list and then resize the Infinite Volume, whether the Infinite Volume uses the aggregates from the new node depends on several variables, including the size of existing constituents and how much the Infinite Volume was increased in size.

How the aggregate list can be filtered

You can filter the aggregate list for the SVM by using advanced parameters that control which aggregates are used for each type of constituent, such as data constituents. Unlike the aggregate list for the SVM, these aggregate-selection parameters apply only to a single operation. For example, if you use the parameter for data constituent aggregates when you create the Infinite Volume and then resize the Infinite Volume without using the parameter, the Infinite Volume uses the SVM aggregate list.

How much space namespace-related constituents require

The namespace constituent and namespace mirror constituents can each consume up to 10 TB of an Infinite Volume's capacity on two or more nodes, depending on the Infinite Volume's size and configuration. The requirements of these namespace-related constituents affect how space is allocated when you create or expand an Infinite Volume.

In most cases, the namespace constituent is 10 TB and never larger than 10 TB.

However, the namespace constituent is smaller in Infinite Volumes that are relatively small. In small Infinite Volumes, the namespace constituent size is configured so that the combined size of the namespace-related constituents (which includes the namespace constituent and one or more namespace mirror constituents) is equal to 25% of the Infinite Volume's size. For example, if an Infinite Volume is 60 TB and has one namespace mirror constituent, the combined size of the namespace constituent and namespace mirror constituent must be 25% of 60 TB, or 15 TB. That makes the namespace constituent 7.5 TB.

If 25% of the Infinite Volume's size would result in namespace-related constituents being larger than 10 TB, the maximum size takes precedence, and each namespace-related constituent is 10 TB.

Related concepts

Space and node requirements for namespace mirror constituents on page 147

How data constituents use aggregate space

Each data constituent is created as large as possible within the following constraints: a hardware-related maximum size, the space available in the containing aggregate, and the requirement to balance capacity across nodes.

Data ONTAP attempts to make every data constituent as big as possible within the following constraints:

- A hardware-related maximum size that is identified in the Hardware Universe.
  - If the Infinite Volume uses nodes from multiple platforms, the smallest value is used for all data constituents on all nodes.

- The available space on its containing aggregate.
  - The available space on an aggregate is determined by the aggregate's size and the space that is used by other constituents and volumes that share the aggregate. If an aggregate is shared with
other Storage Virtual Machines (SVMs), it can already contain FlexVol volumes, other constituents for the same Infinite Volume, and constituents of other Infinite Volumes.

- The requirement to balance the Infinite Volume's capacity across nodes.

For example, if the maximum data constituent size on a given platform is 100 TB, Data ONTAP attempts to make each data constituent 100 TB.

If data constituents cannot be created at the maximum size, Data ONTAP creates smaller data constituents, to a minimum possible size of 1 TB.

**Methods to protect the namespace constituent of an Infinite Volume**

In addition to the namespace mirror constituent, you can also use mirrored aggregates to protect the namespace constituent of an Infinite Volume. When configured correctly, mirrored aggregates can provide zero RPO for the namespace constituent.

With mirrored aggregates, disks are split between two plexes, and data from one plex is automatically mirrored to the second plex. If one plex of the aggregate becomes unavailable, the second plex is automatically used with zero recovery point objective (RPO).

You can use a mirrored aggregate for the namespace constituent of an Infinite Volume. When you create a mirrored aggregate for a namespace constituent, you must split the plexes between two nodes of an HA pair and assign aggregates from both nodes to the Storage Virtual Machine (SVM) with Infinite Volume.

When you create the Infinite Volume, you can specify the mirrored aggregate for the namespace constituent. If the plex that contains the aggregate for the namespace constituent goes offline, the storage system automatically switches to the namespace constituent contained by the plex on the second node of the HA pair with zero RPO.

Although the namespace constituent is always protected by the namespace mirror constituent, using mirrored aggregates is an optional, additional method of protecting the namespace constituent. The namespace mirror constituent provides five-minute RPO to any node in the cluster that the Infinite Volume spans.

**Related concepts**

- *What a namespace mirror constituent is* on page 14

**Related tasks**

- *Recovering individual failed constituents* on page 127

**Related information**

- *Disk and aggregate management*

**How node balancing affects an Infinite Volume's size and aggregate use**

New data constituents of an Infinite Volume are balanced across nodes. This means that the node with the smallest available space determines how much space is used on each node and limits the size of the Infinite Volume that you can create or expand.

**What node balancing is**

When an Infinite Volume is created or expanded, Data ONTAP ensures that the total size of the volume's data constituents is the same on every node that an Infinite Volume uses. For example, in a 6 PB, six-node Infinite Volume, Data ONTAP attempts to distribute the data constituents so that each node holds 1 PB of data constituents.
The node-balancing requirement means that the size of data constituents on each node is limited by the node with the smallest available space. If one node has only 0.5 PB of available space, every other node used by the Infinite Volume can hold only 0.5 PB of data constituents.

**How node balancing affects an Infinite Volume’s size**

Because the node-balancing requirement controls the amount of data constituents on a node, it also significantly restricts the overall size of the Infinite Volume—both when you first create the volume and when you try to expand it.

For example, if you try to create a 6 PB, six-node Infinite Volume but one of the nodes used by the Infinite Volume has only 0.5 PB of available space, each node can hold only 0.5 PB of data constituents, limiting the total size of the Infinite Volume to approximately 3 PB.

You can roughly determine an Infinite Volume’s largest possible size by determining which node has the least amount of available space and multiplying that amount by the number of nodes. You can also determine the largest possible size by running the `volume create` or `volume modify` commands; if the requested size cannot be created, the resulting error message indicates the largest possible size of the Infinite Volume given the existing resources.

**How node balancing affects the expansion of an Infinite Volume**

The node-balancing requirement persists when you expand an Infinite Volume. The source of the new capacity does not necessarily determine where data capacity is placed. Data ONTAP adds data capacity to nodes in a way that equalizes the total size of all data constituents on each node.

In most cases, new data constituents are created. In some circumstances, such as an Infinite Volume that is upgraded from Data ONTAP 8.1, existing data constituents might be expanded evenly until one of them reaches the maximum data constituent size.

**How space is allocated inside a new Infinite Volume**

Several rules govern how space is allocated to constituents when an Infinite Volume is created. Understanding these rules can help you understand the best practices for configuring aggregates for an Infinite Volume.

The following rules govern how space is allocated to constituents when an Infinite Volume is created:

1. The namespace constituent and its mirror copies are created.
   
   Before any space is allocated for data, the namespace constituent and namespace mirror constituents are created as big as possible within their maximum sizes.
   
   a. The namespace constituent is placed on the aggregate with the most available space on any node that the Infinite Volume uses.
   
   b. The first namespace mirror constituent is placed on the aggregate with the next most available space, as long as the aggregate is on a node that meets all of the following conditions:
      
      • The node is used by the Infinite Volume.
      • It does not already contain the namespace constituent.
      • It is preferably not the partner node in the HA pair of the node that contains the namespace constituent.
   
   c. If SnapDiff is enabled, additional namespace mirror constituents are placed on the aggregate with the most available space on each remaining node used by the Infinite Volume.

2. The data capacity is divided equally among the nodes that the Infinite Volume uses.
   
   The data capacity of an Infinite Volume is balanced across nodes. Data capacity is the space remaining from the Infinite Volume’s size after deducting the space required by the namespace-related constituents.
3. Within each node, individual data constituents are made as big as possible within a specified maximum.
   Data constituents are always created as big as they are allowed to be within a specified maximum. Each time that Data ONTAP creates a data constituent, it evaluates all of the aggregates that the Infinite Volume uses on the node and selects the aggregate that has the most available space.

Related concepts

- How much space namespace-related constituents require on page 35
- How node balancing affects an Infinite Volume's size and aggregate use on page 36
- How data constituents use aggregate space on page 35
- Examples of how space is allocated in Infinite Volumes on page 38

Examples of how space is allocated in Infinite Volumes

Reviewing examples of how space is allocated in Infinite Volumes of various configurations can help you understand the layout of the constituents in an Infinite Volume and therefore plan your own Infinite Volume.

### Example of a two-node, 200 TB Infinite Volume

Space is allocated in a two-node, 200 TB Infinite Volume in the following manner, assuming that the platform's maximum data constituent size is 100 TB and that each node has at least two aggregates which each have at least 200 TB of space that is usable by the Infinite Volume:

1. The namespace constituent is 10 TB and the namespace mirror constituent is 10 TB, consuming 20 TB of the 200 TB Infinite Volume.
   In a two-node Infinite Volume, there are two namespace-related constituents: one namespace constituent and one namespace mirror constituent. They are each 10 TB, the absolute maximum size of a namespace constituent.

2. Each node has 90 TB of data constituents.
   After using 20 TB for namespace-related constituents, 180 TB of the Infinite Volume's 200 TB capacity remains. This remaining capacity is divided evenly over the two nodes.

3. Within each node, one aggregate has a 90 TB data constituent.
   The entire 90 TB of data constituent space for the node will be allocated to a single data constituent on a single aggregate. No other aggregates are used.

### Example of a two-node, 250 TB Infinite Volume

Space is allocated in a two-node, 250 TB Infinite Volume in the following manner, assuming that the platform's maximum data constituent size is 100 TB and that each node has at least two aggregates which each have at least 200 TB of space that is usable by the Infinite Volume:

1. The namespace constituent is 10 TB and the namespace mirror constituent is 10 TB, consuming 20 TB.
   In a two-node Infinite Volume, there are two namespace-related constituents: one namespace constituent and one namespace mirror constituent. They are each 10 TB, the absolute maximum size of a namespace constituent.

2. Each node has 115 TB of data constituents.
   After using 20 TB for namespace-related constituents, 230 TB of the Infinite Volume's 250 TB capacity remains. This remaining capacity is divided evenly over the two nodes.

3. Within each node, two data constituents are created—a 100 TB data constituent and a 15 TB data constituent.
   Of the aggregates that the Infinite Volume uses on the node, the one with the most available space receives a data constituent of the maximum possible size, which is 100 TB. Of the
same set of aggregates, the one that now has the most available space receives a data constituent with the remaining data constituent capacity, which is 15 TB.

**Example of a six-node, 6 PB Infinite Volume with SnapDiff enabled**

Space is allocated in a six-node, 6 PB (6144 TB) Infinite Volume in the following manner, assuming that SnapDiff is enabled, that the platform's maximum data constituent size is 100 TB, and that each node has at least four aggregates which each have 400 TB of space that is usable by the Infinite Volume:

1. The namespace constituent is 10 TB and each namespace mirror constituent is 10 TB, consuming 60 TB in total.

   In a six-node Infinite Volume with SnapDiff enabled, there are six namespace-related constituents: one namespace constituent and five namespace mirror constituents. They are each 10 TB, the absolute maximum size of a namespace constituent.

2. Each node has approximately 1014 TB of data constituents.

   After using 60 TB for namespace-related constituents, 6084 TB of the Infinite Volume's capacity remains. This remaining capacity is divided evenly over the 6 nodes.

3. Within each node, 11 data constituents are created—10 that are 100 TB and one that is 14 TB.

   Data ONTAP creates data constituents in the following manner, reevaluating the available space on the aggregates that the Infinite Volume uses on the node before creating each data constituent:

   a. Each of the four aggregates receives a 100 TB data constituent, consuming a total of 400 TB of data constituent capacity.

   b. Each of the four aggregates receives a second 100 TB data constituent, consuming a total of 800 TB of data constituent capacity.

   c. Two aggregates each receive a third 100 TB data constituent, consuming a total of 1000 TB of data constituent capacity.

   d. The third aggregate receives a third data constituent that contains the 14 TB leftover data constituent capacity.

**Related concepts**

*How space is allocated inside a new Infinite Volume* on page 37

**Completing the aggregate planning worksheet**

You can use a worksheet to plan aggregates for Infinite Volumes with and without storage classes. If you plan to create a data protection mirror copy of the Infinite Volume, you can also use the worksheet to plan aggregates for the destination Infinite Volume.

If you plan to create a data protection mirror copy of an Infinite Volume in another cluster, the mirror copy is called the *destination Infinite Volume*, and the original Infinite Volume is called the *source Infinite Volume*.

**Planning aggregates for Infinite Volumes**

You can use the following information to plan aggregates for Infinite Volumes. If you plan to create a data protection mirror copy of the Infinite Volume, you can use the following information to plan aggregates for the source Infinite Volume.
<table>
<thead>
<tr>
<th>Type of information</th>
<th>Consideration</th>
<th>For more information...</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of platforms the Infinite Volume will span</td>
<td>An Infinite Volume can span different platforms in a cluster.</td>
<td>Platforms supported by Infinite Volumes on page 20</td>
</tr>
<tr>
<td>Lowest maximum data constituent size supported by the smallest platform</td>
<td>The platform size can affect the maximum size of data constituents in an Infinite Volume.</td>
<td>How data constituents use aggregate space on page 35</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Constituent size considerations for setting up data protection mirror relationships for Infinite Volumes on page 32</td>
</tr>
<tr>
<td>Number of nodes the Infinite Volume will span in the cluster</td>
<td>An Infinite Volume can span multiple nodes in a cluster.</td>
<td>Node requirements for Infinite Volumes on page 20</td>
</tr>
<tr>
<td>Number of aggregates on each node</td>
<td>An Infinite Volume uses storage from multiple aggregates.</td>
<td>Node requirements for Infinite Volumes on page 20</td>
</tr>
<tr>
<td>Type and size of aggregates on each node</td>
<td>An Infinite Volume requires specific types of aggregates and a minimum aggregate size.</td>
<td>Aggregate requirements for Infinite Volumes on page 22</td>
</tr>
<tr>
<td>Aggregate sharing</td>
<td>Consider whether aggregate space in a cluster will be dedicated to the Infinite Volume or shared with other volumes in the cluster.</td>
<td>How FlexVol volumes and Infinite Volumes share aggregates on page 33</td>
</tr>
<tr>
<td>Features that affect aggregate type</td>
<td>Each storage class used by an Infinite Volume can require a specific type of aggregate.</td>
<td>How storage classes affect which aggregates can be associated with Infinite Volumes on page 199</td>
</tr>
<tr>
<td>Aggregate list for the Storage Virtual Machine (SVM) with Infinite Volume</td>
<td>Unless you are dedicating an entire cluster to the Infinite Volume, an Infinite Volume should have an assigned aggregate list.</td>
<td>How aggregates and nodes are associated with Infinite Volumes on page 34</td>
</tr>
<tr>
<td>Features that affect aggregate space</td>
<td>Some features affect how much aggregate space an Infinite Volume requires.</td>
<td>Considerations when using thin provisioning with Infinite Volumes on page 30</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Space considerations for using incremental tape backup on page 32</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Constituent size considerations for setting up data protection mirror relationships for Infinite Volumes on page 32</td>
</tr>
</tbody>
</table>

**Planning aggregates for destination Infinite Volumes**

If you plan to create a data protection mirror copy of the Infinite Volume, you can use the following information to plan aggregates for the destination Infinite Volume.
<table>
<thead>
<tr>
<th>Type of information</th>
<th>Consideration</th>
<th>For more information...</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of nodes in</td>
<td>A destination Infinite Volume spans multiple nodes in a cluster. You specify which nodes an Infinite Volume uses by assigning aggregates from the nodes to the aggregate list for the SVM with Infinite Volume.</td>
<td><em>Node requirements for Infinite Volumes</em> on page 20</td>
</tr>
<tr>
<td>the destination</td>
<td></td>
<td></td>
</tr>
<tr>
<td>cluster that the</td>
<td></td>
<td></td>
</tr>
<tr>
<td>destination Infinite</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Volume will span</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Number of</td>
<td>You should use the same number of aggregates for the source and destination Infinite Volumes.</td>
<td><em>Aggregate requirements for Infinite Volumes</em> in a data</td>
</tr>
<tr>
<td>aggregates on each</td>
<td></td>
<td>protection mirror relationship* on page 22</td>
</tr>
<tr>
<td>node in the</td>
<td></td>
<td></td>
</tr>
<tr>
<td>destination cluster</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Size of aggregates</td>
<td>The overall size of available aggregates must be large enough to fit a mirror copy of the source Infinite Volume. Namespace mirror constituents are not mirrored to the destination Infinite Volume.</td>
<td></td>
</tr>
<tr>
<td>on each node in the</td>
<td></td>
<td></td>
</tr>
<tr>
<td>destination cluster</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Aggregate sharing</td>
<td>Plan whether the destination Infinite Volume will share aggregate space with other volumes in the destination cluster.</td>
<td><em>How FlexVol volumes and Infinite Volumes share aggregates</em> on page 33</td>
</tr>
<tr>
<td>Features that affect</td>
<td>If the source Infinite Volume uses storage classes, the aggregates in the destination cluster for the destination Infinite Volume must meet the aggregates requirements for the storage classes used by the source Infinite Volume.</td>
<td>*How storage classes affect which aggregates can be</td>
</tr>
<tr>
<td>aggregate type</td>
<td></td>
<td>associated with Infinite Volumes* on page 199</td>
</tr>
<tr>
<td>Aggregate list for</td>
<td>Unless you are dedicating an entire cluster to the destination Infinite Volume, you should specify an aggregate list for the destination SVM with Infinite Volume.</td>
<td><em>Node requirements for Infinite Volumes</em> on page 20</td>
</tr>
<tr>
<td>the destination SVM</td>
<td></td>
<td></td>
</tr>
<tr>
<td>with Infinite Volume</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Features that affect</td>
<td>If you plan to set up incremental tape backup on the destination Infinite Volume, the aggregates for the destination Infinite Volume require enough space to fit all of the namespace mirror constituents. In some cases, this requires the destination SVM with Infinite Volume to have more aggregate space than the source SVM with Infinite Volume.</td>
<td><em>Space considerations for using incremental tape backup</em> on page 32</td>
</tr>
<tr>
<td>aggregate space</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Planning to create Infinite Volumes | 41
Creating aggregates for Infinite Volumes (cluster administrators only)

Before you create an Infinite Volume, you must create aggregates on multiple nodes to provide storage for the volume. If you plan to create a mirror copy of the Infinite Volume, you must also create the aggregates in the destination cluster for the destination Infinite Volume.

Before you begin

• You must know which nodes you want to use in the cluster for the Infinite Volume.

• You should know the maximum data constituent size supported by the platform to create the appropriate size of aggregates for each node.
  If the Infinite Volume will span platforms of different sizes, you must identify the lowest maximum data constituent size supported by the smallest platform because data constituents will be limited to the maximum data constituent size for the smallest platform.

About this task

You should create aggregates that are as large as or bigger than the maximum data constituent size plus 10%. The extra 10% of space enables the system to create the largest possible data constituents when you create the Infinite Volume. For example, if the maximum data constituent size is 100 TB, you should create aggregates of approximately 110 TB or 220 TB, which allows the system to create the largest possible data constituents.

If you plan to create a data protection mirror of the Infinite Volume in another cluster, you should create the same number and size of aggregates in the source and destination clusters.

Step

1. Create aggregates on the nodes that you want to use for the Infinite Volume by using the `storage aggregate create` command.

Related concepts

  * [Node requirements for Infinite Volumes](#) on page 20
  * [Aggregate requirements for Infinite Volumes](#) on page 22
  * [Aggregate requirements for destination Infinite Volumes](#) on page 22

Related tasks

  * [Creating aggregates for Infinite Volumes with storage classes](#) on page 207
  * [Identifying the constituent sizes that platforms support](#) on page 60

Related information

  * [NetApp Hardware Universe](#)
  * [Disk and aggregate management](#)
Creating SVMs to contain Infinite Volumes (cluster administrators only)

Before you can create an Infinite Volume, you must create a specific type of Storage Virtual Machine (SVM) called an SVM with Infinite Volume. You should also assign aggregates to the SVM with Infinite Volume to better control which aggregates in the cluster are used by the Infinite Volume.

Setting up SVMs

Different subtypes of SVMs provide data access, data protection against disasters, and data high availability. You must set up at least one data access SVM per cluster, which involves planning the setup, understanding requirements, completing the setup worksheet, and creating and configuring the SVM.

Planning to create SVMs

You must understand the various aspects of SVMs that help you plan the SVM setup process. You must also understand the purpose for creating the SVM and understand details such as the type of volume the SVM should contain, the environment for data access, and network segregation.

SVMs for data access

You must create one or more SVMs to serve data from the cluster. SVMs can contain either one or more FlexVol Volumes or a single Infinite Volume to serve data to the clients.

You can also use OnCommand System Manager to create and configure SVMs with FlexVol volumes and SVMs with Infinite Volume.

Cluster management using System Manager

You must understand the following SVM attributes when creating SVMs with Infinite Volume for data access:

<table>
<thead>
<tr>
<th>SVM attributes</th>
<th>Possible values</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SVM subtype</td>
<td>default</td>
<td>Data SVMs that are created for data access must have the subtype option set to default.</td>
</tr>
<tr>
<td>Volumes</td>
<td>Infinite Volume</td>
<td>The -is-repository option must be set to true. The default value is false.</td>
</tr>
<tr>
<td>Protocols</td>
<td>• NFS, NFSv3, pNFS, and NFSv4.1 are supported.</td>
<td>Data SVMs can serve data to NAS clients only.</td>
</tr>
<tr>
<td></td>
<td>• CIFS, SMB 1.0 is supported.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SVM attributes</th>
<th>Possible values</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SVM subtype</td>
<td>default</td>
<td>Data SVMs that are created for data access must have the subtype option set to default.</td>
</tr>
<tr>
<td>Volumes</td>
<td>Infinite Volume</td>
<td>The -is-repository option must be set to true. The default value is false.</td>
</tr>
<tr>
<td>Protocols</td>
<td>• NFS, NFSv3, pNFS, and NFSv4.1 are supported.</td>
<td>Data SVMs can serve data to NAS clients only.</td>
</tr>
<tr>
<td></td>
<td>• CIFS, SMB 1.0 is supported.</td>
<td></td>
</tr>
</tbody>
</table>
### Guidelines for creating SVMs

There are naming guidelines, and language and IPspace considerations that you should understand for creating an SVM successfully.

#### SVM naming guidelines

- SVM names must be unique across clusters.
  You must use the fully qualified domain name (FQDN) of the SVM or another convention that ensures unique SVM names across clusters.

- SVM names can have a maximum of 47 characters.
  However, SVMs in a MetroCluster configuration can have a maximum of only 41 characters.

- SVM names are case-sensitive and can contain alphanumeric characters.
  SVM names can contain a period (.), a hyphen (-), or an underscore (_), but must not start with a hyphen, period, or number.

#### Language considerations

The default language setting is `C.UTF-8` (`POSIX.UTF-8`), which is inherited by all of its volumes. You can specify a different language when creating the volume.

When you modify the language of the SVM, the default language setting is modified; the language setting of the existing volumes is not modified.

You should append `.UTF-8` for the language encoding values. For example, for the en_US language, the recommended format is `en_US.UTF-8`.

**Note:** You cannot modify the language of an SVM with Infinite Volume later.

#### IPspace considerations

You must assign an IPspace to SVM when creating the SVM. You cannot modify or remove the IPspace for the SVM later.

#### List of language options

When you create Storage Virtual Machine (SVM), the language is set for the SVM. The language of the SVM determines the default language setting for volumes in that SVM. You can modify the language of an SVM.

You can specify the language for a volume when creating a volume and it can be different from the language of an SVM. If you do not specify the language for a volume then it inherits the language

<table>
<thead>
<tr>
<th>SVM attributes</th>
<th>Possible values</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Root volume security style</td>
<td>• mixed for NFS and CIFS</td>
<td>Based on the clients, you can select the appropriate security style for the SVM root volume.</td>
</tr>
<tr>
<td></td>
<td>• ntfs for CIFS</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• unix for NFS</td>
<td></td>
</tr>
<tr>
<td>IPspace</td>
<td>Any available IPspace</td>
<td>IPspace defines a secured dedicated network path for each SVM.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>You can segregate the network for the SVM by assigning the IPspace when creating the SVM.</td>
</tr>
</tbody>
</table>

---

**For more information, refer to:**

- Page 44: Infinite Volumes Management Guide
Setting of its SVM. After the volume is created, you cannot modify the language of a volume. Therefore, you must be aware of the available language options.

The following table lists the various available language options that helps you choose and enter the correct value when creating an SVM or volume:

<table>
<thead>
<tr>
<th>Language values</th>
<th>Languages</th>
</tr>
</thead>
<tbody>
<tr>
<td>c</td>
<td>POSIX</td>
</tr>
<tr>
<td>C.UTF-8</td>
<td>POSIX with UTF-8</td>
</tr>
<tr>
<td>ar</td>
<td>Arabic</td>
</tr>
<tr>
<td>ar.UTF-8</td>
<td>Arabic with UTF-8</td>
</tr>
<tr>
<td>cs</td>
<td>Czech</td>
</tr>
<tr>
<td>cs.UTF-8</td>
<td>Czech with UTF-8</td>
</tr>
<tr>
<td>da</td>
<td>Danish</td>
</tr>
<tr>
<td>da.UTF-8</td>
<td>Danish with UTF-8</td>
</tr>
<tr>
<td>de</td>
<td>German</td>
</tr>
<tr>
<td>de.UTF-8</td>
<td>German with UTF-8</td>
</tr>
<tr>
<td>en</td>
<td>English</td>
</tr>
<tr>
<td>en.UTF-8</td>
<td>English with UTF-8</td>
</tr>
<tr>
<td>en_us</td>
<td>English (US)</td>
</tr>
<tr>
<td>en_US.UTF-8</td>
<td>US English with UTF-8</td>
</tr>
<tr>
<td>es</td>
<td>Spanish</td>
</tr>
<tr>
<td>es.UTF-8</td>
<td>Spanish with UTF-8</td>
</tr>
<tr>
<td>fi</td>
<td>Finnish</td>
</tr>
<tr>
<td>fi.UTF-8</td>
<td>Finnish with UTF-8</td>
</tr>
<tr>
<td>fr</td>
<td>French</td>
</tr>
<tr>
<td>fr.UTF-8</td>
<td>French with UTF-8</td>
</tr>
<tr>
<td>he</td>
<td>Hebrew</td>
</tr>
<tr>
<td>he.UTF-8</td>
<td>Hebrew with UTF-8</td>
</tr>
<tr>
<td>hr</td>
<td>Croatian</td>
</tr>
<tr>
<td>hr.UTF-8</td>
<td>Croatian with UTF-8</td>
</tr>
<tr>
<td>hu</td>
<td>Hungarian</td>
</tr>
<tr>
<td>hu.UTF-8</td>
<td>Hungarian with UTF-8</td>
</tr>
<tr>
<td>it</td>
<td>Italian</td>
</tr>
<tr>
<td>it.UTF-8</td>
<td>Italian with UTF-8</td>
</tr>
<tr>
<td>ja_v1</td>
<td>Japanese euc-j</td>
</tr>
<tr>
<td>ja_v1.UTF-8</td>
<td>Japanese euc-j with UTF-8</td>
</tr>
<tr>
<td>ja_jp.pck_v2</td>
<td>Japanese PCK (sjis)</td>
</tr>
<tr>
<td>ja_JPPCK_v2.UTF-8</td>
<td>Japanese PCK sjis with UTF-8</td>
</tr>
<tr>
<td>Language values</td>
<td>Languages</td>
</tr>
<tr>
<td>-----------------</td>
<td>----------</td>
</tr>
<tr>
<td>ko</td>
<td>Korean</td>
</tr>
<tr>
<td>ko.UTF-8</td>
<td>Korean with UTF-8</td>
</tr>
<tr>
<td>no</td>
<td>Norwegian</td>
</tr>
<tr>
<td>no.UTF-8</td>
<td>Norwegian with UTF-8</td>
</tr>
<tr>
<td>nl</td>
<td>Dutch</td>
</tr>
<tr>
<td>nl.UTF-8</td>
<td>Dutch with UTF-8</td>
</tr>
<tr>
<td>pl</td>
<td>Polish</td>
</tr>
<tr>
<td>pl.UTF-8</td>
<td>Polish with UTF-8</td>
</tr>
<tr>
<td>pt</td>
<td>Portuguese</td>
</tr>
<tr>
<td>pt.UTF-8</td>
<td>Portuguese with UTF-8</td>
</tr>
<tr>
<td>ro</td>
<td>Romanian</td>
</tr>
<tr>
<td>ro.UTF-8</td>
<td>Romanian with UTF-8</td>
</tr>
<tr>
<td>ru</td>
<td>Russian</td>
</tr>
<tr>
<td>ru.UTF-8</td>
<td>Russian with UTF-8</td>
</tr>
<tr>
<td>sk</td>
<td>Slovak</td>
</tr>
<tr>
<td>sk.UTF-8</td>
<td>Slovak with UTF-8</td>
</tr>
<tr>
<td>sl</td>
<td>Slovenian</td>
</tr>
<tr>
<td>sl.UTF-8</td>
<td>Slovenian with UTF-8</td>
</tr>
<tr>
<td>sv</td>
<td>Swedish</td>
</tr>
<tr>
<td>sv.UTF-8</td>
<td>Swedish with UTF-8</td>
</tr>
<tr>
<td>tr</td>
<td>Turkish</td>
</tr>
<tr>
<td>tr.UTF-8</td>
<td>Turkish with UTF-8</td>
</tr>
<tr>
<td>zh</td>
<td>Simplified Chinese</td>
</tr>
<tr>
<td>zh.UTF-8</td>
<td>Simplified Chinese with UTF-8</td>
</tr>
<tr>
<td>zh.GBK</td>
<td>Simplified Chinese (GBK)</td>
</tr>
<tr>
<td>zh.GBK.UTF-8</td>
<td>Simplified GBK Chinese with UTF-8</td>
</tr>
<tr>
<td>zh_TW</td>
<td>Traditional Chinese euc-tw</td>
</tr>
<tr>
<td>zh_TW.UTF-8</td>
<td>Traditional Chinese euc-tw with UTF-8</td>
</tr>
<tr>
<td>zh_TW.BIG5</td>
<td>Traditional Chinese Big 5</td>
</tr>
<tr>
<td>zh_TW.BIG5.UTF-8</td>
<td>Traditional Chinese Big 5 with UTF-8</td>
</tr>
</tbody>
</table>

**Language configurations**

The language configuration of a Storage Virtual Machine (SVM) or a volume must match the client's language configuration for the file names to appear correctly. If there is a mismatch in the language configuration, then some file names might contain incorrect characters.

The following table helps you identify the language configuration for various clients depending on the client encoding types:
Clients protocol | Client encoding type | Language configuration
---|---|---
CIFS running on Win95/98/ME | ISO 8859-1 | Match non-UTF-8 client locale. Do not append UTF-8 that is 'en_US'.
CIFS running on WinNT 3.1+ | UCS-2 | Unless other clients use non-UTF-8 locales, match UTF-8 client locale. Append UTF-8 that is 'en_US.UTF-8'.
| | | When other clients use non-UTF-8 locales, match non-UTF-8 client locale. Do not append UTF-8 that is 'en_US'.
NFSv2/3 | Non-UTF-8 client locale | Match non-UTF-8 client locale. Do not append UTF-8 that is 'en_US'.
NFSv4 | UTF-8 | Unless other clients use non-UTF-8 locales, match UTF-8 client locale. Append UTF-8 that is 'en_US.UTF-8'.
| | | When other clients use non-UTF-8 locales, match non-UTF-8 client locale. Do not append UTF-8 that is 'en_US'.
FC or iSCSI | | UTF-8 preferred, C/POSIX is acceptable.

**Note:** The default language setting for an SVM is C.UTF-8.

**Creating SVMs with Infinite Volume (cluster administrators only)**

A cluster must have at least one or more SVMs to provide data access from the cluster. You can create SVMs with Infinite Volume to provide data access to large data in the NAS environment.

**Before you begin**

You must have reviewed the planning requirements and understood the guidelines:

- *Planning to create SVMs*
- *Guidelines for creating SVMs*

You must ensure that the following requirements are met:

- The cluster must have at least one non-root aggregate with sufficient space.
- There must be at least 1 GB of space on the aggregate for the SVM root volume.
- The cluster must be synchronized by configuring and enabling NTP to prevent CIFS creation and authentication failures.
- If you want to assign IPspace, you must have created the IPspace.

**About this task**

You can create a maximum of five SVMs simultaneously either by using different SSH sessions or by using a script.
**Note:** It is best to create not more than five SVMs simultaneously to avoid any performance degradation.

You cannot modify the language of an SVM with Infinite Volume.

**Steps**

1. Use the `vserver create` command with the `is-repository` parameter set to `true`.

   **Example**
   
   The following command creates the Storage Virtual Machine (SVM, formerly known as Vserver) with Infinite Volume named vs0.example.com:

   ```
   cluster1::>vserver create -vserver vs1.example.com -rootvolume root_vs0 -aggregate aggr1 -rootvolume-security-style unix -language C.UTF-8 -snapshot-policy default -is-repository true
   
   [Job 2061] Job succeeded: Vserver creation completed
   ```

2. Use the `vserver show` command to verify the status of the newly created SVM.

   **Example**
   
   ```
   cluster1::> vserver show
   Admin      Operational   Root
   Vserver          Type    Subtype    State      State         Volume     Aggregate
   -------          -----   -------    -------   --------     ----------- ----------
   cluster1         admin     -         -           -             -           -
   cluster1-01      node      -         -           -             -           -
   vs1.example.com  data    default    running   running       root_vs0      aggr1
   ```

   SVM creation might fail if intermediate operations such as root volume creation fail, and the SVM will be in the `initializing` state. You must delete the SVM and re-create it.

**Result**

The SVM is created with a root volume of 1 GB, and it is started automatically and is in `running` state. By default, the vsadmin user account is created and is in the `locked` state. The vsadmin role is assigned to the default vsadmin user account.

**Configuring SVMs**

After you create SVMs, you must provision storage, configure the network, services, and protocols to facilitate data access to the clients.

**About this task**

This procedure provides only high-level information about the configuration tasks that you have to perform after creating the SVM. Detailed information about the configuration tasks is available in other clustered Data ONTAP documentation.

*NetApp Documentation: Data ONTAP 8 (current releases)*

**Steps**

1. Specify the aggregates for the SVM for all the volume-related operations that require an aggregate name.

2. Set up a password and unlock the vsadmin user account for delegating the SVM administration.

3. Provide data access to the SVM by performing the following steps:
a. Set up the network interface, such as creating LIFs and routes.

b. Provision storage by creating volumes.

c. Configure the services, such as LDAP, NIS, and DNS.

d. Configure the protocols, such as NFS, CIFS, iSCSI, and FC.

Related tasks

- Creating SVMs with Infinite Volume (cluster administrators only) on page 47
- Assigning shared aggregates to an SVM with Infinite Volume (cluster administrators only) on page 49
- Assigning dedicated aggregates to an SVM with Infinite Volume (cluster administrators only) on page 51

Related information

- System administration
- Network and LIF management
- SMB/CIFS management
- NFS management
- SVM root volume protection express configuration

Assigning shared aggregates to an SVM with Infinite Volume (cluster administrators only)

Before you create an Infinite Volume, you should assign aggregates to the Storage Virtual Machine (SVM) with Infinite Volume. If you want to share aggregates between SVMs in the cluster, you can assign the same aggregate to two or more SVMs in the cluster.

Before you begin

- You must have cluster administrator privileges to perform this workflow.
- You must have created aggregates in the cluster.
- You must have created an SVM with Infinite Volume in the cluster.

Steps

1. Choosing aggregates for an Infinite Volume (cluster administrators only) on page 49
2. Assigning aggregates to an SVM with Infinite Volume on page 50

Choosing aggregates for an Infinite Volume (cluster administrators only)

Before creating an Infinite Volume, you should review the available aggregates and decide which aggregates the Infinite Volume will use.

Steps

1. View all the aggregates in the cluster except for the aggregates that are node root aggregates by using the `storage aggregate show` command with the `-has-mroot false` parameter.

Example

In the following output, the cluster has 6 non-root aggregates:
2. Decide which aggregates the Infinite Volume will use.

When you group the aggregates by node, the total available space on each node should be roughly the same amount. For example, if one node has 112 TB, the second node can have 120 TB, but it should not have 50 TB. If the amount is not balanced, you should consider adding disks to existing aggregates or adding more aggregates.

**Example**

With the aggregates shown in the previous step, you might decide to use four of the available aggregates. If you want to use the same number of aggregates from each node, you can select aggr2 and aggr3 from node 1 and aggr5 and aggr6 from node 2.

3. Of the aggregates that you selected for the Infinite Volume, decide which aggregate will hold the SVM root volume.

You will require the name of this aggregate when you create the Infinite Volume.

**Example**

If you plan to associate the aggregates aggr2, aggr3, aggr5, and aggr6 with the SVM with Infinite Volume, you might decide to place the SVM root volume on the aggr2 aggregate.

**Related information**

*Disk and aggregate management*

### Assigning aggregates to an SVM with Infinite Volume

After creating a Storage Virtual Machine (SVM) for an Infinite Volume, you should assign specific aggregates to it so that the Infinite Volume that you create will use those specific aggregates and not use any aggregate in the cluster.

**Steps**

1. **Assign aggregates to the SVM by using the `vserver add-aggregates` command.**

   You identify which aggregates the Infinite Volume will use by assigning aggregates to its containing SVM with Infinite Volume. If you do not specify the aggregate list for the SVM with Infinite Volume, the Infinite Volume can potentially use all the aggregates in the cluster.

**Example**

The following command assigns several aggregates to the SVM vs0:

```
cluster::> vserver add-aggregates -vserver vs0 -aggregates aggr2,aggr3,aggr5,aggr6
```
2. Verify that the SVM has the assigned aggregates by using the `vserver show` command with the `-instance` and `-vserver` parameters.

   **Example**
   In the following output, the vs0 SVM has its root volume on the aggr2 aggregate and is assigned the aggr2, aggr3, aggr5, and aggr6 aggregates:

   ```
   cluster::> vserver show -instance -vserver vs0
   Vserver: vs0
   ...\
   Aggregate: aggr2
   ...
   List of Aggregates Assigned: aggr2, aggr3, aggr5, aggr6
   ...
   ```

**Assigning dedicated aggregates to an SVM with Infinite Volume (cluster administrators only)**

Before creating an Infinite Volume, you should assign aggregates to the Storage Virtual Machine (SVM) with Infinite Volume. For dedicated aggregates, you must assign the aggregates to the SVM with Infinite Volume, and then ensure that no other SVMs in the cluster use the aggregates.

**Before you begin**

- You must have cluster administrator privileges to perform this workflow.
- You must have created aggregates in the cluster.
- You must have created an SVM with Infinite Volume in the cluster.

**Steps**

1. **Choosing aggregates for an Infinite Volume and all SVMs in the cluster (cluster administrators only)** on page 51
2. **Assigning aggregates for all SVMs in a cluster (cluster administrators only)** on page 53
3. **Assigning aggregates to an SVM with Infinite Volume** on page 54

**Choosing aggregates for an Infinite Volume and all SVMs in the cluster (cluster administrators only)**

Before creating an Infinite Volume with dedicated aggregates, you should review the available aggregates and decide which aggregates every Storage Virtual Machine (SVM) in the cluster will use, including the SVM with Infinite Volume.

**Steps**

1. View all the aggregates in the cluster that can store data by using the `storage aggregate show` command with the `-has-mroot false` parameter.

   Adding the `-has-mroot false` parameter excludes the node root aggregates, which cannot be used by SVMs.

   **Example**
   In the following output, the cluster has 6 non-root aggregates:
You now know which aggregates you can assign to the SVMs.

2. Identify all the data SVMs in the cluster and the aggregates that are assigned to them by using the `vserver show` command with the `-type data` and `-fields aggr-list,aggregate` parameters, and review the output.

For each SVM listed, you must review the `aggr-list` field to determine that aggregates are specified. If the SVM has an unspecified `aggr-list` field, which appears as a dash ("-"), you have to assign aggregates to the SVM.

**Example**

In the following output, the cluster has only one existing data SVM, and its aggregate list must be specified:

```
cluster::> vserver show -type data -fields aggr-list,aggregate
vserver aggregate aggr-list
-------- --------- -------------------------
vs1     aggr1     -
```

3. Decide which aggregates each other SVM will use.

**Example**

With the aggregates and SVMs shown in the previous steps, you might decide to assign aggregates in the following way:

- The existing SVM vs1 uses aggr1 because the SVM root volume is on aggr1.
- The existing SVM vs1 might use aggr4 so that the SVM uses aggregates from two nodes.
- The SVM with Infinite Volume that you plan to create might use the remaining aggregates, which are aggr2, aggr3, aggr5, and aggr6.

4. Decide which aggregates the Infinite Volume will use.

When you group the aggregates by node, the total available space on each node should be roughly the same amount. For example, if one node has 112 TB, the second node can have 120 TB, but it should not have 50 TB. If the amount is not balanced, you should consider adding disks to existing aggregates or adding more aggregates.

**Example**

With the aggregates shown in the previous step, you might decide to use four of the available aggregates. To ensure that you select the same number of aggregates from each node, you can select aggr2 and aggr3 from node 1 and aggr5 and aggr6 from node 2.

5. Of the aggregates that you selected for the Infinite Volume, decide which one will contain the SVM root volume.
You will require the name of this aggregate when you create the Infinite Volume.

Example

If you plan to assign the aggregates aggr2, aggr3, aggr5, and aggr6 to the SVM with Infinite Volume, you might decide to place the SVM root volume on the aggregate aggr2.

Related information

Disk and aggregate management

Assigning aggregates for all SVMs in a cluster (cluster administrators only)

You can ensure that no other Storage Virtual Machines (SVMs) use the aggregates that you want to dedicate to an Infinite Volume by specifying the aggregate lists for all SVMs in the cluster. Without specified aggregate lists, you might not be able to dedicate aggregates to an Infinite Volume.

About this task

This procedure applies when the cluster already contains SVMs. If the cluster does not yet contain any SVMs, you can skip this procedure.

Steps

1. For each SVM in the cluster, specify the aggregates that will be used by the SVM by using the vserver add-aggregates command.

Example

The following command assigns the aggregates aggr1 and aggr4 to the SVM vs1, ensuring that any new FlexVol volumes created on the SVM vs1 can be created only on the aggregates aggr1 or aggr4:

```
cluster::> vserver add-aggregates -vserver vs1 -aggregates aggr1,aggr4
```

2. Verify that all the data SVMs in the cluster have a specified aggregate list by using the vserver show command with the -type data and -fields aggr-list,aggregate parameters.

A specified aggregate list contains the name of an aggregate. An unspecified aggregate list is displayed as a dash ("-").

Example

The following output displays one data SVM with a specified aggregate list:

```
cluster::> vserver show -type data -fields aggr-list,aggregate
vserver aggregate aggr-list
-------- --------- -------------------------
vs1     aggr1     aggr1,aggr4
```

After you finish

Each time that an SVM is created in the future, you must specify its aggregates.
Assigning aggregates to an SVM with Infinite Volume

After creating a Storage Virtual Machine (SVM) for an Infinite Volume, you should assign specific aggregates to it so that the Infinite Volume that you create will use those specific aggregates and not use any aggregate in the cluster.

Steps

1. Assign aggregates to the SVM by using the `vserver add-aggregates` command.
   
   You identify which aggregates the Infinite Volume will use by assigning aggregates to its containing SVM with Infinite Volume. If you do not specify the aggregate list for the SVM with Infinite Volume, the Infinite Volume can potentially use all the aggregates in the cluster.

   Example
   
   The following command assigns several aggregates to the SVM vs0:
   
   ```bash
   cluster::> vserver add-aggregates -vserver vs0 -aggregates aggr2,aggr3,aggr5,aggr6
   ```

   2. Verify that the SVM has the assigned aggregates by using the `vserver show` command with the `-instance` and `-vserver` parameters.

      Example
      
      In the following output, the vs0 SVM has its root volume on the aggr2 aggregate and is assigned the aggr2, aggr3, aggr5, and aggr6 aggregates:
      
      ```bash
      cluster::> vserver show -instance -vserver vs0
      Vserver: vs0
      ...
      Aggregate: aggr2
      ...
      List of Aggregates Assigned: aggr2, aggr3, aggr5, aggr6
      ```
Creating Infinite Volumes

You can create Infinite Volumes with default settings, or you can create Infinite Volumes with specific configurations, such as a dedicated cluster, dedicated aggregates, or a dedicated namespace aggregate. You might also have to create an Infinite Volume that supports data protection on a different platform.

Related tasks

Creating an Infinite Volume with storage classes on page 209

Creating an Infinite Volume using default settings

You can create an Infinite Volume with a simple command, allowing Data ONTAP to determine the size and placement of all the constituents.

Before you begin

• You must have created a Storage Virtual Machine (SVM) with Infinite Volume.
• You must have assigned aggregates to the SVM with Infinite Volume.

About this task

When you create an Infinite Volume without specifying a size, the Infinite Volume is created with the minimum default size of approximately 1.33 TB. The size is approximate due to rounding.

How the Infinite Volume uses aggregates depends on whether you assigned shared or dedicated aggregates to its containing SVM with Infinite Volume.

Steps

1. Create the Infinite Volume by using the `volume create` command with the `-vserver` parameter set for the SVM with Infinite Volume.
   
   If you want to use incremental tape backup, you can include the `-enable-snapdiff` parameter when you use the `volume create` command.

Example

In the following output, an Infinite Volume named ivol is created on the SVM with Infinite Volume named vs0:

```
cluster::> volume create -vserver vs0 -volume ivol -size 200TB
[Job 125] Creating constituent "ivol_ns" on aggregate "aggr2".
[Job 125] Creating constituent "ivol_ns_mirror0001" on aggregate "aggr5".
[Job 125] Creating constituent "ivol_1024_data0001" on aggregate "aggr3"
[Job 125] Creating constituent "ivol_1024_data0002" on aggregate "aggr6" ...
[Job 125] Job succeeded: Created Infinite Volume successfully.
```

After the Infinite Volume is created and is online, it is mounted. By default, it has a junction path of `/NS` and uses thick provisioning.

2. Verify that the Infinite Volume is created and is online by using the `volume show` command with the `-volume` parameter.
Example

The following output shows that the volume ivol is online:

```
cluster::> volume show -volume ivol
Vserver   Volume       Aggregate    State      Type       Size  Available Used%
--------- ------------ ------------ ---------- ---- ---------- ---------- ----- 
vs0       ivol         -            online     RW        200TB      190TB    5%
```

After you finish

You can optionally set up efficiency technologies, and then set up file access.

Related concepts

Creating SVMs to contain Infinite Volumes (cluster administrators only) on page 43

Related tasks

Assigning dedicated aggregates to an SVM with Infinite Volume (cluster administrators only) on page 51
Assigning shared aggregates to an SVM with Infinite Volume (cluster administrators only) on page 49
Managing size-related error messages when you create or expand an Infinite Volume on page 64

Creating an Infinite Volume with a dedicated namespace aggregate

If you want the namespace constituent of an Infinite Volume to have exclusive use of an aggregate, you can use advanced aggregate-selection parameters when you create the Infinite Volume and then modify the aggregate list of all other Storage Virtual Machines (SVMs) to omit the aggregate that is used by the namespace constituent.

Before you begin

- All required aggregates must be created.
- You must have cluster administrator privileges to perform some of the tasks in this workflow.
- You must have created an SVM with Infinite Volume.
- You must have assigned dedicated aggregates to the SVM with Infinite Volume.

About this task

- This workflow applies only to an Infinite Volume without storage classes.
  If you want to create an Infinite Volume with storage classes, you cannot use the command-line interface; you must use OnCommand Workflow Automation instead.
- After using this method of creating an Infinite Volume, the Infinite Volume has exclusive access to its aggregates, and the namespace constituent is on a separate aggregate from all other constituents.

Steps

1. Choosing aggregates for an Infinite Volume's namespace constituent (cluster administrators only) on page 57
2. Creating an Infinite Volume with manually selected aggregates on page 57
3. **Preserving an aggregate for the namespace constituent during expansion (cluster administrators only)** on page 58

**Related concepts**

*Creating SVMs to contain Infinite Volumes (cluster administrators only)* on page 43

**Related tasks**

*Assigning dedicated aggregates to an SVM with Infinite Volume (cluster administrators only)* on page 51

### Choosing aggregates for an Infinite Volume’s namespace constituent (cluster administrators only)

If you want to create an Infinite Volume with a dedicated namespace aggregate, you should first review the aggregates available to the Infinite Volume and decide which ones you want to use for each type of constituent.

**Steps**

1. **Identify an aggregate for the namespace constituent.**
   
   The aggregate must be one that is assigned to the Storage Virtual Machine (SVM) with Infinite Volume. It should have a minimum of 11 TB of space and enough disks to meet your performance requirements.

2. **Review the available space on the remaining aggregates that are assigned to the SVM, and change the aggregates if necessary.**
   
   When you group the remaining aggregates by node, the total available space on each node should be roughly the same amount. For example, if one node has 112 TB, the second node can have 120 TB, but it should not have 50 TB. If the amount is not balanced, you should consider adding disks to existing aggregates or adding more aggregates.

**Related information**

*Disk and aggregate management*

### Creating an Infinite Volume with manually selected aggregates

With advanced privilege, when you create an Infinite Volume, you can specify the aggregates where namespace and data constituents are created. This enables you to dedicate an entire aggregate to the namespace constituent.

**Steps**

1. **Switch to advanced privilege by using the `set privilege advanced` command.**
   
   The command prompt changes to `cluster:*>`.  

2. **Create the Infinite Volume by using the `volume create` command with the `-namespace-aggregate` and `-data-aggr-list` parameters.**
   
   If you want to use incremental tape backup, you can include the `-enable-snapdiff` parameter when you use the `volume create` command.

**Example**

The following command creates an Infinite Volume with the namespace constituent on `aggr1` aggregate and data constituents on the `aggr2`, `aggr3`, `aggr5`, and `aggr6` aggregates:
After the Infinite Volume is created and goes online, it is mounted. By default, it has a junction path of `/NS` and uses thick provisioning.

3. Verify that the Infinite Volume is created and is online by using the `volume show` command with the `-volume` parameter.

**Example**

The following output shows that the volume `ivol` is online:

```
cluster::> volume show -volume ivol

Vserver   Volume       Aggregate    State      Type       Size  Available Used%
--------- ------------ ------------ ---------- ---- ---------- ---------- -----
vs0       ivol         -            online     RW        200TB      190TB    5%
```

4. Verify that the constituents are on the aggregates that you wanted by using the `volume show` command with the `-vserver` and `-is-constituent true` parameters.

5. Return to admin privilege by using the `set privilege admin` command.

The command prompt changes back to `cluster::>`.`

**After you finish**

You can optionally set up efficiency technologies, and then set up file access.

**Related tasks**

- *Managing size-related error messages when you create or expand an Infinite Volume* on page 64
- *Preserving an aggregate for the namespace constituent during expansion* (cluster administrators only)

If you want to stop Data ONTAP from creating new constituents on the aggregate that holds the namespace constituent when the Infinite Volume is resized, you can remove the namespace aggregate from the aggregate list of the Storage Virtual Machine (SVM) with Infinite Volume.

**Steps**

1. Prepare the revised list of aggregates that you will assign to the SVM with Infinite Volume by making a list of all the currently assigned aggregates and removing the namespace aggregate from the list.

**Example**

If the aggregate list of the SVM with Infinite Volume currently includes the aggregates `aggr1`, `aggr2`, `aggr3`, `aggr5`, and `aggr6`, and the namespace constituent is on the aggregate `aggr1`, you assign the aggregates `aggr2`, `aggr3`, `aggr5`, and `aggr6` to the SVM.

2. Assign the new, smaller set of aggregates with the SVM by using the `vserver add-aggregates` command.
Example

The following command assigns the aggregates aggr2, aggr3, aggr5, and aggr6 to the SVM with Infinite Volume:

```
cluster::> vserver add-aggregates -vserver vs0 -aggregates aggr2,aggr3,aggr5,aggr6
```

Result

If the Infinite Volume is later expanded, the existing namespace constituent remains on its aggregate, and new constituents are not created on that aggregate.

Creating an Infinite Volume that supports data protection on a different platform

If you plan to set up a data protection mirror relationship for an Infinite Volume where the platform that contains the destination Infinite Volume is smaller than the platform that contains the source Infinite Volume, you must create the source Infinite Volume with small constituents.

Before you begin

- All required aggregates must be created.
- You must have cluster administrator privileges to perform some of the tasks in this workflow.
- You must have created a Storage Virtual Machine (SVM) with Infinite Volume.
- You must have assigned aggregates to the SVM with Infinite Volume.

About this task

- This workflow applies only to an Infinite Volume without storage classes.
  - If you want to create an Infinite Volume with storage classes, you cannot use the command-line interface; you must use OnCommand Workflow Automation instead.

- By using this method of creating an Infinite Volume, the constituents are kept small enough to allow the Infinite Volume to be in a data protection mirror relationship with a destination Infinite Volume on a system that requires smaller constituents.
  - The size of the constituents for the source Infinite Volume must be equal to or less than the maximum data constituent size for the smaller platform.

Steps

1. Identifying the constituent sizes that platforms support on page 60
2. Creating an Infinite Volume with small constituents on page 60

Related concepts

- Creating SVMs to contain Infinite Volumes (cluster administrators only) on page 43
- Constituent size considerations for data protection mirror relationships for Infinite Volumes on page 32
- Providing disaster recovery on Infinite Volumes using mirroring technology on page 104

Related tasks

- Assigning dedicated aggregates to an SVM with Infinite Volume (cluster administrators only) on page 51
Identifying the constituent sizes that platforms support

To determine whether two platforms support different maximum Infinite Volume constituent sizes, you must identify the maximum FlexVol volume size of each platform and compare the two values.

**Before you begin**

You must know the names of the platforms that you are using.

**Steps**

1. Navigate to the NetApp Hardware Universe.
2. Select your Data ONTAP version.
3. Use the Max Infinite Volume Data Constituent Size (TB) field to identify the maximum data constituent size supported for both of the following platforms:
   - The platform that will hold your source Infinite Volume
   - The platform that will hold your destination Infinite Volume
4. Determine the impact on your Infinite Volume configuration.

<table>
<thead>
<tr>
<th>If the...</th>
<th>Then...</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maximum data constituent size of the destination platform is smaller than that of the source platform</td>
<td>You must specify the maximum constituent size when you create the source Infinite Volume. Record the number for later use when you create the source Infinite Volume, and continue with this workflow.</td>
</tr>
<tr>
<td>Values match</td>
<td>You do not have to specify the maximum constituent size when you create the source Infinite Volume. Instead of completing this workflow, follow the procedure on creating an Infinite Volume using the default settings.</td>
</tr>
<tr>
<td>Maximum data constituent size of the source platform is smaller than that of the destination platform</td>
<td>You do not have to specify the maximum constituent size when you create the source Infinite Volume. Instead of completing this workflow, follow the procedure on an Infinite Volume using the default settings.</td>
</tr>
</tbody>
</table>

**Related tasks**

*Creating an Infinite Volume using default settings* on page 55

Creating an Infinite Volume with small constituents

With advanced privilege, when you create an Infinite Volume, you can specify a maximum size for data constituents so that the data constituents are created small enough to be in a data protection mirror relationship with a destination Infinite Volume on a platform that uses smaller constituent sizes.

**About this task**

The size of the namespace constituent is not relevant, because its maximum size is small enough for any platform.

**Steps**

1. Switch to advanced privilege by using the `set privilege advanced` command.
The command prompt changes to `cluster::*`.

2. Create the Infinite Volume by using the `volume create` command with the `-max-data-constituent-size` parameter.

   If you want to use incremental tape backup, you can include the `-enable-snapdiff` parameter when you use the `volume create` command.

   **Example**

   The following command creates an Infinite Volume with data constituents that are no bigger than 70 TB:

   ```
   cluster::> volume create -vserver vs0 -volume ivol -size 200TB -max-data-constituent-size 70TB
   [Job 125] Creating constituent "ivol_ns" on aggregate "aggr1".
   [Job 125] Creating constituent "ivol_ns_mirror0001" on aggregate "aggr5".
   [Job 125] Creating constituent "ivol_1024_data0001" on aggregate "aggr3"
   [Job 125] Creating constituent "ivol_1024_data0002" on aggregate "aggr6"
   ...
   [Job 125] Job succeeded: Created Infinite Volume successfully.
   ```

   After the Infinite Volume is created and goes online, it is mounted. By default, it has a junction path of `/NS` and uses thick provisioning.

3. Verify that the Infinite Volume is created and is online by using the `volume show` command with the `-volume` parameter.

   **Example**

   The following output shows that the volume `ivol` is online:

   ```
   cluster::> volume show -volume ivol
   Type   Size  Available Used%
   ------- ------- ------- ------
   RW     200TB  190TB   5%
   ```

4. Verify that the constituents are of the sizes that you wanted by using the `volume show` command with the `-vserver` and `-is-constituent true` parameters.

5. Return to admin privilege by using the `set privilege admin` command.

   The command prompt changes back to `cluster::*`.

**After you finish**

You can optionally set up efficiency technologies, and then set up file access.

**Related tasks**

* Managing size-related error messages when you create or expand an Infinite Volume on page 64

**Completing an interrupted Infinite Volume creation**

When you try to create an Infinite Volume and your system reboots before the volume is completely created, you must run the `volume create` command again. You might also have to delete some of the incomplete components or delete the Infinite Volume and start over.

**Steps**

1. Continue the volume creation process by using the `volume create` command with the same parameters that you used when you first tried to create the Infinite Volume.
2. If the command fails, identify whether any constituents of the Infinite Volume were incompletely created, and then delete those constituents:
   
a. Change to advanced privilege by using the `set -privilege advanced` command.

b. Use the `volume show` command with the following parameters to display any constituents of the Infinite Volume that were incompletely created:
   
   • Use the `-vserver` parameter to exclude volumes on other Storage Virtual Machines (SVMs), as well as node root volumes.
   
   • Use the `-volume-style flex` parameter to exclude the Infinite Volume.

   • Use the `-vsroot false` parameter to exclude the root volume for the SVM.

   Do not include the `-is-constituent true` parameter because you do not need to see constituents that were successfully created.

Example

The following command displays an incomplete constituent for the SVM with Infinite Volume named vs0:

```
cluster1*::> volume show -vserver vs0 -volume-style flex -vsroot false

Vserver  Volume       Aggregate    State      Type       Size  Available  Used%
--------- ------------ ------------ ---------- ---- ---------- ----------
        InfiniteVol_1024_data0001 aggr3 online  RW   2GB       1.90GB  5%
```

c. If any volumes were listed, use the `volume offline` and `volume delete` commands to delete them.

Example

In the following example, the InfiniteVol_1024_data001 volume is taken offline and then deleted:

```
cluster1*::> volume offline -vserver vs0 -volume InfiniteVol_1024_data001Volume
"vs0:InfiniteVol_1024_data001Volume" is now offline.
Volume modify successful on volume: InfiniteVol_1024_data001Volume

cluster1*::> volume delete -vserver vs0 -volume InfiniteVol_1024_data001
[Job 27] Job succeeded: Successful
```

d. Use the `volume show` command again to display any constituents of the Infinite Volume that were incompletely created.

Example

The following output indicates that there are no incompletely created constituents in the SVM with Infinite Volume named vs0:

```
cluster1*::> volume show -vserver vs0 -volume-style flex -vsroot false

There are no entries matching your query.
```

e. Return to admin privilege by using the `set -privilege admin` command.

f. Continue the volume creation process by using the `volume create` command with the same parameters and values that you used when you first tried to create the Infinite Volume.

3. If the command fails, perform the following steps to delete the Infinite Volume and re-create it:
a. Unmount the Infinite Volume, take it offline, and delete it.

**Example**

The following example deletes the volume named InfiniteVol on the SVM named vs0:

```
cluster1::> volume unmount -vserver vs0 -volume InfiniteVol
cluster1::> volume modify -vserver vs0 -volume InfiniteVol -state offline
[Job 204] Job succeeded: Volume "vs0"::"InfiniteVol" is now offline.
cluster1::> volume delete -vserver vs0 -volume InfiniteVol
[Job 205] Job succeeded: success
```

b. Re-create the Infinite Volume by using the `volume create` command.

**Result**

When the Infinite Volume is created, the output displays the following message: Job Succeeded.

### Deleting an Infinite Volume

When you delete an Infinite Volume, all of its files, directories, and Snapshot copies are deleted.

**Before you begin**

You must have ensured that no clients or applications are accessing the data in the Infinite Volume.

**Steps**

1. If the volume is mounted, unmount it by using the `volume unmount` command.
2. If the volume is part of a SnapMirror relationship, delete the relationship by using the `snapmirror delete` command.
3. If the volume is online, take it offline by using the `volume offline` command.
4. Delete the volume by using the `volume delete` command.
   The volume and all of its files and directories, including Snapshot copies, are deleted.

**Example of deleting an Infinite Volume**

The following example deletes the volume named InfiniteVol on the Storage Virtual Machine (SVM) named vs0:

```
cluster1::> volume unmount -vserver vs0 -volume InfiniteVol
cluster1::> volume modify -vserver vs0 -volume InfiniteVol -state offline
[Job 204] Job succeeded: Volume "VS0"::"InfiniteVol" is now offline.
cluster1::> volume delete -vserver vs0 -volume InfiniteVol
Warning: This command will permanently destroy all data in the Infinite Volume
"InfiniteVol". Are you sure you wish to proceed? [y|n]: y
[Job 205] Job succeeded: success
```

**After you finish**

You can delete the SVM with Infinite Volume or create a new Infinite Volume.
Error messages and solutions for creating and expanding Infinite Volumes

When you create or expand an Infinite Volume, error messages might indicate that you have to change your configuration before you can continue with the creation or expansion operation.

Related concepts

Error messages and solutions for failed aggregate selection for destination Infinite Volumes on page 135

Managing size-related error messages when you create or expand an Infinite Volume

When you try to create or expand an Infinite Volume and you see error messages indicating that the size is too big for the available resources, you can add resources or reduce the requested size.

Steps

1. If you see an error message that the Infinite Volume cannot be created or expanded at the requested size without adding resources, perform one of the following actions:

   - Add disks to the aggregates associated with the Storage Virtual Machine (SVM) with Infinite Volume, and then rerun the `volume create` or `volume modify` command.
   - Add aggregates to the SVM with Infinite Volume by using the `vserver add-aggregates` command, and then rerun the `volume create` or `volume modify` command.

   If you are an SVM administrator, you must ask the cluster administrator to add resources.

   Example

   In the following output, the Infinite Volume requires more space. When more aggregates are associated with the SVM, the Infinite Volume is created successfully.

   ```bash
   cluster::> volume create -vserver vs0 -volume ivol -size 279.6TB
   [Job 123] Creating Infinite Volume "ivol".
   Error: command failed: [Job 123] Job failed: Cannot create an Infinite Volume of any size for the given configuration and Vserver. Add space to the Vserver by expanding existing aggregates or assigning additional aggregates.
   cluster::> vserver add-aggregates -vserver vs0 -aggregates aggr1,aggr2,aggr3,aggr4, aggr5,aggr6,aggr7,aggr8
   cluster::> volume create -vserver vs0 -volume ivol -size 279.6TB
   [Job 124] Job succeeded: Created Infinite Volume successfully.
   ```

2. If you see an error message that indicates the Infinite Volume's maximum size, rerun the `volume create` or `volume modify` command using a size that is smaller than the maximum indicated in the error message.

   Using a smaller size is especially relevant if the Infinite Volume shares aggregates with other SVMs, which might use some available space in between the time the error message was displayed and the next time that you try to create the Infinite Volume.
Example

In the following output, an Infinite Volume of 279.6 TB is too large for the available space. The maximum possible size is 230 TB. When you reduce the size to 200 TB, the Infinite Volume is created successfully.

```bash
cluster::> volume create -vserver vs0 -volume ivol -size 279.6TB
[Job 124] Verifying create parameters.
Error: command failed: [Job 124] Job failed: The Infinite Volume size cannot be greater than "204800GB" ("219902325555200B"). Requested size is "279.6TB" ("307423451126169B"). To create an Infinite Volume of maximum size use "219902325555200B".

cluster::> volume create -vserver iv.example.com -volume ivol -size 200TB
[Job 125] Job succeeded: Created Infinite Volume successfully.
```

Related concepts

- How node balancing affects an Infinite Volume's size and aggregate use on page 36
- How space is allocated inside a new Infinite Volume on page 37
- How space is allocated when you expand an Infinite Volume on page 168

Troubleshooting a footprint information message when creating or expanding an Infinite Volume

If an error message appears about unavailable footprint information when you try to create or expand an Infinite Volume, you can create or expand the Infinite Volume after a short delay.

About this task

This issue occurs only if an Infinite Volume shares aggregates with other Storage Virtual Machines (SVMs) and only when another volume on a shared aggregate is in a restricted state due to a SnapMirror initialization operation.

Steps

1. Wait until the SnapMirror initialization operation is finished on the other volume.

2. Rerun the `volume create` or `volume modify` command.
Setting up deduplication and compression to increase storage efficiency

When you configure deduplication and data compression to achieve optimal space savings on an Infinite Volume, you can perform the configuration in the same way that you would on a FlexVol volume, but you should understand the unique aspects of how these efficiency technologies work on Infinite Volumes.

If an Infinite Volume uses storage classes, efficiency technologies are not configured at the volume level; they are configured using OnCommand Workflow Automation on individual storage classes.

Related concepts
- How efficiency works on Infinite Volumes with storage classes on page 199

How deduplication works on Infinite Volumes

Deduplication of an Infinite Volume is configured at the level of the Infinite Volume, but the actual deduplication process occurs within each data constituent.

The scope of deduplication

For an Infinite Volume, deduplication occurs within each data constituent—not across the Infinite Volume. For example, if two files on the same data constituent contain the same block, deduplication discards the duplicate block. If two files in separate data constituents contain the same block, deduplication does not discard the duplicate block.

The namespace constituent and namespace mirror constituents of an Infinite Volume are not deduplicated.

How deduplication is configured

Deduplication is configured at the Infinite Volume level. When you enable or disable deduplication on an Infinite Volume, deduplication is enabled or disabled on the data constituents of the Infinite Volume.

You can see whether deduplication is enabled by viewing the State field in the output of the `volume efficiency show` command. (The terms `efficiency state` and `deduplication state` are sometimes used interchangeably.) By default, the field shows the deduplication state of the Infinite Volume as a whole. If the State field contains a dash ("-"), one or more data constituents are offline or have a different deduplication state than the other data constituents. If you add the `-is-constituent true` parameter to the `volume efficiency show` command, the output displays the deduplication state of each individual data constituent.

How deduplication operations are run

When a deduplication operation is run on an Infinite Volume, separate deduplication operations run on each data constituent in the Infinite Volume. For example, if an Infinite Volume has 100 data constituents, a deduplication operation on the Infinite Volume triggers a deduplication operation on each of the 100 data constituents.

Deduplication operations are combined with postprocess compression into a queue of efficiency operations. A maximum of eight efficiency operations per node occur at any one time. If more than eight efficiency operations per node are scheduled to run at any one time, they are queued and run as each operation finishes.
If an operation succeeds overall but fails on one or more constituents, the names of the failed constituents are reported to the event management system, along with the reason for failure.

Information about deduplication operations, such as the status and progress, is not available for an Infinite Volume as a whole. You can see the status and progress of postprocess compression operations on individual data constituents by using the `volume efficiency show` command with the `-is-constituent true` parameter.

**How space savings are reported**

The space savings gained by deduplication on an Infinite Volume reflect the total space savings of all of the volume’s data constituents. You can see the space savings by using the `volume show` command with either the `-instance` or `-fields` parameter.

Space savings information is available only when all data constituents are online.

**How volume state affects efficiency**

You can run efficiency operations and view efficiency information only when an Infinite Volume is online, which means that every constituent in the Infinite Volume must be online.

**Free space required for deduplication**

Deduplication has the following free space requirements:

- Each aggregate that contains deduplication-enabled data constituents or deduplication-enabled FlexVol volumes must have free space that is equivalent to three percent of the total logical data contained within all of the deduplicated data constituents and deduplicated FlexVol volumes on the aggregate.

- Each data constituent in the deduplicated Infinite Volume must have free space that is equivalent to four percent of the data in the data constituent.

If the data constituents or their containing aggregates lack adequate free space, the affected data constituents are skipped while deduplication continues to run on the other data constituents.

**Related information**

*Logical storage management*

**How compression works on Infinite Volumes**

Data compression of an Infinite Volume is configured at the level of the Infinite Volume, but the actual compression processes occur within each data constituent.

**The scope of compression**

Compression runs on the data files within the Infinite Volume, not on the information contained within the namespace constituent.

**How compression and inline compression are configured**

Data compression of an Infinite Volume is configured at the Infinite Volume level. When you enable or disable compression and inline compression on an Infinite Volume, compression and inline compression are enabled or disabled on the data constituents of the Infinite Volume.

You can see whether postprocess compression and inline compression are enabled by viewing the Compression and Inline Compression fields that are displayed when you use the `-instance` parameter with the `volume efficiency show` command. By default, the fields show the
compression state and inline compression state of the Infinite Volume as a whole. If either field contains a dash ("-"), one or more data constituents are offline or have a different state than the other data constituents. If you add the -is-constituent true parameter to the volume efficiency show command, the output displays the postprocess compression and inline compression state of each individual data constituent.

How postprocess compression operations are run

When a postprocess compression operation runs on an Infinite Volume, separate compression operations run on each data constituent in the Infinite Volume. For example, if an Infinite Volume has 100 data constituents, a postprocess compression operation on the Infinite Volume triggers 100 postprocess compression operations on the data constituents.

If an operation succeeds overall but fails on one or more constituents, the names of the failed constituents are reported to the Event Management System, along with the reason for failure.

Information about postprocess compression operations, such as the status and progress, is not available for an Infinite Volume as a whole. You can see the status and progress of post-process compression operations on individual data constituents by using the volume efficiency show command with the -is-constituent true parameter.

How space savings are reported

The space savings gained by compression on an Infinite Volume reflect the total space savings of all of the volume's data constituents. You can see the space savings by using the volume show command. If you add the -is-constituent true parameter to the volume show command, the output displays the space savings of each individual data constituent.

Space savings information is available only when all the constituents are online.

How volume state affects efficiency

You can run efficiency operations and view efficiency information only when an Infinite Volume is online, which means that every constituent in the Infinite Volume must be online.

Related information

Logical storage management

Enabling deduplication and compression on an Infinite Volume

You can improve storage efficiency on an Infinite Volume by enabling deduplication, inline compression, and postprocess compression on the Infinite Volume.

About this task

- This procedure applies only to an Infinite Volume without storage classes.
  The way that you configure deduplication and compression on Infinite Volumes without storage classes is the same way that you configure deduplication and compression on FlexVol volumes.
- This procedure is a summary of one way that you can enable efficiency technologies.

Steps

1. Enable deduplication on the Infinite Volume by using the volume efficiency on command with the -vserver and -volume parameters.
2. Enable inline compression and postprocess compression on the Infinite Volume by using the `volume efficiency modify` command with the `-vserver`, `-volume`, `-compression` and `-inline-compression` parameters.

If you want to enable inline compression, you must also enable postprocess compression.

Example

```
cluster1::> volume efficiency modify -vserver vs0 -volume repo_vol -compression true -inline-compression true
```

3. Define an efficiency policy for the Storage Virtual Machine (SVM) with Infinite Volume with a schedule that meets your requirements.

a. Review the list of possible schedules by using the `job schedule show` command, and decide which schedule you want.

Example

```
cluster1::> job schedule cron show
Name         Description
---------------- -----------------------------------------------------
5min         @:00,:05,:10,:15,:20,:25,:30,:35,:40,:45,:50,:55
8hour        @2:15,10:15,18:15
daily        @0:10
hourly       @:05
weekly       Sun@0:15
5 entries were displayed.
```

b. Create an efficiency policy with the selected schedule by using the `volume efficiency policy create` command.

Example

```
cluster1::> volume efficiency policy create -vserver vs0 -policy dailyIV -schedule daily -enabled true -qos-policy background -duration 8
```

c. Verify the new efficiency policy by using the `volume efficiency policy show` command with the `-vserver` parameter.

Example

```
cluster1::> volume efficiency policy show -vserver vs0
<table>
<thead>
<tr>
<th>Job</th>
<th>Duration</th>
<th>QoS</th>
</tr>
</thead>
<tbody>
<tr>
<td>vs0</td>
<td>daily</td>
<td>best-effort</td>
</tr>
</tbody>
</table>
```

4. Assign the new efficiency policy to the Infinite Volume by using the `volume efficiency modify` command.
Example

Cluster1::> volume efficiency modify -vserver vs0 -volume repo_vol -policy dailyIV
[Job 564] Job succeeded: Modified efficiency settings for volume "repo_vol" in Vserver "vs0" successfully.

5. Verify the efficiency settings by using the `volume efficiency show` command with the `-fields state,compression,inline-compression,schedule,policy` parameter.

Example

cluster1::> volume efficiency show -vserver vs0 -volume repo_vol -fields state,compression,inline-compression,schedule,policy
vserver volume   state   schedule policy  compression inline-compression
------- -------- ------- -------- ------- ----------- ------------------
vs0     repo_vol Enabled -        dailyIV true        true

Related information

*Logical storage management*
Providing file access to Infinite Volumes

Infinite Volumes support access by Windows and UNIX users over multiple protocols, including SMB 1.0, NFSv3, pNFS, and NFSv4.1.

Infinite Volumes supports UNIX, Mixed, NTFS, and Unified security styles.

The way that you set up NFS or SMB access to the Storage Virtual Machine (SVM) with Infinite Volume is similar to that of the SVM with FlexVol volumes.

**Note:** This documentation uses *Infinite Volumes* and *SVMs with Infinite Volume* interchangeably. Although file access is configured at the SVM level, there is a one-to-one relationship between the Infinite Volume and its containing SVM with Infinite Volume. Therefore, configuring access to the SVM is effectively configuring access to the Infinite Volume.

Support for NFS on Infinite Volumes

Infinite Volumes support core NFS versions and features, including file access by using NFSv3, NFSv4.1, and pNFS. However, Infinite Volumes do not support all of the NFS versions and NFS features that Storage Virtual Machines (SVMs) with FlexVol volumes support.

**NFS versions**

The following table identifies the versions or types of NFS that are supported on Infinite Volumes:

<table>
<thead>
<tr>
<th>Supported</th>
<th>Not supported</th>
</tr>
</thead>
<tbody>
<tr>
<td>• NFSv3</td>
<td>• NFSv4.0</td>
</tr>
<tr>
<td>• NFSv4.1</td>
<td></td>
</tr>
<tr>
<td>• pNFS</td>
<td></td>
</tr>
</tbody>
</table>

**NFS features**

Almost all NFS features are supported on Infinite Volumes in the same way that they are supported for FlexVol volumes. For example, NFSv4.1 ACLs are supported.

The following NFS features are not supported on Infinite Volumes:

- Delegations
- Migrations and referrals
- Configurations related to other features that Infinite Volumes do not support, such as quotas or vStorage
- FPolicy, which is used to monitor and manage file access events
- Fsecurity
- Auditing of NAS file access events
  
  Clients can set SACLs (System ACLs) on files, but the SACLs are not currently used.
- Security tracing
Support for CIFS on Infinite Volumes

Infinite Volumes support SMB 1.0 and core CIFS functionality. However, Infinite Volumes do not support all of the SMB versions and CIFS features that Storage Virtual Machines (SVMs) with FlexVol volumes support.

SMB versions

The following table identifies the versions of SMB that are supported and not supported on SVMs with Infinite Volume:

<table>
<thead>
<tr>
<th>Supported</th>
<th>Not supported</th>
</tr>
</thead>
<tbody>
<tr>
<td>• SMB 1.0</td>
<td>• SMB 2.x</td>
</tr>
<tr>
<td></td>
<td>• SMB 3.0</td>
</tr>
</tbody>
</table>

CIFS features

Most CIFS features are supported on SVMs with Infinite Volume in the same way that they are supported for all SVMs.

The following CIFS features are not supported by SVMs with Infinite Volume:

• BranchCache
• Hyper-V over SMB feature
• Remote copy offload
• VSS shadow copy
• Automatic node referrals
• FPolicy, which is used to monitor and manage file access events
• Fsecurity
• Auditing of NAS file access events
  Clients can set SACLs (System ACLs) on files, but the SACLs are not currently used.
• Security tracing
• Antivirus

Change notification is supported on Infinite Volumes but is disabled by default. Enabling change notification on an Infinite Volume share requires advanced privilege level and might require that clients restart their sessions.

Related information

SMB/CIFS management
Comparison of namespaces for Infinite Volumes and FlexVol volumes

The namespace architecture of a Storage Virtual Machine (SVM) with Infinite Volume is different from the namespace architecture of an SVM with FlexVol volumes. Understanding this difference can help you plan the namespace for each SVM.

When you create and mount FlexVol volumes in an SVM with FlexVol volumes, you can create one or more trees of junction paths. Each FlexVol volume can connect to another FlexVol volume or directly to the SVM root volume.

In contrast, when you configure an SVM with Infinite Volume, you need to define only one junction path in the entire SVM—a single junction path for the Infinite Volume. No other junction paths are needed; the Infinite Volume contains all of the data for the SVM. For an Infinite Volume, the namespace for the SVM and the namespace for the Infinite Volume are essentially the same.

Inside both FlexVol volumes and Infinite Volumes, you can create a directory tree of your choosing.

Related concepts

Restrictions on the private namespace of an SVM with Infinite Volume on page 182

Where clients access Infinite Volumes

Clients access an Infinite Volume at or below the Infinite Volume's junction path.

All SMB shares and NFS mounts must occur at the Infinite Volume's junction path or at a directory under the junction path. For example, if the Infinite Volume junction is `/NS`, clients can access `/NS` or any directory created below `/NS`, such as `/NS/admin`, `/NS/video`, and `/NS/eng/images`.

Clients must not access the following locations:

- The Storage Virtual Machine (SVM) root volume, `/`
- Directories created immediately below the SVM root volume, such as `/admin`, `/videos`, or `/eng/images`
- Directories under the private `.system` namespace, such as `/system/constituents`

Although it is possible to create an SMB share in an incorrect location, such as `/system`, files created in such locations are not part of the Infinite Volume or its advanced features.

How access to an Infinite Volume is controlled

To successfully access data on an Infinite Volume, clients pass through one or more layers of security that can include an export policy, a share ACL, and file permissions.

The following table summarizes the security layers and what each one affects:

<table>
<thead>
<tr>
<th>Layer</th>
<th>What it controls</th>
<th>Method of control</th>
<th>Affects NFS access?</th>
<th>Affects CIFS access?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Export policy</td>
<td>Whether a user can mount all or part of the Infinite Volume</td>
<td>Comparing the client's IP and other networking information to export rules in the export policy applied to the entire Infinite Volume</td>
<td>Yes</td>
<td>Not by default</td>
</tr>
<tr>
<td>Layer</td>
<td>What it controls</td>
<td>Method of control</td>
<td>Affects NFS access?</td>
<td>Affects CIFS access?</td>
</tr>
<tr>
<td>---------------------</td>
<td>---------------------------------------------------------</td>
<td>-----------------------------------------------------------------------------------</td>
<td>---------------------</td>
<td>----------------------</td>
</tr>
<tr>
<td>Share ACL</td>
<td>Whether a user can access a share</td>
<td>Comparing the user’s Windows credential to the access control entries (ACEs) in the ACL on a given share</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>File permissions</td>
<td>Whether a user can see and work with specific directories and files</td>
<td>Comparing the user’s credentials—both Windows and UNIX—to the effective file permissions</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>

**Note:** Each of these security layers depends upon security provided by user authentication and authorization processes.

**How export policies affect access to an Infinite Volume**

By default, an Infinite Volume has an export policy that gives clients full read/write access to the entire namespace. You might want to restrict access for clients by revising the policy.

The default export policy for an Infinite Volume is called `repos_namespace_export_policy` and it contains no restrictions on file access. If you want to restrict access, you can add export rules to the existing export policy or you can create a new export policy and assign it to the Infinite Volume.

The export policy of an Infinite Volume affects the entire namespace of the Storage Virtual Machine (SVM) with Infinite Volume because each SVM with Infinite Volume contains only a single volume junction.

By default, export policies apply only to NFS access. If you want an export policy to apply to SMB access, you must use advanced privilege to set the `-is-exportpolicy-enabled` parameter of the `cifs options modify` command.

The SVM with Infinite Volume also has other export policies, which you should leave unchanged:

- The root volume of the SVM with Infinite Volume has an export policy called `repos_root_readonly_export_policy`, which gives clients read-only access to the root volume so that clients can gain access to the Infinite Volume’s junction path. Because clients should not access the SVM root volume, you should not change this policy to give clients write permissions.

- The data constituents have an export policy called `repos_restricted_export_policy`, which restricts all access to the data constituents. Because clients should never access data constituents directly, you must not change this export policy in any way.

**Related tasks**

*Controlling access to an Infinite Volume with IP-based export rules* on page 84

**Related information**

*NFS management*
How share ACLs affect SMB access to Infinite Volumes

Share ACLs control access by SMB clients to each share on an Infinite Volume. Share ACLs apply only to Windows users accessing over SMB, not to NFS clients.

You can create multiple shares on an Infinite Volume, and each share has a default share ACL that allows all Windows users full control. You can configure each share ACL independently of other shares' ACLs.

Share ACLs work the same way on Infinite Volumes and FlexVol volumes.

Related tasks

- Controlling SMB access to an Infinite Volume share with share ACLs on page 85

Related information

- SMB/CIFS management

How file permissions control access to Infinite Volumes

Files and directories on an Infinite Volume can have a variety of effective file permissions, which control a user's access to a specific file based on the user's credentials.

Types of file permissions

Infinite Volumes support all types of file permissions, including the following:

- UNIX permissions (also called *mode bits*)
  NFSv3 and NFSv4.1 clients set mode bits by using the following UNIX commands: `chgrp`, `chown`, and `chmod`. The `chmod` command can be on permissions or on the sticky bit, setuid flag, or setgid flag.

- NTFS file permissions (also called *SMB ACLs*)
  Windows users can set NTFS (NT File System) file permissions by using Windows command-line interface or the Security tab of the Properties dialog box in Windows Explorer. The process of setting NTFS file permissions on a file sets an SMB ACL on the file.

- NFSv4.1 ACLs
  If NFSv4.1 ACLs have been enabled on the Storage Virtual Machine (SVM) with Infinite Volume, NFSv4.1 clients can set NFSv4.1 ACLs using commands specific to each operating system.

Each file or directory can only have one type of file permissions in effect at one time. But different file permissions can be used across files and directories. For example, the Infinite Volume itself might use UNIX permissions, while one subdirectory uses NTFS file permissions, and another subdirectory uses UNIX permissions.

Method of checking access

The general method of determining the user's rights is the same for Infinite Volumes and FlexVol volumes. A user's credentials are compared with the effective file permissions.

On an Infinite Volume, if NTFS file permissions are in effect, the access check has one additional aspect. The NTFS file permissions can include entries for UNIX users, which are then compared with the user's UNIX credential.
What the default permissions are on Infinite Volumes

By default, an Infinite Volume has no restrictions on file access from either the export policy or share ACL. At the file level, an Infinite Volume uses UNIX permissions (mode bits) by default.

**Default export policy**

An Infinite Volume has a default export policy `repos_namespace_export_policy` that contains no restrictions on file access.

**Default share ACL**

Each time a CIFS share is created on an Infinite Volume, it has a default ACL that allows everyone full control.

**Default file permissions**

By default, a new Infinite Volume uses UNIX file permissions (also called `mode bits`). An Infinite Volume has no default NTFS file permissions (also called `SMB ACLs`) or NFSv4.1 ACLs.

When an Infinite Volume is created, the default UNIX permissions are 0755 (`drwxr-xr-x`), which allow the UNIX owner to read, write, and execute his or her own files and allow others to read or execute files. If you want a new Infinite Volume to have UNIX permissions other than 0755, you can use the `-unix-permissions` parameter of the `volume create` command.

How unified security style supports multiprotocol access to Infinite Volumes

The unified security style used by Infinite Volumes enables all users using NFSv3, NFSv4.1, and SMB 1.0 to view and set file permissions, regardless of the file permissions that are currently in effect on a given file or directory. In addition, unified security style supports UNIX principals in NTFS file permissions.

**Setting permissions on all files**

In unified security style, clients can set file permissions regardless of the file permissions that are currently in effect, including the following multiprotocol situations:

- NFS clients can set UNIX permissions bits on a file with NTFS file permissions.
- SMB clients can set NTFS file permissions on a file with UNIX permissions or an NFSv4.1 ACL.
- NFSv4.1 clients can set an NFSv4.1 ACL on a file with NTFS file permissions.

**Displaying permissions on all files**

In unified security style, clients can view file permissions of any file regardless of the file permissions that are currently in effect, including the following multiprotocol situations:

- NFS clients can display UNIX permissions of a file with NTFS file permissions.
- SMB clients can view permissions of a file with UNIX permissions or an NFSv4.1 ACL.
- NFSv4.1 clients can display an equivalent NFSv4.1 ACL for a file with NTFS file permissions.
Support for UNIX principals in NTFS file permissions

In unified security style, NTFS file permissions can contain UNIX principals, which facilitate multiprotocol access in the following ways:

• When a Windows or UNIX user uses any supported protocol to access a file that contains NTFS file permissions, the user's UNIX credential is compared with any UNIX principals in the NTFS file permissions.

• When an SMB client views a file's permissions, any UNIX users in the permissions are displayed without being converted to Windows users.

• When an SMB client sets NTFS file permissions, the SMB client can specify permissions for UNIX principals.

• When an NFS client changes the UNIX permissions, owner, or group of a file that uses NTFS file permissions, UNIX principals are used in the affected permissions while leaving the remaining NTFS file permissions unchanged.

NTFS file permissions are unaffected as long as the v4 ACL Preserve parameter is enabled.

How group mapping supports multiprotocol access to Infinite Volumes

Group mapping improves the accuracy of permissions that appear when NFSv4.1 clients display the ACL of a file or directory that has NTFS file permissions. If an Infinite Volume supports both NFSv4.1 ACLs and SMB, you should configure group mapping, which is similar to user mapping.

Why group mapping is necessary

Groups are often used in ACLs to simplify security management. However, groups in multiple Windows domains cannot be easily translated to the groups of a single NFSv4.1 domain.

Mapping groups from Windows to UNIX ensures that group names appear when NFSv4.1 ACLs are displayed on NFSv4.1 clients.

If a Windows group is not mapped to a UNIX group and a default UNIX group is not configured, the Windows group is displayed to an NFSv4.1 client as nobody (specifically nobody@v4-id-domain).

What group mapping is required

If an Infinite Volume supports both SMB and NFSv4.1 ACLs, you should perform the following configurations:

• Create a Windows-to-UNIX mapping for every Windows group.

• Define a default UNIX group that is used when no mapping exists for a Windows group and the lowercase name of the Windows group is not a valid group name in the UNIX domain.

Comparison of user and group mapping

Group mapping and user mapping share the following similarities:

• They can both be defined either using Data ONTAP or using LDAP.
  In Data ONTAP CLI, the vserver name-mapping commands configure user mapping and the vserver group-mapping commands configure group mapping.

• If they are defined using Data ONTAP, they are defined in a similar way and using the same conversion rules.
  For information about conversion rules in user and group mappings, see either the NFS Reference or the CIFS Reference.

Group mapping is unique in the following ways:
- It is available only on Storage Virtual Machines (SVMs) with Infinite Volume, not SVMs with FlexVol volumes.

- It is necessary only if an SVM is configured for both SMB and NFSv4.1, including NFSv4.1 ACLs.

- It does not affect access; it affects only what NFSv4.1 clients display. During access checks, a user's group membership is determined in the same way on all SVMs.

- It is necessary only in one direction—from Windows to UNIX. UNIX groups do not have to be mapped to Windows groups.

Related tasks

Configuring group mapping on an Infinite Volume on page 83

Setting up file access to an Infinite Volume

After meeting some prerequisites, you can set up file access to an Infinite Volume by configuring user and groups, enabling access to one or two protocols, and then controlling access with a mixture of export policies, share ACLs, and file permissions.

Steps

1. Preparing an Infinite Volume for file access on page 78
2. Configuring name services, user authentication, and user mapping on an Infinite Volume on page 79
3. Setting up basic NFS access to an Infinite Volume on page 80
4. Setting up basic SMB access to an Infinite Volume on page 81
5. Configuring group mapping on an Infinite Volume on page 83
6. Controlling access to an Infinite Volume with IP-based export rules on page 84
7. Controlling SMB access to an Infinite Volume share with share ACLs on page 85
8. Controlling access to Infinite Volumes with file permissions on page 85

Preparing an Infinite Volume for file access

Before setting up file access to an Infinite Volume, you should meet certain prerequisites, verify that the protocol that you want is allowed on the Storage Virtual Machine (SVM) with Infinite Volume, verify that the Infinite Volume is online, and ensure that you know the name of its junction path.

Before you begin

- The cluster must have an NFS license, a CIFS license, or both these licenses.

- The time zone must be set, and the time must be synchronized across the cluster by configuring NTP. This prevents authentication errors and ensures that timestamps in log files are consistent across the cluster.

- The Infinite Volume and its containing SVM with Infinite Volume must have been created.

- Data-access LIFs and routing groups must have been configured on the SVM with Infinite Volume.

Steps

1. Verify that the protocols that you want are allowed on the SVM with Infinite Volume.
a. View the protocols that are allowed on the SVM with Infinite Volume by using the `vserver show` command with the `-vserver` parameter.

For an SVM with Infinite Volume, the allowed protocols can be set to any of the following values:

- nfs
- cifs
- cifs,nfs

By default, both CIFS and NFS are allowed on an SVM with Infinite Volume.

Example

```
cluster1::> vserver show -vserver vs1
Vserver: vs1
...  
  Allowed Protocols: nfs, cifs
  Disallowed Protocols: fcp, iscsi, ndmp
...  
```

b. If the output does not match your requirements, use the `vserver add-protocols` or the `vserver remove-protocols` command.

2. Verify that the Infinite Volume is online and mounted by using the `volume show` command with the `-vserver`, `-volume`, `-fields state,junction-path` parameters.

Example

```
cluster1::> volume show -vserver vs1 -volume ivol -fields state,junction-path
vserver volume state  junction-path
------- ------ ------ --------------
vs1     ivol   online /NS
```

a. If the Infinite Volume is not online, bring it online by using the `volume online` command.

b. If the Infinite Volume does not have a junction path, which indicates it is not mounted, mount it by using the `volume mount` command.

3. Note the junction path that is displayed in the output of the previous step so that you can use it for file access later.

Related information

- System administration
- SMB/CIFS management
- Network and LIF management

Configuring name services, user authentication, and user mapping on an Infinite Volume

You can configure name services, user authentication, user name mapping, and default users for a Storage Virtual Machine (SVM) with Infinite Volume in the same way that you do for an SVM with FlexVol volumes.

Steps

1. If you allow NFS access and you want to use Kerberos for authentication, configure Kerberos.

2. If you allow NFS access, configure either or both of the following name services components:
• NIS name services
• Local UNIX users and groups

3. If you allow SMB access to the Infinite Volume and you want to use local Windows users and groups for authentication and authorization, configure local Windows users and groups.

SMB access uses Active Directory (AD) users and groups. You can configure local Windows users and groups in addition to the AD users and groups.

4. If you allow both SMB and NFS access to the Infinite Volume, configure either or both of the following name translation components:

• User name mapping
• The default UNIX user and the default Windows users

Related information

NFS management
SMB/CIFS management

Setting up basic NFS access to an Infinite Volume

If you want NFS clients to access an Infinite Volume, you must use the `vserver nfs create` command to create an NFS server on the Storage Virtual Machine (SVM) with Infinite Volume and enable NFSv3, NFSv4.1, or both. Then you should test the access using a test UNIX user on an NFS client.

About this task

The way that you set up NFS access is the same for all SVMs.

Steps

1. Plan what type of NFS access you will enable.
   a. Decide whether you are going to enable NFSv3, NFSv4.1, or both.
   b. If you decided to enable NFSv4.1, decide whether you are going to enable NFSv4.1 ACLs.
   c. Determine what optional NFS functionality that you will enable.

2. Create an NFS server by using the `vserver nfs create` command with the parameters based on your choice in the previous step.

   Example

   ```
   cluster1::> vserver nfs create -vserver vs1 -v3 enabled -v4.1 enabled -v4.1-acl enabled
   ```

3. Verify that NFS is enabled by using the `vserver nfs show` command.

   Example

   ```
   cluster1::> vserver nfs show -vserver vs1
   Vserver: vs1
   General NFS Access: true
   NFS v3: enabled
   NFS v4.0: disabled
   UDP Protocol: disabled
   TCP Protocol: enabled
   Spin Authentication: disabled
   ```
4. If you enabled NFSv3 access, mount the client over NFSv3 and create a new directory.

5. If you enabled NFSv4.1, mount the client over NFSv4.1 and create a new directory.

Related concepts

- Support for NFS on Infinite Volumes on page 71

Related information

- NFS management
- ONTAP 9 commands

Setting up basic SMB access to an Infinite Volume

If you want Windows users to access an Infinite Volume over SMB, you must create a CIFS server on the Storage Virtual Machine (SVM) with Infinite Volume and create a CIFS share on the Infinite Volume. You can then test the access from an SMB client, create folders, and create shares to specific folders.

Before you begin

- You must have the name of the domain that you will associate with the CIFS server.
- You must have the name and password of a Windows account with sufficient privileges to add computers to the chosen organizational unit.
- The directory where you want to create the share must already exist.
  If no directories exist, you can create a share on the Infinite Volume as a whole and use it to create the directory structure for future shares.

About this task

The way that you set up SMB access is the same for all SVMs, except that all shares on an SVM with Infinite Volume must be located at or under the Infinite Volume's junction path.

Steps

1. Determine what optional CIFS functionality that you will enable.

2. Configure DNS on the SVM with Infinite Volume by using the vserver services dns create command.

Example

```
cluster::> vserver services dns create -vserver vs1 -domains example.com -name-servers 10.1.1.50,10.1.1.51
```
3. Create a CIFS server and associate it with an Active Directory domain by using the `vserver cifs create` command and then entering the user name and password of the Windows account with sufficient privileges to add computers to the chosen organizational unit.

   **Example**
   
   ```bash
   cluster::> vserver cifs create -vserver vs1 -cifs-server CIFS1 -domain example.com
   ```

4. Verify that CIFS is enabled by using the `vserver cifs show` command.

   **Example**
   
   ```bash
   cluster::> vserver cifs show -vserver vs1
   ``

<table>
<thead>
<tr>
<th>Vserver</th>
<th>Server Name</th>
<th>Domain/Workgroup Name</th>
<th>Authentication Style</th>
</tr>
</thead>
<tbody>
<tr>
<td>vs1</td>
<td>CIFS1</td>
<td>EXAMPLE</td>
<td>domain</td>
</tr>
</tbody>
</table>

5. Create a share on the Infinite Volume by using the `vserver cifs share create` command on the junction path of the Infinite Volume.

   The path of every SMB share must be located at or under the junction path for the Infinite Volume. You must not create a share at the root of the SVM with Infinite Volume or in the `.system` directory.

   **Example**
   
   ```bash
   cluster::> vserver cifs share create -vserver vs1 -share-name InfiniteVol -path /NS
   ```

6. On an SMB client, map a share in either of the following ways:

   - In Windows Explorer, select **Tools > Map Network Drive**, and enter `\cifs_server_name\share_name`.
   - In a command window, enter the following command:
     ```bash
     net use * \cifs_server_name\share_name /user:domain_name\user_name
     ```

7. On the same SMB client, test the access by performing the following actions:

   a. View the share.
   b. Create a folder in the share.
   c. Create a new file in the share.
   d. Modify the file.
   e. Delete the file.

8. Create the directory structure that you want inside the Infinite Volume.

   **Example**
   
   You can create a folder called `Engineering` for the Engineering department.

9. Create shares for specific folders on the Infinite Volume by using the `vserver cifs share create` command.
Example

The following command creates a share on the Engineering folder for the Engineering department:

```
cluster::> vserver cifs share create -vserver vs1 -share-name Engineering -path /ns/Engineering -comment "Engineering Department"
```

Related concepts

- *Support for CIFS on Infinite Volumes* on page 72

Related information

- *SMB/CIFS management*
- *ONTAP 9 commands*

Configuring group mapping on an Infinite Volume

If an Infinite Volume is accessed by both SMB and NFSv4.1 clients, you should ensure that NFSv4.1 clients see accurate security information by mapping Windows groups to UNIX groups and by configuring the default UNIX group.

Before you begin

- The UNIX and Windows groups that you want to map must exist.
- The UNIX group that you want to set as a default must exist.
- You must be aware of the conversion rules that apply to name mapping of both users and groups.

About this task

You should perform this task only if NFSv4.1 ACLs are enabled on the Infinite Volume and the Infinite Volume supports access through both SMB and NFSv4.1.

Steps

1. Create mappings from Windows groups to UNIX groups by using the `vserver group mapping create` command.

   In all the mappings, you should set the `-direction` parameter to `win-unix`, because Windows-to-UNIX mappings are the only ones that are relevant.

2. View the mappings by using the `vserver group mapping show` command, and review the output to evaluate whether the mappings match what you intend.

3. If necessary, fix the mappings with any of the following commands:

   - `vserver group-mapping modify`
   - `vserver group-mapping insert`
   - `vserver group-mapping swap`
   - `vserver group-mapping delete`

4. Set the default UNIX group by using the `vserver cifs options modify` command with the `-default-unix-group` parameter.
Related concepts

*How group mapping supports multiprotocol access to Infinite Volumes* on page 77

Related information

*SMB/CIFS management*

*NFS management*

**Controlling access to an Infinite Volume with IP-based export rules**

The export policy on an Infinite Volume allows full read/write access by default. If you want to restrict access based on IP address, you must define export rules for specific host names, IP addresses, or netgroups in the default export policy called `repos_namespace_export_policy`.

**About this task**

By default, an export policy affects only NFS clients, not SMB clients.

**Steps**

1. View the export rules in the Infinite Volume's export policy by using the `vserver export-policy rule show` command with the `-vserver` and `-policyname` parameters.

   **Example**

   ```
   cluster1::> vserver export-policy rule show -vserver vs1 -policyname repos_namespace_export_policy
   
   Policy          Rule    Access   Client                RO
   Vserver      Name            Index   Protocol Match                 Rule
   ------------ --------------- ------  -------- --------------------- ---------
   vs0          repos_namespace_export_policy 1 0.0.0.0/0             any
   vs0          repos_namespace_export_policy 2 ::0/0                 any
   
   2 entries were displayed.
   ```

2. Define export rules for the export policy by using the `vserver export-policy rule create`, `vserver export-policy rule modify`, `vserver export-policy rule delete`, and `vserver export-policy rule setindex` commands.

   **Example**

   ```
   cluster1::> vserver export-policy rule create -vserver vs1 -policyname repos_namespace_export_policy -ruleindex 3 -protocol any -clientmatch .example.com -rorule any -rwrule "ntlm,krb5,sys" -anon 65534 -allow-suid false -allow-dev false
   ```

3. Verify that the rules match your requirements by using the `vserver export-policy rule show` command with the `-vserver` and `-policyname` parameters.

**Related concepts**

*How export policies affect access to an Infinite Volume* on page 74

**Related information**

*NFS management*
Controlling SMB access to an Infinite Volume share with share ACLs

By default, an SMB share on an Infinite Volume has a share-level ACL that allows all Windows users Full Control within the share. If you want to restrict specific Windows users and groups from accessing the share, you can modify the ACEs in the share ACL.

Before you begin

- The share must already exist.
- You must have the names of the Windows users and groups that you want to restrict.

About this task

You configure share ACLs on a Storage Virtual Machine (SVM) with Infinite Volume the same way you configure them on an SVM with FlexVol volumes.

Steps

1. Display the share ACL on the share by using the `vserver cifs share show` command.
2. Display the ACEs in the share ACL by using the `vserver cifs share access-control show` command.
3. Modify the ACEs in the share ACL by using any of the following commands:
   - `vserver cifs share access-control create` command
   - `vserver cifs share access-control modify` command
   - `vserver cifs share access-control delete` command

Example

If you want everyone to view the share but only people in the Engineering department to work on the share, you can modify the existing ACE so that Everyone has read-only access, and then add an ACE that gives full control to the Engineering group.

4. Verify that the share ACL contains the ACEs you want by using the `vserver cifs share access-control show` command.

Related concepts

*How share ACLs affect SMB access to Infinite Volumes* on page 75

Related information

*SMB/CIFS management*

Controlling access to Infinite Volumes with file permissions

With an Infinite Volume, you can set file permissions using one or more methods, including UNIX permissions, NTFS file permissions, and NFSv4.1 ACLs. You can set file permissions on the Infinite Volume or on directories inside the Infinite Volume.

Before you begin

- In most cases, you must have access to a Windows or UNIX client to set permissions. You can use the CLI to set UNIX permissions on the Infinite Volume. In all other cases, you must use a client to set file permissions.
• You must have decided what method you will use for the Infinite Volume or for the directories inside it.

**About this task**

You need to set file permissions only if you do not want to use the default UNIX permissions that were specified when the Infinite Volume was created.

**Choices**

• If you want to change the default UNIX permissions for the entire Infinite Volume, use the `volume modify` command with the `-unix-permissions` parameter.

**Example**

The following command gives read, write, and execute permissions to the owner and group and no permissions to others:

```
cluster::> volume modify -vserver vsl -volume ivol -unix-permissions 0770
```

The permissions are applied to all newly created files, depending on other configurations such as inheritable ACLs or client-side settings.

• If you want a directory to have UNIX permissions that are different than the default UNIX permissions for the Infinite Volume, use an NFS client, such as a Linux client, to configure permissions for the directory by using the `umask` and `chmod` UNIX commands.

For information about configuring UNIX permissions, see the Linux documentation.

• If you want to use NTFS file permissions (SMB ACLs) on the entire Infinite Volume or a directory inside the Infinite Volume, use a Windows client to set the permissions on the volume or a directory, keeping in mind the following considerations:

  ◦ Similar to when you set NTFS file permissions on any directory or volume, you must decide whether the permissions will be inherited by selecting a value in the **Apply to** box of the Windows Security tab in the Windows Properties window.

  ◦ On an Infinite Volume, you can set permissions for UNIX users and groups, in addition to Windows users and groups.

  ◦ Although you can set SACLs, they have no effect on Infinite Volumes.

The new NTFS file permissions are propagated by the client to existing files in the directory, and used when creating new files, depending on the setting in the **Apply to** box.

• If you want to use NFSv4.1 ACLs on the entire Infinite Volume or a directory inside the Infinite Volume, use an NFSv4.1 client to set an NFSv4.1 ACL either by editing the ACL directly or by using GUI software that is capable of editing NFSv4.1 ACLs.

For information about configuring NFSv4.1 ACLs, see the NFSv4.1 protocol standard or the GUI software documentation.

The NFSv4.1 ACL is set on all new and existing files in the directory. Depending on how you configured the ACL, the entire ACL or ACEs within the ACL can be inherited by child directories.

**Related information**

*SMB/CIFS management*
Advanced Infinite Volume file access concepts

When you configure file access to an Infinite Volume, it is helpful to understand the outcome when each type of client sets permissions, the outcome when each type of client display permissions, and other advanced concepts.

What happens when NFS clients set UNIX permissions on Infinite Volumes

When an NFS client issues a UNIX command that affects permissions on a file on an Infinite Volume, the success of the command depends on several factors, including the effective file permissions, the specific commands, and whether the ACL preservation feature is enabled on the NFS server.

ACL preservation, which is enabled by default, is a configurable feature of each NFS server. It is configured with the `-v4-acl-preserve` parameter of the `nfs modify` command. Although the parameter refers to NFSv4, it also applies to SMB ACLs.

Setting UNIX permissions when UNIX permissions are effective

If an NFS client uses UNIX commands to set permissions on a file where UNIX permissions are effective, the UNIX permissions are updated.

Setting mode bits when NFSv4.1 ACLs are effective

If an NFS client uses UNIX commands to set permissions on a file that has an NFSv4.1 ACL, the NFSv4.1 ACL can be modified or unaffected, as shown in the following table:

<table>
<thead>
<tr>
<th>UNIX command</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>chmod on permissions</td>
<td>If ACL preservation is enabled, the NFSv4.1 ACL remains effective and ACEs for OWNER@, GROUP@, or EVERYONE@ are modified or added to the NFSv4.1 ACL. If ACL preservation is disabled, the new UNIX permissions sent in the command become effective.</td>
</tr>
<tr>
<td>chgrp or chown</td>
<td>There is no effect on the NFSv4.1 ACL.</td>
</tr>
<tr>
<td>chmod on the setuid, setgid, or sticky bit</td>
<td>Inside the NFSv4.1 ACL, the affected special mode bits are added or modified. (The display mode bits are also updated.)</td>
</tr>
</tbody>
</table>

Setting mode bits when NTFS file permissions are effective

If an NFS client uses UNIX commands to set permissions on a file that has NTFS file permissions (which are stored as SMB ACLs), the SMB ACL can be modified or unaffected, as shown in the following table:

<table>
<thead>
<tr>
<th>UNIX command</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>Any of the following commands: chmod on permissions, chgrp, chown</td>
<td>If ACL preservation is enabled, the SMB ACL remains effective and ACEs for OWNER@, GROUP@, or EVERYONE@ are modified or added to the SMB ACL. If ACL preservation is disabled, the new UNIX permissions sent in the command become effective.</td>
</tr>
<tr>
<td>UNIX command</td>
<td>Result</td>
</tr>
<tr>
<td>----------------------</td>
<td>---------------------------------------------</td>
</tr>
<tr>
<td><code>chmod</code> on the setuid, setgid, or sticky bit</td>
<td>There is no effect on the SMB ACL.</td>
</tr>
</tbody>
</table>

**What happens when SMB clients set NTFS file permissions on Infinite Volumes**

When an SMB client sets NTFS file permissions on a file on an Infinite Volume, an SMB ACL is created. If any UNIX users or groups were included in the permissions set by the SMB client, the UNIX names are stored in the resulting SMB ACL.

When an SMB client sets NTFS file permissions on a file on an Infinite Volume, the following occurs:

1. An SMB ACL is set in one of the following ways:

<table>
<thead>
<tr>
<th>If effective permissions are...</th>
<th>The SMB ACL is set in this way</th>
</tr>
</thead>
<tbody>
<tr>
<td>NTFS file permissions</td>
<td>The SMB ACL is updated.</td>
</tr>
<tr>
<td>UNIX permissions</td>
<td>An SMB ACL is created.</td>
</tr>
<tr>
<td>NFSv4.1 ACL</td>
<td>An SMB ACL is created.</td>
</tr>
</tbody>
</table>

2. If any ACEs reference UNIX users or groups, the ACEs are stored in the SMB ACL in NFSv4.1 style, using the UNIX user and group names.

3. Display mode bits are created based on the user in the ACL with the most access.

**What happens when NFS clients set NFSv4.1 ACLs on Infinite Volumes**

When an NFSv4.1 client sets an NFSv4.1 ACL on a file on an Infinite Volume, an NFSv4.1 ACL is created.

When an NFSv4.1 client sets an NFSv4.1 ACL on a file on an Infinite Volume file, the following occurs:

1. An NFSv4.1 ACL is set in one of the following ways:

<table>
<thead>
<tr>
<th>If effective permissions are...</th>
<th>The NFSv4 ACL is set in this way</th>
</tr>
</thead>
<tbody>
<tr>
<td>NTFS file permissions</td>
<td>An NFSv4.1 ACL is created.</td>
</tr>
<tr>
<td>UNIX permissions</td>
<td>An NFSv4.1 ACL is created.</td>
</tr>
<tr>
<td>NFSv4.1 ACL</td>
<td>A new NFSv4.1 ACL is created.</td>
</tr>
</tbody>
</table>

2. Display mode bits are created based on the user in the ACL with the least access.

**How SMB clients set permissions for UNIX users of Infinite Volumes**

On Infinite Volumes, SMB clients are able to set NTFS file permissions for UNIX users and groups.

SMB clients can work with permissions in the following ways:

- Set permissions for any specific user or group in the NFS domain.
- Set permissions for the NFS well-known principals OWNER@, GROUP@, and EVERYONE@.
- Make any specific user or group in the NFS domain the owner of a file.
How file permissions on Infinite Volumes are displayed to clients

File permissions on an Infinite Volume are displayed to SMB and NFS clients irrespective of the type of file permissions that are effective, including NFSv4.1 ACLs, NTFS permissions, and UNIX permissions. The details vary, depending on the exact combination of client and effective file permissions.

Permissions displayed on Windows clients

When a user on a Windows client views the security settings of a file or folder, the following information is displayed:

<table>
<thead>
<tr>
<th>Effective file permissions</th>
<th>What NTFS file permissions are shown</th>
</tr>
</thead>
<tbody>
<tr>
<td>UNIX permissions</td>
<td>Three entries for OWNER@, GROUP@, and EVERYONE@ (using the NFSv4 well-known principals) Three entries for the setuid, setgid, and sticky bit (using principals known only to Data ONTAP). These entries are for informational purposes only.</td>
</tr>
<tr>
<td>NFSv4.1 ACL</td>
<td>Entries for each UNIX user and group, using either the UNIX user and group names or the NFSv4 well-known principals OWNER@, GROUP@, and EVERYONE@</td>
</tr>
<tr>
<td>NTFS file permissions</td>
<td>Standard NTFS file permissions, except that some entries can represent UNIX users and groups, including the NFSv4 well-known principals OWNER@, GROUP@, and EVERYONE@</td>
</tr>
</tbody>
</table>

Permissions displayed when UNIX permissions are requested

When a user displays UNIX permissions by using a `ls -l` command on an NFSv3 or NFSv4.1 client, the following information is displayed:

<table>
<thead>
<tr>
<th>Effective file permissions</th>
<th>What mode bits are shown</th>
</tr>
</thead>
<tbody>
<tr>
<td>UNIX permissions</td>
<td>Standard UNIX permissions</td>
</tr>
<tr>
<td>NFSv4.1 ACL</td>
<td>“Display mode bits”, which represent the least access that is granted to the requesting user based on the ACL.</td>
</tr>
<tr>
<td>NTFS file permissions</td>
<td>“Display mode bits,” represent the most access that is granted to the requesting user based on the NTFS file permissions.</td>
</tr>
</tbody>
</table>

Permissions displayed when NFSv4.1 ACLs are requested

When a user displays an ACL on an NFSv4.1 client—for example, by using the `ls -V` command—the following information is displayed:

<table>
<thead>
<tr>
<th>Effective file permissions</th>
<th>What NFSv4.1 ACL is shown</th>
</tr>
</thead>
<tbody>
<tr>
<td>UNIX permissions</td>
<td>Entries for OWNER@, GROUP@, and EVERYONE@</td>
</tr>
<tr>
<td>NFSv4.1 ACL</td>
<td>Standard NFSv4.1 ACL</td>
</tr>
<tr>
<td>Effective file permissions</td>
<td>What NFSv4.1 ACL is shown</td>
</tr>
<tr>
<td>---------------------------</td>
<td>--------------------------</td>
</tr>
<tr>
<td>NTFS file permissions</td>
<td>Standard NFSv4.1 ACL but all Windows user and group names have been mapped to UNIX user and group names</td>
</tr>
</tbody>
</table>

**Note:** Group mapping must be configured, or "nobody" appears as the group name.

**What happens when NFSv4.1 ACLs contain the user nobody**

If NFSv4.1 clients display ACLs on files that have NTFS file permissions and the ACLs contain access control entries (ACEs) for the user nobody, user or group mapping has failed. These ACLs should not be saved, or the original Windows SID will be accidentally overwritten.

When NTFS file permissions are displayed as an NFSv4.1 ACL, the Windows user and group is mapped to a UNIX user and group. If the mapping fails, the affected Windows SID appears as "nobody@v4-id-domain", where v4-id-domain is the NFSv4 ID mapping domain.

The user or group mapping can fail for the following reasons:

- Use of well-known Windows SIDs, which are translated to nobody instead of well-known NFS principals
- A Windows 2008 client has set permissions for a new well-known Windows principal that a Windows 2003 domain controller cannot understand because the principal was introduced in Windows 2008
- Issues with the configuration of user mapping, group mapping, the default UNIX user, or the default UNIX group

**Why an Infinite Volume’s size appears smaller from a client**

When you use a client to view the size of an entire Infinite Volume, the size appears smaller than the size reported in Data ONTAP, because the namespace mirror constituents are not included in the size that is displayed to clients.

For example, if an Infinite Volume has a size of 100 TB, which includes a 10 TB namespace mirror constituent, the size that is displayed to a client is 90 TB instead of 100 TB.

Namespace mirror constituents are not included because they are data protection mirror copies of the namespace consistent and therefore do not represent the space available for data. The namespace constituent is included because it is part of the mechanism that is used to store data.

Clients can display the size of a volume in various ways. For example, a Windows user with a share for the entire volume can view the share’s properties, or a UNIX user can use the `df` command.

**How locks work on Infinite Volumes**

In Infinite Volumes, the support of locks is tied to the versions of the supported protocols. When you display information about locks, the volume name, file path, and protocol can display unique settings for Infinite Volumes.

**Oplocks on Infinite Volumes**

Traditional oplocks are supported on Infinite Volumes. They can be enabled on an SMB share.

Lease oplocks, which require SMB 2.x, are not supported on Infinite Volumes.

**NFSv4.1 delegations**

NFSv4.1 delegations are not supported on Infinite Volumes.
Output of locks on Infinite Volumes

When you display information about locks for files on an Infinite Volume by using the `vserver locks show` command, the output shows the following values:

- The volume name of a client-held lock, which is the name of the Infinite Volume data constituent that holds the data of the locked file
- The file path of the lock, which is the path to the namespace constituent using the format `/Infinite_Volume_junction_path/optional_directories/filename`—for example, `/NS/Users/Bob/cifsfile.txt`.
- While a file is being removed or if the namespace constituent is unavailable, the path to the data constituent might be displayed—for example, `/.system/constituents/1024_data0002/1647/P48BAAIEIAAAAAAAAI8NAAAWIImIC`.
- The protocol, which can be `crposix`
  A `crposix` lock indicates a transitory lock that is created internally by the Infinite Volume. If a `crposix` lock persists for more than a minute, you should break the lock.

Related information

- **SMB/CIFS management**
- **NFS management**

Comparison of unified and mixed security styles

Understanding the differences between unified and mixed security styles can help you understand how file permissions for these security styles behave on FlexVol volumes and Infinite Volumes.

Access checks

When a client tries to access a file, unified security style handles the access check the same way except that unified security style also supports UNIX principals in NTFS file permissions.

The following table identifies whether the outcome is the same for mixed and unified security styles when clients try to access a file:

<table>
<thead>
<tr>
<th>When...</th>
<th>Of a file where UNIX permissions are effective</th>
<th>Of a file that uses NTFS file permissions</th>
<th>Of a file that has an NFSv4.1 ACL</th>
</tr>
</thead>
<tbody>
<tr>
<td>An NFSv3 or NFSv4.1 client tries to access a file</td>
<td>The outcome is the same: the UNIX credentials are compared to the UNIX permissions.</td>
<td>The outcome is the same: the user's Windows credential (obtained by using user mapping) is compared to the Windows SIDs in the NTFS file permissions. In addition, under unified security style, the user's UNIX credential is compared against any UNIX principals in the NTFS file permissions.</td>
<td>The outcome is the same: the UNIX credentials are compared to the NFSv4.1 ACL.</td>
</tr>
</tbody>
</table>
### Ability to set permissions

The ability to set permissions is similar under both security styles, but unified security style has the following benefits:

- When an NFS client sets UNIX permissions on a file that uses NTFS file permissions, the changes are merged into the NTFS file permissions.

- When an SMB client sets NTFS permissions on a file, the permissions can include UNIX principals.

The following table identifies whether the outcome is the same under mixed and unified security styles when clients set file permissions:

<table>
<thead>
<tr>
<th>When...</th>
<th>Of a file where UNIX permissions are effective</th>
<th>Of a file that uses NTFS file permissions</th>
<th>Of a file that has an NFSv4.1 ACL</th>
</tr>
</thead>
</table>
| An NFSv3 or NFSv4.1 client sets UNIX permissions by changing permissions | The outcome is the same: the UNIX permissions are updated. | The outcome is different:  
  - Under mixed security style, the UNIX permissions become effective.  
  - Under unified security style, the ACEs for OWNER@, GROUP@, and EVERYONE@ are modified or added to the NTFS file permissions. (This occurs only if the v4 ACL preserve parameter is enabled.) | The outcome is the same: ACEs for OWNER@, GROUP@, and EVERYONE@ are modified or added in the NFSv4.1 ACL. (This occurs only if the v4 ACL preserve parameter is enabled.) |
When...

<table>
<thead>
<tr>
<th>Of a file where UNIX permissions are effective</th>
<th>Of a file that uses NTFS file permissions</th>
<th>Of a file that has an NFSv4.1 ACL</th>
</tr>
</thead>
<tbody>
<tr>
<td>An NFSv3 or NFSv4.1 client sets UNIX permissions by changing the owner or group</td>
<td>The outcome is the same: the owner and the primary group of the file are updated.</td>
<td>The outcome is different: • Under mixed security style, the UNIX permissions become effective. • Under unified security style, the Owner SID and the Group SID of the existing NTFS permissions are updated. (This occurs only if the v4 ACL preserve parameter is enabled.)</td>
</tr>
<tr>
<td>An SMB client sets NTFS permissions</td>
<td>The outcome is similar: NTFS file permissions are set. In addition, under unified security style, the NTFS file permissions can include UNIX users.</td>
<td>The outcome is similar: NTFS file permissions replace the existing NTFS file permissions. In addition, under unified security style, the NTFS file permissions can include UNIX users.</td>
</tr>
<tr>
<td>An NFSv4.1 client sets an NFSv4.1 ACL</td>
<td>The outcome is the same: an NFSv4.1 ACL is set.</td>
<td>The outcome is the same: an NFSv4.1 ACL is set.</td>
</tr>
</tbody>
</table>

Viewing permissions

The ability of clients to view permissions is significantly better under unified security style in the following ways:

- An SMB client can show the permissions of a file that has an NFSv4.1 ACL.
- An NFSv4.1 client can display an equivalent NFSv4.1 ACL for a file that has NTFS file permissions.
- When an SMB client shows permissions of a file that uses NTFS file permissions, UNIX principals can appear in the permissions.

The following table identifies whether the outcome is the same or different under mixed and unified security styles when clients show file permissions:

<table>
<thead>
<tr>
<th>When...</th>
<th>Of a file where UNIX permissions are effective</th>
<th>Of a file that uses NTFS file permissions</th>
<th>Of a file that has an NFSv4.1 ACL</th>
</tr>
</thead>
<tbody>
<tr>
<td>An NFS client shows UNIX permissions</td>
<td>The outcome is the same: the UNIX permissions are displayed.</td>
<td>The outcome is the same: the display UNIX permissions (that are calculated each time an ACL is set) are displayed.</td>
<td>The outcome is the same: the display mode bits (that are calculated each time an ACL is set) are displayed.</td>
</tr>
<tr>
<td>When...</td>
<td>Of a file where UNIX permissions are effective</td>
<td>Of a file that uses NTFS file permissions</td>
<td>Of a file that has an NFSv4.1 ACL</td>
</tr>
<tr>
<td>---------</td>
<td>-----------------------------------------------</td>
<td>--------------------------------------------</td>
<td>----------------------------------</td>
</tr>
<tr>
<td>An SMB client shows permissions</td>
<td>The outcome is similar: permissions are displayed for Owner, Group, and Everyone. The sticky bit, setuid flag, and setgid flag are also displayed if they are set. Under mixed security style, effective permissions are also displayed for current user. Under unified security style, permissions for OWNER@, GROUP@, and EVERYONE@ appear as UNIX names when displayed in Windows Explorer.</td>
<td>The outcome is generally the same: the NTFS permissions are displayed. In addition, under unified security style, UNIX users and groups can appear in the permissions.</td>
<td>The outcome is different:  - Under mixed security style, permissions are displayed for Owner, Group, and Everyone, and effective permissions are displayed for the current user. The sticky bit, setuid flag, and setgid flag are also displayed if they are set.  - Under unified security style, effective NTFS file permissions are displayed. All of the users and groups are UNIX principals.</td>
</tr>
<tr>
<td>An NFSv4.1 client displays an NFSv4.1 ACL</td>
<td>The outcome is the same: an NFSv4.1 ACL is displayed, containing ACEs for OWNER@, GROUP@, and EVERYONE@.</td>
<td>The outcome is different:  - Under mixed security style, an NFSv4.1 ACL is displayed, containing ACEs for OWNER@, GROUP@, and EVERYONE@.  - Under unified security style, an NFSv4 ACL is displayed, containing users and groups that have been mapped from Windows to UNIX. (User and group mappings must be configured.)</td>
<td>The outcome is the same: the NFSv4.1 ACL is displayed.</td>
</tr>
</tbody>
</table>
Providing backups of Infinite Volumes using Snapshot copies

You can use Snapshot copies to create read-only images of Infinite Volumes that capture the state of the file system at a point in time, and you can use the Snapshot copies to recover older versions of the file system. You can also use Snapshot policies to manage Snapshot copy schedules.

Related concepts

*Providing backups of Infinite Volumes with storage classes using Snapshot copies* on page 219

Guidelines for working with Snapshot copies of Infinite Volumes

You can create, manage, and restore Snapshot copies of Infinite Volumes. However, you should be aware of the factors affecting the Snapshot creation process and the requirements for managing and restoring the copies.

**Guidelines for creating Snapshot copies of Infinite Volumes**

- The volume must be online.
  You cannot create a Snapshot copy of an Infinite Volume if the Infinite Volume is in a mixed state because a constituent is offline.

- The Snapshot copy schedule should not be less than hourly.
  It takes longer to create a Snapshot copy of an Infinite Volume than of a FlexVol volume. If you schedule Snapshot copies of Infinite Volumes for less than hourly, Data ONTAP tries but might not meet the schedule. Scheduled Snapshot copies are missed when the previous Snapshot copy is still being created.

- Time should be synchronized across all the nodes that the Infinite Volume spans.
  Synchronized time helps schedules for Snapshot copies run smoothly and restoration of Snapshot copies function properly.

- The Snapshot copy creation job can run in the background.
  Creating a Snapshot copy of an Infinite Volume is a volume-level job (unlike the same operation on a FlexVol volume), and the operation spans multiple nodes in the cluster. You can force the job to run in the background by setting the `-foreground` parameter of the `volume snapshot create` command to `false`.

- After you create Snapshot copies of an Infinite Volume, you cannot rename the copy or modify the comment or SnapMirror label for the copy.

**Guidelines for managing Snapshot copy disk consumption**

- You cannot calculate the amount of disk space that can be reclaimed if Snapshot copies of an Infinite Volume are deleted.

- The size of a Snapshot copy for an Infinite Volume excludes the size of namespace mirror constituents.

- To reclaim disk space used by Snapshot copies of Infinite Volumes, you must manually delete the copies.
You cannot use the automatic Snapshot copy deletion feature to automatically delete Snapshot copies of Infinite Volumes. However, you can manually delete Snapshot copies of Infinite Volumes, and you can run the delete operation in the background.

**Guidelines for restoring Snapshot copies of Infinite Volumes**

- You must restore the entire Snapshot copy of the Infinite Volume. You cannot restore single files or parts of files. You also cannot restore a Snapshot copy of a single constituent.

- The Snapshot copy must be in a valid state. You cannot use admin privilege to restore a Snapshot copy of an Infinite Volume if the copy is in a partial or invalid state because the commands require diagnostic privilege. However, you can contact technical support to run the commands for you.

- Restored Snapshot copies inherit the current efficiency settings of the Infinite Volume. If the State field contains a dash (“-“) when you run the `volume snapshot show` command after restoring a Snapshot copy of the Infinite Volume, one or more constituents are offline or have a different state than the other constituents. You can align the State field for constituents in an Infinite Volume by setting efficiency settings again on the Infinite Volume.

**Related information**

*Logical storage management*

**What the default Snapshot policy for an Infinite Volume is**

The default Snapshot policy for an Infinite Volume is named `default-1weekly`, and the Snapshot policy includes a default schedule for creating Snapshot copies of the Infinite Volume. You can modify the default Snapshot policy, or you can assign a custom Snapshot policy to an Infinite Volume.

The default schedule automatically creates a Snapshot copy every hour, once a day, and once a week. A maximum of six hourly, two daily, and one weekly Snapshot copies are retained. Older Snapshot copies are automatically deleted to free up space. You can modify the schedule, or you can add more schedules to the Snapshot policy.

**How Snapshot policies are associated with volumes**

Unless you specify a Snapshot policy when you create a FlexVol volume, a volume inherits the Snapshot policy associated with its containing Storage Virtual Machine (SVM).

When you create the SVM, you can specify a Snapshot policy. If you do not specify a Snapshot policy when you create the SVM, a default Snapshot policy is associated with the SVM. The default Snapshot policy for an SVM is named `default`.

When you create a volume, you can specify a Snapshot policy. If you do not specify a Snapshot policy when you create a volume, the volume inherits the Snapshot policy associated with its containing SVM.
What Snapshot copies you can access

The .snapshot directory on every Infinite Volume contains all of the Snapshot copies created for the volume. You can access only the Snapshot copies in the .snapshot directory that are displayed when you use the \texttt{volume snapshot show} command.

Some Snapshot copies in the .snapshot directory are used only to support internal system processes for the volume, such as data protection of the namespace constituent for an Infinite Volume, and you cannot access these Snapshot copies. You can access any of the Snapshot copies for a volume that are displayed when you use the \texttt{volume snapshot show} command. The command hides the types of Snapshot copies that you cannot access.

\textbf{Note:} If an NFS or CIFS client tries to access any of the Snapshot copies used for internal system processes, the client stops working.

When Snapshot copies of Infinite Volumes are accessible

Snapshot copies of an Infinite Volume are restorable and fully accessible to clients only when the Snapshot copies are in a valid state.

A Snapshot copy of an Infinite Volume consists of information spanning multiple constituents across multiple aggregates. Although a Snapshot copy cannot be created if a constituent is offline, a constituent might be deleted or taken offline after the Snapshot copy is created. If a Snapshot copy of an Infinite Volume references a constituent that is offline or deleted, the Snapshot copy might not be fully accessible to clients or restorable.

The availability of a Snapshot copy of an Infinite Volume is indicated by its state, as explained in the following table:

<table>
<thead>
<tr>
<th>State</th>
<th>Description</th>
<th>Client access to the Snapshot copy</th>
<th>Impact on restore</th>
</tr>
</thead>
<tbody>
<tr>
<td>Valid</td>
<td>The copy is complete.</td>
<td>Fully accessible to clients</td>
<td>Can be restored</td>
</tr>
<tr>
<td>Partial</td>
<td>Data is missing or incomplete.</td>
<td>Partially accessible to clients</td>
<td>Cannot be restored without assistance from technical support</td>
</tr>
<tr>
<td>Invalid</td>
<td>Namespace information is missing or incomplete.</td>
<td>Inaccessible to clients</td>
<td>Cannot be restored</td>
</tr>
</tbody>
</table>

The validity of a Snapshot copy is not tied directly to the state of the Infinite Volume. A valid Snapshot copy can exist for an Infinite Volume with an offline state, depending on when the Snapshot copy was created compared to when the Infinite Volume went offline. For example, a valid Snapshot copy exists before a new constituent is created. The new constituent is offline, which puts the Infinite Volume in an offline state. However, the Snapshot copy remains valid because it references its required preexisting constituents. The Snapshot copy does not reference the new, offline constituent.

When client access to Snapshot copies is temporarily disabled

When a node in a cluster that contains an Infinite Volume loses quorum, client access to the Snapshot copies in the Infinite Volume is disabled.

When a node loses quorum, it cannot communicate with the cluster and receive updated information about the Snapshot copies of the Infinite Volume. While the node is out of quorum, client access to
Snapshot copies in an Infinite Volume is temporarily disabled, and the storage system uses the following methods to indicate that it is inaccessible:

- You receive an event message from the Event Management System when client access to Snapshot copies is temporarily disabled.
- Clients become unresponsive when accessing Snapshot copies in an Infinite Volume.
- You receive an event message from the Event Management System when the node regains quorum and resumes communication with the cluster.

**How some Snapshot and SnapMirror commands run on Infinite Volumes**

You must run most commands sequentially on an Infinite Volume to allow one job to complete before you start the next job. However, you can run the `snapshot delete` and the `snapmirror abort` commands simultaneously with other commands on Infinite Volumes.

When you run a command on an Infinite Volume, the Infinite Volume uses a job to run the command on all of the constituents that it contains. As a result, you only have to run Snapshot and SnapMirror commands on the Infinite Volume, not the individual constituents. Because the Infinite Volume takes each command, creates a job, and runs the command on all of its constituents, it can take some time for each command to complete the operation. If you try to run two commands on the same Infinite Volume, you can create a conflict that results in an error. To avoid conflicts, you must wait for each command to complete the operation before you run the next command.

However, you can run the following commands while other commands are running:

- **snapshot delete**
  
  You can run multiple `snapshot delete` commands at the same time, as long as the commands are not operating on the same Snapshot copy.

- **snapmirror abort**
  
  You can run a `snapmirror abort` command when another command is running, as long as you run the `snapmirror abort` command after you run the other command. If you run the `snapmirror abort` command first, you cannot run another command. The `snapmirror abort` command must complete its operation before you run the next command.

**Related concepts**

*How operations run on Infinite Volumes* on page 180

**Creating Snapshot copies of Infinite Volumes**

You can create Snapshot copies of Infinite Volumes with and without storage classes. Snapshot copies create a backup of the Infinite Volume.

**Step**

1. Create a Snapshot copy of an Infinite Volume by using the `volume snapshot create` command.

**Example**

In the following example, a Snapshot copy named `daily_backup` is created for an Infinite Volume named `repo_vol`: 
After you finish

You can configure Snapshot copy schedules and retention by using the `volume snapshot policy` commands.

Viewing Snapshot copies of constituents of an Infinite Volume

You can view Snapshot copies of constituents of an Infinite Volume by using the `volume snapshot show` command with the `-is-constituent true` parameter. This enables you to view Snapshot copies at the constituent level when you are investigating issues.

About this task

The following example is of an Infinite Volume named `repo_vol` with a Snapshot copy named `daily_backup`. The Infinite Volume contains the following constituents:

- Four data constituents named `repo_vol_1024_data0001`, `repo_vol_1024_data0002`, and so on
- A namespace constituent named `repo_vol_ns`
- A namespace mirror constituent named `repo_vol_ns_mirror`

A Snapshot copy of an Infinite Volume consists of coordinated Snapshot copies of all the constituents of an Infinite Volume.

Choices

- View the Snapshot copies for constituents of an Infinite Volume by using the `volume snapshot show -is-constituent true` command.

Example

In the following output, a Snapshot copy named `daily_backup` is displayed for each constituent of the Infinite Volume:

```
cluster1::> volume snapshot show -is-constituent true

---Blocks---
<table>
<thead>
<tr>
<th>Vserver</th>
<th>Volume</th>
<th>Snapshot</th>
<th>Size</th>
<th>Total%</th>
<th>Used%</th>
</tr>
</thead>
<tbody>
<tr>
<td>vs0</td>
<td>repo_vol_1024_data0001</td>
<td>daily_backup</td>
<td>76KB</td>
<td>0%</td>
<td>31%</td>
</tr>
<tr>
<td></td>
<td>repo_vol_1024_data0002</td>
<td>daily_backup</td>
<td>76KB</td>
<td>0%</td>
<td>31%</td>
</tr>
<tr>
<td></td>
<td>repo_vol_1024_data0003</td>
<td>daily_backup</td>
<td>76KB</td>
<td>0%</td>
<td>31%</td>
</tr>
<tr>
<td></td>
<td>repo_vol_1024_data0004</td>
<td>daily_backup</td>
<td>76KB</td>
<td>0%</td>
<td>31%</td>
</tr>
<tr>
<td></td>
<td>repo_vol_ns</td>
<td>snapmirror.9b6cce82-8845-11e3-ac94-123478563412_2147484675.2014-01-28_195500</td>
<td>96KB</td>
<td>0%</td>
<td>39%</td>
</tr>
<tr>
<td></td>
<td>repo_vol_ns_mirror0001</td>
<td>snapmirror.9b6cce82-8845-11e3-ac94-123478563412_2147484675.2014-01-28_195500</td>
<td>96KB</td>
<td>0%</td>
<td>36%</td>
</tr>
</tbody>
</table>
```
Note: The Snapshot copies with names that start with snapmirror. are used for the data protection mirror relationship between the namespace constituent and the namespace mirror constituent.

- View Snapshot copies of constituents and of volumes by using the `volume snapshot show -is-constituent *` command.

Example

In the following output, a Snapshot copy named daily_backup is displayed for the Infinite Volume and for each constituent of the Infinite Volume. Snapshot copies for the Storage Virtual Machine (SVM) root volume are also displayed:

```
cluster1::> volume snapshot show -is-constituent *

---Blocks---
Vserver  Volume   Snapshot                          Size Total% Used%
-------- -------- ------------------------------------- -------- ------ -----
node1    vol0     hourly.0                            1.97MB  0%   1%
node2    vol0     hourly.0                            1.83MB  0%   1%
node3    vol0     hourly.0                            1.77MB  0%   1%
node4    vol0     hourly.0                            1.64MB  0%   1%
vs0      repo_vol daily_backup                       388KB   0%  32%
         repo_vol_1024_data0001 daily_backup           76KB   0%  31%
         repo_vol_1024_data0002 daily_backup           76KB   0%  31%
         repo_vol_1024_data0003 daily_backup           76KB   0%  31%
         repo_vol_1024_data0004 daily_backup           76KB   0%  31%
         repo_vol_ns daily_backup                      84KB   0%  36%
         repo_vol_ns_daily_backup_9b6cce82-8845-11e3-ac94-123478563412_2147484675.2014-01-28_200500  72KB   0%  33%
         repo_vol_ns_daily_backup_9b6cce82-8845-11e3-ac94-123478563412_2147484675.2014-01-28_200000  80KB   0%  36%
         repo_vol_ns_daily_backup_9b6cce82-8845-11e3-ac94-123478563412_2147484675.2014-01-28_200500  72KB   0%  33%
         vs0_root hourly.2014-01-28_1805              60KB   0%  26%
         hourly.2014-01-28_1905                        64KB   0%  28%
         hourly.2014-01-28_2005                        40KB   0%  19%
```

17 entries were displayed.

Related tasks

- Viewing constituents of an Infinite Volume on page 195
- Viewing the state of Snapshot copies for an Infinite Volume on page 101

Related references

- Naming convention for constituents on page 197
Viewing the state of Snapshot copies for an Infinite Volume

You can view the state of Snapshot copies for an Infinite Volume to help you investigate issues. By default, the state of a Snapshot copy is not displayed.

Step

1. View the state of Snapshot copies for an Infinite Volume by using the `volume snapshot show` command with the `-state` parameter.

Example

In the following example, all Snapshot copies in a valid or partial state are displayed for the Infinite Volume named repo_vol:

```
cluster1::> volume snapshot show -vserver vs1 -volume repo_vol -fields state
vserver volume snapshot state
------- -------- -------------- -----
vs1      repo_vol weekly_snapshot valid
vs1      repo_vol daily_snapshot partial
2 entries were displayed.
```

In the following example, all Snapshot copies in an invalid state are displayed for the Infinite Volume named repo_vol:

```
cluster1::> volume snapshot show -vserver vs1 -volume repo_vol -state invalid
--- Blocks---
Vserver Volume Snapshot       Size Total
% Used%
-------- -------- --------------- --------
------ ----- ----
vs1      repo_vol invalid_snapshot  -
                   -
                 -
                   -
                   -
               invalid_snapshot3  -
                   -
               invalid_snapshot4  -
                   -
4 entries were displayed.
```

After you finish

You can delete Snapshot copies that are in an invalid state.

Related concepts

- *When Snapshot copies of Infinite Volumes are accessible* on page 97

Related tasks

- *Deleting invalid Snapshot copies of Infinite Volumes* on page 102
Deleting invalid Snapshot copies of Infinite Volumes

You cannot restore invalid Snapshot copies of an Infinite Volume, but you can delete invalid Snapshot copies of an Infinite Volume to free up space. You can delete either all Snapshot copies or individual Snapshot copies that are in an invalid state.

About this task

Snapshot copies can have a valid state when you create a Snapshot copy with the same name as an existing Snapshot copy or when a node panics. You should return the node to a healthy state before you delete invalid Snapshot copies. In some cases, when a node returns to a healthy state, the state of the Snapshot copy changes from invalid to valid.

Steps

1. Identify the names of the Snapshot copies in an invalid state for an Infinite Volume by using the `volume snapshot show` command with the `-state` parameter.

   **Example**
   
   In the following example, all Snapshot copies in an invalid state for the Infinite Volume named `repo_vol` are displayed:

   ```
   cluster1::> volume snapshot show -vserver vs1 -volume repo_vol -state invalid
   Blocks---
   Vserver  Volume    Snapshot                      Size  Total
   % Used%
   -------- ------- ----------------------------- ----  -------
   --------- ------ ----------------------------- ----  -------
   vs1      repo_vol invalid_snapshot             -    -
   -        invalid_snapshot2                    -    -
   -        invalid_snapshot3                    -    -
   -        invalid_snapshot4                    -    -
   4 entries were displayed.
   ```

2. Delete invalid Snapshot copies by using the `volume snapshot delete` command with the `-state` parameter.

   You must use brackets with the `snapshot delete` command to specify the `-state` parameter.

   **Example**
   
   In the following example, the Snapshot copy named `invalid_snapshot` is deleted for the Storage Virtual Machine (SVM) with Infinite Volume named `vs1`:

   ```
   cluster1::> volume snapshot delete { -vserver vs1 -state invalid -snapshot invalid_snapshot }
   [Job 624] Job succeeded: success
   ```

   **Example**
   
   In the following example, all Snapshot copies in an invalid state are deleted for the SVM with Infinite Volume named `vs1`:
3. Confirm that all invalid Snapshot copies have been deleted by using the `volume snapshot show` command with the `-state` parameter.

**Example**

In the following example, no Snapshot copies with a state of invalid are found for the Infinite Volume named repo_vol:

```
cluster1::> volume snapshot show -vserver vs1 -volume repo_vol -state invalid
There are no entries matching your query.
```
Providing disaster recovery on Infinite Volumes using mirroring technology

Stored data is susceptible to disaster, either through hardware failure or environmental catastrophe. You can use mirroring technology on Infinite Volumes to create an identical second set of data to replace the primary set of data, in case something happens to the primary set of data.

You can create a data protection mirror relationship from a source Infinite Volume on one cluster to a destination Infinite Volume on a different cluster to provide asynchronous disaster recovery. Infinite Volumes support bidirectional data exchange between two sites and multiple-mirror fanout deployments.

You cannot create a data protection mirror relationship between two Infinite Volumes on the same cluster, and you cannot create a data protection mirror relationship between a FlexVol volume and an Infinite Volume.

Related concepts

Providing disaster recovery on Infinite Volumes with storage classes using mirroring technology on page 219

Creating a data protection mirror copy for an Infinite Volume with dedicated aggregates

When source and destination Infinite Volumes do not share aggregates with other Infinite Volumes or FlexVol volumes in the same cluster, you can create the same number and size of aggregates for the destination volume as are used by the source volume before you create a data protection mirror relationship.

Before you begin

You must understand the aggregate requirements for a destination Infinite Volume.

About this task

Cluster administrators can perform all data protection tasks, and Storage Virtual Machine (SVM) administrators can perform only the following data protection tasks:

- Creating destination Infinite Volumes
- Creating and initializing data protection mirror relationships

This workflow applies only to an Infinite Volume without storage classes. If you want to create a data protection mirror copy of an Infinite Volume with storage classes, you cannot use the command-line interface; you must use OnCommand Workflow Automation instead.

For more detailed aggregate sizes, you can perform this procedure with the units set to KB.

Steps

1. Identifying the size of a source Infinite Volume and aggregate details on page 105
2. Creating dedicated aggregates for a destination cluster (cluster administrators only) on page 106
3. Creating an SVM with Infinite Volume in a destination cluster on page 107
4. Creating cluster and SVM peer relationships on page 107
5. Creating a destination Infinite Volume on page 108
6. Creating and initializing a data protection mirror relationship on page 109

Related concepts

Aggregate requirements for destination Infinite Volumes on page 22

Related tasks

Creating a data protection mirror copy for an Infinite Volume with storage classes on page 22

Identifying the size of a source Infinite Volume and aggregate details

You must identify the size of the source Infinite Volume to create a destination Infinite Volume of the same size or bigger. You must also identify the number and size of aggregates used by the source volume to create the same number and size of aggregates for the destination volume.

Steps

1. Identify the size of the source Infinite Volume by using the `volume show` command.

   The destination Infinite Volume must be equal to or larger than the size of the source Infinite Volume.

   **Example**

   The following command shows that the source volume repo_vol is 420 TB in size:

   ```
   cluster1::> volume show -volume repo_vol
   Vserver   Volume       Aggregate    State      Type       Size  Available Used%
   --------- ------------ ------------ ---------- ---- ---------- ---------- -----  
   vs0       repo_vol     -            online     RW        420TB      401TB    5%
   ```

2. Identify the number of aggregates the source Infinite Volume uses by using the `volume show` command with the `-is-constituent` parameter.

   The destination Infinite Volume should have the same number of aggregates.

   **Example**

   The following command shows that the source volume repo_vol uses four aggregates: aggr1, aggr2, aggr3, and aggr4:

   ```
   cluster1::> volume show -is-constituent true
   Vserver   Volume       Aggregate    State      Type       Size  Available Used%
   --------- ------------ ------------ ---------- ---- ---------- ---------- -----  
   vs0       repo_vol_1024_data0001 aggr1 online  RW        100TB    95TB    5%    
   vs0       repo_vol_1024_data0002 aggr2 online  RW        100TB    95TB    5%    
   vs0       repo_vol_1024_data0003 aggr3 online  RW        100TB    95TB    5%    
   vs0       repo_vol_1024_data0004 aggr4 online  RW        100TB    95TB    5%    
   vs0       repo_vol_ns      aggr1 online     RW        10TB     9.5TB   5%    
   vs0       repo_vol_ns_mirror0001 aggr2 online  DP         10TB     9.5TB   5%    
   6 entries were displayed.
   ```

3. Identify the size of each aggregate that the source Infinite Volume uses by using the `storage aggregate show` command.

   The destination Infinite Volume should have the same number and size of aggregates as the source Infinite Volume. For more detailed aggregate sizes, you can view the aggregate sizes in KB.

   **Example**

   The following command shows that aggr1, aggr2, aggr3, and aggr4 are each 250 TB in size:

   ```
   cluster1::> storage aggregate show
   Aggregate     Size      Available Used% State   #Vols  Nodes            RAID Status
   --------- -------- --------- ----- ------- ------ ---------------- ------------  
   aggr0       1.46TB  69.40GB  95% online   1     node2            raid_dp, normal
   aggr0_node1_0 1.46TB  69.40GB  95% online   1     node1            raid_dp, normal
   ```
Result

You have the following numbers that you require to create dedicated aggregates for the destination Infinite Volume:

- Size of the source Infinite Volume
- Number of aggregates used by the source Infinite Volume
- Size of each aggregate used by the source Infinite Volume

Creating dedicated aggregates for a destination cluster (cluster administrators only)

You can create the same number of aggregates for the destination Infinite Volume as are used by the source Infinite Volume. The aggregates should be the same size or larger than the aggregates used by the source Infinite Volume.

Before you begin

- You must have cluster administrator privileges to perform this task.
- You must know the number and size of aggregates used by the source Infinite Volume.

About this task

Aggregates are dedicated when only one Storage Virtual Machine (SVM) in a cluster uses the aggregates. You use the aggregate list for the SVM to control which aggregates the SVM uses.

Step

1. On the destination cluster, create the same number of aggregates for the destination Infinite Volume as are used by the source Infinite Volume by using the `storage aggregate create` command.

   The aggregates for the destination cluster should be the same size or larger than the aggregates used for the source Infinite Volume. If you cannot create the same number and size of aggregates, you must ensure that the overall aggregate space for the destination Infinite Volume is large enough to fit a mirror copy of the source Infinite Volume.

After you finish

You must create the SVM with Infinite Volume and modify its aggregate list to specify the aggregates to use. You must also ensure that aggregate lists for other SVMs in the cluster do not reference the dedicated aggregates.

Related information

Disk and aggregate management
Creating an SVM with Infinite Volume in a destination cluster

You must create a Storage Virtual Machine (SVM) with Infinite Volume in the destination cluster and specify an aggregate list for the SVM with Infinite Volume to control how volumes in the cluster use the aggregates.

Before you begin

- You must have cluster administrator privileges to perform this task.
- You must have created aggregates in the destination cluster to specify in the aggregate list for the destination SVM with Infinite Volume.

About this task

Each SVM name in all the clusters should be unique because you can set up SVM peer relationships only between SVMs with unique names. You should use a fully qualified domain name for each SVM.

Steps

1. On the destination cluster, create an SVM with Infinite Volume by using the vserver create command with the -is-repository parameter.

   Example

   In the following example, an SVM with Infinite Volume named vs0_dest is created:

   ```
   cluster2::> vserver create -vserver vs0_dest -rootvolume root_vs0 -aggregate aggr1 -ns-switch file -rootvolume-security-style mixed -is-repository true
   ```

2. Modify the SVM with Infinite Volume to specify the aggregate list by using the vserver add-aggregates command:

   Example

   In the following example, an SVM with Infinite Volume named vs0_dest is modified to specify aggr1, aggr2, aggr3, and vs_aggr in the aggregate list for the SVM:

   ```
   cluster2::> vserver add-aggregates -vserver vs0_dest -aggregates aggr1,aggr2,aggr3,vs_aggr
   ```

   If you want the aggregates to be dedicated to the destination Infinite Volume, you must ensure that aggregate lists for other SVMs in the cluster do not reference the dedicated aggregates.

Result

The SVM with Infinite Volume is created in the cluster, and the aggregate list specifies which aggregates to use.

Creating cluster and SVM peer relationships

You must create a cluster peer relationship between the source and destination clusters and a Storage Virtual Machine (SVM) peer relationship between the source and destination SVMs with Infinite Volume before you can create a data protection mirror relationship.

Before you begin

- You must have cluster administrator privileges to perform this task.
• The SVMs with Infinite Volume must have unique names.

**About this task**
You cannot create an SVM peer relationship between an SVM with FlexVol volume and an SVM with Infinite Volume.

**Steps**
1. Create a cluster peer relationship between the source and destination clusters.
2. Create an SVM peer relationship between the SVM with Infinite Volume in the source cluster and the SVM with Infinite Volume in the destination cluster.

**Related information**

*System administration*

**Creating a destination Infinite Volume**
You must create a destination Infinite Volume that is the same size as or bigger than the source Infinite Volume to create a data protection mirror relationship between the source and destination volumes.

**Before you begin**
• You must know the size of the source Infinite Volume.
• If you want to specify an aggregate for the namespace constituent, you must have advanced privilege.

**Step**
1. On the destination cluster in the Storage Virtual Machine (SVM) with Infinite Volume, create a destination Infinite Volume with a type of DP by using the `volume create` command with the `-type` parameter.

**Example**
In the following example, a destination Infinite Volume named `repo_vol_dest` of 420 TB size is created:

```
cluster2:/> volume create -vserver vs0_dest -volume repo_vol_dest -type dp -size 420TB
```

A destination Infinite Volume is created. However, constituents are not created yet.

**After you finish**
Create and initialize a data protection mirror relationship between the source and destination Infinite Volumes.
Creating and initializing a data protection mirror relationship

You must create and initialize the data protection mirror relationship between the source and destination Infinite Volumes to mirror all constituents from the source volume to the destination volume.

Before you begin

You must create and initialize the data protection mirror relationship between the source and destination Infinite Volumes to mirror all constituents from the source volume to the destination volume.

- A cluster peer relationship must exist between the source and destination clusters.
- A Storage Virtual Machine (SVM) peer relationship must exist between the source and destination SVMs with Infinite Volume.
- A SnapMirror license must be installed on the source and destination clusters.

About this task

You can have cluster administrator or SVM administrator privileges to perform this task. However, if you see an error message about insufficient aggregate space, you require cluster administrator privileges to add or modify aggregates.

The following illustration shows what constituents are mirrored from the source Infinite Volume to the destination Infinite Volume after you initialize the data protection mirror relationship. In this example, SnapDiff is disabled on the source and destination Infinite Volumes. The namespace mirror constituent from the source Infinite Volume is not mirrored to the destination Infinite Volume.

The following illustration shows what constituents are mirrored from the source Infinite Volume to the destination Infinite Volume after you initialize the data protection mirror relationship when SnapDiff is enabled on the source Infinite Volume. The namespace mirror constituents from the source Infinite Volume are not mirrored to the destination Infinite Volume.
Steps

1. On the destination Infinite Volume, create a data protection mirror relationship between the source and destination Infinite Volumes by using the `snapmirror create` command.

   Example

   In the following example, a data protection mirror relationship is created between the destination Infinite Volume named repo_vol_dest and the source Infinite Volume named repo_vol:

   ```
   cluster2::> snapmirror create -destination-path vs0_dest:repo_vol_dest -source-path vs0:repo_vol -type DP
   ```

   A data protection mirror relationship is created between the two Infinite Volumes, but no constituents exist yet for the destination Infinite Volume.

2. On the destination Infinite Volume, initialize the data protection mirror relationship by using the `snapmirror initialize` command.

   The `snapmirror initialize` command does not support the `-policy` parameter for Infinite Volumes.

   Example

   In the following example, a data protection mirror relationship is initialized between the destination Infinite Volume named repo_vol_dest and the source Infinite Volume named repo_vol:

   ```
   cluster2::> snapmirror initialize -destination-path vs0_dest:repo_vol_dest
   ```

   Aggregates are automatically identified and evaluated for available space. If aggregates have enough space, the namespace constituent and all of the data constituents from the source Infinite Volume are mirrored to the destination Infinite Volume. If aggregates lack enough space, an error message is displayed, and the data protection mirror relationship is not initialized.

   **Note:** Namespace mirror constituents are not mirrored from the source to the destination Infinite Volume because they are intracluster data protection mirror copies for the source Infinite Volume.
3. You can view constituents in the destination Infinite Volume by using the `volume show` command with the `-is-constituent` parameter.

In the following example, the destination Infinite Volume named `repo_vol_dest` contains four data constituents (repo_vol_dest_1024_data0001, repo_vol_dest_1024_data0002, repo_vol_dest_1024_data0003, and repo_vol_dest_1024_data0004) and one namespace constituent (repo_vol_dest_ns) after the data protection mirror relationship is initialized:

```
cluster2::> volume show -is-constituent true

<table>
<thead>
<tr>
<th>Vserver</th>
<th>Volume</th>
<th>Aggregate</th>
<th>State</th>
<th>Type</th>
<th>Size</th>
<th>Available</th>
<th>Used%</th>
</tr>
</thead>
<tbody>
<tr>
<td>vs0_dest</td>
<td>repo_vol_dest_1024_data0001</td>
<td>aggr3</td>
<td>online</td>
<td>DP</td>
<td>100TB</td>
<td>95TB</td>
<td>5%</td>
</tr>
<tr>
<td>vs0_dest</td>
<td>repo_vol_dest_1024_data0002</td>
<td>aggr4</td>
<td>online</td>
<td>DP</td>
<td>100TB</td>
<td>95TB</td>
<td>5%</td>
</tr>
<tr>
<td>vs0_dest</td>
<td>repo_vol_dest_1024_data0003</td>
<td>aggr2</td>
<td>online</td>
<td>DP</td>
<td>100TB</td>
<td>95TB</td>
<td>5%</td>
</tr>
<tr>
<td>vs0_dest</td>
<td>repo_vol_dest_1024_data0004</td>
<td>aggr1</td>
<td>online</td>
<td>DP</td>
<td>100TB</td>
<td>95TB</td>
<td>5%</td>
</tr>
<tr>
<td>vs0_dest</td>
<td>repo_vol_dest_ns</td>
<td>aggr3</td>
<td>online</td>
<td>DP</td>
<td>10TB</td>
<td>9.5TB</td>
<td>5%</td>
</tr>
</tbody>
</table>
```

5 entries were displayed.

**Related concepts**

*Error messages and solutions for failed aggregate selection for destination Infinite Volumes* on page 135

**Creating a data protection mirror copy for an Infinite Volume with shared aggregates**

When source and destination Infinite Volumes share aggregates with other Infinite Volumes or FlexVol volumes in the same cluster, you must ensure that enough shared aggregate space is available for the destination Infinite Volume before you create a data protection mirror relationship.

**Before you begin**

You must understand the aggregate requirements for a destination Infinite Volume.

**About this task**

Cluster administrators can perform all data protection tasks, and Storage Virtual Machine (SVM) administrators can perform only the following data protection tasks:

- Creating destination Infinite Volumes
- Creating and initializing data protection mirror relationships

This workflow applies only to an Infinite Volume without storage classes. If you want to create a data protection mirror copy of an Infinite Volume with storage classes, you cannot use the command-line interface; you must use OnCommand Workflow Automation instead.

For more detailed aggregate sizes, you can perform this procedure with the units set to KB.

**Steps**

1. **Identifying the size of a source Infinite Volume and constituent details** on page 112
2. **Verifying aggregate space in a destination cluster (cluster administrators only)** on page 113
3. **Creating an SVM with Infinite Volume in a destination cluster** on page 115
4. **Creating cluster and SVM peer relationships** on page 115
5. **Creating a destination Infinite Volume** on page 116
6. **Creating and initializing a data protection mirror relationship** on page 117
Identifying the size of a source Infinite Volume and constituent details

You must identify the size of the source Infinite Volume to create a destination Infinite Volume that is the same size as or larger than the source volume. You must also identify the size of constituents in the source volume to help identify the aggregate space that is required for the destination volume.

Steps

1. Identify the size of the source Infinite Volume by using the `volume show` command:

   The destination Infinite Volume must be equal to or larger in size than the source Infinite Volume.

   **Example**

   In the following example, the source volume is named `repo_vol`, and it is 420 TB in size:

   ```
   cluster1::> volume show -volume repo_vol
   Vserver   Volume       Aggregate    State      Type       Size  Available Used%
   --------- ------------ ------------ ---------- ---- ---------- ---------- ----- 
   vs0       repo_vol     -            online     RW        420TB      401TB    5%
   ```

2. View the constituents in the source Infinite Volume by using the `volume show` command with the `-is-constituent` parameter.

   **Example**

   In the following example, the volume named `repo_vol` has the following constituents:
   - One namespace constituent named `repo_vol_ns`
   - One namespace mirror constituent named `repo_vol_ns_mirror0001`
   - Four data constituents named `repo_vol_1024_data0001`, `repo_vol_1024_data0002`, `repo_vol_1024_data0003`, and `repo_vol_1024_data0004`

   ```
   cluster1::> volume show -is-constituent true
   Vserver   Volume       Aggregate    State      Type       Size  Available Used%
   --------- ------------ ------------ ---------- ---- ---------- ---------- ----- 
   vs0       repo_vol_1024_data0001 aggr1 online RW        100TB       95TB    5%
   vs0       repo_vol_1024_data0002 aggr2 online RW        100TB       95TB    5%
   vs0       repo_vol_1024_data0003 aggr3 online RW        100TB       95TB    5%
   vs0       repo_vol_1024_data0004 aggr4 online RW        100TB       95TB    5%
   vs0       repo_vol_ns aggr1 online           RW        10TB        9.5TB    5%
   vs0       repo_vol_ns_mirror0001 aggr2 online DP        10TB        9.5TB    5%
   6 entries were displayed.
   ```

3. Identify the aggregate space required for the namespace constituent in the source Infinite Volume.

   **Example**

   In the example, the namespace constituent named `repo_vol_ns` is 10 TB, and you require 10 TB of available space on the destination Infinite Volume for a mirror copy of the namespace constituent.

4. Identify how many data constituents exist in the source Infinite Volume.
Example
In the example, the volume named repo_vol has four data constituents named repo_vol_1024_data0001, repo_vol_1024_data0002, repo_vol_1024_data0003, and repo_vol_1024_data0004.

5. Identify the maximum data constituent size for the source Infinite Volume:
   a. Identify the smallest system for the nodes that the source Infinite Volume spans.
      For example, if the source Infinite Volume spans a node for the 6280 system and a node for the 3270 system, the 3270 system is the smaller system.
   b. Identify either the 64-bit maximum FlexVol volume size or the maximum data constituent size for the smaller system by using the Hardware Universe.

6. Identify the overall aggregate space required for all of the data constituents in the source Infinite Volume by multiplying the number of data constituents by the maximum data constituent size.

Example
For example, if the source Infinite Volume has four data constituents, and the maximum data constituent size is 40 TB, you require 160 TB of available space on the destination Infinite Volume for mirror copies of the data constituents.

Result
You have the following numbers that you require to create shared aggregates for the destination Infinite Volume:

• Size of the source Infinite Volume

• Aggregate space required for a mirror copy of the namespace constituent
   The aggregate space required for a mirror copy of the namespace constituent is based on the default maximum size, which is 10 TB.

• Aggregate space required for mirror copies of all data constituents
   The aggregate space required for mirror copies of the data constituents is the number of data constituents multiplied by the maximum data constituent size for the source Infinite Volume.

Note: Namespace mirror constituents are not mirrored to the destination Infinite Volume. You do not have to identify information about namespace mirror constituents.

Related information
NetApp Hardware Universe

Verifying aggregate space in a destination cluster (cluster administrators only)
When an Infinite Volume shares aggregates with other volumes in the cluster, you must have enough available aggregate space in the destination cluster to create a data protection mirror copy of the source Infinite Volume.

Before you begin
• You must have cluster administrator privileges to perform this task.
• You must know the following information about the source Infinite Volume:
  ◦ Size of the Infinite Volume
Aggregate space required for a mirror copy of the namespace constituent
Aggregate space required for mirror copies of all of the data constituents

Steps
1. On the destination cluster, view the available aggregate space by using the `storage aggregate show-space` command.
   
   The space available in the aggregates must be large enough to fit the following items:
   
   • The size of the source Infinite Volume
   • A mirror copy of the namespace constituent from the source Infinite Volume
   • Mirror copies of all of the data constituents from the source Infinite Volume

   When you initialize the data protection mirror relationship, a mirror copy of the namespace constituent and each data constituent in the source Infinite Volume is transferred to the destination Infinite Volume. The size of each mirror copy reflects the current size of each constituent (not the maximum size of each constituent). For example, if the maximum constituent size for data constituents is 70 TB, but the current size of each data constituent is 40 TB, the size of each mirror copy of a data constituent is 40 TB, not 70 TB.

Example

The following example displays the available space for all the aggregates in cluster2:

```
cluster2::> storage aggregate show
Aggregate  Size Available  Used% State #Vols  Nodes   RAID Status
----------  -----------  ----- ------- ------ ------ ------------
aggr0      1.46TB      69.40GB 95% online 1   node6    raid_dp, normal
aggr0_node5_0  1.46TB 69.40GB 95% online 1   node5    raid_dp, normal
aggr0_node7_0  1.46TB 69.40GB 95% online 1   node7    raid_dp, normal
aggr0_node8_0  1.46TB 69.40GB 95% online 1   node8    raid_dp, normal
aggr1      250TB       140TB 44% online 1   node5    raid_dp, normal
aggr2      250TB       140TB 44% online 1   node6    raid_dp, normal
aggr3      250TB       150TB 40% online 2   node7    raid_dp, normal
aggr4      250TB       150TB 40% online 2   node8    raid_dp, normal
8 entries were displayed.
```

For more detailed aggregate sizes, you can view the aggregate sizes in KB.

2. If more aggregate space is required in the destination cluster, add disks or aggregates to the cluster by using the `storage aggregate create` command.

Result

The available aggregate space in the destination cluster is large enough to fit a mirror copy of the source Infinite Volume.

Related information

*Disk and aggregate management*
Creating an SVM with Infinite Volume in a destination cluster

You must create a Storage Virtual Machine (SVM) with Infinite Volume in the destination cluster and specify an aggregate list for the SVM with Infinite Volume to control how volumes in the cluster use the aggregates.

Before you begin

• You must have cluster administrator privileges to perform this task.

• You must have created aggregates in the destination cluster to specify in the aggregate list for the destination SVM with Infinite Volume.

About this task

Each SVM name in all the clusters should be unique because you can set up SVM peer relationships only between SVMs with unique names. You should use a fully qualified domain name for each SVM.

Steps

1. On the destination cluster, create an SVM with Infinite Volume by using the vserver create command with the -is-repository parameter.

Example

In the following example, an SVM with Infinite Volume named vs0_dest is created:

```
cluster2::> vserver create -vserver vs0_dest -rootvolume root_vs0 -aggregate aggr1 -ns-switch file -rootvolume-security-style mixed -is-repository true
```

2. Modify the SVM with Infinite Volume to specify the aggregate list by using the vserver add-aggregates command:

Example

In the following example, an SVM with Infinite Volume named vs0_dest is modified to specify aggr1, aggr2, aggr3, and vs_aggr in the aggregate list for the SVM:

```
cluster2::> vserver add-aggregates -vserver vs0_dest -aggregates aggr1,aggr2,aggr3,vs_aggr
```

If you want the aggregates to be dedicated to the destination Infinite Volume, you must ensure that aggregate lists for other SVMs in the cluster do not reference the dedicated aggregates.

Result

The SVM with Infinite Volume is created in the cluster, and the aggregate list specifies which aggregates to use.

Creating cluster and SVM peer relationships

You must create a cluster peer relationship between the source and destination clusters and a Storage Virtual Machine (SVM) peer relationship between the source and destination SVMs with Infinite Volume before you can create a data protection mirror relationship.

Before you begin

• You must have cluster administrator privileges to perform this task.
• The SVMs with Infinite Volume must have unique names.

**About this task**
You cannot create an SVM peer relationship between an SVM with FlexVol volume and an SVM with Infinite Volume.

**Steps**
1. Create a cluster peer relationship between the source and destination clusters.
2. Create an SVM peer relationship between the SVM with Infinite Volume in the source cluster and the SVM with Infinite Volume in the destination cluster.

**Related information**

*System administration*

**Creating a destination Infinite Volume**
You must create a destination Infinite Volume that is the same size as or bigger than the source Infinite Volume to create a data protection mirror relationship between the source and destination volumes.

**Before you begin**
- You must know the size of the source Infinite Volume.
- If you want to specify an aggregate for the namespace constituent, you must have advanced privilege.

**Step**
1. On the destination cluster in the Storage Virtual Machine (SVM) with Infinite Volume, create a destination Infinite Volume with a type of DP by using the `volume create` command with the `-type` parameter.

**Example**
In the following example, a destination Infinite Volume named `repo_vol_dest` of 420 TB size is created:

```
cluster2::> volume create -vserver vs0_dest -volume repo_vol_dest -type dp -size 420TB
```

A destination Infinite Volume is created. However, constituents are not created yet.

**After you finish**
Create and initialize a data protection mirror relationship between the source and destination Infinite Volumes.
Creating and initializing a data protection mirror relationship

You must create and initialize the data protection mirror relationship between the source and destination Infinite Volumes to mirror all constituents from the source volume to the destination volume.

Before you begin

You must create and initialize the data protection mirror relationship between the source and destination Infinite Volumes to mirror all constituents from the source volume to the destination volume.

- A cluster peer relationship must exist between the source and destination clusters.
- A Storage Virtual Machine (SVM) peer relationship must exist between the source and destination SVMs with Infinite Volume.
- A SnapMirror license must be installed on the source and destination clusters.

About this task

You can have cluster administrator or SVM administrator privileges to perform this task. However, if you see an error message about insufficient aggregate space, you require cluster administrator privileges to add or modify aggregates.

The following illustration shows what constituents are mirrored from the source Infinite Volume to the destination Infinite Volume after you initialize the data protection mirror relationship. In this example, SnapDiff is disabled on the source and destination Infinite Volumes. The namespace mirror constituent from the source Infinite Volume is not mirrored to the destination Infinite Volume.

The following illustration shows what constituents are mirrored from the source Infinite Volume to the destination Infinite Volume after you initialize the data protection mirror relationship when SnapDiff is enabled on the source Infinite Volume. The namespace mirror constituents from the source Infinite Volume are not mirrored to the destination Infinite Volume.
Steps

1. On the destination Infinite Volume, create a data protection mirror relationship between the source and destination Infinite Volumes by using the `snapmirror create` command.

   **Example**
   
   In the following example, a data protection mirror relationship is created between the destination Infinite Volume named repo_vol_dest and the source Infinite Volume named repo_vol:

   ```
   cluster2::> snapmirror create -destination-path vs0_dest:repo_vol_dest -source-path vs0:repo_vol -type DP
   ```

   A data protection mirror relationship is created between the two Infinite Volumes, but no constituents exist yet for the destination Infinite Volume.

2. On the destination Infinite Volume, initialize the data protection mirror relationship by using the `snapmirror initialize` command.

   The `snapmirror initialize` command does not support the `-policy` parameter for Infinite Volumes.

   **Example**
   
   In the following example, a data protection mirror relationship is initialized between the destination Infinite Volume named repo_vol_dest and the source Infinite Volume named repo_vol:

   ```
   cluster2::> snapmirror initialize -destination-path vs0_dest:repo_vol_dest
   ```

   Aggregates are automatically identified and evaluated for available space. If aggregates have enough space, the namespace constituent and all of the data constituents from the source Infinite Volume are mirrored to the destination Infinite Volume. If aggregates lack enough space, an error message is displayed, and the data protection mirror relationship is not initialized.

   **Note:** Namespace mirror constituents are not mirrored from the source to the destination Infinite Volume because they are intracluster data protection mirror copies for the source Infinite Volume.
3. You can view constituents in the destination Infinite Volume by using the `volume show` command with the `-is-constituent` parameter.

In the following example, the destination Infinite Volume named `repo_vol_dest` contains four data constituents (`repo_vol_dest_1024_data0001`, `repo_vol_dest_1024_data0002`, `repo_vol_dest_1024_data0003`, and `repo_vol_dest_1024_data0004`) and one namespace constituent (`repo_vol_dest_ns`) after the data protection mirror relationship is initialized:

```
cluster2::> volume show -is-constituent true
Vserver   Volume       Aggregate    State      Type       Size  Available Used%
--------- ------------ ------------ ---------- ---- ---------- ---------- ----
vs0_dest  repo_vol_dest_1024_data0001 aggr3 online DP    100TB       95TB    5%
vs0_dest  repo_vol_dest_1024_data0002 aggr4 online DP    100TB       95TB    5%
vs0_dest  repo_vol_dest_1024_data0003 aggr2 online DP    100TB       95TB    5%
vs0_dest  repo_vol_dest_1024_data0004 aggr1 online DP    100TB       95TB    5%
vs0_dest  repo_vol_dest_ns aggr3    online     DP         10TB      9.5TB   5%
5 entries were displayed.
```

Related concepts

- Error messages and solutions for failed aggregate selection for destination Infinite Volumes on page 135

Viewing information about destination Infinite Volumes

After you initialize a data protection mirror relationship for an Infinite Volume, you can view information about the destination Infinite Volume.

Related tasks

- Viewing constituents of an Infinite Volume on page 195

When information is available about a destination Infinite Volume

Some information about a destination Infinite Volume is available only after constituents are created. Constituents exist on a destination Infinite Volume after you initialize a data protection mirror relationship between the source and destination Infinite Volumes.

The following table identifies the information that is displayed by some commands and parameters for a destination Infinite Volume before and after you initialize a data protection mirror relationship by using the `snapmirror initialize` command:

<table>
<thead>
<tr>
<th>Command</th>
<th>Before you initialize a data protection mirror relationship</th>
<th>After you initialize a data protection mirror relationship</th>
</tr>
</thead>
<tbody>
<tr>
<td>volume show command with the <code>-state</code> parameter</td>
<td>Displays a temporary expected value</td>
<td>Displays the current value based on the setting for each constituent in the destination Infinite Volume. The current value can be derived only after constituents are created in the destination Infinite Volume.</td>
</tr>
<tr>
<td>volume show command with the <code>-space-guarantee</code> parameter</td>
<td></td>
<td></td>
</tr>
<tr>
<td>volume show command with the <code>-language</code> parameter (requires advanced privilege)</td>
<td>Displays a dash (-)</td>
<td></td>
</tr>
</tbody>
</table>
Protocols supported on destination Infinite Volumes

You can use the NFSv3 protocol to access the active file system on the destination Infinite Volume. Other protocols are not supported because the destination Infinite Volume does not support locks. However, you can use CIFS and NFS protocols to access Snapshot copies on the destination Infinite Volume.

For CIFS access to Snapshot copies on a destination Infinite Volume, you should create a share at or below /NS/.snapshot.

Related concepts

*Where clients access Infinite Volumes* on page 73

When clients can access the active file system on the destination Infinite Volume

For an Infinite Volume in a data protection mirror relationship, the active file system on the destination Infinite Volume is available to clients after the system transfers the latest coordinated Snapshot copy to the destination Infinite Volume.

The storage system automatically directs clients to use the active file system in the latest coordinated Snapshot copy on the destination Infinite Volume. A coordinated Snapshot copy means that Snapshot copies exist for the namespace constituent and all of the data constituents in the Infinite Volume. The latest coordinated Snapshot copy must be transferred before a consistent view of the active file system is available on the destination Infinite Volume. Because the mirror relationship between Infinite Volumes integrates multiple, separate Snapshot copy transfers, attributes of the file system are incrementally updated as each Snapshot copy transfer is complete.

*Note:* After a read/write Infinite Volume is created and is online, the volume is automatically mounted. However, a read-only destination Infinite Volume is not automatically mounted. You must mount the destination Infinite Volume before clients can access the active file system on the volume.

Managing disaster recovery for Infinite Volumes

When an Infinite Volume is in a data protection mirror relationship, you can use the SnapMirror commands on the Infinite Volume to recover the entire Infinite Volume and its constituents. However, you cannot use the SnapMirror commands on individual constituents in an Infinite Volume.

Related tasks

*Reversing the data protection mirror relationship for Infinite Volumes with storage classes when disaster occurs* on page 222

How the snapmirror resync command works for Infinite Volumes and constituents

You can use the `snapmirror resync` command on an Infinite Volume, and the system recovers the Infinite Volume and its constituents—including the namespace constituent and data constituents—from the destination Infinite Volume. However, you cannot use the `snapmirror resync` command on individual constituents.

*Note:* A data protection mirror relationship between two Infinite Volumes excludes the namespace mirror constituent. The namespace mirror constituent is an intracluster data protection mirror copy of the namespace constituent in an Infinite Volume. The data protection mirror relationship for the
namespace constituent is in addition to and separate from the data protection mirror relationship for the Infinite Volume.

You can use the `snapmirror resync` command on the Infinite Volume to recover the entire volume and its constituents. When you use the `snapmirror resync` command, the system uses the current Infinite Volume configuration to determine whether a constituent is newly added, deleted, or failed. The system creates and initializes data protection mirror relationships for newly added constituents during the resynchronization. Deleted constituents are not resynchronized. Failed constituents are not recovered, and you must contact technical support to recover failed constituents.

**Related concepts**

*How some Snapshot and SnapMirror commands run on Infinite Volumes* on page 98

**Reversing the data protection mirror relationship for Infinite Volumes when disaster occurs**

If a disaster disables the source Infinite Volume of a data protection mirror relationship, you can use the destination Infinite Volume to serve data while you repair or replace the source Infinite Volume, and reestablish the original configuration of the systems.

**About this task**

You can have cluster administrator or Storage Virtual Machine (SVM) administrator privileges to perform this task.

In the following example, a data protection mirror relationship exists between a source Infinite Volume on one cluster and a destination Infinite Volume on another cluster. The clusters are in a peer relationship, and the SVMs are in a peer relationship. The original source volume (the one disabled by the disaster) is `cluster1://vs1/volA`, and the original destination volume is `cluster2://vs2/volB`.

In this example, data constituent 3 is offline. All data from the last scheduled data protection mirror update before the source was disabled is preserved, and all the data written to `cluster2://vs2/volB` after you made the volume writable is preserved. Any data written to `cluster1://vs1/volA` is lost.
between the last data protection mirror copy update and the time that `cluster1://vs1/volA` was
disabled is not preserved.

For Infinite Volumes, the `snapmirror initialize` and the `snapmirror resync` commands do not support the `-policy` parameter.

**Steps**

1. Temporarily make the original source volume a read-only destination volume and reverse the data protection mirror relationship to continue to serve data.

   If the source volume (`cluster1://vs1/volA`) is recoverable and its data is intact, complete the following steps:

   a. After the source volume (in this case `cluster1://vs1/volA`) is disabled, use the `snapmirror break` command on the destination volume (`cluster2://vs2/volB`) to make the destination volume writeable.

   **Example**

   ```bash
   cluster2::> snapmirror break vs2:volB
   ```

   The data protection mirror relationship is broken. Volume B changes from read-only to read/write.

   b. Redirect the clients of volume A (`cluster1://vs1/volA`) to the new source volume B (`cluster2://vs2/volB`).

   The former clients of volume A (`cluster1://vs1/volA`) access and write to volume B (`cluster2://vs2/volB`).

   c. Ensure that volume B has a namespace mirror constituent to provide data protection for its namespace constituent.  

      *Creating a namespace mirror constituent on a destination Infinite Volume* on page 125

   d. On the destination volume (`cluster2://vs2/volB`), use the `snapmirror delete` command to remove the data protection mirror relationship between the source volume (`cluster1://vs1/volA`) and the destination volume (`cluster2://vs2/volB`).

   **Example**

   ```bash
   cluster2::> snapmirror delete vs2:volB
   ```

   e. On the source volume (`cluster1://vs1/volA`), use the `snapmirror release` command to remove relationship information from the source.

   **Example**

   ```bash
   cluster1::> snapmirror release vs2:volB
   Warning: Snapshot copies on source volume "vs1:volA" generated by SnapMirror for the purpose of mirroring to destination volume "vs2:volB" will be deleted. Once these Snapshot copies are deleted, it will likely not be possible to re-establish a mirroring relationship between these two volumes.
   Do you want to continue? [y|n]: y
   ```

   f. On the new destination volume (`cluster1://vs1/volA`), use the `snapmirror create` command to create the mirror relationship, but with volume B (`cluster2://vs2/volB`) as the new source and volume A (`cluster1://vs1/volA`) as the new destination.
Example

```
cluster1::> snapmirror create vs2:volB vs1:volA -type DP
```

g. On the new destination volume (cluster1://vs1/volA), use the `snapmirror resync` command to resynchronize volume A (cluster1://vs1/volA) with volume B (cluster2://vs2/volB).

**Note:** If the resynchronization process fails, you must create and resynchronize the data protection mirror relationship again.

Example

```
cluster1::> snapmirror resync vs1:volA
```

Warning: All data newer than Snapshot copy snapmirror.62dd0c95-efcc-11e3-b718-005056975d3f_5_1.2014-06-10_1724 on volume vs1:volA will be deleted.
Do you want to continue? {y|n}: y

If the source volume (cluster1://vs1/volA) is unrecoverable, complete the following steps:

a. After the source volume (in this case, cluster1://vs1/volA) is disabled, use the `snapmirror break` command on the destination volume (cluster2://vs2/volB) to make the destination volume writeable.

Example

```
cluster2::> snapmirror break vs2:volB
```

b. Redirect the clients of volume A (cluster1://vs1/volA) to the new source volume B (cluster2://vs2/volB).

The former clients of volume A (cluster1://vs1/volA) access and write to volume B (cluster2://vs2/volB).

c. On the destination volume (cluster2://vs2/volB), use the `snapmirror delete` command to remove the data protection mirror relationship between volume A (cluster1://vs1/volA) and volume B (cluster2://vs2/volB).

Example

```
cluster2::> snapmirror delete vs2:volB
```

d. On the source volume (cluster1://vs1/volA), use the `snapmirror release` command to remove relationship information from the source volume.

Even though volume A is unrecoverable, the data protection mirror relationship still exists and must be removed.

Example

```
cluster1::> snapmirror release vs2:volB
```

Warning: Snapshot copies on source volume "vs1:volA" generated by SnapMirror for the purpose of mirroring to destination volume "vs2:volB" will be deleted. Once these Snapshot copies are deleted, it will likely not be possible to re-establish a mirroring relationship between these two volumes.
Do you want to continue? {y|n}: y
e. Delete the unrecoverable volume A (cluster1://vs1/volA).

Deleting an Infinite Volume on page 63

f. Create a new data protection destination Infinite Volume named cluster1://vs1/volA by using the volume create command.

Note: Remember to use the -type DP parameter when creating the destination volume, and the destination Infinite Volume must be the same size or larger than the source Infinite Volume.

Example

```
cluster1::> volume create -vserver vs1 -volume volA -type DP -size 200TB
```

g. On the new destination volume (cluster1://vs1/volA), use the snapmirror create command to create the data protection mirror relationship with volume B (cluster2://vs2/volB) as the new source volume and volume A (cluster1://vs1/volA) as the new destination volume.

Example

```
cluster1::> snapmirror create vs1:volA vs2:volB -type DP
```

h. On the new destination volume (cluster1://vs1/volA), use the snapmirror initialize command to create the baseline on the data protection mirror copy. This command also makes volume A (cluster1://vs1/volA) a read-only destination.

Example

```
cluster1::> snapmirror initialize vs1:volA
```

Volume B operates as the read/write source volume, and volume A operates as the read-only destination volume. You can keep this configuration, or you can return the volumes to their original data protection configuration.

2. If you want to return the volumes to their original data protection configuration, perform the following steps:

a. On the new destination volume (cluster1://vs1/volA), update the new destination volume to transfer the latest data from the new source volume (cluster2://vs2/volB) by using the snapmirror update command.

Example

```
cluster1::> snapmirror update vs1:volA
```

b. On the new destination volume (cluster1://vs1/volA), use the snapmirror break command to make volume A (cluster1://vs1/volA) writeable.

Example

```
cluster1::> snapmirror break vs1:volA
```

c. On the new destination volume (cluster1://vs1/volA), use the snapmirror delete command to remove the data protection mirror relationship between volume B (cluster2://vs2/volB) and volume A (cluster1://vs1/volA).
Example

```
cluster1::> snapmirror delete vs1:volA
```

d. On the new source volume (cluster2://vs2/volB), use the **snapmirror release** command to remove the data protection mirror relationship between volume B (cluster2://vs2/volB) and volume A (cluster1://vs1/volA).

```
Example

cluster2::> snapmirror release vs1:volA
```

Warning: Snapshot copies on source volume "vs2:volB" generated by SnapMirror for the purpose of mirroring to destination volume "vs1:volA" will be deleted. Once these Snapshot copies are deleted, it will likely not be possible to re-establish a mirroring relationship between these two volumes.

Do you want to continue? {y|n}: y

e. On the original destination volume (cluster2://vs2/volB), use the **snapmirror create** command to re-create the original data protection mirror relationship with volume A (cluster1://vs1/volA) as the source and volume B (cluster2://vs2/volB) as the destination.

```
Example

cluster2::> snapmirror create vs2:volB vs1:volA -type DP
```

f. On the original destination volume (cluster2://vs2/volB), use the **snapmirror resync** command to resynchronize the original source and original destination volumes.

```
Example

cluster2::> snapmirror resync vs2:volB
```

g. Redirect the clients from volume B (cluster2://vs2/volB) back to their original source volume A (cluster1://vs1/volA).

Related concepts

- What a namespace mirror constituent is on page 14
- Aggregate requirements for destination Infinite Volumes on page 22

Related tasks

- Deleting an Infinite Volume on page 63
- Creating a namespace mirror constituent on a destination Infinite Volume on page 125

Creating a namespace mirror constituent on a destination Infinite Volume

After you break a data protection mirror relationship for two Infinite Volumes, the destination Infinite Volume changes from read-only to read/write. You should create a namespace mirror constituent on the Infinite Volume to provide data protection for the namespace constituent.

Before you begin

The Infinite Volume must not contain storage classes. If you want to create a namespace mirror constituent for an Infinite Volume with storage classes, you cannot use the command-line interface; you must use OnCommand Workflow Automation instead.
About this task

You do not have to create a namespace mirror constituent when SnapDiff is enabled on the destination Infinite Volume because one or more namespace mirror constituents already exist on the destination volume. However, if SnapDiff is disabled on the destination Infinite Volume, no namespace mirror constituents exist on the destination volume, and you should create a namespace mirror constituent.

Steps

1. Identify the size of the namespace constituent in the Infinite Volume by using the `volume show` command with the `-is-constituent` parameter.

Example

In the following example, the namespace constituent named volB_ns is 10 TB and no namespace mirror constituent exists:

```
cluster2::> volume show -is-constituent true

Vserver Volume Aggregate State Type Size Available Used%
------- ------------ --------- ------ ---- ----- --------- ----- 
vs2     volB_1024_data0001 aggr1     online RW   100TB     95TB    5% 
vs2     volB_1024_data0002 aggr2     online RW   100TB     95TB    5% 
vs2     volB_1024_data0003 aggr3     online RW   100TB     95TB    5% 
vs2     volB_1024_data0004 aggr4     online RW   100TB     95TB    5% 
vs2     volB_ns            aggr1     online RW    10TB    9.5TB    5% 
5 entries were displayed.
```

2. Set the Is File System Size Fixed parameter to False for the Infinite Volume by using the `volume modify` command with the `-filesys-size-fixed` parameter.

When the Is File System Size Fixed parameter is set to True, you cannot change the size of the Infinite Volume.

Example

In the following example, the Is File System Size Fixed parameter is set to `false` for the volume named volB:

```
cluster2::> volume modify -vserver vs2 -volume volB -filesys-size-fixed false
```

3. Increase the size of the Infinite Volume by using the `volume modify` command.

At a minimum, you must increase the size of the volume by the size of the namespace constituent. The namespace mirror constituent is automatically created when you increase the size of the Infinite Volume.

Example

In the following example, the size of the volume named volB is increased by the size of the namespace constituent, which is 10 TB:

```
cluster2::> volume modify -vserver vs2 -volume volB -size +10TB

Warning: This operation will expand the Infinite Volume "volB" and potentially add constituents. Since this Infinite Volume is in a SnapMirror relationship the next SnapMirror update will fail unless the SnapMirror destination Infinite Volume is able to provision similar constituents.

Do you want to continue? {y|n}: y
```

A namespace mirror constituent is created for the Infinite Volume.
Recovering individual failed constituents

If a constituent in an Infinite Volume fails and the Infinite Volume is in a data protection mirror relationship, you can use the SnapMirror commands to recover the entire Infinite Volume. You cannot use the SnapMirror commands to recover individual constituents. You must contact technical support to recover only constituents.

About this task

Following are examples of when a constituent in an Infinite Volume can fail:

- An aggregate or shelf used by an Infinite Volume fails.
- The active file system in an Infinite Volume becomes corrupt, and clients cannot read data.

For example, if the namespace constituent fails and you want to recover only the namespace constituent, you must contact technical support.

Step

1. Contact technical support to recover individual, failed constituents.

Related concepts

Methods to protect the namespace constituent of an Infinite Volume on page 36

How namespace mirror constituents affect source and destination Infinite Volumes

Namespace mirror constituents are automatically created for different purposes on the source Infinite Volume and the destination Infinite Volume, and the namespace mirror constituents use aggregate space. Understanding when namespace mirror constituents are automatically created on Infinite Volumes helps you better understand aggregate usage.

Note: The namespace mirror constituents work the same way for an Infinite Volume with or without storage classes.

Related concepts

What a namespace mirror constituent is on page 14
When namespace mirror constituents are created on page 142
How modifying an Infinite Volume affects namespace mirror constituents on page 145
How namespace mirror constituents affect the state of an Infinite Volume on page 146

How Data ONTAP calculates the Infinite Volume size before the transfer of a mirror copy

A mirror copy of a source Infinite Volume is only transferred to the destination Infinite Volume when the mirrored volume size of the destination volume is equal to or bigger than the mirrored volume size of the source volume. How the mirrored volume size is calculated differs between the source and destination volumes.

You must understand the difference between the specified volume size and the mirrored volume size.
The specified volume size is identified when you create an Infinite Volume. The specified size of the destination Infinite Volume must be equal to or bigger than the source Infinite Volume to create and initialize a data protection mirror relationship.

The mirrored volume size is calculated when the system determines whether the destination Infinite Volume has enough space for a mirror copy of the source volume. The mirrored volume size is calculated as follows for the source and destination Infinite Volumes:

- For the source Infinite Volume, the mirrored volume size = (current size of the namespace constituent) + (current sizes of all data constituents).
  For example, a source Infinite Volume contains a 10 TB namespace constituent, a 10 TB namespace mirror constituent, and three 40 TB data constituents. The mirrored volume size of the Infinite Volume is calculated as 130 TB, which is the sum of the namespace constituent and three data constituents.

- For the destination Infinite Volume, the mirrored volume size = (specified volume size) - (the sum of all the namespace mirror constituents sizes on the destination Infinite Volume).
  For example, a destination Infinite Volume has a specified size of 150 TB and contains a 10 TB namespace constituent, two 10 TB namespace mirror constituents (for SnapDiff and tape backup), and three 40 TB data constituents. The mirrored size of the destination Infinite Volume is calculated as 130 TB, which is the size of the destination Infinite Volume (150 TB) minus the size of all the namespace mirror constituents (20 TB).

Note: The size of the namespace mirror constituents is excluded from the mirrored volume size of the source and destination Infinite Volumes because only the namespace constituent and the data constituents are mirrored from the source Infinite Volume to the destination Infinite Volume.

A mirror copy transfer can fail when the specified size of a destination Infinite Volume is bigger than the source Infinite Volume, but namespace mirror constituents on the destination Infinite Volume consume so much space that the destination Infinite Volume lacks enough space for a mirror copy of the source Infinite Volume. This can happen when you enable SnapDiff on the destination Infinite Volume, and the namespace mirror constituents created for SnapDiff consume so much space that a mirror copy of the source Infinite Volume cannot fit on the destination Infinite Volume.
How Data ONTAP selects aggregates for data protection of Infinite Volumes

Aggregates are automatically selected for a destination Infinite Volume when you initialize a data protection mirror relationship. You must understand the factors that affect aggregate selection and how aggregates are selected to correctly create constituents for a destination Infinite Volume.

Related concepts
- Processes used to select aggregates for destination Infinite Volumes with and without storage classes on page 226
- How data protection mirror updates affect aggregate selection on page 169

When Data ONTAP selects aggregates for destination Infinite Volumes

Data ONTAP automatically selects aggregates for a destination Infinite Volume when you initialize a data protection mirror relationship, and not when you create a destination Infinite Volume. When you create a destination Infinite Volume, no aggregates are selected for the destination Infinite Volume because no constituents exist yet.

A pool of available aggregates is identified for the destination Infinite Volume and evaluated for available space. If aggregates have enough space, constituents are mirrored from the source Infinite Volume to aggregates on the destination Infinite Volume, and the data protection mirror relationship is initialized. If aggregates lack enough space, an error message is displayed, and the data protection mirror relationship is not initialized.

Related concepts
- Error messages and solutions for failed aggregate selection for destination Infinite Volumes on page 135

How Data ONTAP automatically selects aggregates for destination Infinite Volumes

Data ONTAP automatically selects aggregates for destination Infinite Volumes. Understanding how aggregates are automatically selected helps you better plan and create aggregates for a destination Infinite Volume.

Methods to control aggregate selection for destination Infinite Volumes

You can use the aggregate list for a destination Storage Virtual Machine (SVM) with Infinite Volume to specify which aggregates in the destination cluster are considered available for automatic selection for a destination Infinite Volume.

You can use the following methods to affect aggregate selection for a destination Infinite Volume:
### How aggregates are selected for constituents in a destination Infinite Volume

When you initialize a data protection mirror relationship for a destination Infinite Volume, Data ONTAP uses the following process to choose aggregates for the namespace constituent and all of the data constituents from the source Infinite Volume that will be mirrored to the destination Infinite Volume:

1. Identifies the pool of aggregates available to the destination Storage Virtual Machine (SVM) with Infinite Volume:
   - If you specified an aggregate for the namespace constituent when you created the destination Infinite Volume, aggregates are divided into two temporary, internal lists: the aggregate specified for the namespace constituent and the aggregates available for all data constituents. The aggregates available for the data constituents are from the aggregate list for the destination SVM with Infinite Volume.
   - If you did not specify an aggregate for the namespace constituent, one temporary, internal list of available aggregates is created from the aggregate list for the destination SVM with Infinite Volume.

2. Chooses an aggregate for the namespace constituent
   - If you specified an aggregate for the namespace constituent, Data ONTAP uses it. Otherwise, Data ONTAP chooses the best aggregate for the namespace constituent. If the aggregate is too small, the initialization operation fails and you receive an error message that informs you how to adjust aggregate space before using the `snapmirror initialize` command again.

3. Chooses aggregates for all of the data constituents
   - The best available aggregates are used for the bigger data constituents. Data ONTAP can place more than one data constituent on the same aggregate, if the aggregate has enough available space.
   - The result of this step determines what happens next:
• If all data constituents can fit on the chosen aggregates, constituents from the source Infinite Volume are mirrored to the chosen aggregates, and the data protection mirror relationship is initialized.

• If all data constituents cannot fit on the chosen aggregates, Data ONTAP returns to step 2. The aggregate that was previously considered for the namespace constituent is returned to the pool of available aggregates. Data ONTAP considers the second best aggregate for the namespace constituent and tries to fit the data constituents on the remaining available aggregates. Data ONTAP continues to try and fit the namespace constituent and all data constituents on the available aggregates until all constituents fit, or until Data ONTAP determines that the constituents cannot fit.

  **Note:** If you specified an aggregate for the namespace constituent and the aggregate is too small, the initialization operation fails because Data ONTAP cannot use any other aggregates for the namespace constituent.

**Related concepts**

*How the best available aggregate is identified* on page 131

*Error messages and solutions for failed aggregate selection for destination Infinite Volumes* on page 135

**How space guarantee for a destination Infinite Volume affects aggregate selection**

The space guarantee for a destination Infinite Volume affects which aggregates are selected for its constituents.

When you create a destination Infinite Volume, you can specify the space guarantee for the Infinite Volume by using the `volume create` command with the `-space-guarantee` parameter as follows:

- When you set `-space-guarantee` to `volume`, the Infinite Volume is thick-provisioned.
- When you set `-space-guarantee` to `none`, the Infinite Volume is thin-provisioned.

The space guarantee for the Infinite Volume applies to all of its data constituents, but not to its namespace constituent. The namespace constituent is always thick-provisioned.

The following space requirements apply to thick-provisioned and thin-provisioned data constituents:

- When data constituents are thick-provisioned, the space available on the aggregate must be equal to or greater than the size of the data constituents.
  You can view the size of a data constituent by using the `volume show` command with the `-is-constituent` and `-size` parameters.

- When data constituents are thin-provisioned, the space available on the aggregate must be equal to or greater than the current used size of the data constituents.
  You can view the size of the data constituent by using the `volume show` command with the `-is-constituent` and `-used` parameters.

**How the best available aggregate is identified**

Whether a constituent is thin-provisioned or thick-provisioned for a destination Infinite Volume affects which aggregates are considered the best choice and selected for its constituents.

For thin-provisioned constituents in a destination Infinite Volume, the best aggregate has the least overcommitment ratio. If two or more aggregates have the same overcommitment ratio, the aggregate with the larger available space is the better aggregate for the constituent.

For thick-provisioned constituents in a destination Infinite Volume, the best aggregate has the largest available size. If two or more aggregates have the same large available size, the larger aggregate is considered better.
How multiple constituents are placed on one aggregate

Data ONTAP uses a calculation to verify the space required to place more than one data constituent on the same aggregate for a destination Infinite Volume.

An aggregate for a destination Infinite Volume has enough space for multiple data constituents when the available space on the aggregate minus $n$ multiplied by the maximum data constituent size is greater than zero. Where $n$ is the number of existing data constituents on the aggregate. The aggregate that contains a namespace constituent can also contain one or more data constituents.

Restrictions on automatic aggregate selection

Shared aggregate space affects automatic aggregate selection for destination Infinite Volumes. You should keep this restriction in mind when creating aggregates for destination Infinite Volumes.

Multiple volumes in a destination cluster can share aggregate space. When you create a destination Infinite Volume in a cluster, the aggregate selection process selects aggregates that allow constituents to grow to the maximum size. However, until the constituents grow to their maximum size and consume space in the aggregates, the space appears available for other volumes to use. As a result, aggregate space that was expected to be available can be consumed by volumes in the cluster, leaving the destination Infinite Volume without enough aggregate space for its constituents to grow.

Examples of automatic aggregate selection for destination Infinite Volumes

The examples illustrate how Data ONTAP selects aggregates on the destination Infinite Volume for mirror copies of constituents from the source Infinite Volume, and initializes a data protection mirror relationship.

<table>
<thead>
<tr>
<th>Typical constituent layout for a destination Infinite Volume</th>
</tr>
</thead>
<tbody>
<tr>
<td>This example represents a typical constituent layout on a destination Infinite Volume. In this example, constituents from the source Infinite Volume are mirrored to aggregates for the destination Infinite Volume, and the data protection mirror relationship is initialized.</td>
</tr>
<tr>
<td>The source Infinite Volume contains the following constituents:</td>
</tr>
<tr>
<td>• Namespace constituent of 10 TB</td>
</tr>
<tr>
<td>• Data constituent 1 of 35 TB</td>
</tr>
<tr>
<td>• Data constituent 2 of 20 TB</td>
</tr>
<tr>
<td>The aggregate list for the destination Storage Virtual Machine (SVM) with Infinite Volume contains the following aggregates:</td>
</tr>
<tr>
<td>• Aggregate 1 has 50 TB of available aggregate space.</td>
</tr>
<tr>
<td>• Aggregate 2 has 25 TB of available aggregate space.</td>
</tr>
<tr>
<td>Constituents from the source Infinite Volume are mirrored to aggregates for the destination Infinite Volume as follows:</td>
</tr>
<tr>
<td>• Namespace constituent is created on aggregate 1.</td>
</tr>
<tr>
<td>• Data constituent 1 is created on aggregate 1.</td>
</tr>
<tr>
<td>• Data constituent 2 is created on aggregate 2.</td>
</tr>
<tr>
<td>The following diagram illustrates the placement of constituents on aggregates for the destination Infinite Volume:</td>
</tr>
</tbody>
</table>
Each constituent for a destination Infinite Volume is placed on a different aggregate

In this example, constituents from the source Infinite Volume are mirrored to aggregates for the destination Infinite Volume, and the data protection mirror relationship is initialized.

The source Infinite Volume contains the following constituents:

- Namespace constituent of 5 TB
- Data constituent 1 of 30 TB
- Data constituent 2 of 20 TB

The aggregate list for the destination SVM with Infinite Volume contains the following aggregates, and no aggregate is specified for the namespace constituent:

- Aggregate 1 has 10 TB of available aggregate space.
- Aggregate 2 has 20 TB of available aggregate space.
- Aggregate 3 has 30 TB of available aggregate space.

The constituents from the source Infinite Volume are mirrored to aggregates for the destination Infinite Volume as follows:

- Aggregate 1 (10 TB) contains the namespace constituent (5 TB).
- Aggregate 2 (20 TB) contains data constituent 2 (20 TB).
- Aggregate 3 (30 TB) contains data constituent 1 (30 TB).

The following diagram illustrates the placement of constituents on aggregates for the destination Infinite Volume:
Multiple constituents for a destination Infinite Volume are placed on one aggregate

In this example, constituents from the source Infinite Volume are mirrored to aggregates for the destination Infinite Volume, and the data protection mirror relationship is initialized. Multiple data constituents are placed on the same aggregate in the destination Infinite Volume.

The source Infinite Volume contains the following constituents:

- Namespace constituent of 10 TB
- Data constituent 1 of 20 TB
- Data constituent 2 of 20 TB

For the destination Infinite Volume, the maximum data constituent size is 20 TB. The aggregate list for the destination SVM with Infinite Volume contains 50 TB of available aggregate space on one aggregate, and no aggregate is specified for the namespace constituent. All of the constituents from the source Infinite Volume are mirrored to the single aggregate for the destination Infinite Volume.

The following diagram illustrates the placement of constituents on aggregates for the destination Infinite Volume:
Constituents for a destination Infinite Volume are too big for the available aggregate space

In this example, constituents from the source Infinite Volume are not mirrored to aggregates for the destination Infinite Volume, and the data protection mirror relationship is not initialized. Although aggregates for the destination Infinite Volume contain enough overall available space, you have specified an aggregate for the namespace constituent, and the remaining aggregates lack enough space for the data constituents.

The source Infinite Volume contains the following constituents:

- Namespace constituent of 10 TB
- Data constituent 1 of 35 TB
- Data constituent 2 of 20 TB

The aggregate list for the destination SVM with Infinite Volume contains 70 TB of available aggregate space on the following aggregates, and aggregate 1 (40 TB) is specified for the namespace constituent:

- Aggregate 1 has 40 TB of available aggregate space.
- Aggregate 2 has 10 TB of available aggregate space.
- Aggregate 3 has 20 TB of available aggregate space.

The constituents from the source Infinite Volume are not mirrored to aggregates for the destination Infinite Volume. Because you specified aggregate 1 (40 TB) for the namespace constituent in the destination Infinite Volume, the remaining aggregates are too small for the two data constituents. As a result, you cannot initialize the data protection mirror relationship.

Error messages and solutions for failed aggregate selection for destination Infinite Volumes

When you initialize a data protection mirror relationship for a destination Infinite Volume, aggregates are automatically selected for constituents. If aggregates lack enough space, an error message is displayed that identifies the problem and provides a solution. Cluster administrators can adjust aggregate space to correct the problem.

Insufficient space on destination for SnapMirror source Infinite Volume namespace constituent

Message

Error: command failed: [Job number] Job failed: Insufficient space on destination for SnapMirror source Infinite Volume namespace constituent "Vserver name:namespace constituent name" (Destination aggregate "aggregate name" needs numberunit more to store snapmirror source Infinite Volume constituent "Vserver name:namespace constituent name").

Cause

The aggregate is too small for the namespace constituent.
**Corrective action**

Add more disks to the existing aggregate to create more space for the namespace constituent.

**Size exceeds maximum (namespace constituent)**

**Message**

```
Error: command failed: [Job number] Job failed: Size exceeds maximum. The Infinite Volume namespace constituent "Vserver name:namespace constituent name" on SnapMirror source is larger than the maximum allowed size for the namespace constituent as determined by the configuration of the destination Infinite Volume. Source namespace constituent size: numberunit. Maximum allowed: numberunit.
```

**Cause**

The platform for the destination Infinite Volume is smaller than the platform for the source Infinite Volume. The two platforms support different maximum constituent sizes.

The maximum constituent size for an Infinite Volume corresponds to the maximum FlexVol volume size for a platform. Because the maximum constituent size for the destination Infinite Volume is smaller than the maximum constituent size for the source Infinite Volume, the namespace constituent from the source Infinite Volume is too big for the destination Infinite Volume. The namespace constituent cannot fit into the aggregates for the destination Infinite Volume.

You would only encounter this error if you set the `-max-namespace-constituent-size` parameter to greater than 10 TB.

**Corrective action**

Choose one of the following options:

- Choose a different cluster with a bigger platform for the destination Infinite Volume.
- Contact technical support for help with setting up the destination Infinite Volume.

**Related concepts**

*Constituent size considerations for data protection mirror relationships for Infinite Volumes* on page 32

**Insufficient space for the following source data constituents on destination Vserver**

**Message**

```
Error: command failed: [Job number] Job failed: Insufficient space for the following source data constituents on destination Vserver "Vserver name": data constituent name(sizeunit). Add aggregates of sufficient size for each constituent, or increase the size of existing aggregates for the Vserver.
```

**Cause**

The Storage Virtual Machine (SVM) with Infinite Volume in the destination cluster specifies too few aggregates to mirror all of the data constituents from the source Infinite Volume to the destination Infinite Volume.
Corrective action

Choose one of the following options:

- Add more disks to the existing aggregate to create enough space to fit two or more data constituents on one aggregate.
- Add more aggregates to the destination cluster for the SVM with Infinite Volume. Add an aggregate for each data constituent in the source Infinite Volume to allow each data constituent to be mirrored to its own aggregate on the destination Infinite Volume.

Insufficient space on SnapMirror destination for source Infinite Volume data constituents

Message

Error: command failed: [Job number] Job failed: Insufficient space on SnapMirror destination for source Infinite Volume data constituent "Vserver name: data constituent name". Destination aggregate "aggregate name" needs sizeunit more to store snapmirror source Infinite Volume constituent "Vserver name: data constituent name".

Cause

The aggregates being considered for data constituents have some space, but not enough space.

Corrective action

Add more disks to the existing aggregates to create more space.

Size exceeds maximum (data constituents)

Message

Error: command failed: [Job number] Job failed: Size exceeds maximum. The Infinite Volume data constituent "Vserver name: data constituent name" on SnapMirror source is larger than the maximum allowed size for a single data constituent as determined by the configuration of the destination Infinite Volume. Source data constituent size: numberunit. Maximum allowed: numberunit.

Cause

The platform for the destination Infinite Volume is smaller than the platform for the source Infinite Volume. The two platforms support different maximum constituent sizes.

The maximum constituent size for an Infinite Volume corresponds to the maximum FlexVol volume size for a platform. Because the maximum constituent size for the destination Infinite Volume is smaller than the maximum constituent size for the source Infinite Volume, the data constituents from the source Infinite Volume are too big for the destination Infinite Volume. The data constituents cannot fit into the aggregates for the destination Infinite Volume.

Corrective action

Choose one of the following options:

- Choose a different cluster with a bigger platform for the destination Infinite Volume.
- Contact technical support for help with setting up the destination Infinite Volume.
Related concepts

Constituent size considerations for data protection mirror relationships for Infinite Volumes on page 32
Providing tape backup and restore of Infinite Volumes

You can incrementally back up data in Infinite Volumes to tape and restore data from tape by using a SnapDiff-supported backup application that supports backup and restore over a volume mounted with the NFS or CIFS protocols. NDMP is not supported for Infinite Volumes.

The cluster administrator can perform all of the tape backup tasks. The Storage Virtual Machine (SVM) administrator can perform only the following tape backup tasks:

- Enabling SnapDiff on an Infinite Volume
- Selecting a Snapshot copy of an Infinite Volume for SnapDiff to use
- Creating a baseline for incremental tape backup
- Backing up only changed files to tape
- Restoring files to disk
- Creating SnapDiff resources after expanding a destination Infinite Volume onto new nodes
- Disabling SnapDiff on an Infinite Volume

Related concepts

Providing tape backup and restore of Infinite Volumes with storage classes on page 227

Tape backup and restore workflow for Infinite Volumes

You can perform incremental tape backup and restore operations for an Infinite Volume mounted with the NFS or CIFS protocol by using a SnapDiff-supported backup application. The workflow provides an overview of tasks required for incremental tape backup and restore operations.

Note: Infinite Volumes do not support NDMP.

You can incrementally back up the data in Infinite Volumes to tape and restore the data from tape using any backup application (also called Data Management Applications or DMAs) that meets the following requirements:

- Supports SnapDiff
- Backs up and restores data over a volume mounted with the NFS or CIFS protocol

Note: The SnapDiff API must be configured for the Infinite Volume.

The high-level tasks that are required to set up and use incremental tape backup for Infinite Volumes are as follows:

1. Set up a tape library configuration to meet the requirements specified by the backup application for a volume mounted with the NFS or CIFS protocol.
   For more information about tape library requirements for the backup application, see your backup application documentation.

2. Configure the backup application to back up the mounted Infinite Volume.

3. Set up incremental tape backup of Infinite Volumes:
   a. Verify space requirements in the Infinite Volume for SnapDiff resources.
b. Enable SnapDiff on the Infinite Volume.
c. Select a Snapshot copy of the Infinite Volume for SnapDiff to use.
d. Configure SnapDiff to use the Snapshot copy.
e. Back up all data in the Infinite Volume to create a baseline for SnapDiff to use in the future.
   For more information about using the backup application to perform backup operations, see
   your backup application documentation.

4. Back up only changed files in the Infinite Volume for each subsequent tape backup:
   a. Use a backup application to request from SnapDiff a list of changed files in the Infinite
      Volume since the last tape backup.
   b. Use a backup application to back up the changed files to tape.

5. Restore files from tape by using the backup application.
   For more information about using the backup application to perform restore operations, see
   your backup application documentation.

Protocols for tape backup and restore of Infinite Volumes

You can use some NFS and CIFS protocols to back up files in Infinite Volumes to tape and restore
files from tape. However, some protocols cannot back up the streams or access control lists (ACLs)
associated with the files.

You should consider the following implications when you choose a protocol to back up and restore
files for Infinite Volumes:

• If you are using NFSv3 to back up and restore files that use ACLs, NFSv3 backs up and restores
  only the files, not the ACLs.
  You should use another supported protocol to back up and restore both the files and the ACLs.

• If you want to back up and restore the files and the streams, you should use CIFS.
  When a file uses streams, only CIFS can back up and restore the files and the streams associated
  with the files.

Related concepts

- Support for CIFS on Infinite Volumes on page 72
- Support for NFS on Infinite Volumes on page 71
- Protocol considerations for restoring files to Infinite Volumes with storage classes on page 227

Tape backup topology to use for Infinite Volumes

For Infinite Volumes, you must use the tape backup topology specified by the backup application for
volumes mounted with the NFS or CIFS protocol. You cannot use an NDMP-supported tape backup
topology for Infinite Volumes.

Tape backup topologies for mounted volumes do not have any components installed on NetApp
controllers and use the network to communicate with Data ONTAP. For example, when you back up
data for Infinite Volumes, the data flows as follows:

1. The NetApp controller sends data over the network to the backup application client.
2. The backup application client communicates over the network with the backup application.
3. The backup application communicates over a direct connection with the tape device.
The following diagram illustrates the data flow:

![Diagram showing data flow]

Each backup application specifies how to set up tape libraries or tape drives to back up data over a mounted volume. For more information about tape library and tape drive requirements for backup applications, see your backup application documentation.

**How incremental tape backup uses SnapDiff and Snapshot copies**

The storage capacity potential of an Infinite Volume is larger than what a traditional file-scanning backup application can back up in a reasonable time. An incremental backup of Infinite Volumes to tape by using SnapDiff and Snapshot copies is the only viable solution for large Infinite Volumes.

**What SnapDiff is**

SnapDiff is an internal Data ONTAP engine that quickly identifies the file and directory differences between two Snapshot copies.

By finding the differences between two Snapshot copies, SnapDiff eliminates the file scanning requirements of a traditional backup application during an incremental backup, which reduces the backup processing to only the time it takes to write the changed or added data.

When incrementally backing up an Infinite Volume to tape using SnapDiff, the backup application uses the SnapDiff application programming interfaces (APIs) to communicate with the SnapDiff engine to identify new, changed, and deleted files between two Snapshot copies of the active file system in an Infinite Volume. The differencing process uses the namespace constituent and namespace mirror constituents in an Infinite Volume to determine names for the list of new, changed, and deleted files. When these changes are identified, the backup application backs up the identified data from the list produced during the differencing process.

**Related concepts**

How incremental tape backup uses namespace-related constituents on page 142

**SnapDiff support for backup applications**

You must enable SnapDiff on an Infinite Volume for a SnapDiff-supported backup application to incrementally back up the data to tape. Only supported third-party backup applications can use SnapDiff.

Because the backup application requires integration with the SnapDiff APIs to use SnapDiff for incremental tape backup of Infinite Volumes, only supported third-party backup applications can use SnapDiff for incremental tape backup of Infinite Volumes.
Guidelines for using LIFs with SnapDiff

You should connect backup applications to the data LIF for the Storage Virtual Machine (SVM) with Infinite Volume to allow SnapDiff to communicate large amounts of information. You should not connect backup applications to the cluster-management LIF or to the node-management LIF.

SnapDiff differs from other application programming interfaces (APIs) in the amount of information handled. Where APIs typically retrieve a small amount of control and management information, SnapDiff retrieves information about each file in an Infinite Volume. As a result, the amount of information reported by SnapDiff to the backup application can be large, depending on the number of files in the Infinite Volume. Because of the potentially large amount of information, you should connect the backup application to the data LIF for the SVM with Infinite Volume. For this network configuration, you must change the firewall settings on the data LIF to allow SnapDiff communication.

You should not connect the backup application to the cluster-management LIF for the SVM with Infinite Volume. However, if you do connect to the cluster-management LIF, backup applications must include the name of the SVM and the name of the Infinite Volume in the SnapDiff APIs to identify which SVM to use.

You should not connect the backup application to the node-management LIF because node-management LIFs are aware of nodes, not clusters, and an Infinite Volume spans multiple nodes.

What Snapshot copies you should back up to tape

You can back up any of the Snapshot copies that display a valid state for an Infinite Volume when you use the `volume snapshot show` command with the `-state` parameter.

**Note:** When using CIFS to back up Snapshot copies on an Infinite Volume, you should create a share at `/NS/.snapshot` or below.

How incremental tape backup uses namespace-related constituents

Incremental tape backup of Infinite Volumes uses the namespace constituent and namespace mirror constituents. You should understand space and node requirements for namespace mirror constituents, when namespace mirror constituents are created, and how namespace mirror constituents affect the state of an Infinite Volume.

Related concepts

- What a namespace constituent is on page 14
- What a namespace mirror constituent is on page 14

When namespace mirror constituents are created

When you create an Infinite Volume, one namespace mirror constituent is automatically created to provide data protection for the namespace constituent. When you enable SnapDiff for an Infinite Volume, additional namespace mirror constituents are automatically created for SnapDiff for incremental tape backup of Infinite Volumes.

When you create a read/write Infinite Volume that spans two or more nodes in a cluster, one namespace mirror constituent is automatically created, and a data protection mirror relationship is automatically created between the namespace constituent and the namespace mirror constituent. The data protection mirror relationship is updated every five minutes. The data protection mirror relationship is an automatic process for an Infinite Volume. You cannot use SnapMirror commands to
modify or manage the data protection mirror relationship between the namespace constituent and the namespace mirror constituent.

The following illustration shows the namespace (NS) constituent and the namespace mirror (NSm) constituent for an Infinite Volume:

When you enable SnapDiff on an Infinite Volume that spans three or more nodes, additional namespace mirror constituents are automatically created for SnapDiff to use for incremental tape backup of Infinite Volumes. A namespace mirror constituent is created on each node with a data constituent, except the node with the namespace constituent and the node with the namespace mirror constituent that was created to provide data protection for the namespace constituent. Namespace mirror constituents created to support SnapDiff are updated daily or as configured for SnapDiff. A SnapMirror license is not required to enable SnapDiff.

The following illustration shows the namespace constituent and the namespace mirror constituents for an Infinite Volume with SnapDiff enabled:

When you create a destination Infinite Volume for a data protection mirror relationship, a namespace mirror constituent is not created on the destination Infinite Volume. SnapDiff is disabled on the source and the destination Infinite Volumes. With SnapDiff disabled, a namespace mirror constituent is created on the source Infinite Volume to provide data protection for the namespace constituent, and no other namespace mirror constituents are created.

The following illustration shows an Infinite Volume in a data protection mirror relationship, with SnapDiff disabled:
However, if you enable SnapDiff on a destination Infinite Volume, namespace mirror constituents are automatically created for use by SnapDiff. You must initialize the data protection mirror relationship between the source and destination Infinite Volumes before you can enable SnapDiff. SnapDiff is disabled on the source Infinite Volume and enabled on the destination Infinite Volume. With SnapDiff enabled on the destination Infinite Volume, namespace mirror constituents are created on each node with a data constituent, except the node with the namespace constituent.

The following illustration shows an Infinite Volume in a data protection mirror relationship, with SnapDiff enabled on the destination Infinite Volume:
Related concepts

*What a namespace constituent is* on page 14

**How modifying an Infinite Volume affects namespace mirror constituents**

Namespace mirror constituents are automatically created or deleted when you modify Infinite Volumes, depending on the SnapDiff setting for the volume. The behavior differs slightly between a read/write Infinite Volume and a read-only destination Infinite Volume in a data protection mirror relationship.

When you modify a read/write Infinite Volume to increase its size and create data constituents on new nodes, and SnapDiff is already enabled on the Infinite Volume, namespace mirror constituents are automatically created on new nodes as required. When you modify a read/write or read-only Infinite Volume, and SnapDiff is already disabled, all namespace mirror constituents—except one namespace mirror constituent—are automatically deleted. One namespace mirror constituent is retained to provide data protection for the namespace constituent.

The following table summarizes when namespace mirrors are automatically created and deleted when you modify a read/write or read-only Infinite Volume:
<table>
<thead>
<tr>
<th>Infinite Volume</th>
<th>SnapDiff setting</th>
<th>Task</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>Read/write Infinite Volume</td>
<td>Enabled</td>
<td>Add new nodes to the cluster and increase volume size by using volume modify</td>
<td>Namespace mirror constituents are automatically created on new nodes as required.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Note: You must increase the size enough to create new data constituents on the new nodes.</td>
<td></td>
</tr>
<tr>
<td>Read-only destination Infinite Volume in a data protection mirror relationship</td>
<td></td>
<td></td>
<td>NameSpace mirror constituents are not automatically created on new nodes, although SnapDiff is enabled on the destination Infinite Volume. You must expand the destination Infinite Volume onto new nodes and update the data protection mirror relationship before you reenable SnapDiff to automatically create namespace mirror constituents on new nodes.</td>
</tr>
<tr>
<td>Read/write Infinite Volume and read-only destination Infinite Volume in a data protection mirror relationship</td>
<td>Disabled</td>
<td>Modify the volume by using volume modify</td>
<td>All namespace mirror constituents—except one namespace mirror constituent—are automatically deleted.</td>
</tr>
</tbody>
</table>

**How namespace mirror constituents affect the state of an Infinite Volume**

Infinite Volumes with SnapDiff enabled temporarily display a mixed state while namespace mirror constituents are created. After the namespace mirror constituents are created, Infinite Volumes automatically display an online state. You cannot start new operations until the namespace mirror constituents are created, and the Infinite Volume displays an online state.

The amount of time required to create all of the namespace mirror constituents depends on the size of the namespace constituent and the number of nodes an Infinite Volumes spans.

If you enable SnapDiff when you create an Infinite Volume, the namespace constituent is empty, and little time is required to create namespace mirror constituents on all of the nodes that an Infinite Volume spans. As a result, the Infinite Volume displays a mixed state for a small amount of time.

If you enable SnapDiff after an Infinite Volume has existed for some time, the namespace constituent contains data, and more time is required to create namespace mirror constituents. The Infinite Volume displays a mixed state until all of the namespace mirror constituents are created and initialized on all of the nodes with data constituents. The number of nodes that the Infinite Volume spans also affects the amount of time it takes to create and initialize namespace mirror constituents.

When you resize an Infinite Volume with SnapDiff enabled, namespace mirror constituents are created and initialized on all the new nodes with data constituents. Depending on the size of the namespace constituent and the number of new nodes, it can take some time to create and initialize a namespace mirror constituent on each new node. The Infinite Volume displays a mixed state while the namespace mirror constituents are being initialized.
Before you start a tape backup operation, you should wait until all of the namespace mirror constituents are created and initialized, and the Infinite Volume displays an online state.

### How aggregates are selected for namespace mirror constituents

One process is used to automatically select an aggregate for the namespace mirror constituent created to provide data protection of the namespace constituent in an Infinite Volume. Another process is used to automatically select aggregates for the namespace mirror constituents created for SnapDiff and incremental tape backup of Infinite Volumes.

Depending on the number of nodes, Data ONTAP executes the following process to select an aggregate for the namespace mirror constituent used for data protection of the namespace constituent:

<table>
<thead>
<tr>
<th>Number of nodes</th>
<th>Aggregate selection process</th>
</tr>
</thead>
<tbody>
<tr>
<td>For Storage Virtual Machines (SVMs) with Infinite Volume that use aggregates from two nodes</td>
<td>Data ONTAP considers the node without the namespace constituent and chooses an aggregate with the largest available space.</td>
</tr>
<tr>
<td>For SVMs with Infinite Volume that use aggregates from multiple nodes</td>
<td>Data ONTAP considers all nodes that contain a data constituent and chooses an aggregate with the largest available space. The aggregate selection process excludes the node that contains the namespace constituent. The aggregate selection process prefers to also exclude the HA pair of the node that contains the namespace constituent, but uses the node when no other nodes are available.</td>
</tr>
</tbody>
</table>

Data ONTAP executes the following process to select aggregates for namespace mirror constituents used for SnapDiff and incremental tape backup:

1. Identifies all the nodes in the cluster that an Infinite Volume spans
2. Identifies the nodes that have a data constituent for the Infinite Volume
3. Creates a namespace mirror constituent on each node that has a data constituent for the Infinite Volume, except for the node that has the namespace constituent and the node in which the first namespace mirror constituent is created

When you expand an Infinite Volume with SnapDiff enabled onto new nodes, additional namespace mirror constituents are automatically created, and aggregates are automatically selected for the new namespace mirror constituents.

**Related concepts**

*How space is allocated inside a new Infinite Volume* on page 37

### Space and node requirements for namespace mirror constituents

Up to 25% of the total size of an Infinite Volume can be used for the namespace constituent and namespace mirror constituents. The number of namespace mirror constituents created depends on whether SnapDiff is enabled for the Infinite Volume and the number of nodes the Infinite Volume spans.

An Infinite Volume must span two or more nodes before a namespace mirror constituent is created to provide data protection for the namespace constituent. An Infinite Volume must span three or more nodes before additional namespace mirror constituents are created for SnapDiff to use.

The following table summarizes the impact of nodes and SnapDiff on the number of namespace mirror constituents created for an Infinite Volume:
When an Infinite Volume spans... | The following constituents are created...
---|---
Two or more nodes in a cluster and SnapDiff is disabled for the Infinite Volume | • One namespace constituent  
• One namespace mirror constituent for data protection

Two nodes in a cluster and SnapDiff is enabled for the Infinite Volume | • One namespace constituent  
• One namespace mirror constituent for data protection  
SnapDiff can also use this namespace mirror constituent.  
• No additional namespace mirror constituents  
SnapDiff has no additional namespace mirror constituents to use.

Three or more nodes in a cluster and SnapDiff is enabled for the Infinite Volume | • One namespace constituent  
• One namespace mirror constituent for data protection  
• One namespace mirror constituent on each node that contains a data constituent, except for the node that contains the namespace constituent and the node that contains the namespace mirror constituent for data protection

Each namespace mirror constituent is the same size as the namespace constituent. The maximum size for the namespace constituent is 10 TB. The space available in an Infinite Volume is divided evenly among the namespace constituent and the namespace mirror constituents. The sum of the space used by the namespace constituent and each namespace mirror constituent can consume up to 25% of the total size of the Infinite Volume or 100 TB of space. For example, consider an Infinite Volume that is 80 TB in size, spans two nodes, and has SnapDiff disabled. The Infinite Volume contains a 10 TB namespace constituent and a 10 TB namespace mirror constituent. The namespace constituent and the namespace mirror constituent consume 25% of the total size of the Infinite Volume.

**Note:** The total size of the namespace constituent and namespace mirror constituents should not exceed 100 TB of space because an Infinite Volume should span a maximum of 10 nodes, and each namespace constituent or namespace mirror constituent can be a maximum of 10 TB. As a result, 10 x 10 TB = 100 TB of space.

**Related concepts**

*How much space namespace-related constituents require* on page 35
Setting up incremental tape backup of Infinite Volumes

Setting up incremental tape backup involves verifying that the Infinite Volume has sufficient space for SnapDiff resources, enabling SnapDiff on the Infinite Volume, and creating a baseline for incremental tape backup.

Steps

1. Deciding whether to verify space requirements on page 149
2. Verifying space requirements in an Infinite Volume for SnapDiff resources (cluster administrators only) on page 149
3. Enabling SnapDiff on an Infinite Volume on page 151
4. Selecting a Snapshot copy of an Infinite Volume for SnapDiff to use on page 152
5. Creating a baseline for incremental tape backup on page 153

Deciding whether to verify space requirements

In most cases, you should verify space requirements before enabling SnapDiff to ensure that the Infinite Volume has enough space for the namespace mirror constituents. However, in some cases, you can enable SnapDiff without first verifying space requirements. You should understand when to verify space requirements.

Step

1. Deciding whether to verify space requirements in an Infinite Volume before enabling SnapDiff:

<table>
<thead>
<tr>
<th>If...</th>
<th>Then...</th>
</tr>
</thead>
<tbody>
<tr>
<td>The Infinite Volume spans two nodes in a cluster</td>
<td>Enable SnapDiff without verifying space requirements. In this case, no new namespace mirror constituents are created. Instead SnapDiff uses the namespace constituent and the namespace mirror constituent created for data protection of the namespace constituent.</td>
</tr>
<tr>
<td>The Infinite Volume spans three or more nodes</td>
<td>Verify the space requirements. Additional namespace mirror constituents are created when you enable SnapDiff, and you should verify that the Infinite Volume has enough space.</td>
</tr>
<tr>
<td>The Infinite Volume is in a data protection mirror relationship and you plan to enable SnapDiff on a destination Infinite Volume that spans two or more nodes</td>
<td>Verify the space requirements. Additional namespace mirror constituents are created for destination Infinite Volumes that span two or more nodes when you enable SnapDiff. When a destination Infinite Volume spans two or more nodes, you should verify that the destination Infinite Volume has enough space before enabling SnapDiff.</td>
</tr>
</tbody>
</table>

Verifying space requirements in an Infinite Volume for SnapDiff resources (cluster administrators only)

Before you enable SnapDiff on an Infinite Volume, you must ensure that the Infinite Volume has enough space for all of the namespace mirror constituents that will be created for SnapDiff to use for tape backup.

Before you begin

You must have cluster administrator privileges.
Steps

1. View the size of the namespace constituent on the Infinite Volume by using the `volume show` command with the `-is-constituent true` parameter.

   Each namespace mirror constituent is the same size as the namespace constituent.

   **Example**

   In the following example, the namespace constituent is named `repo_vol_ns`, and it is 10 TB in size. The namespace mirror constituent created for data protection of the namespace constituent is named `repo_vol_ns_mirror0001`, and it is 10 TB in size.

   ```
   cluster1::> volume show -is-constituent true
   Vserver Volume                 Aggregate State  Type Size  Available Used%  
   ------- ------------           --------- ------ ---- ----- --------- ----- 
   vs0     repo_vol_1024_data0001 aggr1     online RW   100TB    95TB    5%  
   vs0     repo_vol_1024_data0002 aggr2     online RW   100TB    95TB    5%  
   vs0     repo_vol_1024_data0003 aggr3     online RW   100TB    95TB    5%  
   vs0     repo_vol_1024_data0004 aggr4     online RW   100TB    95TB    5%  
   vs0     repo_vol_ns            aggr1     online RW    10TB   9.5TB    5%  
   vs0     repo_vol_ns_mirror0001 aggr2     online DP    10TB   9.5TB    5%  
   6 entries were displayed.
   ```

2. Make a list of the aggregates that have a data constituent.

   In the preceding example, the data constituents are named `repo_vol_1024_data0001`, `repo_vol_1024_data0002`, `repo_vol_1024_data0003`, and `repo_vol_1024_data0004`, and the data constituents use the following aggregates: `aggr1`, `aggr2`, `aggr3`, and `aggr4`.

3. Identify which of the aggregates that contain a data constituent also have a namespace constituent or a namespace mirror constituent.

   In the preceding example, the following aggregates with data constituents also contain either a namespace constituent or a namespace mirror constituent:

   - `aggr1` contains a data constituent named `repo_vol_1024_data0001` and a namespace constituent named `repo_vol_ns`.
   - `aggr2` contains a data constituent named `repo_vol_1024_data0002` and a namespace mirror constituent named `repo_vol_ns_mirror0001`.

   The aggregates named `aggr3` and `aggr4` contain data constituents, but no namespace constituent or namespace mirror constituent.

4. View the aggregates to identify the nodes that contain them by using the `storage aggregate show` command.

   **Example**

   In the following example, aggregates `aggr3` and `aggr4` have data constituents, but do not have a namespace constituent or a namespace mirror constituent, and `aggr3` is on node3 and `aggr4` is on node4. As a result, node3 and node4 require namespace mirror constituents for SnapDiff to use.

   ```
   cluster1::> storage aggregate show
   Aggregate  Size   Available Used% State #Vols Nodes  RAID Status
   -------     ------ --------- ----- ------ ------ ------ ------------
   aggr0       1.46TB  69.40GB   95%   online   1   node2  raid_dp,normal
   aggr0_node_0 1.46TB  69.40GB   95%   online   1   node1  raid_dp,normal
   aggr0_node3_0 1.46TB  69.40GB   95%   online   1   node3  raid_dp,normal
   aggr0_node4_0 1.46TB  69.40GB   95%   online   1   node4  raid_dp,normal
   aggr1       250TB  140TB    44%   online   2   node1  raid_dp,normal
   aggr2       250TB  140TB    44%   online   2   node2  raid_dp,normal
   aggr3       250TB  150TB    40%   online   1   node3  raid_dp,normal
   aggr4       250TB  150TB    40%   online   2   node4  raid_dp,normal
   8 entries were displayed.
   ```
You have a list of the nodes with data constituents that require namespace mirror constituents for SnapDiff to use for tape backup.

5. For each node with a data constituent that requires a namespace mirror constituent for SnapDiff to use, ensure that at least one aggregate contained by the node has enough available space for a namespace mirror constituent by using the `storage aggregate show` command.

The namespace mirror constituent will be the same size as the namespace constituent. In this example, node3 and node4 require a namespace mirror constituent, and the namespace mirror constituent will be 10TB. As a result, node3 and node4 must contain aggregates with enough space for a namespace mirror constituent that is 10 TB in size.

**Example**

In the following example, aggr3 is 250 TB in size, and only 100 TB of its space is used; therefore, aggr3 has enough space for a namespace mirror constituent.

```
cluster1::> storage aggregate show -aggregate aggr3

Aggregate: aggr3
...
Size: 250TB
State: online
Used Size: 100TB
Number Of Volumes: 1
Volume Style: flex
```

6. Increase the aggregate space if required.

**Related concepts**

- *Space and node requirements for namespace mirror constituents* on page 147

**Related information**

- *Disk and aggregate management*

**Enabling SnapDiff on an Infinite Volume**

You must enable SnapDiff on an Infinite Volume to set up incremental tape backup.

**Before you begin**

- The Infinite Volume must span two or more nodes in the cluster.
- All nodes in the cluster for the Infinite Volume must be running Data ONTAP 8.2 or later.

**About this task**

You can have cluster administrator or Storage Virtual Machine (SVM) administrator privileges to perform this task. However, if you receive a message about insufficient aggregate space, you require cluster administrator privileges to add and modify aggregates. With cluster administrator and advanced privileges, you can identify the list of aggregates to use for the namespace mirror constituents by using the `volume modify` command with the `-ns-mirror-aggr-list` parameter.

You can enable SnapDiff when you create an Infinite Volume (recommended) or when you modify an Infinite Volume. When you modify an Infinite Volume to enable or disable SnapDiff, you cannot modify any other parameters at the same time. For example, you cannot increase the size of the Infinite Volume at the same time as you enable or disable SnapDiff. After you enable or disable SnapDiff, you can modify other parameters on the Infinite Volume.
Step

1. Enable SnapDiff on the Infinite Volume by using the `volume modify` command with the `-enable-snapdiff` parameter.

Example

In the following example, SnapDiff is enabled for an Infinite Volume named `repo_vol` that spans four nodes:

```bash
cluster1::> volume modify repo_vol -enable-snapdiff true
[Job 39] Job succeeded: Modified Infinite Volume successfully.
```

Before enabling SnapDiff, the Infinite Volume contains one namespace mirror constituent named `repo_vol_ns_mirror0001`:

```bash
cluster1::> volume show -is-constituent true
```

<table>
<thead>
<tr>
<th>Vserver Volume</th>
<th>Aggregate State</th>
<th>Type</th>
<th>Size</th>
<th>Available</th>
<th>Used%</th>
</tr>
</thead>
<tbody>
<tr>
<td>vs0 repo_vol_1024_data0001</td>
<td>aggr1</td>
<td>online</td>
<td>RW</td>
<td>100TB</td>
<td>95TB</td>
</tr>
<tr>
<td>vs0 repo_vol_1024_data0002</td>
<td>aggr2</td>
<td>online</td>
<td>RW</td>
<td>100TB</td>
<td>95TB</td>
</tr>
<tr>
<td>vs0 repo_vol_1024_data0003</td>
<td>aggr3</td>
<td>online</td>
<td>RW</td>
<td>100TB</td>
<td>95TB</td>
</tr>
<tr>
<td>vs0 repo_vol_1024_data0004</td>
<td>aggr4</td>
<td>online</td>
<td>RW</td>
<td>100TB</td>
<td>95TB</td>
</tr>
<tr>
<td>vs0 repo_vol_ns</td>
<td>aggr1</td>
<td>online</td>
<td>DP</td>
<td>10TB</td>
<td>9.5TB</td>
</tr>
<tr>
<td>vs0 repo_vol_ns_mirror0001</td>
<td>aggr2</td>
<td>online</td>
<td>DP</td>
<td>10TB</td>
<td>9.5TB</td>
</tr>
</tbody>
</table>

6 entries were displayed.

After enabling SnapDiff, the Infinite Volume contains three namespace mirror constituents named `repo_vol_ns_mirror0001`, `repo_vol_ns_mirror0002`, and `repo_vol_ns_mirror0003`:

```bash
cluster1::> volume show -is-constituent true
```

<table>
<thead>
<tr>
<th>Vserver Volume</th>
<th>Aggregate State</th>
<th>Type</th>
<th>Size</th>
<th>Available</th>
<th>Used%</th>
</tr>
</thead>
<tbody>
<tr>
<td>vs0 repo_vol_1024_data0001</td>
<td>aggr1</td>
<td>online</td>
<td>RW</td>
<td>100TB</td>
<td>95TB</td>
</tr>
<tr>
<td>vs0 repo_vol_1024_data0002</td>
<td>aggr2</td>
<td>online</td>
<td>RW</td>
<td>100TB</td>
<td>95TB</td>
</tr>
<tr>
<td>vs0 repo_vol_1024_data0003</td>
<td>aggr3</td>
<td>online</td>
<td>RW</td>
<td>100TB</td>
<td>95TB</td>
</tr>
<tr>
<td>vs0 repo_vol_1024_data0004</td>
<td>aggr4</td>
<td>online</td>
<td>RW</td>
<td>100TB</td>
<td>95TB</td>
</tr>
<tr>
<td>vs0 repo_vol_ns</td>
<td>aggr1</td>
<td>online</td>
<td>DP</td>
<td>10TB</td>
<td>9.5TB</td>
</tr>
<tr>
<td>vs0 repo_vol_ns_mirror0001</td>
<td>aggr2</td>
<td>online</td>
<td>DP</td>
<td>10TB</td>
<td>9.5TB</td>
</tr>
<tr>
<td>vs0 repo_vol_ns_mirror0002</td>
<td>aggr3</td>
<td>online</td>
<td>DP</td>
<td>10TB</td>
<td>9.5TB</td>
</tr>
<tr>
<td>vs0 repo_vol_ns_mirror0003</td>
<td>aggr4</td>
<td>online</td>
<td>DP</td>
<td>10TB</td>
<td>9.5TB</td>
</tr>
</tbody>
</table>

8 entries were displayed.

Namespace mirror constituents are created for SnapDiff to use.

Related concepts

*How namespace mirror constituents affect the state of an Infinite Volume* on page 146

*What Snapshot copies you should back up to tape* on page 142

Selecting a Snapshot copy of an Infinite Volume for SnapDiff to use

You must select a Snapshot copy of an Infinite Volume for SnapDiff to use for incremental tape backup. SnapDiff uses the Snapshot copy to help identify new, changed, or deleted files in an Infinite Volume.

About this task

You can use a Snapshot copy created on a schedule by Data ONTAP, or you can use a Snapshot copy created by a backup application. The Snapshot copy of the Infinite Volume must be in a valid state for SnapDiff to use it.
Steps

1. Perform one of the following actions:

<table>
<thead>
<tr>
<th>To...</th>
<th>Do this...</th>
</tr>
</thead>
<tbody>
<tr>
<td>Use a Snapshot copy created by Data ONTAP on a regular schedule for an Infinite Volume</td>
<td>Select a Snapshot copy in a valid state from the available Snapshot copies by running the <code>volume snapshot show</code> command with the <code>-state</code> parameter.</td>
</tr>
<tr>
<td>Create a regularly scheduled Snapshot copy for SnapDiff to use</td>
<td>Create a Snapshot copy of the Infinite Volume by using the <code>volume snapshot create</code> command.</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> The Infinite Volume must be in an online state to create a Snapshot copy.</td>
</tr>
<tr>
<td>Use the backup application to initiate the creation of a Snapshot copy of the Infinite Volume</td>
<td>See the documentation provided with the backup application.</td>
</tr>
</tbody>
</table>

You have selected a Snapshot copy of an Infinite Volume to use with SnapDiff.

2. Configure SnapDiff to use the Snapshot copy.

   For more information about SnapDiff configuration, see the documentation provided with the backup application.

Related tasks

- [Viewing the state of Snapshot copies for an Infinite Volume](#) on page 101

Creating a baseline for incremental tape backup

With SnapDiff enabled, you can use the NFS or CIFS protocol to mount an Infinite Volume, and you can use a backup application to back up all the files from the mounted Infinite Volume to tape. The tape backup creates a baseline for SnapDiff to use.

Before you begin

- A backup application must be installed and running, and it must support tape backup of files from a volume that is mounted with a supported version of the NFS or CIFS protocol.
- The server running the backup application must be on the same network as the Storage Virtual Machine (SVM) with Infinite Volume that contains the mounted Infinite Volume.
- SnapDiff API must be configured for the Infinite Volume.
- SnapDiff must be enabled on the Infinite Volume.
- A Snapshot copy for the SnapDiff API to use must exist on the Infinite Volume, and the SnapDiff API must be configured to use the Snapshot copy.
  
  For more information about SnapDiff configuration, see the documentation provided with the backup application.

About this task

The backup application backs up data from the entire Infinite Volume to tape. You cannot configure the backup application to back up data from specific, individual data constituents to tape.
Steps

1. Use the NFS or CIFS protocol to mount the Infinite Volume.
2. Ensure that the server running the backup application can read and write data to the Infinite Volume.
   
   If you are backing up a destination Infinite Volume, ensure that the server running the backup application can read data from the destination Infinite Volumes. The server running the backup application cannot write data because the active file system on a destination Infinite Volume is read-only.
3. Configure the backup application to back up the mounted Infinite Volume to tape.
   
   For more information about backing up files to tape, see the documentation provided with the backup application.

Result

All of the files in the Infinite Volume are backed up to tape, and a baseline is created for SnapDiff to use.

After you finish

You can now use subsequent backups to back up only the new and changed files in the Infinite Volume.

Related tasks

* Incrementally backing up changed files from an Infinite Volume to tape on page 156

Setting up incremental tape backup of destination Infinite Volumes

When an Infinite Volume is in a data protection mirror relationship, you can use a backup application and SnapDiff to back up to tape a Snapshot copy on the destination Infinite Volume rather than on the source Infinite Volume. This setup moves the processing workload to the destination volume.

Steps

1. Transferring a Snapshot copy to the destination Infinite Volume on page 154
2. Creating a baseline for incremental tape backup on page 155

Transferring a Snapshot copy to the destination Infinite Volume

You can create a Snapshot copy on the source Infinite Volume and use the `snapmirror update` command to transfer the Snapshot copy to the destination Infinite Volume. Backup applications can use the Snapshot copy on the destination volume for incremental tape backup.

Before you begin

- The source and destination Infinite Volumes must be in a data protection mirror relationship, and the data protection mirror relationship must be initialized.
- The destination Infinite Volume must span two or more nodes.
- The destination Infinite Volume must have enough space for SnapDiff resources.
- SnapDiff must be enabled on the destination Infinite Volume.
Steps

1. On the source Infinite Volume, create a Snapshot copy to use for tape backup by using the `volume snapshot create` command.

   **Example**

   In the following example, a Snapshot copy named `tape_backup` is created for a source Infinite Volume named `repo_vol`:

   ```bash
   cluster1::> volume snapshot create -volume repo_vol -snapshot tape_backup
   ```

2. On the destination cluster, manually update the destination Infinite Volume by using the `snapmirror update` command.

   Alternatively, you can wait for the regularly scheduled data protection mirror update to occur to transfer the Snapshot copy to the destination Infinite Volume.

   **Example**

   In the following example, the data protection mirror relationship is updated for a destination Infinite Volume named `repo_vol_dest` on a destination Storage Virtual Machine (SVM) with Infinite Volume named `vs0_dest`:

   ```bash
   cluster2::> snapmirror update -destination-path vs0_dest:repo_vol_dest
   ```

   The latest data protection mirror copy of the source Infinite Volume is transferred to the destination Infinite Volume, and the transfer includes the latest Snapshot copy that you created.

Related concepts

- Providing disaster recovery on Infinite Volumes using mirroring technology on page 104

Related tasks

- Verifying space requirements in an Infinite Volume for SnapDiff resources (cluster administrators only) on page 149
- Enabling SnapDiff on an Infinite Volume on page 151

Creating a baseline for incremental tape backup

With SnapDiff enabled, you can use the NFS or CIFS protocol to mount an Infinite Volume, and you can use a backup application to back up all the files from the mounted Infinite Volume to tape. The tape backup creates a baseline for SnapDiff to use.

Before you begin

- A backup application must be installed and running, and it must support tape backup of files from a volume that is mounted with a supported version of the NFS or CIFS protocol.

- The server running the backup application must be on the same network as the Storage Virtual Machine (SVM) with Infinite Volume that contains the mounted Infinite Volume.

- SnapDiff API must be configured for the Infinite Volume.

- SnapDiff must be enabled on the Infinite Volume.

- A Snapshot copy for the SnapDiff API to use must exist on the Infinite Volume, and the SnapDiff API must be configured to use the Snapshot copy.

For more information about SnapDiff configuration, see the documentation provided with the backup application.
About this task

The backup application backs up data from the entire Infinite Volume to tape. You cannot configure the backup application to back up data from specific, individual data constituents to tape.

Steps

1. Use the NFS or CIFS protocol to mount the Infinite Volume.

2. Ensure that the server running the backup application can read and write data to the Infinite Volume.
   
   If you are backing up a destination Infinite Volume, ensure that the server running the backup application can read data from the destination Infinite Volumes. The server running the backup application cannot write data because the active file system on a destination Infinite Volume is read-only.

3. Configure the backup application to back up the mounted Infinite Volume to tape.
   
   For more information about backing up files to tape, see the documentation provided with the backup application.

Result

All of the files in the Infinite Volume are backed up to tape, and a baseline is created for SnapDiff to use.

After you finish

You can now use subsequent backups to back up only the new and changed files in the Infinite Volume.

Incrementally backing up changed files from an Infinite Volume to tape

Backup applications can use SnapDiff to identify new, modified, and deleted files in an Infinite Volume since the last tape backup, and then back up to tape only the changed files instead of all of the files in the Infinite Volume.

Before you begin

- You must have set up incremental tape backup to create a baseline against which SnapDiff can identify changed files in the Infinite Volume since the last tape backup.

Steps

1. Use a backup application to request from SnapDiff a list of changed files in the Infinite Volume since the last tape backup.

2. Use a backup application to back up the changed files to tape.

Related tasks

Setting up incremental tape backup of Infinite Volumes on page 149
Setting up incremental tape backup of destination Infinite Volumes on page 154
Restoring files from tape for an Infinite Volume

You can use the NFS or CIFS protocol to mount the Infinite Volume and restore files from a tape backup to the mounted Infinite Volume.

Before you begin

- A backup application must be installed and running, and it must support the retrieval of files from a tape backup to a volume that is mounted with a supported version of the NFS or CIFS protocol.
- The server running the backup application must be on the same network as the Storage Virtual Machine (SVM) with Infinite Volume that contains the mounted Infinite Volume.

About this task

Some but not all versions of NFS and CIFS are supported for Infinite Volumes.

When an Infinite Volume is in a data protection mirror relationship, you can only restore files to the source Infinite Volume because the destination Infinite Volume is read-only. If you want to restore files to the destination Infinite Volume, you must first break the data protection mirror relationship to change the destination Infinite Volume from read-only to read/write.

Steps

1. Use the NFS or CIFS protocol to mount the Infinite Volume.
2. Ensure that the server running the backup application can read and write data to the Infinite Volume.
3. Restore files from the tape backup to the mounted Infinite Volume.
   
   For more information about restoring files, see the documentation provided with the backup application.

Related concepts

Support for CIFS on Infinite Volumes on page 72
Support for NFS on Infinite Volumes on page 71
Protocol considerations for restoring files to Infinite Volumes with storage classes on page 227

Disabling SnapDiff for an Infinite Volume

You can disable SnapDiff on an Infinite Volume to disable incremental tape backup. With SnapDiff disabled, the namespace mirror constituents that are created for SnapDiff are deleted, and backup applications no longer receive a list of files that were changed since the last tape backup.

About this task

When you disable SnapDiff, all namespace mirror constituents, except one namespace mirror constituent, are automatically deleted from the nodes in the cluster. One namespace mirror constituent is retained to provide data protection for the namespace constituent.

Step

1. Disable SnapDiff on the Infinite Volume by using the `volume modify` command with the `–enable-snapdiff` parameter.
Example

In the following example, SnapDiff is disabled for the Infinite Volume named repo_vol:

```
cluster1::> volume modify repo_vol -enable-snapdiff false
    [Job 50] Job succeeded: Modified Infinite Volume successfully.
```

Before disabling SnapDiff, the Infinite Volume spans four nodes and contains three namespace mirror constituents named repo_vol_ns_mirror0001, repo_vol_ns_mirror0002, and repo_vol_ns_mirror0003:

```
cluster1::> volume show -is-constituent true
          Vserver Volume                 Aggregate State   Type  Size  Available Used%
------- ------------           --------- ------- ---- ----- --------- ----- 
vs0     repo_vol_1024_data0001 aggr1     online  RW   100TB      95TB   5%
vs0     repo_vol_1024_data0002 aggr2     online  RW   100TB      95TB   5%
vs0     repo_vol_1024_data0003 aggr3     online  RW   100TB      95TB   5%
vs0     repo_vol_1024_data0004 aggr4     online  RW   100TB      95TB   5%
vs0     repo_vol_ns            aggr1     online  RW    10TB     9.5TB   5%
vs0     repo_vol_ns_mirror0001 aggr2     online  DP    10TB     9.5TB   5%
vs0     repo_vol_ns_mirror0002 aggr3     online  DP    10TB     9.5TB   5%
vs0     repo_vol_ns_mirror0003 aggr4     online  DP    10TB     9.5TB   5%
8 entries were displayed.
```

After disabling SnapDiff, the Infinite Volume spans four nodes and contains one namespace mirror constituent named repo_vol_ns_mirror0001, which is used for data protection of the namespace constituent named repo_vol_ns:

```
cluster1::> volume show -is-constituent true
          Vserver Volume                 Aggregate State   Type  Size  Available Used%
------- ------------           --------- ------- ---- ----- --------- ----- 
vs0     repo_vol_1024_data0001 aggr1     online  RW   100TB      95TB   5%
vs0     repo_vol_1024_data0002 aggr2     online  RW   100TB      95TB   5%
vs0     repo_vol_1024_data0003 aggr3     online  RW   100TB      95TB   5%
vs0     repo_vol_1024_data0004 aggr4     online  RW   100TB      95TB   5%
vs0     repo_vol_ns            aggr1     online  RW    10TB     9.5TB   5%
vs0     repo_vol_ns_mirror0001 aggr2     online  DP    10TB     9.5TB   5%
vs0     repo_vol_ns_mirror0002 aggr3     online  DP    10TB     9.5TB   5%
vs0     repo_vol_ns_mirror0003 aggr4     online  DP    10TB     9.5TB   5%
6 entries were displayed.
```

All namespace mirror constituents are deleted, except the namespace mirror constituent used for data protection of the namespace constituent.


### Monitoring space usage in Infinite Volumes

You can monitor how much space is available in an Infinite Volume, how many files the Infinite Volume contains, and how much space is available in the Infinite Volume's aggregates. This information can help you decide when to expand an Infinite Volume.

**Related tasks**

*Monitoring space usage in Infinite Volumes with storage classes* on page 228

### Commands for monitoring space usage in an Infinite Volume

You can monitor the capacity of an Infinite Volume by displaying information about how full the Infinite Volume is and how full the Infinite Volume's aggregates are.

<table>
<thead>
<tr>
<th>To display information about...</th>
<th>Use this command...</th>
</tr>
</thead>
<tbody>
<tr>
<td>How full an Infinite Volume is</td>
<td><code>volume show -fields size,used,available,percent-used</code></td>
</tr>
<tr>
<td>How full an aggregate is</td>
<td><code>storage aggregate show -fields size,usedsize,availsize,percent-used</code></td>
</tr>
<tr>
<td>How much space is used by volumes and constituents on an aggregate</td>
<td><code>storage aggregate show-space -fields volume-footprints,used-including-snapshot-reserve</code></td>
</tr>
<tr>
<td>How much space is used on all aggregates by each Infinite Volume constituent</td>
<td><code>volume show -is-constituent true -fields aggregate,size,used,available,percent-used</code></td>
</tr>
</tbody>
</table>

For detailed information about these commands, see the appropriate man page.

**Note:** The `volume show-space` and `volume show-footprint` commands display information about Infinite Volume constituents as if the constituents were FlexVol volumes. The `volume show-space` and `volume show-footprint` commands do not display information about the Infinite Volume as a whole.

### Viewing efficiency space savings on an Infinite Volume

You can see the amount of space that is saved through deduplication and compression in the Infinite Volume as a whole or in each data constituent in an Infinite Volume by using the `volume show` command.

**Choices**

- View space savings in the entire Infinite Volume by using the `volume show` with the `-volume` parameter and either the `-instance` parameter or the `-fields` parameter.

**Example**

The following example uses the `-volume` and `-instance` parameters to show the space saved by storage efficiency for an Infinite Volume.
View space savings in each data constituents by using the `volume show` command with the `-vserver`, `-is-constituent true`, and `-fields` parameters.

For the `fields` parameter, the relevant values are:

- `-sis-space-saved,-sis-space-saved-percent` for savings from both deduplication and compression
- `-dedupe-space-saved,-dedupe-space-saved-percent,-dedupe-space-shared` for savings from deduplication only
- `-compression-space-saved,-compression-space-saved-percent,-used` for savings from compression only

**Example**

The following example uses the `-fields` parameter to show the space saved by storage efficiency for each constituent in the Infinite Volume. Most of the entries are for data constituents, and therefore show space savings. The last constituents are the namespace constituent and namespace mirror constituents, which do not use deduplication or compression.

```
cluster1::> volume show -volume repo_vol -instance
         Vserver Name: vs0
         Volume Name: repo_vol
...  
      Space Saved by Storage Efficiency: 802689024B
         Percentage Saved by Storage Efficiency: 34%
      Space Saved by Deduplication: 61317120B
         Percentage Saved by Deduplication: 3%
      Space Shared by Deduplication: 4988928B
      Space Saved by Compression: 796557312B
         Percentage Space Saved by Compression: 36%
... 
```

The system output displays the requested information for each constituent in the Infinite Volume individually.

**Related concepts**

*How efficiency works on Infinite Volumes with storage classes* on page 199

**How you can determine and control a volume's space usage in the aggregate**

You can determine which FlexVol volumes are using the most space in the aggregate and specifically which features within the volume. The `volume show-footprint` command provides information about a volume's footprint, or its space usage within the containing aggregate.

The `volume show-footprint` command shows details about the space usage of each volume in an aggregate, including offline volumes. This command bridges the gap between the output of the
volume show-space and aggregate show-space commands. All percentages are calculated as a percent of aggregate size.

The following example shows the volume show-footprint command output for a volume called testvol:

```
cluster1::> volume show-footprint testvol
Vserver : thevs
Volume  : testvol

<table>
<thead>
<tr>
<th>Feature</th>
<th>Used</th>
<th>Used%</th>
</tr>
</thead>
<tbody>
<tr>
<td>Volume Data Footprint</td>
<td>120.6MB</td>
<td>4%</td>
</tr>
<tr>
<td>Volume Guarantee</td>
<td>1.88GB</td>
<td>71%</td>
</tr>
<tr>
<td>Flexible Volume Metadata</td>
<td>11.38MB</td>
<td>0%</td>
</tr>
<tr>
<td>Delayed Frees</td>
<td>1.36MB</td>
<td>0%</td>
</tr>
<tr>
<td>Total Footprint</td>
<td>2.01GB</td>
<td>76%</td>
</tr>
</tbody>
</table>
```

The following table explains some of the key rows of the output of the volume show-footprint command and what you can do to try to decrease space usage by that feature:

<table>
<thead>
<tr>
<th>Row/feature name</th>
<th>Description/contents of row</th>
<th>Some ways to decrease</th>
</tr>
</thead>
<tbody>
<tr>
<td>Volume Data Footprint</td>
<td>The total amount of space used in the containing aggregate by a volume's data in the active file system and the space used by the volume's Snapshot copies. This row does not include reserved space.</td>
<td>• Deleting data from the volume.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Deleting Snapshot copies from the volume.</td>
</tr>
<tr>
<td>Volume Guarantee</td>
<td>The amount of space reserved by the volume in the aggregate for future writes. The amount of space reserved depends on the guarantee type of the volume.</td>
<td>Changing the type of guarantee for the volume to none.</td>
</tr>
<tr>
<td>Flexible Volume Metadata</td>
<td>The total amount of space used in the aggregate by the volume's metadata files.</td>
<td>No direct method to control.</td>
</tr>
<tr>
<td>Delayed Frees</td>
<td>Blocks that ONTAP used for performance and cannot be immediately freed.</td>
<td>No direct method to control.</td>
</tr>
<tr>
<td></td>
<td>For SnapMirror destinations, this row has a value of 0 and is not displayed.</td>
<td></td>
</tr>
<tr>
<td>File Operation Metadata</td>
<td>The total amount of space reserved for file operation metadata.</td>
<td>No direct method to control.</td>
</tr>
<tr>
<td>Total Footprint</td>
<td>The total amount of space that the volume uses in the aggregate. It is the sum of all of the rows.</td>
<td>Any of the methods used to decrease space used by a volume.</td>
</tr>
</tbody>
</table>

Related concepts

- How to determine space usage in an aggregate on page 162
- Methods to create space in an Infinite Volume or its aggregates on page 163
Related information

*NetApp Technical Report 3483: Thin Provisioning in a NetApp SAN or IP SAN Enterprise Environment*


What the volume footprint is

A volume footprint is the amount of space a volume is using within the aggregate. Understanding what is included in the volume footprint helps you understand the space requirements for the volume.

The volume footprint consists of the space used by user data and metadata, including metadata that resides in the aggregate rather than within the volume itself. For this reason it can be larger than the volume size, as shown in the following diagram:

![Diagram showing volume footprint](image)

How to determine space usage in an aggregate

You can view space usage by all volumes in one or more aggregates with the `aggregate show-space` command. This helps you see which volumes are consuming the most space in their containing aggregates so that you can take actions to free more space.

The used space in an aggregate is directly affected by the space used in the FlexVol volumes it contains. Measures that you take to increase space in a volume also affect space in the aggregate.

The following rows are included in the `aggregate show-space` command output:

- **Volume Footprints**
  The total of all volume footprints within the aggregate. It includes all of the space that is used or reserved by all data and metadata of all volumes in the containing aggregate.

- **Aggregate Metadata**
  The total file system metadata required by the aggregate, such as allocation bitmaps and inode files.

- **Snapshot Reserve**
  The amount of space reserved for aggregate Snapshot copies, based on volume size. It is considered used space and is not available to volume or aggregate data or metadata.

- **Snapshot Reserve Unusable**
  The amount of space originally allocated for aggregate Snapshot reserve that is unavailable for aggregate Snapshot copies because it is being used by volumes associated with the aggregate. Can occur only for aggregates with a non-zero aggregate Snapshot reserve.
• Total Used
  The sum of all space used or reserved in the aggregate by volumes, metadata, or Snapshot copies.

• Total Physical Used
  The amount of space being used for data now (rather than being reserved for future use). Includes space used by aggregate Snapshot copies.

The following example shows the aggregate show-space command output for an aggregate whose Snapshot reserve is 5%. If the Snapshot reserve was 0, the row would not be displayed.

```
cluster1::> storage aggregate show-space

Aggregate : wqa_gx106_aggr1

<table>
<thead>
<tr>
<th>Feature</th>
<th>Used</th>
<th>Used%</th>
</tr>
</thead>
<tbody>
<tr>
<td>Volume Footprints</td>
<td>101.0MB</td>
<td>0%</td>
</tr>
<tr>
<td>Aggregate Metadata</td>
<td>300KB</td>
<td>0%</td>
</tr>
<tr>
<td>Snapshot Reserve</td>
<td>5.98GB</td>
<td>5%</td>
</tr>
<tr>
<td>Total Used</td>
<td>6.07GB</td>
<td>5%</td>
</tr>
<tr>
<td>Total Physical Used</td>
<td>34.82KB</td>
<td>0%</td>
</tr>
</tbody>
</table>
```

Related concepts

- How you can determine and control a volume’s space usage in the aggregate on page 160
- Methods to create space in an Infinite Volume or its aggregates on page 163

Methods to create space in an Infinite Volume or its aggregates

If an Infinite Volume is full, you can respond either by creating space in the Infinite Volume or by expanding it. If you want to expand the Infinite Volume but its aggregates are full, you can either create space in its aggregates or add storage.

Methods of creating space in an Infinite Volume

You can create space in an Infinite Volume in the following ways:

• Enable storage efficiency technologies, such as deduplication and compression.

• Reduce the size of the Snapshot reserve if the df command shows that the Snapshot reserve is not 100 percent full.
  This makes space available to the active file system.

• Delete volume Snapshot copies if the Snapshot reserve is 100 percent full and Snapshot copies are spilling into the active file system.

• Increase the size the Infinite Volume.
  You can expand the Infinite Volume if there is space available on the aggregates that the Infinite Volume uses or if you add storage capacity.

Methods of creating space in aggregates that an Infinite Volume uses

If an Infinite Volume shares aggregates with other volumes and the aggregates are full, you can create space in the aggregates in the following ways:

• Move the FlexVol volumes to other aggregates.
• Shrink the FlexVol volumes.
• Delete unneeded Infinite Volumes or FlexVol volumes.
• Use thin provisioning in other Infinite Volumes or FlexVol volumes that use thick provisioning.

Methods of adding storage to expand an Infinite Volume
You can add storage to an Infinite Volume in the following ways:
• Add disks to the aggregates that the Infinite Volume uses.
• Provide the Infinite Volume more aggregates to use by associating the Storage Virtual Machine (SVM) with Infinite Volume with more aggregates from existing nodes.
• Provide the Infinite Volume more nodes to use by associating the SVM with Infinite Volume with aggregates from additional nodes.

After adding more storage, you can resize the Infinite Volume so that it uses the added storage.

Related concepts
- How you can determine and control a volume's space usage in the aggregate on page 160
- How to determine space usage in an aggregate on page 162
- Expanding Infinite Volumes on page 168
- Expanding Infinite Volumes with storage classes on page 228

Unsupported space management capabilities for Infinite Volumes
Some of the space management capabilities that are used with FlexVol volumes are not supported for Infinite Volumes.

Infinite Volumes do not support the following space management capabilities:
• Reservations, which are also called space reservations, file reservations, or LUN reservations
• Fractional reserve
• Automatic free space preservation, which automatically resizes a volume or deletes Snapshot copies
• Autosizing functionality, including autogrow and autoshrink
• Volume fullness alerts

How an Infinite Volume distributes data across its constituents
When new files are added to an Infinite Volume, the files are distributed to data constituents in a manner that aims to balance the capacity on each data constituent. When files are distributed across data constituents, data also tends to be distributed across disks and nodes.

Data constituents are selected for incoming files in a weighted round-robin fashion with accommodations for nearly full and nearly empty constituents. In general, new files are created in the following manner:
• Data constituents with more than 80 percent used space generally do not receive incoming files.
• Data constituents with less than 80 percent used space each receive incoming files in sequence.
• Data constituents that are empty or nearly empty receive most of the incoming files.
When choosing which data constituent to write a file to, the Infinite Volume tries to create the file on multiple data constituents and returns an error only if none of the attempts succeeds.

**Note:** If an Infinite Volume uses storage classes, this capacity-balancing mechanism occurs within each storage class, not across the entire Infinite Volume.

An interval schedule named RepositoryBalanceMonitorJobSchedule is used to monitor the used space of all data constituents in an Infinite Volume to help choose which data constituents to write a file to. If a cluster administrator changes the interval schedule of RepositoryBalanceMonitorJobSchedule for a cluster by using the `job schedule interval modify` command, the new schedule applies to all Storage Virtual Machines (SVMs) with Infinite Volume in the cluster. You cannot have different interval schedules for RepositoryBalanceMonitorJobSchedule for each SVM with Infinite Volume in a cluster.

### When fullness warning messages are generated

Data ONTAP issues event messages with a severity level of warning when data constituents of an Infinite Volume are running out of space. You can take corrective action by rebalancing the used capacity of data constituents or by providing more space for the full data constituents.

The event messages are triggered at two thresholds: nearly full and full. The thresholds are based on percentage levels of used space in the data constituents and the number of data constituents that are full. You cannot configure or disable the Infinite Volume fullness thresholds.

An Infinite Volume is considered nearly full when one or more, but not all, data constituents have more than 80 percent used space. An Infinite Volume is considered full when all data constituents have more than 80 percent used space.

Similarly, a storage class of an Infinite Volume is considered nearly full when one or more, but not all, data constituents have more than 80 percent used space. A storage class is considered full when all data constituents have more than 80 percent used space.

The following table shows the fullness threshold, when the EMS event is triggered, the corrective action, and the name of the EMS event.

<table>
<thead>
<tr>
<th>Fullness threshold</th>
<th>Corrective action</th>
<th>EMS event name</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nearly full (volume)</td>
<td>• Rebalance the used capacity across all data constituents of an Infinite Volume.</td>
<td>cr.abv.volume.filling</td>
</tr>
<tr>
<td></td>
<td>• Provide more space for the Infinite Volume.</td>
<td></td>
</tr>
<tr>
<td>Full (volume)</td>
<td>• Provide more space for the Infinite Volume.</td>
<td>cr.abv.volume.full</td>
</tr>
<tr>
<td>Nearly full (storage class)</td>
<td>• Rebalance the used capacity across all data constituents in the storage class of the Infinite Volume.</td>
<td>cr.abv.service.filling</td>
</tr>
<tr>
<td></td>
<td>• Provide more space for the storage class.</td>
<td></td>
</tr>
<tr>
<td>Full (storage class)</td>
<td>• Provide more space for the storage class of the Infinite Volume.</td>
<td>cr.abv.service.full</td>
</tr>
</tbody>
</table>
Every hour, an event message is sent if a threshold for a data constituent is crossed for an Infinite Volume. The event messages appear on your system console and are written to the cluster's event log. You can view the event messages by using the `event log` commands. You can configure the Event Management System (EMS) to further control event messages.

**Related tasks**

- *Rebalancing the used capacity in an Infinite Volume* on page 187
- *Rebalancing the used capacity in a storage class of an Infinite Volume* on page 231

**Related information**

- *System administration*

### Maximum number of files an Infinite Volume can store

In most cases, an Infinite Volume can hold up to 2 billion files. If an Infinite Volume is relatively small, its maximum number of files might be less than 2 billion.

The maximum number of files that an Infinite Volume can hold is determined by the size of its namespace constituent. If the namespace constituent is 10 TB, the Infinite Volume can hold 2 billion files. If the namespace constituent is less than 10 TB, the Infinite Volume can hold proportionally fewer files.

The size of the namespace constituent is roughly proportional to the size of the Infinite Volume, depending on several factors, such as the namespace constituent's 10 TB maximum size, the available space in the aggregate that holds the namespace constituent, and the SnapDiff setting.

For a two-node Infinite Volume or a multi-node Infinite Volume without SnapDiff enabled, setting the Infinite Volume to a size of 80 TB or greater typically creates a namespace constituent of 10 TB.

The file count not only includes regular files, but also other file system structures, such as directories and symbolic links.

For information about the total files attribute and the files used attribute, you can use the `volume show` command on the Infinite Volume.

**Related tasks**

- *Monitoring the number of files in an Infinite Volume* on page 166

### Monitoring the number of files in an Infinite Volume

You can periodically compare the number of files that an Infinite Volume contains with the maximum number that it can store. If the Infinite Volume is nearing capacity, you can consider expanding the Infinite Volume.

**Steps**

1. Identify the number of files that the Infinite Volume currently contains by using the `volume show` command with the `-fields files,files-used` parameter.

2. Compare the value in the Total Files field, which represents the maximum number of files that the Infinite Volume can contain, to the value in the Files Used field.

   If the value in the Files Used is close to the value in the Total Files field, the Infinite Volume has almost reached the maximum number files that it can contain, given the existing resources.
3. Decide whether adding more resources to the Infinite Volume will increase the number of files that it can contain by comparing the value in the Total Files field with the 2 billion maximum value.

If the value in the Total Files field is less than 2 billion, you can increase the Total Files by expanding the Infinite Volume.

Related concepts

*Maximum number of files an Infinite Volume can store* on page 166
Expanding Infinite Volumes

You can expand an Infinite Volume by making more aggregate space available to the Infinite Volume, and then increasing the size of the Infinite Volume. When the Infinite Volume is in a data protection mirror relationship, you must expand both the source and the destination Infinite Volumes.

All methods of making more aggregate space available apply to Infinite Volumes with and without storage classes. However, what tool you use to resize the Infinite Volume depends on whether the Infinite Volume uses storage classes:

- You can resize an Infinite Volume by using the command-line interface or OnCommand Workflow Automation.
- You can resize an Infinite Volume with storage classes by using only OnCommand Workflow Automation.

Related concepts

Expanding Infinite Volumes with storage classes on page 228

How space is allocated when you expand an Infinite Volume

Several rules govern how space is allocated to constituents when an Infinite Volume is expanded. Understanding these rules can help you expand an Infinite Volume more effectively.

The following rules govern how space is allocated to constituents when an Infinite Volume is expanded:

1. Namespace-related constituents are created or expanded.
   a. If necessary, namespace mirror constituents are created.
      If a node has been added to the Infinite Volume and SnapDiff is enabled on the Infinite Volume, a namespace mirror constituent is created on the new node. This new namespace mirror constituent is the same size as the namespace constituent.
   b. If possible, the namespace constituent and the namespace mirror constituents are expanded.
      If the namespace constituent is not at the maximum size, its size is increased as much as possible within a specified maximum. At the same time, the namespace mirror constituents are increased by the same amount.

2. The new total data capacity is divided equally among the nodes that the Infinite Volume uses.
   When you add capacity to an Infinite Volume, its new total data capacity is the space remaining from the Infinite Volume's new total size after deducting the space required by the namespace-related constituents. The data capacity is allocated equally across nodes in the following way to ensure that data capacity remains balanced across nodes:
   - If the Infinite Volume uses the same number of nodes as it did previously, each node is allocated an equal amount of new data capacity.
   - If the Infinite Volume uses more nodes than it did previously, the new nodes receive a disproportionate amount of new data capacity so that their data capacity approaches the amount of data capacity that existing nodes have.

3. Within each node, individual data constituents are made as big as possible within a specified maximum.
Data constituents are always created as big as they are allowed to be within a specified maximum. Each time that Data ONTAP creates a data constituent, it evaluates all of the aggregates that the Infinite Volume uses on the node and selects the aggregate that has the most available space.

Related concepts

*Error messages and solutions for creating and expanding Infinite Volumes* on page 64

**How data protection mirror updates affect aggregate selection**

During manual or scheduled updates of a data protection mirror relationship between two Infinite Volumes, additional aggregate space is used, and sometimes new aggregates are automatically selected for a destination Infinite Volume.

When you increase the size of a source Infinite Volume, you must also increase the size of the destination Infinite Volume. The destination volume must be the same size or bigger than the source volume. You should increase the size of the destination Infinite Volume before you increase the size of the source Infinite Volume to avoid a scheduled data protection mirror update from starting before you finish increasing the size of the destination Infinite Volume.

When you increase the size of the source Infinite Volume, and new data constituents are created, aggregates must be selected on the destination Infinite Volume for mirror copies of the new data constituents. Before the next manual or scheduled update to the data protection mirror copy, you must increase the size of the destination volume to provide enough space for the mirror copies to transfer from the source to the destination Infinite Volume.

During manual or scheduled updates to the data protection mirror relationship, mirror copies of the namespace constituent and all data constituents—including new data constituents—are transferred to the destination volume, and aggregate space is selected for the new data constituents. New aggregates are automatically selected from the aggregate list assigned to the destination Storage Virtual Machine (SVM) with Infinite Volume. Alternatively, previously selected aggregates can be used for the new data constituents when the aggregates have enough space.

Note: During a data protection mirror update, constituents are not moved between different aggregates. When an aggregate is selected for a namespace constituent or a data constituent, the constituent remains on the aggregate.

Related concepts

*How Data ONTAP automatically selects aggregates for destination Infinite Volumes* on page 129
*How Data ONTAP calculates the Infinite Volume size before the transfer of a mirror copy* on page 127
*Constituent size considerations for data protection mirror relationships for Infinite Volumes* on page 32

**Determining which aggregates an Infinite Volume uses**

Before you expand an Infinite Volume, you must identify the aggregates that it uses. You can then evaluate the aggregates used by the Infinite Volume instead of evaluating all of the aggregates in the cluster.

**Step**

1. Determine the aggregates that are currently used by the Infinite Volume's constituents by using the `volume show` command with the `-vserver` and the `-is-constituent true` parameters.
Preparing to resize an Infinite Volume

You have several options for resizing an Infinite Volume, each of which requires a small amount of preparation.

Choices

- Determining space availability in an Infinite Volume's aggregates on page 170
- Adding disks to the aggregates that an Infinite Volume uses on page 171
- Associating more aggregates with an Infinite Volume on page 172
- Associating more nodes with an Infinite Volume on page 174

Determining space availability in an Infinite Volume's aggregates

One method of adding capacity to an Infinite Volume is to increase the Infinite Volume's size to use more of its existing aggregate space. However, before you resize the Infinite Volume, you must determine whether space is available on existing aggregates.

Before you begin

- You must know which aggregates are associated with the Infinite Volume.
- When the Infinite Volume is in a data protection mirror relationship, enough aggregate space must be available on both the source and destination clusters.
  If the Infinite Volume you are expanding is the source volume in a data protection mirror relationship, you must be prepared to expand the destination Infinite Volume before the next SnapMirror update.
- To view information about aggregates, you must have cluster administrator privileges.

About this task

- These steps apply only if the Infinite Volume or the storage class uses thick provisioning.
  If the Infinite Volume or storage class uses thin provisioning, you can increase its size even if the associated aggregates do not have enough available space.

Steps

1. Display information about used and available space on some or all of the Infinite Volume's aggregates by using the `storage aggregate show` command with the `-aggregate` and `-fields size,usedsize,availsize,percent-used` parameters.
Example

In the following output, three of the aggregates have between 40 TB and 70 TB of available space:

```
class1::> storage aggregate show -fields size,usedsize,availsize,percent-used -aggregate aggr1,aggr2,aggr3
aggregate availsize percent-used size   usedsize
--------- --------- ------------ ------ --------
aggr1     68.59TB   14%          79.76TB 11.17TB
aggr2     68.59TB   14%          79.76TB 11.17TB
aggr3     40.68TB   49%          79.76TB 39.08TB
3 entries were displayed.
```

2. Interpret the output with the following guidelines:

   • If the aggregates have available space, you can resize the Infinite Volume.

   • If the aggregates do not have available space, you need to consider adding disks or aggregates before you can expand the Infinite Volume.

Related tasks

- Resizing an Infinite Volume on page 175
- Resizing an Infinite Volume with storage classes on page 230
- Resizing an Infinite Volume by using OnCommand Workflow Automation on page 176
- Resizing source and destination Infinite Volumes on page 176

Adding disks to the aggregates that an Infinite Volume uses

One method of adding capacity to an Infinite Volume is to add disks to one or more of the aggregates that are already assigned to the Storage Virtual Machine (SVM) with Infinite Volume and then resize the Infinite Volume.

Before you begin

   • You must know which aggregates are associated with the Infinite Volume.

   • You must have decided the amount of capacity to add.
      The best practice is to add roughly an equal amount of capacity to each node that the Infinite Volume uses. For example, if you add 200 TB of disks to an aggregate on one node, you should add 200 TB of disks to an aggregate on all other nodes that the Infinite Volume uses.

   • When the Infinite Volume is in a data protection mirror relationship, enough aggregate space must be available on both the source and destination clusters.
      When you add disks to the aggregates used by the source Infinite Volume, you must also determine whether you should add disks to the aggregate used by the destination Infinite Volume. If the Infinite Volume you are expanding is the source volume in a data protection mirror relationship, you must be prepared to expand the destination Infinite Volume before the next SnapMirror update.

About this task

For an Infinite Volume with storage classes, you must add disks that meet the requirements of the storage class definitions. For storage class definitions, see OnCommand Workflow Automation.

Step

1. Add disks to one or more aggregates associated with the Infinite Volume.
After you finish
Resize the Infinite Volume to increase its use of existing aggregates.

Related tasks
Resizing an Infinite Volume on page 175
Resizing an Infinite Volume with storage classes on page 230
Resizing an Infinite Volume by using OnCommand Workflow Automation on page 176
Resizing source and destination Infinite Volumes on page 176

Related information
Disk and aggregate management

Associating more aggregates with an Infinite Volume

One method of adding capacity to an Infinite Volume is to assign additional aggregates from the Infinite Volume's existing nodes to the Storage Virtual Machine (SVM) with Infinite Volume and then resize the Infinite Volume.

Before you begin

- You must know which aggregates are associated with the Infinite Volume.
- You must have decided the amount of capacity to add.
  The best practice is to add roughly an equal amount of capacity to each node that the Infinite Volume uses. For example, if you add a 200 TB aggregate from one node, you should add a 200 TB aggregate from each other node that the Infinite Volume uses.
- The aggregates must already be created and added to the nodes.
- When the Infinite Volume is in a data protection mirror relationship, enough aggregate space must be available on both the source Infinite Volume and the destination clusters.
  When you associate more aggregates with the source Infinite Volume, you must also determine whether you should associate more aggregates with the destination Infinite Volume. If the Infinite Volume you are expanding is the source volume in a data protection mirror relationship, you must be prepared to expand the destination Infinite Volume before the next SnapMirror update.
- For an Infinite Volume with storage classes, you must add aggregates that meet the requirements of the storage class definitions. For storage class definitions, see OnCommand Workflow Automation.

About this task
This procedure applies only when the containing SVM with Infinite Volume has a specified aggregate list. If the aggregate list is unspecified—for example, if the entire cluster is dedicated to the Infinite Volume—you can skip this procedure.

Steps
1. Determine which nodes the Infinite Volume uses:
   a. Display all of the aggregates in the cluster by using the storage aggregate show -fields aggregate,nodes command.
   b. Compare the output with the list of aggregates assigned to the SVM with Infinite Volume.
Example

If aggr1 is used by the Infinite Volume, the following output indicates that node1 is used by the Infinite Volume:

```
cluster::> storage aggregate show -fields aggregate,nodes
aggregate nodes
---------- ------------
aggr0     node2
aggr0_node1 node1
aggr1     node1
aggr2     node2
aggr3     node1
...
```

2. Determine which aggregates are available on the nodes that the Infinite Volume currently uses:
   a. For each node that the Infinite Volume uses, display the aggregates on the node by using the `storage aggregate show` command with the `-nodes` and `-fields aggregate` parameters.
   b. In the output, identify the aggregates that the Infinite Volume does not currently use.

Example

The following output indicates aggr5, which is not already used by the Infinite Volume, is on node1:

```
cluster::> storage aggregate show -node node1 -fields aggregate
aggregate
---------------------
aggr0_node1
aggr1
aggr3
aggr5
...
```

3. Assign the new aggregates to the SVM with Infinite Volume by using the `volume add-aggregates` command.

   You should include all of the existing aggregates from existing nodes as well as the new aggregates from the new nodes.

Result

The aggregates are ready for the Infinite Volume to use the next time that the Infinite Volume is resized.

After you finish

Resize the Infinite Volume to use more aggregates.

Related tasks

- Resizing an Infinite Volume on page 175
- Resizing an Infinite Volume with storage classes on page 230
- Resizing an Infinite Volume by using OnCommand Workflow Automation on page 176
- Resizing source and destination Infinite Volumes on page 176
Associating more nodes with an Infinite Volume

One method of expanding an Infinite Volume is to assign aggregates from one or more new nodes to the Storage Virtual Machine (SVM) with Infinite Volume and then resize the Infinite Volume.

Before you begin

- You must know which aggregates are associated with the Infinite Volume.
- You must have decided the number of nodes to add and the amount of capacity to add from each node.
  You must not use more than 10 nodes for each Infinite Volume. For each new node, the best practice is to add aggregates that have a total size that roughly equals the amount of capacity that the Infinite Volume currently uses on each existing node. For example, if an Infinite Volume currently uses 500 TB on each node, you should add roughly 500 TB of aggregate space from each new node.
- The nodes must have been added to the cluster.
- You must know the names of the new nodes.
- When the Infinite Volume is in a data protection mirror relationship, enough aggregate space must be available on both the source Infinite Volume and the destination Infinite Volume.
  When you associate more nodes with the source Infinite Volume, you must also determine whether you should associate more nodes with the destination Infinite Volume. If the Infinite Volume you are expanding is the source volume in a data protection mirror relationship, you must be prepared to expand the destination Infinite Volume before the next SnapMirror update.
- For an Infinite Volume with storage classes, the aggregates on the new nodes must meet the requirements of the storage class definitions.
  For storage class definitions, see OnCommand Workflow Automation.

About this task

This procedure applies only when the containing SVM with Infinite Volume has a specified aggregate list. If the aggregate list is unspecified—for example, if the entire cluster is dedicated to the Infinite Volume—you can skip this procedure.

Steps

1. Determine which aggregates you will use from the new nodes:
   a. Identify the names of the aggregates on the new nodes by using the `storage aggregate show` command with the `-node` parameter.
   b. Decide which aggregates you will make available to the Infinite Volume.
       You can choose some or all. You must choose at least one aggregate from each node, or the Infinite Volume will not use the node.

2. Assign aggregates from the new nodes to the SVM with Infinite Volume by using the `vserver add-aggregates` command.
   You should include all the existing aggregates from existing nodes as well as the new aggregates from the new nodes.

Result

The aggregates are ready for the Infinite Volume to use the next time that the Infinite Volume is resized.
After you finish

Resize the Infinite Volume to use more nodes.

Related tasks

- Resizing an Infinite Volume on page 175
- Resizing an Infinite Volume with storage classes on page 230
- Resizing an Infinite Volume by using OnCommand Workflow Automation on page 176
- Resizing source and destination Infinite Volumes on page 176

Resizing an Infinite Volume

You can expand an Infinite Volume by setting its size to a larger value.

Before you begin

- The Infinite Volume must not contain storage classes.
  If you want to increase the size of an Infinite Volume with storage classes, you cannot use the command-line interface; you must use OnCommand Workflow Automation instead.
- The cluster must contain enough aggregate space to expand the Infinite Volume.

About this task

You can have cluster administrator or Storage Virtual Machine (SVM) administrator privileges to perform this task. However, if you require more aggregate space, you must have cluster administrator privileges to add and modify aggregates.

If SnapDiff is enabled to support tape backup, and you resize the Infinite Volume onto new nodes, there might be a delay before you can run operations on the Infinite Volume. After the resize operation finishes, Data ONTAP initializes the new namespace mirror constituent that it created on each new node.

During the initialization process, client operations continue unaffected, but you should not perform operations on the Infinite Volume. The length of time required for initialization to complete depends on the number of files that the Infinite Volume currently holds and the number of nodes that are being added. For example, if you add two nodes to an Infinite Volume that already contains 1 billion files, it might take hours for the initialization to complete.

If you also want to change the Infinite Volume's space guarantee or SnapDiff setting, you can do so before or after resizing the Infinite Volume. Changes to the size, space guarantee, or SnapDiff setting of an Infinite Volume must occur in separate operations.

Steps

1. Increase the size of the Infinite Volume by using the `volume modify` command with the `-size` parameter.

2. If the Infinite Volume is in a data protection mirror relationship, perform the following actions:
   a. Enter `y` when you see the following warning:

   Warning: This operation will expand the InfiniteVol "slv1" and potentially add constituents. Since this InfiniteVol is in a SnapMirror relationship the next "SnapMirror update" will fail unless the SnapMirror destination InfiniteVol is able to provision similar constituents.

   Are you sure you wish to proceed? {y|n}:

Expanding Infinite Volumes | 175
b. Expand the destination Infinite Volume by the same amount.

**Related concepts**

- How modifying an Infinite Volume affects namespace mirror constituents on page 145
- How namespace mirror constituents affect the state of an Infinite Volume on page 146

**Related tasks**

- Resizing an Infinite Volume by using OnCommand Workflow Automation on page 176
- Resizing an Infinite Volume with storage classes on page 230

### Resizing an Infinite Volume by using OnCommand Workflow Automation

When you want to increase the size of an Infinite Volume, you can use a workflow in OnCommand Workflow Automation. If the Infinite Volume is in a data protection mirror relationship, the workflow automatically increases the size of the source and destination Infinite Volumes at the same time.

**Before you begin**

- The Infinite Volume must not contain storage classes.
- You must have installed OnCommand Workflow Automation to perform this procedure.
- You must know how to use workflows in OnCommand Workflow Automation.
- The cluster must contain enough aggregate space to expand the Infinite Volume.
  
## Resizing source and destination Infinite Volumes

When Infinite Volumes are in a data protection mirror relationship, you must increase the size of both the source and destination volumes, and the data protection mirror copy must be updated to transfer mirror copies of existing and new constituents from the source to the destination volume.

**Before you begin**

- The Infinite Volume must not contain storage classes.
  
## About this task

You can have cluster administrator or Storage Virtual Machine (SVM) administrator privileges to perform this task. However, if you require more aggregate space, you must have cluster administrator privileges to add and modify aggregates.
You must increase the size of both the source and destination Infinite Volumes before the next scheduled update of the data protection mirror copy. Otherwise, the next scheduled update of the data protection mirror copy will fail. You can modify the schedule if required to allow time for increasing the size of the source and destination Infinite Volumes. You should increase the size of the destination Infinite Volume before you increase the size of the source Infinite Volume.

**Steps**

1. Increase the size of the destination Infinite Volume by using the `volume modify` command. The destination Infinite Volume must be equal to or larger than the source Infinite Volume.

   **Example**

   In the following example, a destination Infinite Volume named `repo_vol_dest` is increased to a size of 200 TB:

   ```
   cluster2::> volume modify -vserver vs0_dest -volume repo_vol_dest -size 200TB
   ```

   The size of the destination Infinite Volume is increased, but mirror copies of the constituents are not yet transferred from the source to the destination volume.

2. Increase the size of the source Infinite Volume by using the `volume modify` command.

   **Example**

   In the following example, a source Infinite Volume named `repo_vol` is increased to a size of 200 TB:

   ```
   cluster1::> volume modify -vserver vs0 -volume repo_vol -size 200TB
   ```

   The size of the source Infinite Volume and the existing constituents is increased. If required, new data constituents are created on new aggregates.

3. Perform one of the following actions:
   - Wait for the next scheduled update of the data protection mirror copy.
   - Manually update the data protection mirror copy.

   **Example**

   In the following example, the data protection mirror relationship is manually updated for the SVM with Infinite Volume named `vs0_dest:repo_vol_dest`:

   ```
   cluster2::> snapmirror update -destination-path vs0_dest:repo_vol_dest
   ```

   Mirror copies of the namespace constituent and all of the data constituents—including any new data constituents—are transferred from the source Infinite Volume to the destination Infinite Volume. Namespace mirror constituents are not transferred.

**Related concepts**

- *Error messages and solutions for failed aggregate selection for destination Infinite Volumes* on page 135
- *How data protection mirror updates affect aggregate selection* on page 169

**Related tasks**

- *Reenabling SnapDiff after expanding a destination Infinite Volume onto new nodes* on page 173
Resizing an Infinite Volume with storage classes on page 230

Reenabling SnapDiff after expanding a destination Infinite Volume onto new nodes

If you are using SnapDiff for incremental tape backup of a Snapshot copy on a destination Infinite Volume, you must manually reenable SnapDiff after you expand the destination volume onto a new node to automatically create namespace mirror constituents on the new node. Otherwise, tape backup operations might fail.

**Before you begin**

- The Infinite Volume must not contain storage classes.
- SnapDiff must be enabled on the destination Infinite Volume.
- You must have increased the size of the destination Infinite Volume to expand the volume onto one or more new nodes.
- You must have updated the data protection mirror relationship to transfer new data constituents from the source Infinite Volume to the new nodes for the destination Infinite Volume.
- An aggregate on each new node must contain enough space to fit a namespace mirror constituent. The aggregate should contain at least 10 TB of available space.

**Step**

1. On the destination Infinite Volume, reenable SnapDiff by using the `volume modify` command with the `-enable-snapdiff` parameter.

**Example**

In the following example, SnapDiff is reenabled for the Infinite Volume repo_vol:

```
cluster2::> volume modify repo_vol_dest -enable-snapdiff true
```

Namespace mirror constituents are automatically created on the new nodes for SnapDiff to use for incremental tape backup of the Infinite Volume.

Reenabling SnapDiff after expanding a destination Infinite Volume with storage classes onto new nodes

If you are using SnapDiff for incremental tape backup of a Snapshot copy on a destination Infinite Volume with storage classes, you must manually reenable SnapDiff and specify the aggregates to use after you expand the destination volume onto new nodes. This process creates the namespace mirror constituents required by SnapDiff for tape backup.

**Before you begin**

- SnapDiff must be enabled on the destination Infinite Volume with storage classes.
- You must have increased the size of the destination Infinite Volume to expand the volume onto one or more new nodes.
- You must have updated the data protection mirror relationship to transfer new data constituents from the source Infinite Volume to the new nodes for the destination Infinite Volume.
• You must know the name of the aggregate that you want to use for the namespace mirror constituent on each new node. Each node that contains a data constituent requires a namespace constituent or a namespace mirror constituent to support SnapDiff.

• The aggregate on each new node must contain enough space to fit a namespace mirror constituent. The aggregate should contain enough available space to fit at a minimum the size of the namespace constituent.

Steps

1. Switch to advanced privilege by using the `set -privilege advanced` command.

2. On the destination Infinite Volume, reenable SnapDiff and specify the aggregates on the new nodes to use for namespace mirror constituents by using the `volume modify` command with the `-enable-snapdiff` parameter.

Example

In the following example, the destination Infinite Volume with storage classes is expanded onto two new nodes. You have identified an aggregate on each node, aggr9 and aggr13, to contain the namespace mirror constituents. You reenable SnapDiff for the destination Infinite Volume with storage classes named repo_vol_dest and specify two aggregates (one on each new node):

```
cluster2::> volume modify repo_vol_dest -enable-snapdiff true -ns-mirror-aggr-list aggr9 aggr13
```

Namespace mirror constituents are created on the new nodes for SnapDiff to use for incremental tape backup of the destination Infinite Volume with storage classes.
Managing Infinite Volumes

When you manage an existing Infinite Volume, you can obtain information about active operations and events, rebalance the used capacity of data constituents, move constituents from an aggregate, create a new root volume, and obtain information about how the Infinite Volume is structured internally at the constituent level.

Related concepts

Managing Infinite Volumes with storage classes on page 231

Monitoring operations and events on Infinite Volumes

When you get information about operations and events on Infinite Volumes, you should understand how operations and events can apply to constituents versus the entire Infinite Volume so that you can get information you require.

How operations run on Infinite Volumes

Many volume-level operations run on Infinite Volumes in a different way than they do on FlexVol volumes. Understanding the unique characteristics of Infinite Volume operations can help you understand why the Infinite Volume is not available for subsequent operations and help you interpret error messages.

Characteristics of volume-level jobs on Infinite Volumes

Volume-level operations on Infinite Volumes have the following characteristics:

• They run as jobs even if the same operation is not a job on a FlexVol volume.
  For example, bringing a volume online is a job on an Infinite Volume, but is not a job on a FlexVol volume.

• They lock the Infinite Volume, preventing other volume-level operations from running until the first operation is complete.
  For example, if you change the Infinite Volume's size, you must wait for the job to finish before you can enable deduplication.

• They trigger separate operations on each constituent in the Infinite Volume.
  For example, if an Infinite Volume has 60 data constituents, a postprocess compression operation on the Infinite Volume triggers 60 postprocess compression operations on the data constituents. The volume-level job is not complete until all constituent-level jobs are complete.

How Infinite Volume jobs can succeed overall but fail on individual constituents

An operation on an Infinite Volume depends on the success of the operation on each constituent. Operations can fail on individual constituents—for example, if the constituents' containing aggregate is offline.

If the operation succeeds on some constituents but fails on one or more constituents, the operation succeeds with warnings. A message indicates that the job succeeded but that the operation did not succeed on one or more constituents.

For example, in the following output, efficiency is started on the Infinite Volume as a whole but is not started on one data constituent that is on an offline aggregate:
Job succeeded: Started the efficiency operation for volume "repo_vol" of Vserver "vs0" successfully.

Warning: Cannot modify the efficiency settings on one or more data constituents.

Constituent-level failures are reported to the Event Management System and are identified with the string mgmt.cr.constituent.failure.

**Volume-level operations that have unique characteristics on Infinite Volumes**

The volume-level operations that have some or all of these characteristics include the following operations:

- Changing a volume's state, such as taking it offline
- Efficiency operations, such as enabling deduplication and running postprocess compression operations
- Most Snapshot and SnapMirror operations, although some exceptions exist
- Aggregate relocation (ARL) operations
  ARL operations share some, but not all, of these characteristics.
  When you run ARL operations on an Infinite Volume, the operation must finish before you can run any other operations.

**Related concepts**

*How some Snapshot and SnapMirror commands run on Infinite Volumes* on page 98

**Related information**

*Disk and aggregate management*

**Viewing the status of efficiency operations on an Infinite Volume**

You can determine whether efficiency is enabled for the Infinite Volume as a whole or on individual data constituents. You can also view the status and progress of the efficiency operations on individual data constituents, although status and progress information is not available for the Infinite Volume as a whole.

**Choices**

- Use the `volume efficiency show` command with the `-volume` and `-instance` parameters to view the state of all types of efficiency—including deduplication, inline compression, and postprocess compression—for the Infinite Volume at a whole.

**Example**

The following command displays the overall efficiency state (which represents the deduplication state), the state of compression, and the state of inline compression for an Infinite Volume named repo_vol.

```
cluster1::> volume efficiency show -volume repo_vol -instance
Vserver Name: vs0
Volume Name: repo_vol
Volume Path: /vol/repo_vol
State: Enabled
Status: -
Progress: -
```
• Use the `volume efficiency show` command with the `-is-constituent true` parameter to view the state, status, and progress of efficiency on the individual data constituents of an Infinite Volume.

**Example**

The following command shows efficiency information about the data constituents in the Infinite Volume named repo_vol. The output shows that efficiency (specifically, deduplication) is enabled on all of them, efficiency operations are currently running on all of them, and the progress of postprocess efficiency operations on each data constituent.

```
cluster::> volume efficiency show -is-constituent true

<table>
<thead>
<tr>
<th>Vserver</th>
<th>Volume</th>
<th>State</th>
<th>Status</th>
<th>Progress</th>
<th>Policy</th>
</tr>
</thead>
<tbody>
<tr>
<td>vs0</td>
<td>repo_vol_1024_data0001</td>
<td>Enabled</td>
<td>Active</td>
<td>42756 KB Scanned</td>
<td>-</td>
</tr>
<tr>
<td>vs0</td>
<td>repo_vol_1024_data0002</td>
<td>Enabled</td>
<td>Active</td>
<td>27712 KB Scanned</td>
<td>-</td>
</tr>
<tr>
<td>vs0</td>
<td>repo_vol_1024_data0003</td>
<td>Enabled</td>
<td>Active</td>
<td>37890 KB Searched</td>
<td>-</td>
</tr>
</tbody>
</table>

...  
```

**Restrictions on the private namespace of an SVM with Infinite Volume**

You should not run management operations on or allow client access to the private namespace of a Storage Virtual Machine (SVM) with Infinite Volume.

The private namespace of the SVM with Infinite Volume is located under `/system`, which you should protect in the following ways:

• Do not perform any management operations on this area or configure it in any way.
  You should run operations only on the Infinite Volume as a whole.

• Ensure that clients never access this area.
  Clients must access each SVM with Infinite Volume only at the Infinite Volume's junction path or below.

This area contains junctions for each of the Infinite Volume's data constituents. While the junctions are hidden by default, they are visible using the `volume show` command with the `-junction` and `-is-constituent true` parameters. Showing constituent junctions might be helpful when you are troubleshooting—for example, if a file cannot be accessed. Data constituents do not have junctions if they go offline, for example, because their aggregate goes offline.

**Related concepts**

*Comparison of namespaces for Infinite Volumes and FlexVol volumes* on page 73

**Viewing information about events on Infinite Volume constituents (cluster administrators only)**

You can view information about events on Infinite Volume constituents, which is helpful if an operation on an Infinite Volume succeeds overall but fails on one or more constituents. By identifying the constituents where the operation failed, you can investigate possible issues with the aggregate that contains the affected constituents.

• View information about operations that failed on Infinite Volume constituents by using the `event log show` command with the `-event mgmt.cr.constituent.failure` parameter.
Example

```
cluster1:> event log show -event mgmt.cr.constituent.failure
Time                Node             Severity      Event
------------------- ---------------- ------------- ---------------------------
6/7/2012 10:11:48   cluster1-3     ERROR         mgmt.cr.constituent.failure:
Repository job "Start Volume Efficiency"
run on Infinite Volume "repo_vol" in Vserver "vs0" failed for constituent
volume "repo_vol_default_data0007": entry
doesn't exist
```

After you finish

You can use the constituent’s name to investigate why the operation failed—for example, by viewing the Infinite Volume’s constituents to identify which aggregate contains the affected constituents.

Rebalancing the used capacity of data constituents

Over time, some data constituents in an Infinite Volume might have more used capacity than other data constituents. You can use Auto Balance Volume to rebalance used capacity across data constituents in an Infinite Volume or across data constituents in a storage class of an Infinite Volume to maintain efficient operations.

Related concepts

- When fullness warning messages are generated on page 165

Related tasks

- Rebalancing the used capacity in a storage class of an Infinite Volume on page 231
- Viewing the names of storage classes on page 233

Commands for rebalancing used capacity of Infinite Volumes

You can use the Auto Balance Volume commands to rebalance the used capacity of data constituents in an Infinite Volume or data constituents in a storage class of an Infinite Volume. You can also view the progress of the operation and stop the operation if necessary.

When using Auto Balance Volume for an Infinite Volume with storage classes, you must specify a storage class by using the `autobalance volume rebalance` command with the `-storage-service` parameter.

<table>
<thead>
<tr>
<th>To...</th>
<th>Use this command...</th>
<th>For more information...</th>
</tr>
</thead>
<tbody>
<tr>
<td>View the used capacity of data constituents</td>
<td><code>volume show with the -is-constituent parameter</code></td>
<td></td>
</tr>
<tr>
<td>Start an Auto Balance Volume operation</td>
<td><code>autobalance volume rebalance start</code></td>
<td></td>
</tr>
<tr>
<td>View the progress of the Auto Balance Volume operation</td>
<td><code>autobalance volume rebalance show</code></td>
<td></td>
</tr>
<tr>
<td>Stop the Auto Balance Volume operation</td>
<td><code>autobalance volume rebalance stop</code></td>
<td></td>
</tr>
</tbody>
</table>
Related information

ONTAP 9 commands

Examples of how used capacity is rebalanced for Infinite Volumes

Reviewing examples of how data is moved between data constituents of an Infinite Volume or between data constituents of a storage class of an Infinite Volume helps you understand how Auto Balance Volume works.

When you perform the rebalance operation, data is moved between data constituents to rebalance used capacity. You can rebalance used capacity for all data constituents in an Infinite Volume without storage classes, or in all data constituents in a storage class of an Infinite Volume. When the operation finishes, the used capacity should be roughly equal across all data constituents in the Infinite Volume or across all data constituents in the storage class.

Note: Data is moved on a best-effort basis when the system is not busy. When the system is busy, priority is given to client access, not to data movement, and data is moved over time when the system is less busy.

Example of Auto Balance Volume and an Infinite Volume

In this example, an Infinite Volume contains the following data constituents:

- Data constituent 1 with a used capacity of 30%
- Data constituent 2 with a used capacity of 40%
- Data constituent 3 with a used capacity of 60%

When you run the `autobalance volume rebalance start` command for the Infinite Volume, data is moved from data constituent 3, which is 60% full, to data constituents 1 and 2, which are less full. The goal of the operation is to rebalance the used capacity for all data constituents in the Infinite Volume to be as even as possible.

The following diagram illustrates how used capacity is rebalanced for data constituents 1, 2, and 3 when you run Auto Balance Volume for an Infinite Volume:

![Example of Auto Balance Volume and an Infinite Volume](image)

Example of Auto Balance Volume and an Infinite Volume with storage classes

In this example, an Infinite Volume contains the following two storage classes:

- A gold storage class that contains the following data constituents:
  - Data constituent 1 with a used capacity of 30%
  - Data constituent 2 with a used capacity of 40%
  - Data constituent 3 with a used capacity of 60%
A silver storage class that contains the following data constituents:

- Data constituent 1 with a used capacity of 30%
- Data constituent 2 with a used capacity of 60%

When you run the `autobalance volume rebalance start` command for the gold storage class in the Infinite Volume, data is moved from data constituent 3, which is 60% full, to data constituents 1 and 2, which are less full. The operation rebalances used capacity only for the data constituents in the gold storage class.

If you also want to rebalance used capacity for the silver storage class, you must run the `autobalance volume rebalance start` command for the silver storage class too.

The following diagram illustrates how used capacity is rebalanced for a gold storage class that contains data constituents 1, 2, and 3 when you run Auto Balance Volume for the gold storage class:

![Diagram showing used capacity rebalancing](image)

**Scenarios where used capacity is unchanged**

In some situations, the used capacity of data constituents can remain unbalanced after the operation finishes, such as the following:

- Some data constituents contain very large files, and the large files cannot fit into other data constituents. As a result, the large files are not moved between data constituents.
- Some data constituents were offline when you ran the used-capacity rebalancing operation. As a result, the offline data constituents were excluded from the used-capacity rebalancing operation, leaving the used capacity unchanged for the offline data constituents.

If the used capacity of the data constituents remains unbalanced after the operation finishes, and the constituents are online, you can rerun the `autobalance volume rebalance start` command to try and further rebalance the used capacity.

**Note**: Files with locks are not moved between data constituents when you rebalance used capacity.
How offline data constituents affect the status of Auto Balance Volume operations

Offline constituents of an Infinite Volume affect how Auto Balance Volume operations run. You should understand how the operation works, how offline data constituents affect the status of the operation, and what actions you can perform.

When you use Auto Balance Volume by running the `autobalance volume rebalance start` command, the operation proceeds in two phases:

- **Planning phase**
  During this phase, all online data constituents in the Infinite Volume or all online data constituents in the storage class of the Infinite Volume are analyzed, and plans are made to move data from data constituents with a higher used capacity to data constituents with a lower used capacity. If a data constituent is offline during the planning phase, it is excluded from the planning phase and from the subsequent execution phase.

- **Execution phase**
  During this phase, the plans are used to move data between data constituents. If a data constituent goes offline during the execution phase, the status of the operation changes, depending on a number of factors.

The following table lists the possible statuses of the Auto Balance Volume operation and the actions you can perform:

<table>
<thead>
<tr>
<th>Status</th>
<th>Description</th>
<th>Actions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Active</td>
<td>All data constituents are online, and data is being moved.</td>
<td>No action required.</td>
</tr>
<tr>
<td>Partial</td>
<td>One or more data constituents went offline during the execution phase.</td>
<td>Without stopping the operation, you can bring the offline data constituents online. If you can bring them online in time, the data constituents might be included in the execution phase.</td>
</tr>
<tr>
<td>Blocked</td>
<td>One or more data constituents went offline during the execution phase, and the offline data constituents are blocking all movement of data between the data constituents. As a result, the operation cannot continue.</td>
<td>Without stopping the operation, you can bring the offline data constituents online. If you can bring them online in time, the data constituents might be included in the execution phase. However, if you cannot bring the data constituents online, the operation remains blocked, until you stop the operation by using the <code>autobalance volume rebalance stop</code> command.</td>
</tr>
<tr>
<td>Stopped</td>
<td>The operation was stopped by running the <code>autobalance volume rebalance stop</code> command.</td>
<td>You can start the operation again by running the <code>autobalance volume rebalance start</code> command. The operation starts at the planning phase.</td>
</tr>
<tr>
<td>Complete</td>
<td>The operation is completed.</td>
<td>No action required.</td>
</tr>
</tbody>
</table>
Viewing the used capacity of data constituents in Infinite Volumes

You can view the used capacity of data constituents in an Infinite Volume or that of data constituents in a storage class for an Infinite Volume to identify how full the data constituents are.

Choices

- View the used capacity for data constituents in an Infinite Volume without storage classes by using the `volume show` command with the `-is-constituent` parameter.

Example

In the following example, the Infinite Volume contains several data constituents, and the used capacity is reported for each data constituent:

```
cluster1::> volume show -is-constituent true -fields percent-used
vserver volume                 percent-used
------- ---------------------- ------------
vs0     repo_vol_1024_data0001 35%
vs0     repo_vol_1024_data0002 38%
vs0     repo_vol_1024_data0003 45%
vs0     repo_vol_1024_data0004 42%
vs0     repo_vol_ns            10%
vs0     repo_vol_ns_mirror0001 10%
6 entries were displayed.
```

- View the used capacity for data constituents in a storage class of an Infinite Volume:

  1. Switch to advanced privilege by using the `set -privilege advanced` command.

  2. Use the `volume show` command with the following parameters: `-is-constituent` and `-storage-service`.

Example

In the following example, the gold storage class contains four data constituents (repos_1025_data0001, repos_1025_data0002, repos_1026_data0001, repos_1026_data0002), and two of the data constituents are more than 69% full while two data constituents are less than 11% full:

```
cluster1::*> volume show -is-constituent true -fields percent-used, storage-service
vserver volume              percent-used storage-service
------- ------------------- ------------ 
vs0     repos_1024_data0001 5%            gold
vs0     repos_1025_data0001 70%           silver
vs0     repos_1025_data0002 75%           silver
vs0     repos_1026_data0001 5%            bronze
vs0     repos_1026_data0002 10%           bronze
vs0     repos_ns            10%           -
vs0     repos_ns_mirror0001 33%          -
7 entries were displayed.
```

Rebalancing the used capacity in an Infinite Volume

You can rebalance used capacity across all data constituents in an Infinite Volume by using Auto Balance Volume to maintain roughly equal amounts of used space in all data constituents.

Before you begin

- The Infinite Volume must not contain storage classes.
About this task

You can view the used capacity of data constituents before you start the operation. You should rebalance capacity in an Infinite Volume in the following situations:

- You increased the size of the Infinite Volume, and the operation created new data constituents. The new data constituents are empty, and the pre-existing data constituents contain data.
- You have used the Infinite Volume for some time and notice that some data constituents have greater used capacity than other data constituents.

You cannot rebalance the used capacity across data constituents of a destination Infinite Volume. You can only rebalance the used capacity across data constituents of a source Infinite Volume in a data protection mirror relationship.

Steps

1. Switch to advanced privilege by using the `set -privilege advanced` command.

2. Rebalance the used capacity between data constituents of the Infinite Volume by running the `autobalance volume rebalance start` command.

Example

In the following example, used capacity rebalancing is started for a Storage Virtual Machine (SVM) named vs0 and an Infinite Volume named repo_vol:

```
cluster1::*> autobalance volume rebalance start -vserver vs0 -volume repo_vol
Info: Auto Balance Volume operation started. Use the "autobalance volume rebalance show -vserver vs0 -volume repo_vol" command to view the status of this operation.
```

3. View the state of the operation by using the `autobalance volume rebalance show` command.

Example

In the following example, the state is `running`:

```
cluster1::*> autobalance volume rebalance show -vserver vs0 -volume repo_vol
Vserver: vs0
Volume: repo_vol
Storage Service State Transferred Target Transferred Percent
----------- ---------- ---------- --------- -----------
- running 14.18TB 47.28TB 30%
```

Related tasks

*Rebalancing the used capacity in a storage class of an Infinite Volume* on page 231
Moving constituents of an Infinite Volume from an aggregate

You can use a single command to move all constituents of an Infinite Volume from a source aggregate to a destination aggregate to prepare the source aggregate for decommissioning.

Before you begin

- You must have cluster administrator privileges to perform this task.
- The destination aggregate must be on a platform supported by Infinite Volumes.
- The destination aggregate must be assigned to the aggregate list for the Storage Virtual Machine (SVM) with Infinite Volume.
- The destination aggregate must have enough space to fit all of the constituents from the source aggregate.
- If the Infinite Volume uses storage classes, the destination aggregate must meet the storage class requirements of the data constituents being moved. The system does not validate that the destination aggregate meets the storage class requirements when you run the operation.

About this task

- The `volume aggregate vacate` command runs in the background, allowing you to perform additional operations.
- If the Infinite Volume is a source volume in a data protection mirror relationship, and you are initializing the data protection mirror relationship, you must wait until the `snapmirror initialize` command finishes before you run the `volume aggregate vacate` command. Otherwise the `snapmirror initialize` command will fail.
- The `volume aggregate vacate` command moves only constituents of an Infinite Volume from the aggregate; it does not move FlexVol volumes from the aggregate. You must use the `volume move` command to move FlexVol volumes from the aggregate that you want to decommission.

Steps

1. Move constituents of an Infinite Volume from a source aggregate to a destination aggregate by using the `volume aggregate vacate` command.

Example

In the following example, constituents of an Infinite Volume named repo_vol are moved from the source aggregate named aggr1 to the destination aggregate named aggr2:

```
cluster:/> volume aggregate vacate -vserver vs0 -volume repo_vol -source-aggregate aggr1 -destination-aggregate aggr2
```

All constituents for the specified Infinite Volume are moved from the source aggregate to the destination aggregate.

2. Repeat this procedure as required for other Infinite Volumes until no more constituents remain on the source aggregate.

3. Move any FlexVol volumes from the aggregate by using the `volume move` command.
How SnapDiff affects moving constituents from an aggregate

When you move constituents of an Infinite Volume with SnapDiff enabled from an aggregate, the command completes the operation only when the constituents can be moved from a source aggregate to a destination aggregate without affecting SnapDiff and tape backup.

If the operation cannot be completed because moving the constituents affects how SnapDiff works for incremental tape backup of Infinite Volumes, an error message is displayed, and you must contact technical support to complete the operation.

How the mixed state of an Infinite Volume affects its availability

Unlike other volumes, an Infinite Volume can go into a mixed state, which means its constituents are not all in the same state. When an Infinite Volume is in a mixed state, you cannot perform operations on it, and file access might be interrupted.

Causes of mixed state

Mixed state occurs whenever any constituent in the Infinite Volume has a state that differs from the state of other constituents.

A mixed state typically occurs when most constituents are online but one constituent is offline. For example, if you take an aggregate offline that contains constituents, you also cause the constituents to go offline.

An Infinite Volume cannot be set to a mixed state. Mixed is a read-only state.

The impact of mixed state on administrative operations

When an Infinite Volume is in a mixed or restricted state, you cannot run operations on the volume.

The impact of mixed state on file access

The impact of mixed state on file access depends on which constituent is unavailable, as explained in the following table:
<table>
<thead>
<tr>
<th>Type of constituent that is unavailable</th>
<th>Impact on file access</th>
</tr>
</thead>
<tbody>
<tr>
<td>The namespace constituent</td>
<td>Disruptive:</td>
</tr>
<tr>
<td></td>
<td>• If the node that contains the namespace constituent is not available, all file access is disrupted</td>
</tr>
<tr>
<td></td>
<td>• If the node that contains the namespace constituent is available, clients can continue accessing data that they recently accessed but cannot access new data</td>
</tr>
<tr>
<td>Data constituents</td>
<td>Partially disruptive:</td>
</tr>
<tr>
<td></td>
<td>• Data in the affected data constituents is unavailable</td>
</tr>
<tr>
<td></td>
<td>• Data in other data constituents is available</td>
</tr>
<tr>
<td>A namespace mirror constituent</td>
<td>No impact</td>
</tr>
</tbody>
</table>

**How to respond to a mixed state**

If an Infinite Volume is in a mixed state, the way that you respond depends on the cause.

You should view the state of all constituents to determine which one is in a restricted or offline state. You can view the state of the constituents in an Infinite Volume by running the `volume show` command with the `-is-constituent true` parameter.

If the mixed state is caused by a new namespace mirror constituent that is in restricted state while it is first being initialized, you do not need to respond. You can wait until the SnapMirror initialization finishes and the Infinite Volume automatically returns to an online state.

If a namespace constituent or data constituent is offline, you should investigate the cause and resolve the issue.

**Forcing constituents of an Infinite Volume into the same state**

When an aggregate used by an Infinite Volume is offline, you can force constituents of an Infinite Volume that are contained by the remaining online aggregates into the same state of online, offline, or restricted. You should then use the state of the constituents to determine the Infinite Volume’s availability.

**Before you begin**

An aggregate used by the Infinite Volume must be offline.

**About this task**

When an aggregate used by an Infinite Volume is offline, all constituents contained by the offline aggregate are unavailable, and the Infinite Volume displays a mixed state because its constituents have different states.

You can force constituents of the Infinite Volume that are contained by the online aggregates into the same state of online, offline, or restricted. However, the overall state of the Infinite Volume remains mixed because of the unavailable constituents contained by the offline aggregate. When an Infinite Volume is in a mixed state, you should use the state of the constituents to determine the current availability of the Infinite Volume.
Steps

1. View the state of the Infinite Volume by using the `volume show` command.

   **Example**

   The following output shows that the Infinite Volume `repo_vol` is in a mixed state:

   ```
   cluster1::> volume show -volume repo_vol
   Vserver       Volume       Aggregate    State      Type       Size
   Available Used%  --------- ------------ ------------ ---------- ---- ----------
   ---------- ----- ----- ----- ----- ----- ----- ----- ----- ----- ----- ----- ----
   vs0         repo_vol     -            mixed      RW        200TB
   190TB     5%
   Notice: The NVFAIL setting of some constituents does not match that of the Infinite Volume "repo_vol".
   ```

2. View the constituents of the Infinite Volume to better understand the mixed state by using the `volume show` command with the `-is-constituent true` parameter.

   **Example**

   The following output displays the constituents of the Infinite Volume `repo_vol`. Data constituent `repo_vol_1024_data0002` is offline, and data constituent `repo_vol_1024_data0003` is unavailable because of an offline aggregate:

   ```
   cluster1::> volume show -is-constituent true
   Vserver       Volume       Aggregate    State      Type       Size
   Available Used%  --------- ------------ ------------ ---------- ---- ----------
   ---------- ----- ----- ----- ----- ----- ----- ----- ----- ----- ----- ----- ----
   vs0         repo_vol_1024_data0001 aggr1    online     RW        100TB
   95TB      5%
   vs0         repo_vol_1024_data0002 aggr2    offline    RW
   100TB
   vs0         repo_vol_1024_data0003 aggr3    -          RW
   _          _          _          _          _          _          _          _
   vs0         repo_vol_1024_data0004 vs_aggr online     RW        100TB
   95TB      5%
   vs0         repo_vol_ns aggr1    online     RW        10TB
   9.5TB      5%
   vs0         repo_vol_ns_mirror0001 aggr2    online     DP        10TB
   9.5TB      5%
   6 entries were displayed.
   ```

3. Force the constituents of the Infinite Volume that are contained by online aggregates into an online, offline, or restricted state by using the appropriate command with the `-force` parameter.

   **Example**

   The following command forces constituents of the Infinite Volume `repo_vol` that are contained by online aggregates into an offline state. However, an error message is displayed for Infinite Volume `repo_vol` because the operation cannot force constituents contained by offline aggregates into an offline state. As a result, the operation succeeds on constituents contained by online aggregates, fails on constituents contained by offline aggregates, and fails on Infinite Volume `repo_vol` because the state of the volume cannot be changed from mixed to offline because its constituents have different states:
Warning: Forcing offline volume repo_vol in Vserver vs0 can potentially disrupt access to files in other volumes.
Do you want to continue? {y|n}: y

Notice: The NVFAIL setting of some constituents does not match that of the Infinite Volume "repo_vol".

cluster1::> volume show -is-constituent true

<table>
<thead>
<tr>
<th>Vserver</th>
<th>Volume</th>
<th>Aggregate</th>
<th>State</th>
<th>Type</th>
<th>Size</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>vs0</td>
<td>repo_vol_1024_data0001</td>
<td>aggr1</td>
<td>offline</td>
<td>RW</td>
<td>100TB</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>vs0</td>
<td>repo_vol_1024_data0002</td>
<td>aggr2</td>
<td>offline</td>
<td>RW</td>
<td>100TB</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>vs0</td>
<td>repo_vol_1024_data0003</td>
<td>aggr3</td>
<td>-</td>
<td>RW</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>vs0</td>
<td>repo_vol_1024_data0004</td>
<td>vs_aggr</td>
<td>offline</td>
<td>RW</td>
<td>100TB</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>vs0</td>
<td>repo_vol_ns</td>
<td>aggr1</td>
<td>offline</td>
<td>RW</td>
<td>10TB</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>vs0</td>
<td>repo_vol_ns_mirror0001</td>
<td>aggr2</td>
<td>offline</td>
<td>DP</td>
<td>10TB</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The state of the constituents determines the availability of an Infinite Volume in a mixed state. In this case, all constituents contained by online aggregates are offline, which means that the Infinite Volume is operating as an offline volume, although it is in a mixed state.

5. View the state of the Infinite Volume by using the `volume show` command.

Example

The following output shows that the Infinite Volume repo_vol is in a mixed state:

<table>
<thead>
<tr>
<th>Vserver</th>
<th>Volume</th>
<th>Aggregate</th>
<th>State</th>
<th>Type</th>
<th>Size</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>vs0</td>
<td>repo_vol</td>
<td>-</td>
<td>mixed</td>
<td>RW</td>
<td>200TB</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
The Infinite Volume is in a mixed state because its constituents have different states. When an Infinite Volume is in a mixed state, you should use the state of its constituents to determine the availability of the Infinite Volume.

**Related concepts**

*How the mixed state of an Infinite Volume affects its availability* on page 190

## Creating a new root volume on an SVM with Infinite Volume

If the root volume for a Storage Virtual Machine (SVM) with Infinite Volume becomes unavailable, you must create a new root volume on a specific aggregate to replace the unavailable root volume. You cannot promote an existing volume as the root volume.

**About this task**

When you create a new root volume for an SVM with Infinite Volume, the old, unavailable root volume is automatically deleted.

The new root volume is automatically created at a size that is appropriate for an SVM with Infinite Volume. For an SVM with Infinite Volume, the root volume is not required to be 1 GB in size.

**Steps**

1. Set advanced privilege by using the `set -privilege` command.

2. Create a new root volume for the SVM by using the `volume make-vsroot` command with the `-aggregate` parameter.
   
   A new root volume is created, and the old, unavailable root volume is deleted.

3. Mount the Infinite Volume by using the `volume mount` command.

---

The following example shows how to create a new root volume for the SVM with Infinite Volume vs0 and mount Infinite Volume repo_vol:

```
cluster1::> set -privilege advanced

Warning: These advanced commands are potentially dangerous; use them only when directed to do so by technical support.
Do you want to continue? {y|n}: y

cluster1::> volume make-vsroot -vserver vs0 -volume vs0_root1 -aggregate aggr1

Warning: Creating a new Vserver root volume will delete the old Vserver root volume "vs0_root".
Do you want to continue? {y|n}: y

cluster1::> volume show -volume vs0_root1 -instance

Vserver Name: vs0
Volume Name: vs0_root1
... Junction Path: / ...
Vserver Root Volume: true ...

cluster1::> volume mount -vserver vs0 -volume repo_vol -junction-path /NS -active true -policy-override false

[Job 69] Job succeeded: Volume "repo_vol" on Vserver "vs0" is
```
Viewing constituents of an Infinite Volume

You can view constituents of an Infinite Volume—and their containing aggregates—by using the `volume show` command with the `-is-constituent true` parameter. This enables you to identify the aggregates that contain each constituent when you are investigating issues or managing your physical storage.

**Choices**

- View only the Infinite Volume constituents by using the `volume show -is-constituent true` command.

**Example**

```bash
cluster1::> volume show -is-constituent true
Vserver   Volume       Aggregate    State      Type       Size  Available Used%
--------- ------------ ------------ ---------- ---- ---------- ---------- -----
vs0       repo_vol_1024_data0001 aggr3 online  RW        100TB       95TB    5%
vs0       repo_vol_1024_data0002 vs_aggr online RW       100TB       95TB    5%
vs0       repo_vol_1024_data0003 aggr4 online  RW        100TB       95TB    5%
...        ...
...        ...
vs0       repo_vol_ns aggr1 online    RW       10TB      9.5TB    5%
vs0       repo_vol_ns_mirror0001 aggr2 online DP       10TB      9.5TB    5%
100 entries were displayed.
```

- View both constituents and volumes by using the `volume show -is-constituent *` command.

**Example**

In the following output, the first two lines show the node root volumes, the third line shows the Infinite Volume, and the last line shows the Storage Virtual Machine (SVM) root volume:

```bash
cluster1::> volume show -is-constituent *
Vserver   Volume       Aggregate    State      Type       Size  Available Used%
--------- ------------ ------------ ---------- ---- ---------- ---------- -----
node1     vo10         aggr0_node1  online     RW       1.38TB     1.30TB    5%
node2     vo10         aggr0         online     RW       1.38TB     1.30TB    5%
vs0       repo_vol    -            online     RW       9.59PB     8.74PB    5%
vs0       repo_vol_1024_data0001 aggr3 online  RW        100TB       95TB    5%
vs0       repo_vol_1024_data0002 vs_aggr online RW       100TB       95TB    5%
vs0       repo_vol_1024_data0003 aggr4 online  RW        100TB       95TB    5%
...        ...
...        ...
vs0       repo_vol_ns aggr1 online    RW       10TB      9.5TB    5%
vs0       repo_vol_ns_mirror0001 aggr2 online DP       10TB      9.5TB    5%
vs0       vs0_root     vs_aggr      online     RW         20MB    18.87MB    5%
104 entries were displayed.
```
Related tasks

Viewing Snapshot copies of constituents of an Infinite Volume on page 99

Viewing constituents of a specific role in an Infinite Volume

Instead of viewing all the constituents of an Infinite Volume, you can view only constituents of a specific role by using the `volume show -constituent-role` command. For example, you can view only data constituents or namespace mirror constituents of an Infinite Volume.

Choices

- View only the data constituents of an Infinite Volume by using the `volume show -constituent-role` command:

  **Example**

  In the following example, only data constituents are displayed:

  ```
  cluster1::> volume show -constituent-role data
  Vserver  Volume       Aggregate    State      Type       Size
  Available Used%
  --------- ------------ ------------ ---------- ---- ----------
  ---------- ----- 
  vs0       repo_vol_1024_data0001 aggr1  online  RW 384MB
  364.7MB  5% 
  vs0       repo_vol_1024_data0002 aggr2  online  RW 384MB
  364.7MB  5% 
  vs0       repo_vol_1024_data0003 aggr3  online  RW 384MB
  364.7MB  5% 
  vs0       repo_vol_1024_data0004 vs_aggr online  RW 384MB
  364.7MB  5% 
  4 entries were displayed.
  ```

- View only the namespace constituent of an Infinite Volume by using the `volume show -constituent-role` command:

  **Example**

  In the following example, only the namespace constituent is displayed:

  ```
  cluster1::> volume show -constituent-role namespace
  Vserver  Volume       Aggregate    State      Type       Size
  Available Used%
  --------- ------------ ------------ ---------- ---- ----------
  ---------- ----- 
  vs0       repo_vol_ns  aggr1  online  RW 256MB
  243.1MB  5% 
  ```

- View only the namespace mirror constituent of an Infinite Volume by using the `volume show -constituent-role` command:

  **Example**

  In the following example, only the namespace mirror constituent is displayed:

  The number of namespace mirror constituents of an Infinite Volume depends on whether SnapDiff is enabled.

  **Example**

  In the following example, only the namespace mirror constituent is displayed:
### Naming convention for constituents

The constituents of an Infinite Volume are named automatically according to a naming convention that starts with the Infinite Volume's name followed by a suffix that indicates their role. The naming convention helps you distinguish namespace-related constituents from data constituents.

Constituents have the following naming convention:

<table>
<thead>
<tr>
<th>Constituent type</th>
<th>Naming convention</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Namespace constituent</td>
<td>$IV_{Name}_ns$</td>
<td>repo_vol_ns</td>
</tr>
<tr>
<td></td>
<td>$IV_{Name}$ is the name of the Infinite Volume.</td>
<td></td>
</tr>
<tr>
<td>Data constituent</td>
<td>$IV_{Name}_{SCID}<em>data</em>{DCID}$</td>
<td>repo_vol_1024_data_0001</td>
</tr>
<tr>
<td></td>
<td>• $SCID$ is the storage class ID.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• If the Infinite Volume does not use storage classes, the value is always 1024.</td>
<td>repo_vol_1024_data_0002</td>
</tr>
<tr>
<td></td>
<td>• $DCID$ is the data constituent index.</td>
<td>repo_vol_1024_data_0003</td>
</tr>
<tr>
<td>Namespace mirror</td>
<td>$IV_{Name}_{ns_mirr}_NMID$</td>
<td>repo_vol_ns_mirror_0001</td>
</tr>
<tr>
<td>constituent</td>
<td>$NMID$ is the namespace mirror constituent index.</td>
<td></td>
</tr>
</tbody>
</table>
Using Infinite Volumes with storage classes

You can use an Infinite Volume with storage classes to provide different levels of storage in one Infinite Volume and to automatically place data written to the Infinite Volume in specific storage classes. You must use OnCommand Workflow Automation and OnCommand Unified Manager to perform most of the operations.

Related concepts

- How Infinite Volumes use storage classes and data policies on page 15
- Tool requirements for Infinite Volumes with storage classes on page 23
- Deciding whether to use storage classes on page 24

Related tasks

- Setting up and managing an Infinite Volume with storage classes on page 27

Planning to create Infinite Volumes with storage classes

Before you create an Infinite Volume with storage classes, you should understand the default storage class definitions to learn how storage classes affect aggregate requirements, thin provisioning, efficiency settings, and tool requirements. You can create custom storage class definitions if necessary.

Definitions for the default storage classes

OnCommand Workflow Automation includes workflows for the default storage classes, performance and capacity. You should understand the definitions of the workflows for the default storage classes to determine whether you want to use the default storage classes or create storage classes with customized settings.

<table>
<thead>
<tr>
<th>Workflow name</th>
<th>Workflow definition for the storage class</th>
</tr>
</thead>
</table>
| Add or expand performance storage class to Infinite Volume | • Required aggregate type: SAS  
• Space guarantee setting: volume  
• Deduplication setting: enabled  
• Volume efficiency setting: QoS background  
• Volume Efficiency Policy: weekends 6 am to midnight |
| Add or expand capacity storage class to Infinite Volume | • Required aggregate type: SATA  
• Space guarantee setting: volume  
• Deduplication setting: enabled  
• Compression: enabled / inline  
• Volume Efficiency Policy: midnight to 6 am |
How storage classes affect which aggregates can be associated with Infinite Volumes

Each storage class definition specifies an aggregate type. When you create an Infinite Volume with a storage class, only the type of aggregate specified for the storage class can supply storage for the volume. You must understand storage class definitions to create aggregates that are appropriate for the storage class.

Storage class definitions are available only in OnCommand Workflow Automation. After you understand the aggregate requirements for each storage class, you can use the command-line interface or OnCommand Workflow Automation to create aggregates for storage classes. However, you must use OnCommand Workflow Automation, not the command-line interface, to create an Infinite Volume with one or more storage classes.

When you use OnCommand Workflow Automation to create an Infinite Volume with a storage class, OnCommand Workflow Automation automatically filters the aggregates available in the cluster based on the storage class that you want to use. If no aggregates meet the requirements of the storage class, you cannot create an Infinite Volume with that storage class.

How efficiency works on Infinite Volumes with storage classes

When an Infinite Volume uses storage classes, space-saving technologies are configured at the storage-class level to reflect the service-level objectives of each storage class. This affects the way that efficiency is configured and the way that efficiency information is displayed.

How efficiency is configured on an Infinite Volume with storage classes

Instead of configuring deduplication and compression technologies for the entire Infinite Volume, deduplication and compression settings are configured within the storage classes that an Infinite Volume uses.

For example, an Infinite Volume might have two storage classes—one for movie files and another for music files. While storage classes might have deduplication enabled, compression might be enabled only on the storage class for movie files. The storage class for music files might have a schedule that runs efficiency operations daily at off-peak hours, while the storage class for movie files might have a schedule that runs efficiency operations only on weekends.

To configure storage classes, you must use OnCommand Workflow Automation. You can use the command-line interface to create and modify efficiency policies, but you must use OnCommand Workflow Automation to apply an efficiency policy to a storage class.

How efficiency state is displayed for an Infinite Volume with storage classes

When an Infinite Volume uses storage classes, efficiency state information is not available at the Infinite Volume level. For an Infinite Volume with storage classes, the state of deduplication, compression, and in-line compression are displayed as a dash (-).

Instead, you can display efficiency state information for each individual data constituent.

How space savings information is displayed for an Infinite Volume with storage classes

When an Infinite Volume uses storage classes, space savings information is available both for the entire Infinite Volume and for individual data constituents.

When you view space savings information at the Infinite Volume level, you should keep in mind that the information reflects a summary of the savings across all data constituents in all storage classes. If efficiency technologies are not enabled on all storage classes, the savings for the entire Infinite Volume might be lower than you expect. For example, if one storage class has 50% space savings and
another storage class has 0% space savings (because efficiency technologies are disabled), the savings for the entire Infinite Volume are 25%.

**Related concepts**

*Definitions for the default storage classes* on page 198

**Commands disabled for Infinite Volumes with storage classes**

Some commands and parameters are disabled for an Infinite Volume with storage classes. You must use OnCommand Workflow Automation to perform the operations.

The following parameters for the `volume modify` command are disabled for an Infinite Volume with storage classes:

- `-size`
- `-space-guarantee`
- `-percent-snapshot-space`

Some `volume efficiency` commands are disabled for an Infinite Volume with storage classes.

Instead of using the commands, you must use OnCommand Workflow Automation and storage classes to define efficiency settings and space-guarantee settings for an Infinite Volume with storage classes. In addition, you must use OnCommand Workflow Automation to increase the size of an Infinite Volume with storage classes. You cannot use the `volume modify` or the `volume size` command to increase the size of an Infinite Volume with storage classes.

**Related references**

*Support for efficiency commands on Infinite Volumes with storage classes* on page 200

*This command is not supported for Infinite Volumes that are managed by storage services* on page 201

**Support for efficiency commands on Infinite Volumes with storage classes**

Commands for configuring efficiency are disabled on Infinite Volumes with storage classes. You can use commands that show volume-level efficiency information and work with efficiency policies, but you must use OnCommand Workflow Automation to set up efficiency on an Infinite Volume with storage classes.

<table>
<thead>
<tr>
<th>Supported on all Infinite Volumes</th>
<th>Not supported on Infinite Volumes with storage classes</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>volume efficiency revert-to</code></td>
<td><code>volume efficiency check</code></td>
</tr>
<tr>
<td><code>volume efficiency show</code></td>
<td><code>volume efficiency modify</code></td>
</tr>
<tr>
<td><code>volume efficiency policy create</code></td>
<td><code>volume efficiency off</code></td>
</tr>
<tr>
<td><code>volume efficiency policy delete</code></td>
<td><code>volume efficiency on</code></td>
</tr>
<tr>
<td><code>volume efficiency policy modify</code></td>
<td><code>volume efficiency start</code></td>
</tr>
<tr>
<td><code>volume efficiency policy show</code></td>
<td><code>volume efficiency stop</code></td>
</tr>
<tr>
<td></td>
<td><code>volume efficiency undo</code></td>
</tr>
</tbody>
</table>

The `volume efficiency stat` command is not supported on any Infinite Volume.
This command is not supported for Infinite Volumes that are managed by storage services

Message

Error: command failed: This command is not supported for Infinite Volumes that are managed by storage services. Use OnCommand Unified Manager to manage this Infinite Volume.

Cause

Some commands are disabled for Infinite Volumes with storage classes.

Corrective action

Use OnCommand Unified Manager or OnCommand Workflow Automation to perform the operation.

Related concepts

Commands disabled for Infinite Volumes with storage classes on page 200
How storage classes relate to storage services on page 201

How storage classes relate to storage services

A storage service is equivalent to a storage class in an Infinite Volume. Data ONTAP uses the term storage service, and OnCommand Workflow Automation uses the term storage class. If an Infinite Volume uses a storage class, it is considered managed by storage services, and you cannot use some commands for the Infinite Volume.

When you create an Infinite Volume without storage classes, the storage-service label in Data ONTAP displays a dash (-). An Infinite Volume without storage classes is considered not managed by storage services, and you can use Data ONTAP commands or OnCommand Unified Manager to monitor and manage the capacity of the Infinite Volume.

Note: Advanced privilege is required to view the storage-service label in Data ONTAP. For example, with advanced privilege, you can use the `volume show` command with the `-storage-service` parameter.

However, when you create an Infinite Volume with one or more storage classes, the storage-service label in Data ONTAP contains the names of the storage classes, and the Infinite Volume is considered managed by storage services. When an Infinite Volume is managed by storage services, some commands are disabled. Instead of using the commands, you must use OnCommand Workflow Automation to manage the storage classes, and you must use OnCommand Unified Manager to monitor and manage the capacity of the storage classes in the Infinite Volume. When you try to use a disabled command for an Infinite Volume with storage classes, an error message is displayed.

Note: When you upgrade an Infinite Volume from Data ONTAP 8.1.1 and later to Data ONTAP 8.2 and later, the Infinite Volume automatically contains an unnamed storage class. The unnamed storage class is not managed by storage services.

Related concepts

Commands disabled for Infinite Volumes with storage classes on page 200

Related tasks

Viewing the names of storage classes on page 233
**Related references**

- *This command is not supported for Infinite Volumes that are managed by storage services* on page 201
- *Support for efficiency commands on Infinite Volumes with storage classes* on page 200

**Tasks and tools for managing Infinite Volumes with storage classes**

You must use OnCommand Workflow Automation and OnCommand Unified Manager to perform most of the operations for Infinite Volumes with storage classes. However, you can use the command-line interface for some operations, such as creating aggregates or modifying a Storage Virtual Machine (SVM) with Infinite Volume.

<table>
<thead>
<tr>
<th>To...</th>
<th>Use this tool or command...</th>
<th>For more information...</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create storage classes by using the predefined storage class workflows</td>
<td>OnCommand Workflow Automation</td>
<td></td>
</tr>
<tr>
<td>Create storage classes by customizing predefined storage class workflows</td>
<td>OnCommand Workflow Automation</td>
<td></td>
</tr>
<tr>
<td>Create aggregates for storage classes</td>
<td>OnCommand Workflow Automation or the <code>storage aggregate</code> command</td>
<td></td>
</tr>
<tr>
<td>Create an Infinite Volume with one or more storage classes</td>
<td>OnCommand Workflow Automation</td>
<td></td>
</tr>
<tr>
<td>Edit a data policy for the SVM with Infinite Volume to add rules for filtering data into specific storage classes and to select storage classes</td>
<td>OnCommand Unified Manager</td>
<td></td>
</tr>
<tr>
<td>Create and initialize a data protection mirror relationship for an Infinite Volume with storage classes</td>
<td>OnCommand Workflow Automation</td>
<td></td>
</tr>
<tr>
<td>Monitor the capacity of storage classes for an Infinite Volume</td>
<td>OnCommand Unified Manager</td>
<td></td>
</tr>
<tr>
<td>Increase the size of existing storage classes for an Infinite Volume or add more storage classes to an Infinite Volume to increase capacity</td>
<td>OnCommand Workflow Automation</td>
<td></td>
</tr>
</tbody>
</table>
To... | Use this tool or command... | For more information...
---|---|---
Modify protocols and network settings for the SVM with Infinite Volume | vserver nfs modify, vserver cifs modify, or the network interface create command |  
Modify volume settings for storage classes, such as deduplication and efficiency settings | OnCommand Workflow Automation |  
Modify efficiency policy settings for storage classes | volume efficiency policy modify command |  

Related information

ONTAP 9 commands

Tools for modifying Infinite Volumes with storage classes

When you create an Infinite Volume with storage classes by using workflows in OnCommand Workflow Automation, default settings are used. However, you can further configure the Infinite Volume by using OnCommand System Manager or the command-line interface. You can also create custom workflows in OnCommand Workflow Automation.

For example, the “Create and configure a Storage Virtual Machine with Infinite Volume” workflow configures a Storage Virtual Machine (SVM) with Infinite Volume with specific versions of NAS protocols. If you want to use different versions of the NAS protocols, you can further configure the SVM with Infinite Volume by using OnCommand System Manager or the command-line interface.

How custom storage classes affect predefined workflows

When you use custom storage class settings for an Infinite Volume, you must modify some predefined workflows before you can use them. You must modify settings in the workflows to reflect the current storage class settings. Otherwise you cannot use the workflow for the custom storage classes.

When you use custom storage class settings for an Infinite Volume, you have to first modify the following workflows before using them:

- Add or expand capacity storage class to Infinite Volume
- Add or expand performance storage class to Infinite Volume

Before you can use the workflows for an Infinite Volume with custom storage class settings, you must clone the workflows and update the storage class settings. The storage class settings in the workflow must reflect the storage class settings in an Infinite Volume. Otherwise you cannot use the workflows. After you update the workflows to reflect settings for custom storage classes, you can use the workflows for an Infinite Volume with the same type of storage classes.

You must remember to modify workflows when you create an Infinite Volume with custom storage classes or when you modify an existing Infinite Volume to change its storage class settings.

Related tasks

Customizing predefined workflows for storage classes (basic) on page 204
Customizing predefined workflows for storage classes (advanced) on page 205
How customized predefined workflows support array LUNs

When you use array LUNs to provide storage for storage classes of an Infinite Volume, you must customize the predefined workflows distributed with OnCommand Workflow Automation to use a DISK_TYPE of LUN, and the array LUN must provide storage for only one storage class of the Infinite Volume.

When you use array LUNs to provide storage for an Infinite Volume without storage classes, you are not required to use OnCommand Workflow Automation or to customize the predefined workflows.

Related tasks

- Customizing predefined workflows for storage classes (basic) on page 204
- Customizing predefined workflows for storage classes (advanced) on page 205

Customizing predefined workflows for storage classes (basic)

You can create a clone of a storage class workflow to customize settings, such as thin provisioning, deduplication, and compression. You can also specify what type of aggregate disk to use. You can use this customization method for single-tier aggregates (aggregates composed of only HDDs or only SSDs).

Before you begin

- You must have installed OnCommand Workflow Automation to perform this procedure.
- You must know how to use workflows in OnCommand Workflow Automation.
- You cannot use this customization method for Flash Pool aggregates.

About this task

This is one way to create a workflow for a custom storage class. You can use other methods to create a workflow for a custom storage class. See OnCommand Workflow Automation.

The examples in this procedure describe how to customize the predefined workflow named “Add or expand capacity storage class to Infinite Volume”. It’s important to identify the name of the custom storage class because the data policy uses the name of the custom storage class to store data for incoming files to the correct storage class.

Steps

1. Clone a predefined workflow for storage classes, such as Add or expand capacity storage class to Infinite Volume or Add or expand performance storage class to Infinite Volume:
   a. Click Designer > Workflows.
   
   b. Select the workflow that you want to clone, and click on the toolbar.
   
   c. Click Setup.
   
   d. In the Setup Details dialog box, update the Workflow Description field, and type a name for the workflow in the Workflow Name field.
You should include the name of the storage class in workflow name and the workflow description.

2. Customize the name of the storage class and the type of disks for the storage class by editing constants:
   
a. Click the **Constants** tab, and type a new description and value for the DISK_TYPE parameters:
   
   For example, for the value of DISK_TYPE, you can type one of the following:
   
   •  'SATA'
   •  'SAS'
   
   b. For the label/tag of the storage class, change the name and the value.
   
   You must identify the name of the custom storage class because the data policy uses the name of the custom storage class when filtering data into the correct storage class. For example, to change a capacity storage class to a gold storage class, in the Name box, change CAPACITY_CLASS_LABEL to GOLD_CLASS_LABEL, and in the Value box, change 'capacity_class' to 'gold_class'.
   
   c. Click **OK**.

3. Customize the volume settings for the storage class by editing the Create Data Constituents command:
   
a. Under the command in the workflow named Create Data Constituents, click the commands.
   
   For example, click **new_constituent**.
   
   b. Set one or more of the following parameters:
      
      •  compression
      •  space_guarantee
      •  dedup_enabled
   
   c. Click **OK**, and then click **Yes** to confirm the changes.

4. Click **Save**.
   
   You have a workflow with custom storage class settings.

**Related concepts**

*How custom storage classes affect predefined workflows* on page 203

**Customizing predefined workflows for storage classes (advanced)**

You can create a clone of a storage class workflow to customize settings, such as thin provisioning, deduplication, and compression. You can also specify what type of aggregate disk to use. You can use this customization method for Flash Pool aggregates.

**Before you begin**

- You must use OnCommand Workflow Automation to perform this procedure.
- You must know how to use workflows in OnCommand Workflow Automation.
About this task

This is one way to create a workflow for a custom storage class. You can use other methods to create a workflow for a custom storage class. See OnCommand Workflow Automation.

Steps

1. Create a finder that selects the type of aggregates that you want for the storage class:
   a. Click Designer > Finders.
   b. Click on the toolbar.
   c. In the Name box, type a name for the finder.
   d. In the Type list, select Aggregate (cm_storage).
   e. On the Filters tab, move filters from the Available Filters list to the Selected Filters list.
      For example, move the filter named “Filter aggregates by disk type” to the Selected Filters list.
   f. Click Save.
      You have an aggregate finder.

2. Create a volume template with the settings that you want for the storage class:
   a. Click Designer > Templates.
   b. Click on the toolbar.
   c. In the Name box, type a name for the template.
   d. In the Type list, select Volume (cm_storage).
   e. In the Attributes area, set values for the following attributes:
      - compression
      - dedup_enabled
      - is_managed_by_service
      - space_guarantee
      - storage_class
   f. Click Save.
      You have a template for the volume settings.

3. Clone a predefined workflow for a storage class, such as Add or expand capacity storage class to Infinite Volume or Add or expand performance storage class to Infinite Volume:
   a. Click Designer > Workflows.
   b. Select the workflow that you want to clone, and then click on the toolbar.
   c. Click Setup.
   d. In the Setup Details dialog box, update the Workflow Description field, and then type a name for the workflow in the Workflow name field.
e. Click OK.

4. Select the volume template that you created:
   a. Under Create Data Constituents, click `new_constituent`.
   b. On the Constituent tab, in the Template list, select the volume template that you created.
   c. Click OK.

5. Select the finder that you created:
   a. At the start of row 4, click Edit row loop.
   b. In the Finder list, select the finder that you created.
   c. Click OK.

6. Click Save.

You have a workflow with custom storage class definitions.

Related concepts

*How custom storage classes affect predefined workflows* on page 203

**Creating aggregates for Infinite Volumes with storage classes**

If you want to create an Infinite Volume with storage classes, you must create aggregates that meet the storage class requirements. If you plan to create a mirror copy of the Infinite Volume, the aggregates for the destination Infinite Volume must also meet the storage class requirements.

**Before you begin**

- You must know what type of aggregate are required by the storage classes that you want to use. For storage class definitions, see OnCommand Workflow Automation.
- You must know which nodes you want to use in the cluster for each storage class, so you can create aggregates on the appropriate nodes.

**About this task**

If you plan to create a data protection mirror of the Infinite Volume in another cluster, you must create enough aggregate space in the destination cluster to fit a mirror copy of the Infinite Volume, and the aggregate disk type must meet the requirements of the storage classes used by the source Infinite Volume.

**Step**

1. Create aggregates of the correct type on the nodes that you want to use for each storage class in the Infinite Volume by using the `storage aggregate create` command.

**Related concepts**

*Node requirements for Infinite Volumes* on page 20
*Aggregate requirements for Infinite Volumes* on page 22
*Aggregate requirements for destination Infinite Volumes* on page 22
*How storage classes affect which aggregates can be associated with Infinite Volumes* on page 199
Converting to an Infinite Volume with storage classes

When you add a storage class to an Infinite Volume without storage classes by using a workflow in OnCommand Workflow Automation, the workflow converts the Infinite Volume without storage classes to an Infinite Volume with storage classes. You should be aware of how the workflow changes the Infinite Volume.

**Before you begin**

- You must have installed OnCommand Workflow Automation to perform this procedure.
- You must have created the type of aggregates required by the storage class on the nodes that you want to add to the Infinite Volume.
  
  For storage class definitions, see OnCommand Workflow Automation.
- If the Storage Virtual Machine (SVM) has an assigned aggregate list, you must have assigned any aggregates that you want to use to the aggregate list.
- You must be aware that you cannot convert from an Infinite Volume with storage classes to an Infinite Volume without storage classes.
  
  When you convert to an Infinite Volume with storage classes, the change is permanent.

**About this task**

When you add a storage class to an Infinite Volume without storage classes, aggregates are affected as follows:

- All aggregates used by the Infinite Volume before the conversion are not required to meet storage class definitions.
- All aggregates required for the new storage class must meet the requirements of the storage class that you want to use.

After you convert an Infinite Volume without storage classes to an Infinite Volume with storage classes, some commands in the command-line interface are disabled, and you must use OnCommand Workflow Automation and OnCommand Unified Manager to perform some operations.

**Steps**

1. Add a storage class to the Infinite Volume by using one of the following workflows in OnCommand Workflow Automation:
   
   - Add or expand capacity storage class to Infinite Volume
   - Add or expand performance storage class to Infinite Volume

   The workflow prompts you to create a name for the legacy storage class that will contain all of the existing data constituents. The workflow also adds a new storage class to the Infinite Volume and creates new data constituents for the storage class. The result is an Infinite Volume with two storage classes.

2. Modify the data policy for the SVM with Infinite Volume to add rules for filtering data into the storage classes and to select storage classes by using OnCommand Unified Manager.

   If you do not modify the data policy, all new content is stored in the legacy storage class. See the OnCommand Unified Manager Online Help.
3. Customize one of the following predefined workflows to specify the settings for the legacy storage class:
   • Add or expand capacity storage class to Infinite Volume
   • Add or expand performance storage class to Infinite Volume

Related tasks
   Customizing predefined workflows for storage classes (basic) on page 204
   Customizing predefined workflows for storage classes (advanced) on page 205

Creating an Infinite Volume with storage classes
You must use workflows in OnCommand Workflow Automation to create an Infinite Volume with storage classes. You must use several workflows in the correct order to complete the operation.

About this task
After you create an Infinite Volume with storage classes, you can further configure the Infinite Volume by using OnCommand System Manager or the command-line interface.

Steps
1. Creating an SVM and Infinite Volume with storage classes on page 209
2. Adding a storage class to an Infinite Volume on page 210
3. Configuring a Snapshot policy for an Infinite Volume with storage classes on page 211

Creating an SVM and Infinite Volume with storage classes
When you create a Storage Virtual Machine (SVM) with Infinite Volume, you can choose whether to enable storage classes for the Infinite Volume that the SVM will contain. If you want to create an Infinite Volume with storage classes, you must enable storage classes when you create the SVM with Infinite Volume.

Before you begin
• You must have installed OnCommand Workflow Automation to perform this procedure.
• You must know how to use workflows in OnCommand Workflow Automation.
• You must have created aggregates on the nodes that you want to use for the Infinite Volume, and the type of aggregate must meet the requirements of the storage classes that you want to use. If the Infinite Volume
• You must know the names of the aggregates that you want to use for all storage classes.

Steps
1. Create an SVM and an Infinite Volume with storage classes enabled by using the Create and configure a Storage Virtual Machine with Infinite Volume workflow with the following options in OnCommand Workflow Automation:
   a. Optional: Select the Use specific aggregates for the SVM check box, and select aggregates.
      If you select aggregates, you should select all of the aggregates on all of the nodes that you want to use for all of the storage classes.
   b. In the Infinite Volume Details area, in the Volume type list, select Primary (rw).
c. In the Infinite Volume Details area, select the **Create volume with classes of storage** check box.

The **Volume size (Tera Bytes)** option is disabled because you specify the size later when you add storage classes.

**Note:** The Snapshot policy is disabled. You can set up the Snapshot policy after you create the Infinite Volume with storage classes by using a workflow or by using the command-line interface.

2. Finish selecting the options, and click **Execute**.

An SVM with Infinite Volume is created, and an Infinite Volume is created, but no storage classes exist yet. You cannot store data until you add storage classes to the Infinite Volume.

### Adding a storage class to an Infinite Volume

You can add a storage class to an Infinite Volume by using a workflow in OnCommand Workflow Automation. You can add multiple different storage classes to an Infinite Volume, but you must add each storage class one at a time.

**Before you begin**

- You must have installed OnCommand Workflow Automation to perform this procedure.
- You must know how to use workflows in OnCommand Workflow Automation.
- You must have created aggregates in the cluster that meet the requirements of the storage class that you want to add to the Infinite Volume.
- If you specified an aggregate list when you created the Storage Virtual Machine (SVM) with Infinite Volume, you must have included in the list the aggregates that you want to use.
  
  Before you run the workflow, you can modify the aggregate list to specify aggregates by using the command-line interface.

**About this task**

The storage class definition includes settings for thin provisioning, deduplication, and compression. If you want to use different settings, you must customize a predefined workflow for storage classes.

**Steps**

1. Add a storage class to the Infinite Volume by using one of the following workflows in OnCommand Workflow Automation:
   
   - **Add or expand capacity storage class to Infinite Volume**
   - **Add or expand performance storage class to Infinite Volume**
   - A customized workflow for a storage class if you created one

   If the Infinite Volume is in a data protection mirror relationship, the storage class workflow automatically adds the same storage class to the source and destination Infinite Volumes. If the destination cluster lacks enough space for the new storage class, or if the available aggregates do not meet the requirements of the storage class, the workflow fails at the planning stage.

   The Infinite Volume contains the storage class. If the Infinite Volume is in a data protection mirror relationship, constituents for the storage classes are transferred during the next scheduled update to the data protection mirror.

2. Repeat this procedure to add another storage class to the Infinite Volume.
Related tasks

Assigning aggregates to an SVM with Infinite Volume on page 50

Configuring a Snapshot policy for an Infinite Volume with storage classes

When you create an Infinite Volume with storage classes, the Snapshot policy is disabled. After you create the Infinite Volume with storage classes, you can configure a Snapshot policy by using a workflow.

Before you begin

- You must have installed OnCommand Workflow Automation to perform this procedure.
- You must know how to use workflows in OnCommand Workflow Automation.
- You must have created a Snapshot policy.

About this task

Instead of using a workflow, you can configure the Snapshot policy for an Infinite Volume with storage classes by using the `volume snapshot policy` command.

Steps

1. Select a Snapshot policy for an Infinite Volume with storage classes by using the Setup data protection for an Infinite Volume workflow with the following options in OnCommand Workflow Automation:
   a. In the Primary Cluster list, select the cluster that contains the Infinite Volume with storage classes.
   b. If the Infinite Volume is in a data protection mirror relationship, you must select the source Infinite Volume; you cannot select the destination Infinite Volume.
   c. In the Primary Vserver list, select the Storage Virtual Machine (SVM).
   d. In the Snapshot Policy list, select the Snapshot policy.
   e. Clear the Create SnapMirror relationship check box.

2. Finish selecting the options, and click Execute.

The Snapshot policy is configured for the Infinite Volume with storage classes.

Configuring file access support for data policies

Rules in a data policy can filter data based on users or directory path. You must ensure that you configure users and directories before you modify the data policy to create rules that reference the users and directories. Otherwise you cannot activate the data policy.

How storage classes affect file access to Infinite Volumes

File access is configured in the same way for an Infinite Volume regardless of whether it uses storage classes. However, if an Infinite Volume uses storage classes, you must also create any users and directories that you plan to use in the Infinite Volume's data policy.

An Infinite Volume with storage classes supports the same file access and requires the same file access configuration as an Infinite Volume that does not use storage classes.
You can configure file access to an Infinite Volume with storage classes by using either the command-line interface or OnCommand Workflow Automation. Even if you use OnCommand Workflow Automation to create an Infinite Volume and configure file access, you can further configure file access by using the command-line interface.

For more information about data policies, see the *OnCommand Unified Manager Online Help*.

**Related concepts**

*Providing file access to Infinite Volumes* on page 71

**Configuring users or directories for rules in a data policy**

After you set up file access to an Infinite Volume with storage classes, you should configure any users or directories that rules in the data policy will reference. Otherwise, you cannot activate or import the data policy that references the users and directories.

**Before you begin**

You must have configured file access to the Infinite Volume.

**Step**

1. Configure the following functionality before you import or activate the data policy for the SVM with Infinite Volume:
   - Any users that you plan to use in rules that filter data based on file owner
   - Any directories that you plan to use in rules that filter data based on directory path
     Use a client to create directories in the Infinite Volume namespace.

   **Note:** If a policy uses the user ID and you do a lookup in the name-mapping table, the first mapping with this user ID is displayed. This mapping does not have any IP-qualifier configured and does not even have the IP-qualifier configured with 0.0.0.0/0.

**Related information**

*SMB/CIFS management*

*NFS management*

**Modifying the data policy for an SVM with Infinite Volume**

After adding two or more storage classes to an Infinite Volume, you should modify the data policy for a Storage Virtual Machine (SVM) with Infinite Volume to create rules that filter data written to the Infinite Volume into all the storage classes. Otherwise data might be stored in wrong storage class.

**Editing rules in a data policy for an SVM with Infinite Volume**

You should edit rules and select storage classes in a data policy any time that you add two or more storage classes to an Infinite Volume to ensure that data is filtered into the correct storage class. Otherwise data is not filtered into the correct storage classes.

**About this task**

Changes made to rules in a data policy affect only incoming data and not data that is already stored in the Infinite Volume.
You should use OnCommand Unified Manager to edit rules in a data policy for a Storage Virtual Machine (SVM) with Infinite Volume. You can also use the `vserver data-policy` command to edit rules in a data policy, but the commands require you to work with the data policy and its rules in JSON format.

**Steps**

1. Edit the rules and select storage classes in a data policy by using OnCommand Unified Manager.
2. Activate changes made to the data policy by using OnCommand Unified Manager.
   
   In OnCommand Unified Manager, you can choose when to activate the changes made to the data policy. For more information, see the [OnCommand Unified Manager Online Help](#).
3. If the Infinite Volume is in a data protection mirror relationship, export the updated data policy from the source SVM with Infinite Volume, and import the data policy to the destination SVM with Infinite Volume by using OnCommand Unified Manager.

   Data policies on the source and destination Infinite Volumes are identical.

**Managing JSON data policies for SVMs with Infinite Volume**

You should use OnCommand Unified Manager to create and modify rules for a data policy associated with a Storage Virtual Machine (SVM) with Infinite Volume. Commands are supported, but not recommended because the commands require you to work with the data policy in JSON format.

**Commands for managing JSON data policies for SVMs with Infinite Volume**

You can use the `vserver data-policy` commands to import, export, and validate data policies in JSON format for Storage Virtual Machines (SVMs) with Infinite Volume.

**Note:** The `vserver data-policy` commands are not supported for SVMs with FlexVol volumes.

<table>
<thead>
<tr>
<th>To...</th>
<th>Use this command...</th>
<th>For more information...</th>
</tr>
</thead>
<tbody>
<tr>
<td>Export a data policy from the SVM with Infinite Volume in JSON format</td>
<td><code>vserver data-policy export</code></td>
<td></td>
</tr>
<tr>
<td>Ensure the JSON in a data policy is valid before importing the data policy to the SVM with Infinite Volume</td>
<td><code>vserver data-policy validate</code></td>
<td></td>
</tr>
<tr>
<td>Import a data policy in JSON format to the SVM with Infinite Volume</td>
<td><code>vserver data-policy import</code></td>
<td></td>
</tr>
</tbody>
</table>

**Related information**

*ONTAP 9 commands*
Importing JSON data policies to SVMs with Infinite Volume

You can manually import a data policy in JSON format into Storage Virtual Machines (SVMs) with Infinite Volume to ensure that the SVM with Infinite Volume has the latest data policy. The imported data policy immediately starts filtering data written to the Infinite Volume.

Before you begin

• The Infinite Volume contained by the SVM must be mounted.
• The data policy must be in JSON format, and the JSON must be valid as defined for data policies.
• The data policy must contain a rule named *default*, and the default rule must be the last rule in the data policy.
• If the data policy contains rules that filter data based on directory location, the referenced directories must exist in the Infinite Volume's namespace before you import the data policy.
• If the data policy contains rules that filter data based on users, the user names must exist in the SVM with Infinite Volume or the name service before you import the data policy. For example, if Data ONTAP uses Windows Active Directory authentication to resolve user names, the user name must exist in Windows Active Directory before you import the data policy.

About this task

The entire existing data policy on the SVM with Infinite Volume is replaced when you import a data policy.

If the Infinite Volume is in a data protection mirror relationship, you must import the changed data policy into the source and the destination SVMs with Infinite Volume to ensure that both SVMs reference the same data policy.

Steps

1. Import the data policy into the SVM with Infinite Volume by using the `vserver data-policy import` command.

   The data policy is imported and is active for any new data written to the Infinite Volume. The updated data policy does not affect existing data in the Infinite Volume.

2. If the Infinite Volume is in a data protection mirror relationship, import the data policy to the destination SVM with Infinite Volume by using the `vserver data-policy import` command.

   The data policies on the source and destination Infinite Volumes are identical.

Related concepts

*Considerations for valid JSON formatting in data policies* on page 214

Considerations for valid JSON formatting in data policies

Data policies are in JavaScript Object Notation (JSON) format, and the JSON format must be valid as defined for data policies. You should be aware of what makes JSON formatting valid for data policies.

You should use OnCommand Unified Manager to edit data policies. OnCommand Unified Manager uses valid JSON formatting for data policies.

Consider the following criteria when working with data policies in JSON format:

• A data policy supports a maximum of 100 rules.
• A rule in a data policy supports a maximum of 50 conditions.

• The `storageservice` key identifies the name of the storage class.
  You must use OnCommand Workflow Automation to create storage classes that data policies reference.

  **Note:** The `storageservice` key contains a dash (−) when you create an Infinite Volume without storage classes, or when you upgrade an Infinite Volume created in Data ONTAP 8.1.1 and later to Data ONTAP 8.2 and later.

• The `parentURI` key must end in a forward slash (/).
  The `parentURI` key uses the following format: `/namespace name/directory name/`—for example, `/NS/users/alice/`.

• The `parentURI` key supports scope matching with `==`, `!=`, `starts`, and `!starts`.

  **Note:** The `parentURI` key does not support `ends`/`!ends`.

• The `metadata/cdmi_owner` key identifies the owner of a file.
  For example, the CIFS protocol uses a `user name` to identify the owner of a file.

• The `metadata/cdmi_owner` key supports scope matching with `==` and `!=`.

• The `metadata/cdmi_owner` key must contain a user name that is equal to or less than 64 characters.

**Related information**

[JSON (JavaScript Object Notation)](https://www.json.org/)


**Example of JSON data policy: filtering based on directory**

A data policy in JSON format can filter data written to the Infinite Volume into different storage classes based on the directory location of the file.

The data policy filters data as follows:

1. If the file is created in or beneath `/NS/users/alice`, store the data in the gold storage class.
2. If the file is created in and not beneath `/NS/users/bob/important`, store the data in the gold storage class.
3. If the file is created in or beneath `/NS/users/bob`, store the data in the silver storage class.
4. Otherwise, store the data in the bronze storage class.

Following is an example of the JSON for the data policy:

```json
{
   "ruleset_format_version" : "1.0",
   "rules" : [
      {
         "rule_label" : "Alice’s stuff",
         "rule_id" : "63cd19f1-74d4-4754-8aca-e0223f6c3923",
         "rule_scope" : [ { "parentURI" : "starts /NS/users/alice/" } ],
         "rule_epoch" : { "epoch_reference" : "cdmi_ctime" },
         "rule_epocs" : {
            "0" : {
               "local" : {
                  "metadata" : {
                     "storageservice" : "gold"
                  }
               }
            }
         }
      }
   ]
}
```
Example of JSON data policy: filtering based on file name

A data policy in JSON format can filter data written to the Infinite Volume into different storage classes based on file name.

The data policy filters data as follows:

1. If the file name ends in `.doc` or `.xls`, store its data in the gold storage class.
2. If the file name ends in `.mp3`, `.wav`, or `.ogg`, store its data in the silver storage class.
3. Otherwise, store its data in the bronze storage class.
Following is an example of the JSON for the data policy:

```json
{
    "ruleset_format_version": "1.0",
    "rules": [
        {
            "rule_label": "office stuff",
            "rule_id": "63cd19f1-74d4-4754-8aca-e0223f6c3923",
            "rule_scope": [
                { "objectName": "ends .doc" },
                { "objectName": "ends .xls" }
            ],
            "rule_epoch": { "epoch_reference": "cdmi_ctime" },
            "rule_epochs": {
                "0": {
                    "local": {
                        "metadata": {
                            "storageservice": "gold"
                        }
                    }
                }
            }
        },
        {
            "rule_label": "media stuff",
            "rule_id": "7c007dea-60ee-41d2-9f64-88e18637941c",
            "rule_scope": [
                { "objectName": "ends .mp3" },
                { "objectName": "ends .ogg" },
                { "objectName": "ends .wav" }
            ],
            "rule_epoch": { "epoch_reference": "cdmi_ctime" },
            "rule_epochs": {
                "0": {
                    "local": {
                        "metadata": {
                            "storageservice": "silver"
                        }
                    }
                }
            }
        },
        {
            "rule_label": "default",
            "rule_id": "00fc0534-10dd-7812-ec8c-a9356fa8cd00",
            "rule_scope": [],
            "rule_epoch": { "epoch_reference": "cdmi_ctime" },
            "rule_epochs": {
                "0": {
                    "local": {
                        "metadata": {
                            "storageservice": "bronze"
                        }
                    }
                }
            }
        }
    ]
}
```

**Example of JSON data policy: filtering based on file owner**

A data policy in JSON format can filter data written to the Infinite Volume into different storage classes based on the owner of the file.

The data policy filters data as follows:

1. If Alice owns the file, store the data in the gold storage class.
2. If Bob owns the file, store the data in the silver storage class.
3. If users other than Alice and Bob own the file, store the data in the bronze storage class.
4. Otherwise, store the data in the bronze storage class.

Following is an example of the JSON for the data policy:

```json
{
  "ruleset_format_version" : "1.0",
  "rules" : [
    {
      "rule_label" : "Alice's stuff",
      "rule_id" : "63cd19f1-74d4-4754-8aca-e0223f6c3923",
      "rule_scope" : [
        { "metadata" : { "cdmi_owner" : "== alice" } }
      ],
      "rule_epoch" : { "epoch_reference" : "cdmi_ctime" },
      "rule_epochs" : {
        "0" : {
          "local" : {
            "metadata" : {
              "storageservice" : "gold"
            }
          }
        }
      }
    },
    {
      "rule_label" : "Bob's stuff",
      "rule_id" : "7c007dea-60ee-41d2-9f64-88e18637941c",
      "rule_scope" : [
        { "metadata" : { "cdmi_owner" : "== bob" } }
      ],
      "rule_epoch" : { "epoch_reference" : "cdmi_ctime" },
      "rule_epochs" : {
        "0" : {
          "local" : {
            "metadata" : {
              "storageservice" : "silver"
            }
          }
        }
      }
    },
    {
      "rule_label" : "If not user1, user2, or user3",
      "rule_id" : "963fcb66-1d1c-464e-b08a-6b8b473fac8a",
      "rule_scope" : [
        { "metadata" : { "cdmi_owner" : "!= user1" } },
        { "metadata" : { "cdmi_owner" : "!= user2" } },
        { "metadata" : { "cdmi_owner" : "!= user3" } }
      ],
      "rule_epoch" : { "epoch_reference" : "cdmi_ctime" },
      "rule_epochs" : {
        "0" : {
          "local" : {
            "metadata" : {
              "storageservice" : "bronze"
            }
          }
        }
      }
    },
    {
      "rule_label" : "default",
      "rule_id" : "00fc0534-10dd-7812-ec8c-a9356fa8cd00",
      "rule_scope" : [],
      "rule_epoch" : { "epoch_reference" : "cdmi_ctime" },
      "rule_epochs" : {
      }
    }
  ]
}
```
Providing backups of Infinite Volumes with storage classes using Snapshot copies

You can use Snapshot copies to provide backups of Infinite Volumes with storage classes.

How storage classes affect Snapshot copies of Infinite Volumes

Snapshot copy creation and management is the same for an Infinite Volume with or without storage classes. You must create Snapshot copies for the entire Infinite Volume. You cannot create Snapshot copies for individual storage classes in Infinite Volumes.

Related concepts

- Providing backups of Infinite Volumes using Snapshot copies on page 95

Related tasks

- Configuring a Snapshot policy for an Infinite Volume with storage classes on page 211

Providing disaster recovery on Infinite Volumes with storage classes using mirroring technology

You can use mirroring technology on Infinite Volumes with storage classes to create an identical second set of data to replace the primary set of data, in case something happens to the primary set of data. However, you must use OnCommand Workflow Automation to perform most operations.

Tasks and tools for managing data protection mirror copies for Infinite Volumes with storage classes

You must use OnCommand Workflow Automation and OnCommand Unified Manager to perform operations for Infinite Volumes with storage classes in a data protection mirror relationship.

<table>
<thead>
<tr>
<th>To...</th>
<th>Use this tool or command...</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create and initialize data protection mirror relationships for an Infinite Volume with one or more storage classes</td>
<td>OnCommand Workflow Automation</td>
</tr>
<tr>
<td>Manually transfer the data policy for the source Infinite Volume with storage classes to the destination Infinite Volume with storage classes</td>
<td>OnCommand Unified Manager</td>
</tr>
</tbody>
</table>
Creating a data protection mirror copy for an Infinite Volume with storage classes

You can create a data protection mirror copy for an Infinite Volume with storage classes in another cluster by creating a destination Infinite Volume and initializing a data protection mirror relationship. You should also export the data policy from the source Infinite Volume to the destination Infinite Volume.

Before you begin
You must understand the aggregate requirements for a destination Infinite Volume.

About this task
You cannot create a data protection mirror copy between an Infinite Volume without storage classes and an Infinite Volume with storage classes.

Steps
1. Creating an SVM and destination Infinite Volume with storage classes on page 220
2. Creating and initializing a data protection mirror relationship on page 221
3. Exporting a data policy to a destination SVM with Infinite Volume on page 222

Related concepts
Aggregate requirements for destination Infinite Volumes on page 22

Creating an SVM and destination Infinite Volume with storage classes

You can create a Storage Virtual Machine (SVM) with Infinite Volume and a destination Infinite Volume that is ready to contain one or more storage classes in another cluster.

Before you begin
• You must have installed OnCommand Workflow Automation to perform this procedure.
• You must know how to use workflows in OnCommand Workflow Automation.
• You must have created enough aggregate space on the nodes in the destination cluster to fit a mirror copy of the Infinite Volume with storage classes, and the aggregate disk type must meet the requirements of the storage classes used by the source Infinite Volume. For storage class definitions, see OnCommand Workflow Automation.
• You must know the names of the aggregates that you want to use for all storage classes in the destination cluster.

Steps
1. Create the SVM and a destination Infinite Volume in another cluster by using the Create and configure a Storage Virtual Machine with Infinite Volume workflow with the following options in OnCommand Workflow Automation:
a. Optional: Select the **Use specific aggregates for the SVM** check box, and select aggregates. If you select aggregates, you should select all of the aggregates on all of the nodes that you want to use for all of the storage classes.

b. In the Infinite Volume Details area, in the Volume type list, select **Secondary (DP)**.

c. In the Infinite Volume Details area, select the **Create volume with classes of storage** check box.

   The **Volume size (Tera Bytes)** option is disabled. The volume size is automatically calculated based on the size of the source Infinite Volume with storage classes.

2. Finish selecting the options, and click **Execute**.

   The SVM with Infinite Volume is created, and a destination Infinite Volume is created, but no storage classes exist yet.

**Creating and initializing a data protection mirror relationship**

You must create and initialize a data protection mirror relationship for an Infinite Volume with storage classes by using a workflow in OnCommand Workflow Automation. You can also use the workflow to optionally configure a Snapshot policy or enable SnapDiff for the source Infinite Volume.

**Before you begin**

- You must have installed OnCommand Workflow Automation to perform this procedure.
- You must know how to use workflows in OnCommand Workflow Automation.
- The destination cluster must contain enough aggregate space to mirror constituents from the source Infinite Volume to the destination Infinite Volume, and the aggregates must meet the requirements of the storage classes used by the source Infinite Volume.
- The intercluster LIF must be created on both the source and destination clusters to allow the workflow to automatically set up a cluster peer relationship.

   You can create the intercluster LIF by using a workflow in OnCommand Workflow Automation or by using the command-line interface.

**About this task**

The “Set up data protection for an Infinite Volume” workflow automatically creates the cluster peer and Storage Virtual Machine (SVM) peer relationships that the data protection mirror relationship requires.

**Note:** If a cluster peer relationship exists between the two clusters, the workflow skips the commands for creating a cluster peer relationship, and moves to the next set of commands configured in the workflow.

**Steps**

1. Configure several settings on the source Infinite Volume and create a data protection mirror relationship for the source Infinite Volume by using the **Setup data protection for an Infinite Volume** workflow by setting the following options:

<table>
<thead>
<tr>
<th>To...</th>
<th>Do this...</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create a data protection mirror relationship</td>
<td>In the Primary Volume To Protect area, select the <strong>Create SnapMirror relationship</strong> check box, and select the options in the SnapMirror Relationship Settings area.</td>
</tr>
</tbody>
</table>
To... | Do this...
---|---
Optional: Set up a Snapshot policy on the source Infinite Volume | In the Primary Volume To Protect area, in the *Snapshot policy* box, select the Snapshot policy that you want to assign to the source Infinite Volume.
Optional: Create namespace mirror constituents on the source Infinite Volume to support SnapDiff and incremental tape backup | In the Primary Volume To Protect area, select the *Enable tape backups* check box.

2. Finish selecting the options, and click **Execute**.

A data protection mirror relationship is created and initialized. Storage classes from the source Infinite Volume are mirrored to the destination Infinite Volume.

## Exporting a data policy to a destination SVM with Infinite Volume

The destination Storage Virtual Machine (SVM) with Infinite Volume should always reference the same data policy as the source SVM with Infinite Volume. Therefore, you should manually export the data policy after you create a data protection mirror relationship and at any time that you modify the policy in the source SVM.

### About this task

When the destination SVM with Infinite Volume always references the same data policy as the source SVM with Infinite Volume, you are better prepared for disaster recovery. During disaster recovery, you might be required to direct clients to use the destination Infinite Volume while you repair the source Infinite Volume, and during this time, you want no change in data policy behavior.

### Steps

1. Export the data policy from the source SVM with Infinite Volume by using OnCommand Unified Manager.  
   See the *OnCommand Unified Manager Online Help*.

2. Import the data policy to the destination SVM with Infinite Volume by using OnCommand Unified Manager.  
   See the *OnCommand Unified Manager Online Help*.

Both Infinite Volumes reference the same data policy and its rules.

## Reversing the data protection mirror relationship for Infinite Volumes with storage classes when disaster occurs

If a disaster disables the source Infinite Volume with storage classes of a data protection mirror relationship, you can use the destination Infinite Volume to serve data while you repair the source Infinite Volume. Some situations require you to contact technical support for help.

### Before you begin

You must have exported the data policy from the source Infinite Volume with storage classes to the destination Infinite Volume with storage classes by using OnCommand Unified Manager.

### About this task

In the following example, a data protection mirror relationship exists between a source Infinite Volume with storage classes on one cluster and a destination Infinite Volume with storage classes on another cluster. The clusters are in a peer relationship, and the Storage Virtual Machines (SVMs) are
in a peer relationship. The original source volume (the one disabled by the disaster) is 
cluster1://vs1/volA, and the original destination volume is cluster2://vs2/volB. 

In this example, a data constituent of the source Infinite Volume is offline. All data from the last 
scheduled data protection mirror update before the source was disabled is preserved, and all the data 
written to cluster2://vs2/volB after you made the volume writable is preserved. Any data 
written to cluster1://vs1/volA between the last data protection mirror copy update and the time 
that cluster1://vs1/volA was disabled is not preserved. 

The following situations require you to contact technical support to help you resynchronize the data 
protection mirror relationship for an Infinite Volume with storage classes: 

• The source Infinite Volume with storage classes is unrecoverable. 
  After you make the destination Infinite Volume read/write to serve data, you must contact 
technical support to help you delete the source Infinite Volume, create a new Infinite Volume with 
storage classes, and create a data protection mirror relationship. 

• The size of the Infinite Volume with storage classes increased after you broke the data protection 
mirror relationship and before you resynchronized the data protection mirror relationship. 
  After you break the data protection mirror relationship, you must not change the size of the 
Infinite Volumes with storage classes before you resynchronize the data protection mirror 
relationship. For example, if volume B was 200 TB when you broke the data protection mirror 
relationship, and volume B remains 200 TB when you are ready to resynchronize the data 
protection mirror relationship, you can use the snapmirror resync command. However, if you 
broke the data protection mirror relationship and then increased the size of volume B to 300 TB, 
you cannot use the snapmirror resync command without the help of technical support. 

Steps 

1. Temporarily make the original source volume a read-only destination volume and reverse the data 
   protection mirror relationship to continue to serve data. 
   
If the source volume (cluster1://vs1/volA) is recoverable and its data is intact, complete the 
following steps: 

   a. After the source volume (in this case cluster1://vs1/volA) is disabled, use the 
snapmirror break command on the destination volume (cluster2://vs2/volB) to 
   make the destination volume writeable. 

   Example 

   cluster2::> snapmirror break vs2:volB 

   The data protection mirror relationship is broken. Volume B changes from read-only to read/ 
write. 

   b. Redirect the clients of volume A (cluster1://vs1/volA) to the new source volume B 
(cluster2://vs2/volB). 

   The former clients of volume A (cluster1://vs1/volA) access and write to volume B 
(cluster2://vs2/volB). 

   c. Ensure that volume B has a namespace mirror constituent to provide data protection for its 
namespace constituent. 

   You can create a namespace mirror constituent by enabling SnapDiff on the destination 
Infinite Volume with storage classes. 

   Reenabling SnapDiff after expanding a destination Infinite Volume with storage classes onto 
new nodes on page 178
d. On the destination volume (cluster2://vs2/volB), use the snapmirror delete command to remove the data protection mirror relationship between the source volume (cluster1://vs1/volA) and the destination volume (cluster2://vs2/volB).

**Example**

```
cluster2::> snapmirror delete vs2:volB
```

e. On the source volume (cluster1://vs1/volA), use the snapmirror release command to remove relationship information from the source.

**Example**

```
cluster1::> snapmirror release vs2:volB
```

Warning: Snapshot copies on source volume "vs1:volA" generated by SnapMirror for the purpose of mirroring to destination volume "vs2:volB" will be deleted. Once these Snapshot copies are deleted, it will likely not be possible to re-establish a mirroring relationship between these two volumes.

Do you want to continue? {y|n}: y

f. On the new destination volume (cluster1://vs1/volA), use the snapmirror create command to create the mirror relationship, but with volume B (cluster2://vs2/volB) as the new source and volume A (cluster1://vs1/volA) as the new destination.

**Example**

```
cluster1::> snapmirror create vs2:volB vs1:volA -type DP
```

g. On the new destination volume (cluster1://vs1/volA), use the snapmirror resync command to resynchronize volume A (cluster1://vs1/volA) with volume B (cluster2://vs2/volB).

**Note:** If the resynchronization process fails, you must create and resynchronize the data protection mirror relationship again.

**Example**

```
cluster1::> snapmirror resync vs1:volA
```

Warning: All data newer than Snapshot copy snapmirror.62dd0c95-efcc-11e3-b718-005056975d3f_5_1.2014-06-10_1724 on volume vs1:volA will be deleted.

Do you want to continue? {y|n}: y

If the source volume (cluster1://vs1/volA) is unrecoverable, complete the following steps:

a. After the source volume (in this case, cluster1://vs1/volA) is disabled, use the snapmirror break command on the destination volume (cluster2://vs2/volB) to make the destination volume writeable.

**Example**

```
cluster2::> snapmirror break vs2:volB
```

b. Redirect the clients of volume A (cluster1://vs1/volA) to the new source volume B (cluster2://vs2/volB).
The former clients of volume A (cluster1://vs1/volA) access and write to volume B (cluster2://vs2/volB).

c. On volume B (cluster2://vs2/volB), use the `snapmirror delete` command to remove the data protection mirror relationship between the volume A (cluster1://vs1/volA) and volume B (cluster2://vs2/volB).

```
Example
cluster2::> snapmirror delete vs2:volB
```

d. On the source volume (cluster1://vs1/volA), use the `snapmirror release` command to remove relationship information from the source volume.

Even though volume A is unrecoverable, the data protection mirror relationship still exists and must be removed.

```
Example
cluster1::> snapmirror release vs2:volB
```

Warning: Snapshot copies on source volume "vs1:volA" generated by SnapMirror for the purpose of mirroring to destination volume "vs2:volB" will be deleted. Once these Snapshot copies are deleted, it likely not be possible to re-establish a mirroring relationship between these two volumes.

Do you want to continue? {y|n}: y

e. Contact technical support for help with the following steps:

- Delete the unrecoverable volume A (cluster1://vs1/volA).
- Create a new Infinite Volume with storage classes named cluster1://vs1/volA.
- Create the data protection mirror relationship between volume B and volume A.
- Initialize the data protection mirror relationship.

Volume B operates as the read/write source volume, and volume A operates as the read-only destination volume. You can keep this configuration, or you can return the volumes to their original data protection configuration.

2. If you want to return the volumes to their original data protection configuration, perform the following steps:

a. On the new destination volume (cluster1://vs1/volA), update the new destination volume to transfer the latest data from the new source volume (cluster2://vs2/volB) by using the `snapmirror update` command.

```
Example
cluster1::> snapmirror update vs1:volA
```

b. On the new destination volume (cluster1://vs1/volA), use the `snapmirror break` command to make volume A (cluster1://vs1/volA) writeable.

```
Example
cluster1::> snapmirror break vs1:volA
```
c. On the new destination volume (cluster1://vs1/volA), use the `snapmirror delete` command to remove the data protection mirror relationship between volume B (cluster2://vs2/volB) and volume A (cluster1://vs1/volA).

Example

```
cluster1::> snapmirror delete vs1:volA
```

d. On the new source volume (cluster2://vs2/volB), use the `snapmirror release` command to remove the data protection mirror relationship between volume B (cluster2://vs2/volB) and volume A (cluster1://vs1/volA).

Example

```
cluster2::> snapmirror release vs1:volA
```

Warning: Snapshot copies on source volume "vs2:volB" generated by SnapMirror for the purpose of mirroring to destination volume "vs1:volA" will be deleted. Once these Snapshot copies are deleted, it will likely not be possible to re-establish a mirroring relationship between these two volumes.

Do you want to continue? {y|n}: y

e. On the original destination volume (cluster2://vs2/volB), use the `snapmirror create` command to re-create the original data protection mirror relationship with Volume A (cluster1://vs1/volA) as the source and volume B (cluster2://vs2/volB) as the destination.

Example

```
cluster2::> snapmirror create vs2:volB vs1:volA -type DP
```

f. On the original destination volume (cluster2://vs2/volB), use the `snapmirror resync` command to resynchronize the original source and original destination volumes.

Example

```
cluster2::> snapmirror resync vs2:volB
```

g. Redirect the clients from volume B (cluster2://vs2/volB) back to their original source volume A (cluster1://vs1/volA).

Related tasks

- Reenabling SnapDiff after expanding a destination Infinite Volume with storage classes onto new nodes on page 178

Processes used to select aggregates for destination Infinite Volumes with and without storage classes

The aggregate selection process that is used depends on whether the Infinite Volume uses storage classes and on the tool that is used to create the data protection mirror relationship for the Infinite Volume.

For Infinite Volumes without storage classes, you can use the command-line interface or OnCommand Workflow Automation to create and initialize data protection mirror relationships. In this case, the aggregate selection process provided by Data ONTAP is used.
For Infinite Volumes with storage classes, you must use OnCommand Workflow Automation to create and initialize data protection mirror relationships. You cannot use the command-line interface. In this case, the aggregate selection process provided by OnCommand Workflow Automation is used because it can factor storage classes into its aggregate selection process.

Related concepts

How Data ONTAP selects aggregates for data protection of Infinite Volumes on page 129

Providing tape backup and restore of Infinite Volumes with storage classes

You can incrementally back up an Infinite Volume with storage classes to tape, and you can restore data from tape to an Infinite Volume with storage classes.

How storage classes affect tape backup and restore of Infinite Volumes

Setup and management of incremental tape backup is the same for Infinite Volumes with or without storage classes. You must configure tape backup for the entire Infinite Volume. You cannot configure tape backup for individual storage classes in Infinite Volumes.

Related concepts

Providing tape backup and restore of Infinite Volumes on page 139

Protocol considerations for restoring files to Infinite Volumes with storage classes

When you restore files from a tape backup to an Infinite Volume with multiple storage classes by using the NFS or CIFS protocol with a backup application, the current data policy determines the storage class into which each file is placed. You should understand how the protocols work with the data policy.

For both NFS and CIFS, rules in the current data policy for the Storage Virtual Machine (SVM) with Infinite Volume determine the storage class for restored files based on directory path, file name, and file owner.

You should be aware of how a data policy with rules based on file owner works with the NFS protocol when you use a backup application with root instead of file owner to restore files to an Infinite Volume with storage classes. When you use root instead of file owner, the data policy rules might apply to root, even if the backup application subsequently changes the file owner. You should understand when the backup application changes the file owner to ensure that the data policy with rules based on file owner works as expected when you restore files over NFS to an Infinite Volume with storage classes.

Related concepts

Protocols for tape backup and restore of Infinite Volumes on page 140

Enabling SnapDiff on an Infinite Volume with storage classes

You can enable SnapDiff on an Infinite Volume with storage classes to support incremental tape backup.

Before you begin

- You must have installed OnCommand Workflow Automation to perform this procedure.
• You must know how to use workflows in OnCommand Workflow Automation.
• The Infinite Volume with storage classes must span two or more nodes in the cluster.
• The aggregates used by the Infinite Volume must have enough space to fit the required namespace mirror constituents.

**About this task**

Instead of using a workflow, you can enable SnapDiff for an Infinite Volume with storage classes by using the `volume modify` command. If you want to enable SnapDiff for a destination Infinite Volume, you cannot use a workflow; instead you must use the `volume modify` command.

**Steps**

1. Enable SnapDiff for an Infinite Volume with storage classes by using the **Setup data protection for an Infinite Volume** workflow with the following options in OnCommand Workflow Automation:
   a. In the Primary Cluster list, select the cluster that contains the Infinite Volume with storage classes.
      
      If the Infinite Volume is in a data protection mirror relationship, you must select the source Infinite Volume; you cannot select the destination Infinite Volume.
   b. In the Primary Vserver list, select the Storage Virtual Machine (SVM).
   c. Select the **Enable tape backups** check box.
   d. Clear the **Create SnapMirror relationship** check box.

2. Finish selecting the options, and click **Execute**.

SnapDiff is enabled for the Infinite Volume with storage classes and the required namespace mirror constituents are created.

**Related tasks**

*Enabling SnapDiff on an Infinite Volume* on page 151

**Monitoring space usage in Infinite Volumes with storage classes**

If an Infinite Volume uses storage classes, you can determine how full each storage class is by using OnCommand Unified Manager.

**Step**

1. Use OnCommand Unified Manager to view the capacity of the SVM with Infinite Volume.

   For more information, see *OnCommand Unified Manager Online Help*.

   When you view the capacity of the SVM with Infinite Volume, you can also view the used and free space of individual storage classes.

**Expanding Infinite Volumes with storage classes**

You can expand an Infinite Volume with storage classes by making more aggregate space available to the Infinite Volume, and then increasing the size of the Infinite Volume with storage classes. When
the Infinite Volume with storage classes is in a data protection mirror relationship, the destination Infinite Volume is automatically resized.

**Methods to increase aggregate space for Infinite Volumes with storage classes**

Before you increase the size of existing storage classes or add more storage classes to an Infinite Volume by using a workflow, you must make more aggregate space available. The workflows do not add more aggregate space; the workflows only expand the Infinite Volume to use more aggregate space.

You can use the following methods to make more aggregate space available:

- Add more disks to one or more of the aggregates that are already assigned to the Storage Virtual Machine (SVM) with Infinite Volume
- Assign more aggregates from existing nodes to the SVM with Infinite Volume
- Assign aggregates from one or more new nodes to the SVM with Infinite Volume

You must ensure that you add disks or aggregates that meet the storage class requirements.

**Related tasks**

- Determining space availability in an Infinite Volume’s aggregates on page 170
- Associating more aggregates with an Infinite Volume on page 172
- Adding disks to the aggregates that an Infinite Volume uses on page 171
- Associating more nodes with an Infinite Volume on page 174

**How expansion works for source and destination Infinite Volumes with storage classes**

When you use OnCommand Workflow Automation to expand an Infinite Volume with storage classes that is in a data protection mirror relationship, the process automatically expands the source and destination Infinite Volumes.

For an Infinite Volume with storage classes that is in a data protection mirror relationship, you can expand the Infinite Volume by either increasing the size of existing storage classes or by adding new storage classes. This process automatically expands the source and destination Infinite Volumes with storage classes as follows:

- The overall size of the source Infinite Volume is increased, and new data constituents are created for the storage classes.
  - If SnapDiff is enabled on the source Infinite Volume, new namespace mirror constituents are automatically created.

- The overall size of the destination Infinite Volume is increased. New constituents are created, but data protection mirror relationships are not yet created and initialized for the new data constituents. Existing constituents are not yet expanded.
  - When the next scheduled mirror update runs, data protection mirror relationships are created and initialized for new data constituents, and existing data constituents are expanded. Namespace mirror constituents on the source Infinite Volume are not affected.
  - If SnapDiff is enabled on the destination Infinite Volume with storage classes to support incremental tape backup, and the destination Infinite Volume with storage classes is expanded onto new nodes, you must reenable SnapDiff on the destination Infinite Volume with storage classes after expanding the volume onto new nodes and waiting for the data protection mirror update to transfer the latest constituents.
Resizing an Infinite Volume with storage classes

You can expand an Infinite Volume with storage classes by adding more storage classes or by increasing the size of existing storage classes. If the Infinite Volume is a source volume in a data protection mirror relationship, the resizing process automatically resizes the destination Infinite Volume too.

Before you begin

- You must have installed OnCommand Workflow Automation to perform this procedure.
- You must know how to use workflows in OnCommand Workflow Automation.
- The cluster must contain enough aggregate space to expand the Infinite Volume. If the Infinite Volume is in a data protection mirror relationship, both the source and destination clusters must contain enough aggregate space to expand the source and destination Infinite Volumes.
- If the Infinite Volume uses custom storage classes, you must have cloned the predefined workflows and modified the storage class settings to reflect the custom storage class settings before you use the workflows to resize an Infinite Volume with custom storage classes.

About this task

You must use OnCommand Workflow Automation to increase the size of an Infinite Volume with storage classes. You cannot use the command-line interface.

Steps

1. Increase the size of the existing storage classes or add new storage classes to the Infinite Volume by using one of the following workflows in OnCommand Workflow Automation:
   - Add or expand capacity storage class to Infinite Volume
   - Add or expand performance storage class to Infinite Volume
   - A customized workflow for a storage class if you created one
   The overall size of the Infinite Volume is increased and new data constituents are created for the storage classes.

2. If you added new storage classes to the Infinite Volume, modify the data policy for the SVM with Infinite Volume to create rules for filtering data into the new storage classes by using OnCommand Unified Manager.
   For more information, see the OnCommand Unified Manager Online Help.

3. If you updated the data policy for the SVM with Infinite Volume, export the updated data policy from the source SVM with Infinite Volume, and import the data policy to the destination SVM with Infinite Volume by using OnCommand Unified Manager.
   For more information, see the OnCommand Unified Manager Online Help.

Related concepts

- Methods to increase aggregate space for Infinite Volumes with storage classes on page 229
- How custom storage classes affect predefined workflows on page 203
Managing Infinite Volumes with storage classes

When you manage an existing Infinite Volume with storage classes, you can modify storage class settings, rebalance the used capacity of data constituents, and move constituents from an aggregate.

Modifying storage class settings for an Infinite Volume

You can modify volume settings, such as thin provisioning, deduplication, and compression, for the storage classes of an existing Infinite Volume.

Before you begin

- You must have installed OnCommand Workflow Automation to perform this procedure.
- You must know how to use workflows in OnCommand Workflow Automation.

About this task

When you modify the storage class settings for an Infinite Volume, you create custom storage classes for the Infinite Volume. You cannot use predefined workflows with custom storage classes without first cloning and modifying the predefined workflows to update the storage class settings.

You can modify the efficiency policy for a storage class by using the `volume efficiency policy` command.

Steps

1. Create a workflow by using OnCommand Workflow Automation.
2. Modify the storage class settings for an Infinite Volume by using the workflow in OnCommand Workflow Automation.

Related concepts

- How custom storage classes affect predefined workflows on page 203

Rebalancing the used capacity in a storage class of an Infinite Volume

You can rebalance used capacity across all data constituents in a storage class of an Infinite Volume by using Auto Balance Volume to maintain roughly equal amounts of used space in all data constituents in the storage class. You must run the operation on each storage class in the Infinite Volume.

Before you begin

- The Infinite Volume must contain storage classes.
- You must have viewed the used capacity of the Infinite Volume with storage classes to identify the name of the storage class.

About this task

You should rebalance capacity in a storage class in the following situations:

- You increased the size of the storage class, and the operation created new data constituents.
The new data constituents are empty, and the pre-existing data constituents contain data.

- You have used the storage class for some time and notice that some data constituents have greater used capacity than other data constituents.

You cannot rebalance the used capacity across data constituents in a storage class of a destination Infinite Volume. You can only rebalance the used capacity across data constituents in a storage class of a source Infinite Volume in a data protection mirror relationship.

**Steps**

1. Switch to advanced privilege by using the `set -privilege advanced` command.

2. Rebalance capacity between the data constituents of the storage class by using the `autobalance volume rebalance start` command with the `storage-service` parameter.

**Example**

In the following example, used capacity rebalancing is started for a Storage Virtual Machine (SVM) named vs0 and an Infinite Volume named repo_vol with a gold storage class:

```
cluster1::*> autobalance volume rebalance start -vserver vs0 -volume repo_vol -storage-service gold
```

Info: Auto Balance Volume operation started. Use the "autobalance volume rebalance show -vserver vs0 -volume repo_vol -storage-service gold" command to view the status of this operation.

3. View the state of the operation by using the `volume balance capacity show` command.

**Example**

In the following example, the status of the operation is `running`:

```
cluster1::*> autobalance volume rebalance show -vserver vs0 -volume repo_vol -storage-service gold
```

```
Vserver: vs0
Volume Name: repo_vol
Storage Service: gold
State: running
Progress:
  Amount Transferred: 14.18TB
  Target Amount: 47.28TB
  Percentage Transferred: 30%
```

**Related concepts**

- *When fullness warning messages are generated* on page 165
- *Rebalancing the used capacity of data constituents* on page 183

**Moving constituents of an Infinite Volume with storage classes from an aggregate**

Before moving constituents of an Infinite Volume with storage classes from an aggregate, you must know which storage classes reference the constituents and ensure that the destination aggregate meets
the storage class requirements. Otherwise constituents might be moved to an aggregate that is inapposite for the storage class.

The system does not validate that the destination aggregate meets the storage class requirements when you move constituents from the aggregate.

Steps
1. Viewing the names of storage classes on page 233
2. Identifying aggregate requirements for storage classes on page 233
3. Moving constituents of an Infinite Volume with storage classes from an aggregate on page 234

Related concepts
How SnapDiff affects moving constituents from an aggregate on page 190

Viewing the names of storage classes
You can view the names of the storage classes for an Infinite Volume by viewing the storage-service field. You can then use the storage class names with some commands to perform operations on the storage class.

About this task
You can also view the names of the storage classes for an Infinite Volume by using OnCommand Unified Manager.

Steps
1. Switch to advanced privilege by using the set -privilege advanced command.
2. View the names of the storage classes in an Infinite Volume by using the volume show command with the -is-constituent and -storage-service parameters.

Example
In the following example, an Infinite Volume on the Storage Virtual Machine (SVM) named vs0 contains gold, silver, and bronze storage classes:

```plaintext
cluster1::*> volume show -is-constituent true -fields percent-used, storage-service
vserver volume percent-used storage-service
---------- ----------- ------------
vs0 repos_1024_data0001 5% gold
vs0 repos_1025_data0001 70% silver
vs0 repos_1025_data0002 75% silver
vs0 repos_1026_data0001 0% bronze
vs0 repos_1026_data0002 10% bronze
vs0 repos_ns 10% -
vs0 repos_ols0001 33% -
7 entries were displayed.
```

Identifying aggregate requirements for storage classes
You can identify the aggregate requirements for a storage class by reviewing the workflow in OnCommand Workflow Automation that was used to create the storage class.

Before you begin
- You must have installed OnCommand Workflow Automation to perform this procedure.
- You must know how to use workflows in OnCommand Workflow Automation.
Step

1. View the disk type that is required for aggregates in the workflow that you used to create the storage class for the Infinite Volume by using OnCommand Workflow Automation.

For example, if you used a predefined workflow for storage classes, such as **Add or expand capacity storage class to Infinite Volume** or **Add or expand performance storage class to Infinite Volume**, you can click the Help icon for the workflow, and read the Goal section to determine what type of aggregates are required.

For more information, see OnCommand Workflow Automation.

Moving constituents of an Infinite Volume with storage classes from an aggregate

You can use a single command to move all constituents of an Infinite Volume from a source aggregate to a destination aggregate to prepare the source aggregate for decommissioning.

Before you begin

- You must have cluster administrator privileges to perform this task.
- The destination aggregate must be on a platform supported by Infinite Volumes.
- The destination aggregate must be assigned to the aggregate list for the Storage Virtual Machine (SVM) with Infinite Volume.
- The destination aggregate must have enough space to fit all of the constituents from the source aggregate.
- If the Infinite Volume uses storage classes, the destination aggregate must meet the storage class requirements of the data constituents being moved. The system does not validate that the destination aggregate meets the storage class requirements when you run the operation.

About this task

- The `volume aggregate vacate` command runs in the background, allowing you to perform additional operations.
- If the Infinite Volume is a source volume in a data protection mirror relationship, and you are initializing the data protection mirror relationship, you must wait until the `snapmirror initialize` command finishes before you run the `volume aggregate vacate` command. Otherwise the `snapmirror initialize` command will fail.
- The `volume aggregate vacate` command moves only constituents of an Infinite Volume from the aggregate; it does not move FlexVol volumes from the aggregate. You must use the `volume move` command to move FlexVol volumes from the aggregate that you want to decommission.

Steps

1. Move constituents of an Infinite Volume from a source aggregate to a destination aggregate by using the `volume aggregate vacate` command.

Example

In the following example, constituents of an Infinite Volume named repo_vol are moved from the source aggregate named aggr1 to the destination aggregate named aggr2:
cluster::> volume aggregate vacate -vserver vs0 -volume repo_vol -source-aggregate aggr1 -destination-aggregate aggr2

All constituents for the specified Infinite Volume are moved from the source aggregate to the destination aggregate.

2. Repeat this procedure as required for other Infinite Volumes until no more constituents remain on the source aggregate.

3. Move any FlexVol volumes from the aggregate by using the `volume move` command.

**Related information**

*NetApp Hardware Universe*
Copyright information

Copyright © 1994–2017 NetApp, Inc. All rights reserved. Printed in the U.S.

No part of this document covered by copyright may be reproduced in any form or by any means—graphic, electronic, or mechanical, including photocopying, recording, taping, or storage in an electronic retrieval system—without prior written permission of the copyright owner.

Software derived from copyrighted NetApp material is subject to the following license and disclaimer:

**THIS SOFTWARE IS PROVIDED BY NETAPP "AS IS" AND WITHOUT ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, WHICH ARE HEREBY DISCLAIMED. IN NO EVENT SHALL NETAPP BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.**

NetApp reserves the right to change any products described herein at any time, and without notice. NetApp assumes no responsibility or liability arising from the use of products described herein, except as expressly agreed to in writing by NetApp. The use or purchase of this product does not convey a license under any patent rights, trademark rights, or any other intellectual property rights of NetApp.

The product described in this manual may be protected by one or more U.S. patents, foreign patents, or pending applications.

RESTRIC TED RIGHTS LEGEND: Use, duplication, or disclosure by the government is subject to restrictions as set forth in subparagraph (c)(1)(ii) of the Rights in Technical Data and Computer Software clause at DFARS 252.277-7103 (October 1988) and FAR 52-227-19 (June 1987).
Trademark information

Active IQ, AltaVault, Arch Design, ASUP, AutoSupport, Campaign Express, Clustered Data ONTAP, Customer Fitness, Data ONTAP, DataMotion, Fitness, Flash Accel, Flash Cache, Flash Pool, FlexArray, FlexCache, FlexClone, FlexGroup, FlexPod, FlexScale, FlexShare, FlexVol, FPolicy, Fueled by SolidFire, GetSuccessful, Helix Design, LockVault, Manage ONTAP, MetroCluster, MultiStore, NetApp, NetApp Insight, OnCommand, ONTAP, ONTAPI, RAID DP, RAID-TEC, SANscreen, SANshare, SANtricity, SecureShare, Simplicity, Simulate ONTAP, Snap Creator, SnapCenter, SnapCopy, SnapDrive, SnapIntegrator, SnapLock, SnapManager, SnapMirror, SnapMover, SnapProtect, SnapRestore, Snapshot, SnapValidator, SnapVault, SolidFire, SolidFire Helix, StorageGRID, SyncMirror, Tech OnTap, Unbound Cloud, and WAFL and other names are trademarks or registered trademarks of NetApp, Inc., in the United States, and/or other countries. All other brands or products are trademarks or registered trademarks of their respective holders and should be treated as such. A current list of NetApp trademarks is available on the web.

How to send comments about documentation and receive update notifications

You can help us to improve the quality of our documentation by sending us your feedback. You can receive automatic notification when production-level (GA/FCS) documentation is initially released or important changes are made to existing production-level documents.

If you have suggestions for improving this document, send us your comments by email.

doccomments@netapp.com

To help us direct your comments to the correct division, include in the subject line the product name, version, and operating system.

If you want to be notified automatically when production-level documentation is released or important changes are made to existing production-level documents, follow Twitter account @NetAppDoc.

You can also contact us in the following ways:

• NetApp, Inc., 495 East Java Drive, Sunnyvale, CA 94089 U.S.
• Telephone: +1 (408) 822-6000
• Fax: +1 (408) 822-4501
• Support telephone: +1 (888) 463-8277
Index

A
access
controlling based on IP address on Infinite Volumes 84
controlling with file permissions on Infinite Volumes 85
access checks
Infinite Volumes 75
ACLs
file permissions, SMB 77
share, controlling SMB access to Infinite Volumes 85
active file systems
protocols supported on destination Infinite Volumes 120
when available on destination Infinite Volumes 120
when clients can access on destination Infinite Volume 120
aggregate lists
specifying for SVMs with Infinite Volume (dedicated aggregates) 107
specifying for SVMs with Infinite Volume (shared aggregates) 115
aggregate overcommitment
for Infinite Volumes, about 30
aggregate show-space command
how to determine aggregate space usage by using 162
aggregates
adding disks to, to expand Infinite Volumes 171
adding to a destination cluster 106
assigning dedicated aggregates to SVMs with Infinite Volume 51
assigning shared aggregates to SVMs with Infinite Volume 49
assigning to SVMs with Infinite Volume 53, 220
assigning to SVMs with Infinite Volume (dedicated aggregates) 54, 107
assigning to SVMs with Infinite Volume (shared aggregates) 50, 115
associated with Infinite Volumes 34
associating with all SVMs in the cluster 53
automatically selected during mirror copy updates 169
automatically selected for destination Infinite Volumes 129
automatically selected for destination Infinite Volumes, examples of 132
calculation used to very space to fit multiple constituents 132
commands for displaying space information 159
considerations when planning space 30
constraining data constituents 35
creating for Infinite Volumes 42
creating for Infinite Volumes with storage classes 207
customizing type to use for Infinite Volumes with storage classes 204, 205
determining those used by Infinite Volumes 169
derror messages for destination Infinite Volumes, examples of 135
expanding Infinite Volumes by associating more 172
expanding Infinite Volumes to use more space 176
how associated with SVMs with Infinite Volume 34
how automatically selected for destination Infinite Volumes 130
how Infinite Volumes use 36
how provisioning of constituents affects aggregate selection for Infinite Volumes 131
how SnapDiff affects moving constituents for Infinite Volumes 190
how storage classes use for Infinite Volumes 199
how to determine and control volume space usage in 160
how to determine space usage in 162
identifying how many a source Infinite Volume uses 105
identifying number required for destination Infinite Volumes 105
identifying space required for destination Infinite Volumes 112
impact of space guarantee for destination Infinite Volumes on 131
Infinite Volume constituents 35
methods of creating space in 163
methods to control selection for destination Infinite Volumes 129
methods to create more space available 229
methods used to select for destination Infinite Volumes 226
mirrored, using to protect namespace constituents 36
moving constituents for Infinite Volumes 189
moving constituents for Infinite Volumes (with storage classes) 234
namespace constituent, dedicated to 56
preparing to move constituents of Infinite Volumes with storage classes 232
preserving, for the namespace constituent 58
requirements for data protection mirror relationships 22
requirements for Infinite Volumes 22
requirements for storage classes 199
restrictions for automatic selection 132
selecting for an Infinite Volume 49
selecting for an Infinite Volume and all the SVMs 51
selecting, for constituents 57
selection, for namespace mirror constituents 147
setting up data protection for Infinite Volumes using dedicated aggregates 104
setting up data protection for Infinite Volumes using shared aggregates 111
sharing between FlexVol volumes and Infinite Volumes 33
SVM with Infinite Volume, assigning (dedicated aggregates) 54
SVM with Infinite Volume, assigning (shared aggregates) 50
SVM with Infinite Volume, creating with 51
SVM with Infinite Volume, creating with dedicated aggregates 54
understanding selection for destination Infinite Volumes 129
verifying space in a destination cluster 113
viewing requirements for storage classes 233
when automatically selected for destination Infinite Volumes 129
where to find more information about 19
worksheet for planning 39
aggregates to expand the Infinite Volume
assigning from new nodes to SVMs with Infinite Volume 174
aggregates, Infinite Volume
determining space availability before resizing 170
array LUNs
how customized predefined workflows support 204
assigning
aggregates 34, 49, 51
auditing
Infinite Volumes support 71, 72
Auto Balance Volume
commands for Infinite Volumes 183
examples of used capacity rebalancing 184
impact of offline constituents on operations 186
rebalancing the used capacity
across data constituents 187
of data constituents for a storage class 231
autogrow
unsupported by Infinite Volumes 164
automatic free space preservation
unsupported by Infinite Volumes 164
automatic node referrals
Infinite Volumes not support 72
autoshrink
unsupported by Infinite Volumes 164
autosizing
unsupported by Infinite Volumes 164
B
backup applications
guidelines for using network interfaces for SnapDiff 142
selecting a Snapshot copy for SnapDiff 152
SnapDiff support for 141
supported tape backup topology for mounted volumes 140
backups
creating Snapshot copies of Infinite Volumes 98
using Snapshot copies 95
See also tape backup
BranchCache
Infinite Volumes support 72
C
capabilities
Infinite Volumes 11
capacity
how it is allocated in new Infinite Volumes 37
how rebalancing the used capacity of data constituents works 184
rebalancing the used capacity
across data constituents 187
of data constituents for a storage class 231
rebalancing the used capacity of data constituents 183
viewing the used capacity of data constituents 187
when warning messages are generated for full constituents 165
capacity, Infinite Volume
determining aggregate space availability before resizing 170
change notifications
Infinite Volumes support 72
CIFS
considerations for tape backup and restore of Infinite Volumes 140
using to back up Infinite Volumes to tape 139
using to restore files to Infinite Volumes 157
where to find more information about 19
See also SMB
CIFS access
share ACLs 75
CIFS servers
creating for Infinite Volumes 81
client access
Infinite Volumes 73
setting up NFS access to Infinite Volumes 80
setting up SMB access to Infinite Volumes 81
size of Infinite Volumes 90
to Snapshot copies of Infinite Volumes, temporarily disabled 97
what Snapshot copies to access 97
See also file access
cluster peering
creating peer relationships between clusters
(dedicated aggregates) 107
creating peer relationships between clusters (shared aggregates) 115
command line
aggregate selection process for destination Infinite Volumes 226
commands
disabled for Infinite Volumes with storage classes 200
for displaying aggregate space information 159
for displaying Infinite Volume space information 159
for rebalancing used capacity 183
how they run on Infinite Volumes 98
comments
how to send feedback about documentation 238
compression
customizing settings for Infinite Volumes with storage classes 204, 205
enabling 68
how it works on storage classes 199
how it works with Infinite Volumes 67
Infinite Volumes, viewing details 181
modifying setting for storage classes 231
where to find more information about 19
configuring
SVMs 48
considerations
aggregate space used by tape backup resources 32
constituent sizes for data protection for Infinite Volumes 32
deciding whether to use storage classes 25
protocols for restoring files to Infinite Volumes with storage classes 227
valid JSON formatting in data policies 214
when using thin provisioning with Infinite Volumes 30
constituents
and Snapshot copies 97
creating for destination Infinite Volumes (dedicated aggregates) 108, 109
creating for destination Infinite Volumes (shared aggregates) 116, 117
data constituents, recovering after failure 127
described 13
determining state of Infinite Volumes 190
effect on information availability about destination Infinite Volumes 119
failures 180
for infinite Volumes 14
forcing to online, offline, or restricted state 191
how Infinite Volumes are composed of 13
how Infinite Volumes place on aggregates 36
how provisioning affects aggregate selection 131
identifying how many a source Infinite Volume uses 112
identifying size of 112
identifying the maximum size supported by platforms 60
impact of moving from aggregates when SnapDiff enabled 190
impact on Auto Balance Volume when offline 186
managing disaster recovery 120
moving from aggregates used by Infinite Volumes 189
moving from aggregates used by Infinite Volumes (with storage classes) 234
namespace constituent, recovering after failure 127
naming convention 197
size considerations for data protection mirror copies for Infinite Volumes 32
verifying that Infinite Volumes have enough space for namespace mirror 149
viewing 195
viewing constituents with a specific role for an Infinite Volume 196
viewing Snapshot copies of 99
cr.abv.service.filling corrective action for the event 165
cr.abv.service.full corrective action for the event 165
cr.abv.volume.filling corrective action for the event 165
cr.abv.volume.full corrective action for the event 165
creating aggregates 42
aggregates for Infinite Volumes with storage classes 207
an Infinite Volume using default setting 55
data protection mirror copy of an Infinite Volume 111
data protection mirror copy of Infinite Volumes 104
data protection mirror copy of Infinite Volumes with storage classes 220
destination Infinite Volumes with storage classes 220
Infinite Volumes with dedicated namespace aggregate 56
Infinite Volumes with small constituents 60
Infinite Volumes with storage classes 209
SVMs with Infinite Volume 47

D
data constituents

calculation used to verify space for multiple constituents on one aggregate 132
commands for rebalancing used capacity 183
definition 14
described 13
how event messages warn you when full 165
how used capacity is rebalanced across an Infinite Volume 184
rebalancing the used capacity of 183, 187, 231
size of 35
viewing for an Infinite Volume 196
viewing the used capacity of 187

Data ONTAP
where to find information about features in 19
data policies
commands for managing in JSON format 213
configuring users or directories for rules in 212
creating Infinite Volumes with 208
defined 16
editing rules 212
exporting from a source to a destination SVM with Infinite Volume 222
file access configuration to support 211
filtering based on directory 215
filtering based on file name 216
filtering based on file owner 217
how the default rule works 16
how they filter files for SVMs with Infinite Volume 16
importing into SVMs with Infinite Volume in JSON format 214
introduction to 15
JSON considerations for importing 214
managing for Infinite Volumes with storage classes 213
modifying for SVM with Infinite Volume 212
tools for editing for Infinite Volumes 202
when applied to files in Infinite Volumes 17
data protection
creating a namespace mirror constituent, Infinite Volumes 125
where to find more information about 19
data protection mirror copies
active file systems for 120
creating aggregates to provide storage for 207
namespace constituent in Infinite Volumes, of 14
providing disaster recovery for Infinite Volumes 104
when namespace mirror constituents are created in Infinite Volumes 142
data protection mirror relationships
aggregate requirements 22
aggregate requirements for Infinite Volumes 22
between volumes on platforms of different sizes 32
creating aggregates to provide storage for 42
creating for Infinite Volumes with storage classes 220, 221
creating Infinite Volume on a platform supporting smaller constituents 59
creating relationships for Infinite Volumes (dedicated aggregates) 109
creating relationships for Infinite Volumes (shared aggregates) 117
destination Infinite Volumes, when information is available 119
expanding Infinite Volumes to use more aggregate space 176
expansion on Infinite Volumes with storage classes 228
how expansion works for Infinite Volumes with storage classes 229
how maximum constituent size affects Infinite Volumes 32
how updates affect aggregate selection 169
importing a data policy for Infinite Volumes in 214
increasing the size of Infinite Volumes with storage classes 230
initialization errors for Infinite Volumes 135
initializing data protection mirror relationships for Infinite Volumes (dedicated aggregates) 109
initializing data protection mirror relationships for Infinite Volumes (shared aggregates) 117
retrieving data from Infinite Volumes during a disaster 121
retrieving data from storage classes during disaster recovery 222
setting up for Infinite Volume with shared aggregates 111
setting up for Infinite Volumes with dedicated aggregates 104
troubleshooting errors for Infinite Volumes 135
viewing information about destination Infinite Volumes 119
when aggregates are selected for Infinite Volumes 129
data protection mirrors
Infinite Volume size requirements for mirror copy transfers 127
understanding aggregate selection for destination Infinite Volumes 129
deduplication
compression
viewing space savings 159
customizing settings for Infinite Volumes with storage classes 204, 205
enabling 68
how it works on storage classes 199
how it works with Infinite Volumes 66
Infinite Volumes
compression, viewing space savings 159
deduplication, viewing space savings 159
viewing space savings 159
modifying setting for storage classes 231
on Infinite Volumes, viewing details 181
viewing space savings 159
where to find more information about 19
default storage classes
definitions for 198
default UNIX users
Infinite Volumes 79
default Windows users
Infinite Volumes 79
delegations
Infinite Volumes support 71
deleting
Infinite Volumes 63
destination Infinite Volumes
adding dedicated aggregates to a cluster for 106
aggregate requirements for 22
aggregates are automatically selected for 129
aggregates, method used to select for 226
availability of information about 119
client access to Snapshot copies on 120
creating (dedicated aggregates) 108
creating (shared aggregates) 116
creating a data protection mirror relationship for (dedicated aggregates) 109
creating a data protection mirror relationship for (shared aggregates) 117
creating aggregates for 42
creating constituents for (dedicated aggregates) 109
creating constituents for (shared aggregates) 117
examples of automatic aggregate selection for 132
how mirror copy updates affect aggregate selection 169
identifying aggregate space required for 112
identifying number of aggregates required for 105
identifying size required 105
namespace mirror constituents on 127
protocols supported for client access 120
storage classes
creating 220
creating aggregates for 207
transferring Snapshot copies to 154
understanding aggregate selection for 129
verifying aggregate space for shared aggregates 113
viewing information about 119
when aggregates are automatically selected for 129
worksheet for planning aggregates 39
destination Infinite Volumes with storage classes
aggregates, method used to select for 226
impact of increasing the size of 229
directories
configuring for rules in a data policy 212
for Snapshot copies of volumes 97
disaster recovery
for Infinite Volumes using mirroring 104
Infinite Volumes with storage classes, for 222
disasters
managing recovery for Infinite Volumes 120
disks
adding to aggregates to expand Infinite Volumes 171
DMAs
incremental tape backups of Infinite Volumes, using Snapshot copies and SnapDiff 141
documentation
how to receive automatic notification of changes to 238
how to send feedback about 238

E

efficiency
commands disabled on Infinite Volumes with storage classes 200
compression on Infinite Volumes 67
deduplication on Infinite Volumes 66
enabling 68
how it works on storage classes 199
modifying setting for storage classes 231
on Infinite Volumes, viewing details 181
EMS
fullness warning messages for Infinite Volumes 165
error messages
failed Infinite Volume creation 64
failed Infinite Volume expansion 64
for failed aggregate selection 135
insufficient space for the following source data constituents on destination SVM 136
insufficient space on destination for namespace constituent 135
insufficient space on SnapMirror destination for source Infinite Volume data constituents 137
size exceeds maximum, data constituents 137
size exceeds maximum, namespace constituent 136
this command is not supported for Infinite Volumes 201
when creating Infinite Volumes 64
events
Infinite Volume constituents 182
when fullness warning messages are generated for Infinite Volumes 165
expanding
data protection mirror relationships for Infinite Volumes with storage classes 228
Infinite Volumes by adding disks to aggregates 171
Infinite Volumes by associating more aggregates 172
Infinite Volumes by associating more nodes 174
Infinite Volumes by resizing 176
Infinite Volumes in a data protection mirror relationship by resizing 176
Infinite Volumes with storage classes
by resizing 230
in a data protection mirror relationship by resizing 230
Infinite Volumes, determining aggregate space availability before resizing 170
Infinite Volumes, methods of 168
methods to create more aggregate space before 229
reenabling SnapDiff after expanding destination
Infinite Volumes 178
reenabling SnapDiff after expanding destination
Infinite Volumes with storage classes 178
expanding Infinite Volumes
how capacity is allocated 168
export policies
controlling access to Infinite Volumes using 84
how they work on Infinite Volumes 74
Infinite Volume default 76
export rules
controlling access to Infinite Volumes 84

F

features
unsupported on Infinite Volumes 17
feedback
how to send comments about documentation 238
file access
configuration support for data policies 211
configuring users or directories for rules in a data policy 212
Infinite Volumes 71, 78
Infinite Volumes with storage classes 211
mixed state of Infinite Volumes 190
preparing Infinite Volumes for 78
with export policies 74
See also client access
file locks
how they work 90
file permissions
CIFS ACLs 83
comparison of unified and mixed security styles 91
configuring on Infinite Volumes 85
Infinite Volume defaults 76
Infinite Volumes 75
NFSv4 ACLs 83
NFSv4.1 ACLs 77
SMB ACLs 77
SMB clients setting for UNIX users 88
types 75
what happens when NFS clients set NFSv4.1 ACLs on Infinite Volumes 88
what happens when NFS clients set UNIX permissions on Infinite Volumes 87
what happens when SMB clients set NTFS file permissions on Infinite Volumes 88
file reservations
unsupported by Infinite Volumes 164
file system
data policy, impact on 16
Infinite Volumes with storage classes 24
See also namespace
files
how Infinite Volumes balance space usage when writing 164
maximum number on Infinite Volume 166
monitoring number on Infinite Volumes 166
FlexVol volumes
aggregate sharing with Infinite Volumes 33
comparison with Infinite Volumes 12
how maximum size affects Infinite Volumes in data protection mirror relationships 32
how to determine and control space usage in aggregates 160
where to find more information about 19
FlexVol volumes and Infinite Volumes
behaviour of security style 91
footprint
volume, described 162
footprint issue
creating or expanding Infinite Volumes, troubleshooting 65

FPolicy
Infinite Volumes support 71, 72
fractional reserve
unsupported by Infinite Volumes 164

Fsecurity
Infinite Volumes support 71, 72
full threshold corrective action for Infinite Volume capacity 165

G

group mappings
configuring on Infinite Volumes 83
Infinite Volumes 77
guidelines
for creating SVMs 44
for working with Snapshot copies of Infinite Volumes 95

I

importing
data policy in JSON format 214
incremental tape backup
aggregate selection for namespace mirror constituents 147
disabling for Infinite Volumes 157
enabling SnapDiff 151
for Infinite Volumes 156
how namespace-related constituents are used 142 of Infinite Volumes 141
setup, destination Infinite Volumes 154
setup, Infinite Volumes 149
SnapDiff support
of backup application 141
tape backup topology for 140
See also tape backup

Infinite Volumes
access location 73
aggregate requirements 22
generate requirements for data protection mirror relationships 22
associated aggregates 34
backing up files incrementally to tape 156
backing up over NAS protocols to tape 153
backing up over NAS protocols to tape (destination Infinite Volumes) 155
backup applications supported by SnapDiff 141
calculation used to verify space on one aggregate for multiple constituents 132
capabilities of 11
capacity allocation 37, 168
choosing aggregates for 49
CIFS support 72
client access 73
commands for displaying space information 159
commands for rebalancing used capacity 183
comparison with FlexVol volumes 12
completing an interrupted creation operation 61
compression, enabling 68
configuring file permissions 85
configuring group mappings 83
configuring IP-based access control 84
considerations before creating 30
constituent naming convention 197
constituent placement 57
constituents
how you can recover 120
constituents described 13
controlling access with share ACLs 85
controlling client access 73
converting to an Infinite Volume with storage classes and data policy 208
creating a namespace mirror constituent for 125
creating CIFS servers 81
creating cluster peer relationships (dedicated aggregates) 107
creating cluster peer relationships (shared aggregates) 115
creating NFS servers 80
creating shares 81
creating Snapshot copies of 98
creating SVM peer relationships (dedicated aggregates) 107
creating SVM peer relationships (shared aggregates) 115
creating using default settings 55
creating with dedicated namespace aggregate 56
creating with manually selected aggregates 57
creating with small constituents 60
creating, examples of 38
creating, for data protection on a different platform 59
data constituent size 35
data constituents 14
data policies for filtering data into storage classes 211
See also Infinite Volumes with storage classes
deciding whether to verify space requirements for SnapDiff 149
deduplication operations, viewing details 181
deduplication, enabling 68
default junction for 23
default permissions 76
default Snapshot policy for 96
default user 79
definition of 11
deleting 63
deleting invalid Snapshot copies 102
deleting namespace mirror constituents used by SnapDiff 157
determining aggregate space availability before resizing 170
determining aggregates used 169
disabling incremental tape backups 157
disabling SnapDiff 157
effect of namespace mirror constituents on the state of 146
enabling SnapDiff 151
error messages for creation and expansion 64
error messages for failed aggregate selection 135
events 182
unsupported features 17
user name mapping 79
verifying space requirements for SnapDiff resources 149
viewing constituents 195
viewing constituents with a specific role 196
viewing Snapshot copies of constituents 99
viewing the state of Snapshot copies for 101
viewing the used capacity of data constituents 187
volume size calculation for mirror copy transfer 127
what happens when NFS clients set NFSv4.1 ACLs on 88
what happens when NFS clients set UNIX permissions on Infinite Volumes 87
what happens when SMB clients set NTFS file permissions on 88
when fullness warning messages are generated 165
when to use 11
why size appears smaller 90
workflow for tape backup and restore 139

Infinite Volumes with storage classes
adding capacity 230
adding storage classes 210
configuring Snapshot policies 211
configuring users or directories for rules in a data policy 212
considerations for deciding whether to use 25
converting to 208
creating 209
creating a data protection mirror copy for 220
creating aggregates for 207
creating data protection mirror relationships 221
customizing predefined workflows for storage classes 204, 205
data policies and rules, introduction to 15
deciding whether to use 24
definitions of default storage classes 198
disabled commands for 200
editing rules in a data policy 212
effect on aggregates 24
enabling SnapDiff 227
enabling SnapDiff for 221
expanding 228
exporting a data policy 222
file access configuration to support data policies 211
how compression works on 199
how customized settings affect predefined workflows 203
how deduplication works on 199
how expansion works for destination volumes 229
how offline constituents affect Auto Balance Volume 186
how used capacity is rebalanced across data constituents 184
impact on file access 211
impact on moving constituents from aggregates when SnapDiff enabled 190
increasing the size when in a mirror relationship 230
introduction to 15
JSON data policy example, filtering data based on file owner 217
managing 231
methods to create more aggregate space for 229
modifying data policies 212
modifying storage class settings 231
monitoring space usage 228
moving constituents from aggregates 234
planning to create 198
preparing to move constituents from aggregates 232
protocol considerations for tape restore 227
rebalancing the used capacity
of data constituents for a storage class 231
reenabling SnapDiff 178
requirements 20
retrieving data during disaster recovery 222
setting up Snapshot policies for 221
Snapshot copies, impact on 219
tape backup and restore for 227
tape backup, impact of storage classes on 227
tasks for setting up and managing 27
tools for creating and managing 202
tools for creating mirror copies 219
tools for modifying 203
tools required to create and manage 23
using 198
using mirroring technology on 219
viewing the names of 233
viewing the used capacity data constituents in a storage class 187
when to use 24
worksheet for planning aggregates 39

InfiniteVol
See Infinite Volumes information
how to send feedback about improving documentation 238
invalid state
Snapshot copies 97
IPSpace
considerations when creating SVMs 44

J

JSON
commands for managing data policies in format of 213
considerations for data policies 214
data policy example, filtering based on directory 215
data policy example, filtering based on file name 216
data policy example, filtering based on file owner 217
importing data policies in format of 214
junction path
Infinite Volumes 73
junctions
for SVMs with Infinite Volume 23, 73, 182

K

Kerberos
Infinite Volumes 79
L
language considerations
for creating SVMs 44
LDAP
Infinite Volumes 79
LIFs
guidelines for using with SnapDiff 142
local Windows users and groups
Infinite Volumes 79
locks
how they work on Infinite Volumes 90
supported for the active file system on destination
Infinite Volumes 120
LUN reservations
unsupported by Infinite Volumes 164

M
management tasks
for Infinite Volume with storage classes 27
for Infinite Volumes 25
maximum FlexVol volume size
and data constituents 35
MetroCluster configuration
naming guidelines for
SVMs
guidelines for creating 44
mirror constituents
verifying that Infinite Volumes have enough space
for namespace 149
mirror relationships
tools for creating for Infinite Volumes with storage
classes 219
mirrored aggregates
using to protect namespace constituents 36
mixed security style
compared to unified security style 91
mode bits
on Infinite Volumes 89
what happens when set by NFS clients on Infinite
Volumes 87
monitoring
space usage in Infinite Volumes with storage classes
228
mount points
for Infinite Volumes 23
for SVMs with Infinite Volume 73, 182
mounting
Infinite Volumes 73, 80
multiprotocol access
Infinite Volumes 71

N
name mapping
Infinite Volumes 79
names
viewing for storage classes 233
namespace
data policy, impact on 16
for SVMs with Infinite Volume 23, 73, 182
Infinite Volumes with storage classes 24
See also file system
namespace constituents
and maximum files 166
and thin provisioning 30
definition 14
described 13
how they use aggregate space 35
methods of protecting 36
preserving an aggregate for 58
tape backup, how it uses 142
viewing for an Infinite Volume 196
namespace mirror constituents
creating for destination Infinite Volumes 125
deciding whether to verify space requirements for
SnapDiff 149
definition of 14
deleting by disabling SnapDiff 157
described 13
effect of modifying the Infinite Volume 145
effect on Infinite Volume state 146
how aggregates are selected for 147
how they use aggregate space 35
impact on source and destination Infinite Volumes
127
space requirements for 147
tape backup, how it uses 142
verifying that Infinite Volumes have enough space
for 149
viewing for an Infinite Volume 196
NAS protocols
using to back up Infinite Volumes to tape 153
using to back up Infinite Volumes to tape
(destination Infinite Volumes) 155
nearly full threshold
corrective action when for Infinite Volume capacity
165
network interfaces
guidelines for using with SnapDiff 142
NFS
protocols for tape backup and restore of Infinite
Volumes 140
support for SVMs with Infinite Volume 71
using to back up Infinite Volumes to tape 139
using to restore files to Infinite Volumes 157
where to find more information about 19
NFS access
setting up for Infinite Volumes 80
NFS clients
what happens when setting NFSv4.1 ACLs on
Infinite Volumes 88
what happens when they set UNIX permissions on
Infinite Volumes 87
NFS servers
creating for Infinite Volumes 80
NFSv4.1 ACLs
on Infinite Volumes 89
what happens when set by NFS clients, on Infinite
Volumes 88
NFSv4.1 clients
showing ACLs 90
NIS name services
Infinite Volumes 79
nodes
- balancing capacity for Infinite Volumes 20, 36
- expanding Infinite Volumes by associating more 174
- Infinite Volumes, requirements 20
- losing quorum affects Snapshot copies of Infinite Volumes 97
- requirements for SVMs with Infinite Volume 20

NTFS file permissions
- appearance on NFSv4.1 clients 90
- Infinite Volume defaults 76
- Infinite Volumes 89
- what happens when set by Windows clients, on Infinite Volumes 88

O
- offline
  - constituents 190
  - forcing constituents to a state of 191

OnCommand Unified Manager
- See data policies

OnCommand Workflow Automation
- See workflows

online
- forcing constituents to a state of 191
- operations
  - and mixed state of Infinite Volume 190
  - constituents 182
  - Infinite Volumes 182
  - on Infinite Volumes 180

P
- partial state
  - Snapshot copies 97
  - permissions
    - Infinite Volume defaults 76
- planning
  - Infinite Volumes with storage classes 198
  - Infinite Volumes, setup of 30
  - to create SVMs 43
- platforms
  - data constituent size 35
  - identifying supported Infinite Volume constituent size 60
  - supported for Infinite Volumes 20
- policies, data
  - configuring users or directories for rules in 212
- protocols
  - CIFS support on Infinite Volumes 72
  - considerations for tape restore to storage classes 227
  - NFS support on Infinite Volumes 71
  - supported on destination Infinite Volumes 120

R
- referrals
  - Infinite Volumes support 71
- remote copy offload
  - Infinite Volumes support 72
- repository Vserver
  - See SVMs with Infinite Volume
- requirements
  - for Infinite Volumes 21
  - Infinite Volumes 20
  - Infinite Volumes with storage classes, tools for 23
  - Infinite Volumes, aggregate 22
  - Infinite Volumes, node 20
- resizing
  - Infinite Volumes 176
  - Infinite Volumes with storage classes 210
  - Infinite Volumes, determining aggregate space availability before resizing 170
- restoring
  - files from tape to an Infinite Volume 157
  - restricted
    - forcing constituents to a state of 191
  - reversing
    - a data protection mirror relationship for Infinite Volumes with storage classes 222
    - data protection mirror relationship for Infinite Volumes 121
- root volumes
  - restoring for SVMs with Infinite Volume 194
- rules
  - defined 16

S
- SACLs
  - Infinite Volumes support 71, 72
- schedules
  - Snapshot copies of Infinite Volumes, default for 96
- security
  - Infinite Volumes support 71, 72
  - security styles
    - comparison of unified and mixed 91
    - supports Infinite Volumes 71
    - unified 76
- setting up
  - SVMs 43
- setup
  - of Infinite Volumes, planning for 30
- setup tasks
  - for Infinite Volume with storage classes 27
  - for Infinite Volumes 25
- share ACLs
  - controlling SMB access to Infinite Volumes 85
  - how they work on Infinite Volumes 75
  - Infinite Volume default 76
- shares
  - creating on Infinite Volumes 81
  - Infinite Volumes 73
- SMB
  - controlling access to Infinite Volumes with share ACLs 85
  - Infinite Volumes support 72
  - See also CIFS
- SMB 3 over Hyper-V
  - Infinite Volumes support 72
- SMB access
  - setting up for Infinite Volumes 81
- SMB clients
  - setting permissions of UNIX users 88
- SnapDiff
aggregate selection for namespace mirror constituents 147
backup applications
  requirements to use 141
deciding whether to verify space requirements in Infinite Volumes 149
defined 141
disabling for Infinite Volumes 157
effect on moving constituents from an aggregate 190
enabling for Infinite Volumes with storage classes 221, 227
enabling for tape backup of Infinite Volumes 151
guidelines for using network interfaces 142
how expansion works for Infinite Volumes with storage classes 229
how it works with namespace mirror constituents 141
incremental tape backups of Infinite Volumes, using 141
namespace mirror constituents creation for tape backup of Infinite Volumes 142
namespace mirror constituents for 14 nodes
  number required for namespace mirror constituents 147
reennabling it after expanding destination Infinite Volume 178
reennabling it after expanding destination Infinite Volumes with storage classes 178
required for incremental tape backup of Infinite Volumes 139
selecting a Snapshot copy for 152
space considerations 32
space requirements for namespace mirror constituents 147
verifying space requirements in Infinite Volumes 149
workflow for tape backup and restore of Infinite Volumes 139
See also tape backup
Snapshot policies
  configuring for Infinite Volumes with storage classes 211
  enabling for Infinite Volumes with storage classes 221
  for Snapshot schedules 95
  how volumes inherit from SVMs 96
  Infinite Volumes, default for 96
source Infinite Volumes
  namespace mirror constituents on 127
space
  how it is allocated in new Infinite Volumes 37
  methods of creating in an aggregate 163
  methods of creating in an Infinite Volume 163
space allocation
  when expanding Infinite Volumes 168
space allocation examples
  for Infinite Volumes 38
space guarantee
  how affects aggregate selection for destination
  Infinite Volumes 131
space information
  commands for displaying 159
space management
  unsupported features on Infinite Volumes 164
space reservations
  unsupported by Infinite Volumes 164
space savings
  on Infinite Volumes through compression 67
  on Infinite Volumes through deduplication 66
  with deduplication, viewing 159
space usage
  how to determine and control volume, in aggregates 160
  how to determine in an aggregate 162
  monitoring storage classes for 228
states
  deleting invalid Snapshot copies of Infinite Volumes 102
  forcing Infinite Volumes to online, offline, or restricted 191
  mixed state of Infinite Volumes 190
  Snapshot copies 97
  viewing for Snapshot copies of Infinite Volumes 101
storage classes
  adding capacity for Infinite Volumes with 230
  adding to an Infinite Volumes without storage classes 208
  adding to Infinite Volumes 210
  aggregate overcommitment in Infinite Volumes with 30
  commands for rebalancing used capacity 183
  configuring Snapshot copies for volumes with 98
  configuring Snapshot policies for Infinite Volumes with 211
  considerations for deciding whether to use 25
  creating aggregates for Infinite Volumes with 207
  creating destination Infinite Volumes with 220
  creating Infinite Volumes with 209
  creating mirror copies of Infinite Volumes with 220
  creating SVMs with Infinite Volume to support 209
  customizing predefined workflows for 204, 205
deciding whether to use 24

Snapshot copies
  available for tape backup 142
  client access on destination Infinite Volumes 120
  creating for Infinite Volumes 98
  default schedule for Infinite Volumes 96
  deleting when invalid 102
directory for 97
  for backups 95
guidelines for Infinite Volumes 95
  incremental tape backups of Infinite Volumes, using 141
  Infinite Volumes 97
  Infinite Volumes with storage classes, impact on 219
  selecting for SnapDiff to use 152
  states 97
  used by SnapDiff to identify file and directory differences 141
  viewing for constituents of an Infinite Volume 99
  viewing the state of 101
  what copies to access 97
  where to find more information about 19

Snapshot copies of Infinite Volumes
  client access is temporarily disabled 97
guidelines for 95
default definitions 198
definition and example of 15
disabled commands for Infinite Volumes with 200
editing rules in a data policy 212
effect on aggregates 24
enabling SnapDiff for Infinite Volumes with 227
how array LUNs are supported 204
how customized settings affect predefined workflows 203
how data policies work 16
how efficiency works on 199
how expansion works for Infinite Volumes with 229
how offline constituents affect Auto Balance Volume 186
how they relate to storage services for Infinite Volumes 201
how used capacity is rebalanced across data constituents of 184
impact on file access to Infinite Volumes 211
impact on moving constituents from aggregates when SnapDiff enabled 190
increasing the size of Infinite Volumes when in a mirror relationship 230
Infinite Volumes with, introduction to 15
managing 231
managing a data policy in JSON format for 213
managing with tools and commands 213
methods to create more aggregate space for 229
modifying a data policy to filter data into 212
modifying settings for 231
moving constituents from aggregates for 234
planning to create Infinite Volumes with 198
preparing to move constituents off aggregates 232
protocol considerations for tape restore 227
rebalancing the used capacity across data constituents of 231
reenabling SnapDiff after expanding a destination Infinite Volume with 178
requirements for setting up file access 211
resizing 210
retrieving data from Infinite Volumes with 222
setup and management tasks for Infinite Volumes with 27
Snapshot copies of Infinite Volumes, impact on 219
tape backup of Infinite Volumes, impact on 227
tools for creating and managing Infinite Volumes with 202
tools for creating mirror relationships for Infinite Volumes with 219
tools required to create and manage Infinite Volumes with 23
using mirroring technology on Infinite Volumes with 219
using with Infinite Volumes 198
viewing aggregate requirements for 233
viewing the names of 233
viewing the used capacity of data constituents 187
when to use 24
worksheet for planning aggregates 39
See also Infinite Volumes with storage classes
storage classes, Infinite Volumes with
storage efficiency
enabling 68
on Infinite Volumes, viewing details 181
where to find more information about 19
storage services
definition of 201
how it relates to storage classes for Infinite Volumes 201
See also storage classes
suggestions
how to send feedback about documentation 238
SVM peering
creating peer relationships between SVMs with Infinite Volume (dedicated aggregates) 107
creating peer relationships between SVMs with Infinite Volume (shared aggregates) 115
SVMs
assigning aggregates for all SVMs in the cluster 53
configuring 48
creating Infinite Volumes using default settings 55
default Snapshot policies associated with 96
guidelines for creating 44
language configurations 46
language options 44
planning to create 43
requirements for Infinite Volumes 21
setting up 43
where to find information about features in 19
SVMs with Infinite Volume
aggregate requirements 22
assigning aggregates 34, 220
assigning aggregates (dedicated aggregates) 54, 107
assigning aggregates (shared aggregates) 50, 115
assigning dedicated aggregates 51
assigning shared aggregates 49
CIFS support 72
configuration requirements 20, 21
creating 43, 47
creating (dedicated aggregates) 107
creating (shared aggregates) 115
creating a new root volume for 194
creating for destination Infinite Volumes with storage classes 220
creating Infinite Volumes 64
creating Infinite Volumes with manually selected aggregates 57
creating Infinite Volumes with smaller constituents for 60
creating with storage classes enabled 209
data policies, JSON considerations for 214
data policies, when applied to files 17
default junction for 73, 182
delegation 34
exporting a data policy 222
file permissions 75
guidelines for using network interfaces for SnapDiff 142
how a data policy filters file in 16
how export policies affect access to 74
importing a data policy in JSON format 214
methods of aggregate selection for destination Infinite Volumes 129
mount point 73, 182
namespace 23, 73, 182
NFS support 71
node requirements 20
resizing the Infinite Volume by adding disks to aggregates 171
resizing the Infinite Volume by associating more aggregates 172
resizing the Infinite Volume by associating more nodes with 174
specifying an aggregate list (dedicated aggregates) 107
specifying an aggregate list (shared aggregates) 115
vserver data-policy commands 213

tape backup
(incremental), setting up for destination Infinite Volumes 154
creating baseline for incremental 153
creating baseline for incremental (destination Infinite Volumes) 155
deciding whether to verify space requirements for namespace mirror constituents 149
effect of SnapDiff on moving constituents from an aggregate 190
enabling SnapDiff for 151, 227
how namespace mirror constituents support SnapDiff for 142
how namespace-related constituents are used 142
Infinite Volumes with storage classes, impact on 227
of Infinite Volume, unsupported features 139
of Infinite Volume, using CIFS or NFS 139
reenabling SnapDiff after expanding destination Infinite Volumes 178
reenabling SnapDiff after expanding destination Infinite Volumes with storage classes 178
selecting a Snapshot copy of Infinite Volumes for 152
setting up incremental 149
Snapshots copies available for 142
transferring Snapshot copies to destination Infinite Volumes 154
tape backup and restore
CIFS and NFS considerations 140
Infinite Volumes, workflow for 139
where to find more information about 19
tape drives
supported topology for Infinite Volumes 140
tape libraries
supported topology for Infinite Volumes 140
tape restore
protocol considerations for storage classes 227
restoring files to an Infinite Volume 157
thin provisioning
customizing settings for Infinite Volumes with storage classes 204, 205
for Infinite Volumes 30
modifying setting for storage classes 231
tools
used to modify settings for storage classes 203
transfers
mirror copy, size calculation of Infinite Volumes for 127
troubleshooting
creating SVMs with Infinite Volume fails 47
footprint information not available 65
Twitter
how to receive automatic notification of documentation changes 238

U
Unified Manager
See data policies
unified security style
compared to mixed security style 91
Infinite Volumes 76
set file permissions 76
view file permissions 76
UNIX clients
what happens when they set permissions on Infinite Volumes 87
UNIX file permissions
Infinite Volume defaults 76
UNIX users
setting permissions in SMB clients 88
unsupported features
Infinite Volumes 17
on Infinite Volumes for space management 164
used space
how to determine and control in aggregates, by volume 160
how to determine in aggregate 162
user mapping
Infinite Volumes 79
users
configuring for rules in a data policy 212

V
vacating
aggregates of constituents 189, 234
valid state
Snapshot copies 97
volume footprint
described 162
volume guarantees
Infinite Volumes 164
volume show-footprint command
understanding output 160
volumes
comparison of FlexVol and Infinite 12
creating a new root volume for SVMs with Infinite Volume 194
how Snapshot policies are associated with 96
how space usage is balanced 164
how to determine space usage of, in aggregates 160
Infinite Volume defined 11
sharing of aggregates 33
See also aggregates 33
Volumes, Infinite
determining aggregate space availability before resizing 170
expanding by adding disks to aggregates 171
expanding by associating more aggregates 172
expanding by associating more nodes with 174
Vservers
See SVMs
VSS shadow copy
Infinite Volumes support 72

W

Windows users
what happens when SMB clients set NTFS file permissions on Infinite Volumes 88
Workflow Automation
See workflows
workflows
adding storage classes to an Infinite Volume 210
configuring Snapshot policies for Infinite Volumes with storage classes 211, 221
converting to Infinite Volumes with storage classes 208
creating and initializing data protection mirror relationships 221
creating destination Infinite Volumes with storage classes 220
creating Infinite Volumes with storage classes 209
creating SVMs 209
customizing predefined storage class workflows 204, 205
definitions of default storage classes 198
enabling SnapDiff 221
enabling SnapDiff for tape backup 227
expanding Infinite Volumes with storage classes 230
how customized storage classes affect 203
how expansion works for source and destination Infinite Volumes with storage classes 229
how storage classes support array LUNs 204
modifying storage class settings with 231
overview of setting up Infinite Volumes 25
overview of setting up Infinite Volumes with storage classes 27
resizing an Infinite Volume 176
resizing Infinite Volumes with storage classes 230
worksheets
planning aggregates 39
writing files
how Infinite Volumes balance space usage when 164