



## FlashArray User Guide

*Thursday, August 22, 2019 17:35*

## Contents

<b>About the Guide .....</b>	<b>vii</b>
Organization of the Guide .....	vii
A Note on Format and Content .....	vii
Related Documentation .....	vii
Contact Us .....	viii

## Part 1: FlashArray Overview

<b>Chapter 1: FlashArray Storage Capacity and Utilization .....</b>	<b>11</b>
Array Capacity and Storage Consumption .....	11
Volume and Snapshot Storage Consumption .....	12
FlashArray Data Lifecycle .....	15
<b>Chapter 2: FlashArray Concepts and Features .....</b>	<b>19</b>
Arrays .....	19
Hardware Components .....	20
Network Interface .....	20
Storage .....	23
Users and Security .....	34
Industry Standards .....	35
Troubleshooting and Logging .....	36
<b>Chapter 3: Conventions .....</b>	<b>39</b>
Object Names .....	39
Volume Sizes .....	39
IP Addresses .....	41
Storage Network Addresses .....	41

## Part 2: Using the GUI to Administer a FlashArray

<b>Chapter 4: GUI Overview .....</b>	<b>45</b>
GUI Navigation .....	46
GUI Login .....	50
<b>Chapter 5: Dashboard .....</b>	<b>53</b>
Capacity .....	54
Recent Alerts .....	55
Hardware Health .....	55
Performance Charts .....	55

<b>Chapter 6: Storage .....</b>	<b>59</b>
Array .....	60
Hosts and Host Groups .....	74
Volumes .....	84
Protection Groups .....	102
Pods .....	119
<b>Chapter 7: Analysis .....</b>	<b>123</b>
Performance .....	125
Capacity .....	129
Replication .....	131
<b>Chapter 8: Health .....</b>	<b>133</b>
Hardware .....	133
Alerts .....	138
Connections .....	141
<b>Chapter 9: Settings .....</b>	<b>145</b>
System .....	145
Network .....	165
Users .....	173
Software .....	187

## Part 3: Using the CLI to Administer a FlashArray

<b>Chapter 10: CLI Overview .....</b>	<b>197</b>
CLI Command Syntax and Conventions .....	197
CLI Output .....	199
CLI Login .....	205
CLI Help .....	206
<b>Chapter 11: CLI Getting Started .....</b>	<b>501</b>
Creating Volumes .....	501
Creating Hosts .....	501
Connecting Hosts and Volumes .....	502
Resizing Volumes .....	503
Destroying Volumes .....	505
Ongoing Administration of Hosts .....	507
Monitoring I/O Performance .....	508
Using listobj Output .....	508

## Part 4: Supplementary Information

**Appendix A: The Pure Storage Glossary ..... 513**



## About the Guide

The Pure Storage® FlashArray User Guide is written for array administrators who view and manage the Pure Storage FlashArray storage system.

FlashArray arrays are administered through the Purity for FlashArray (Purity//FA) graphical user interface (GUI) or command line interface (CLI). Users should be familiar with system, storage, and networking concepts, and have a working knowledge of Windows or UNIX.

## Organization of the Guide

The guide is organized into the following major sections:

### **Part 1, “FlashArray Overview” [9]**

Defines FlashArray storage capacity measurement and describes the physical and virtual objects managed by a FlashArray administrator.

### **Part 2, “Using the GUI to Administer a FlashArray” [43]**

Describes the use of the browser-based Purity//FA graphical user interface (GUI) to administer a FlashArray.

### **Part 3, “Using the CLI to Administer a FlashArray” [195]**

Describes the Purity//FA command line interface (CLI) and its use in FlashArray administration.

## A Note on Format and Content

FlashArray technology is evolving rapidly. As with all advanced information technologies, once basic architecture is in place, implementation develops at different rates in its different facets, each preceded by feature-by-feature detailed design.

This edition of the guide describes the properties and behavior of arrays that run the Purity//FA 5.3.0 release. It may include information about planned capabilities whose external form has been specified at the time of the release. Such material is included to provide users of this release with information for design planning purposes, and is subject to change as new functionality is implemented. Material relating to not-yet-implemented functionality is identified as such in the text.

## Related Documentation

Refer to the following related guides to learn more about the FlashArray:

- **Pure Storage REST API Guide.** The Pure Storage REpresentational State Transfer (REST) API uses HTTP requests to interact with the FlashArray resources. The Pure Storage REST API Guide provides an overview of the REST API and a list of all available resources.
- **Pure Storage SMI-S Provider Guide.** Purity//FA includes the Pure Storage Storage Management Initiative Specification (SMI-S) provider, which allows FlashArray administrators to manage the array using an SMI-S client over HTTPS. The Pure Storage SMI-S Provider Guide describes functionality the provider supports and information on connecting to the provider.

- **Third-party plugin guides.** Pure Storage packages and plug-ins extend the functionality of the FlashArray. Available packages and plug-ins include, but are not limited to, VSS Hardware Provider, FlashArray OpenStack Cinder Volume Driver, FlashArray Storage Replication Adapter, Management Plugin for vSphere, and vRealize Operations Management Pack.

All related guides are available in the Pure1 Knowledge site at <https://support.purestorage.com>.

## Contact Us

Pure Storage is always eager to hear from you.

### Documentation Feedback

We welcome your feedback about Pure Storage documentation and encourage you to send your questions and comments to <[documentfeedback@purestorage.com](mailto:documentfeedback@purestorage.com)>.

### Product Support

If you are a registered Pure Storage user, log in to the Pure Storage Support website at <https://support.purestorage.com> to browse our knowledge base, view the status of your open support cases, and view the details of past support cases.

You can also contact Pure Storage Support at <[support@purestorage.com](mailto:support@purestorage.com)>.

### General Feedback

For all other questions and comments about Pure Storage, including products, sales, service, and just about anything that interests you about data storage, email <[info@purestorage.com](mailto:info@purestorage.com)>.

# Part 1: FlashArray Overview



## Chapter 1. FlashArray Storage Capacity and Utilization

The two keys to FlashArray cost-effectiveness are highly efficient provisioning and data reduction. One of an array administrator's primary tasks is understanding and managing physical and virtual storage capacity. This chapter describes the ways in which physical storage and virtual capacity are used and measured.

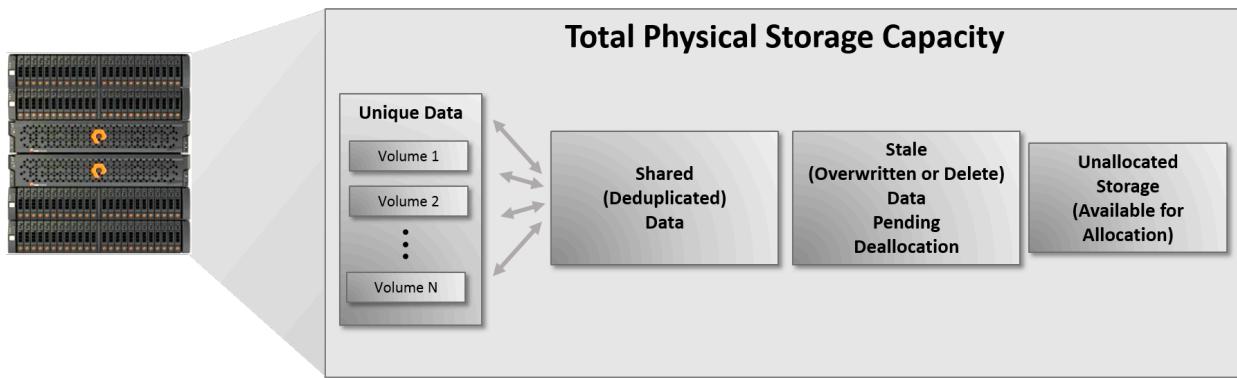
### Array Capacity and Storage Consumption

Administrators monitor physical storage consumption and manage it by adding storage capacity or relocating data sets when available (unallocated) storage becomes dangerously low.

#### Physical Storage States

In a FlashArray, the physical storage that holds data can be in one of four states: unique, shared, stale, and unallocated.

**Figure 1.1: FlashArray Physical Storage States**



- **Unique data.** Reduced host-written data that is not duplicated elsewhere in the array and descriptive metadata.
- **Shared data.** Deduplicated data. Data that comprises the contents of two or more sector addresses in the same or different volumes (FlashArray deduplication is array-wide).
- **Stale data.** Overwritten or deleted data. Data representing the contents of virtual sectors that have been overwritten or deleted by a host or by an array administrator. Such storage is deallocated and made available for future use by the continuous storage reclamation process, but because the process runs asynchronously in the background, deallocation is not immediate.
- **Unallocated storage.** Available for storing incoming data.

#### Reporting Array Capacity and Storage Consumption

Array physical storage capacity and the amount of storage occupied by data and metadata is displayed through the GUI (**Storage > Dashboard**) and CLI (`purearray list --space`). For example,

```
$ purearray list --space
Name   Capacity ... System Shared Space Volumes Snapshots Total
FLASH  10.05T   ... 1.77T  2.78T      2.09T  409.08G  7.04T
```

## Volume and Snapshot Storage Consumption

FlashArrays present disk-like *volumes* to connected hosts. They also maintain immutable *snapshots* of volume contents. As with conventional disks, a volume's storage capacity is presented as a set of consecutively numbered 512-byte *sectors* into which data can be written and from which it can be read. Hosts read and write data in *blocks*, which are represented as consecutively-numbered sequences of sectors.

Purity//FA allocates (also known as "*provisions*") storage for data written by hosts, and *reduces* the data before storing it.

### Provisioning

The *provisioned size* of a volume is its capacity as reported to hosts. As with conventional disks, the size presented by a FlashArray volume is nominally fixed, although it can be increased or decreased by an administrator. To optimize physical storage utilization, however, FlashArray volumes are thin and micro provisioned.

- **Thin provisioning.** Like conventional arrays that support thin provisioning, FlashArrays do not allocate physical storage for volume sectors that no host has ever written, or for *trimmed* (expressly deallocated by host or array administrator command) sector addresses.
- **Micro provisioning.** Unlike conventional thin provisioning arrays, FlashArrays allocate only the exact amount of physical storage required by each host-written block after reduction. In FlashArrays, there is no concept of allocating storage in "chunks" of some fixed size.

### Data Reduction

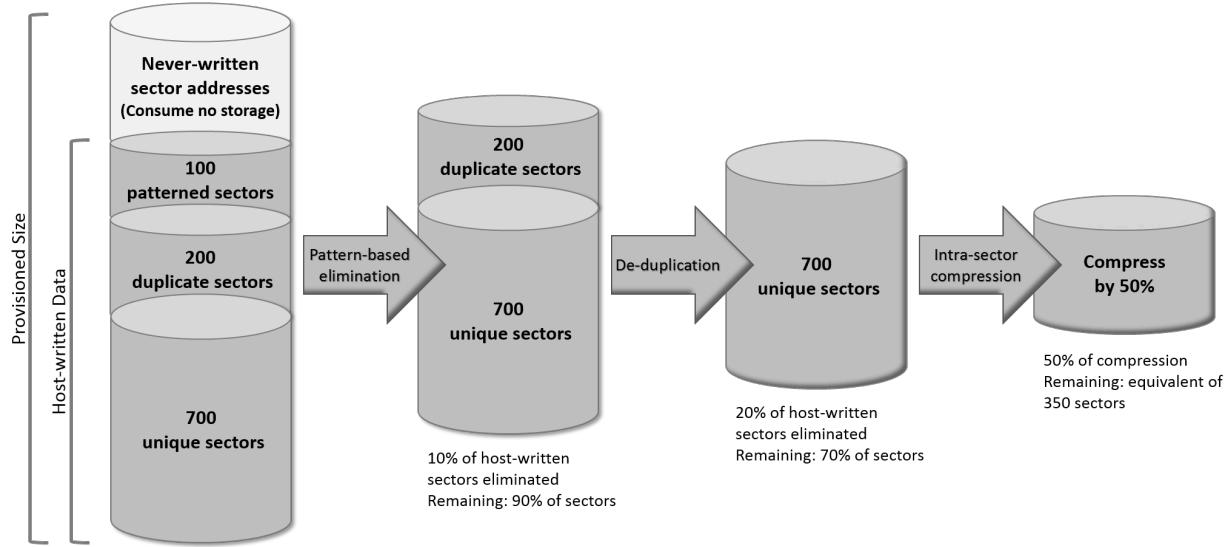
The second key to FlashArray cost effectiveness is *data reduction*, which is the elimination of redundant data through pattern elimination, duplicate elimination, and compression.

- **Pattern elimination.** When Purity//FA detects sequences of incoming sectors whose contents consist entirely of repeating patterns, it stores a description of the pattern and the sectors that contain it rather than the data itself. The software treats zero-filled sectors as if they had been trimmed—no space is allocated for them.
- **Duplicate elimination.** Purity//FA computes a hash value for each incoming sector and attempts to determine whether another sector with the same hash value is stored in the array. If so, the sector is read and compared with the incoming one to avoid the possibility of aliasing. Instead of storing the incoming sector redundantly, Purity//FA stores an additional reference to the single data representation. Purity//FA deduplicates data *globally* (across an entire array), so if an identical sector is stored in an array, it is a deduplication candidate, regardless of the volume(s) with which it is associated.
- **Compression.** Purity//FA attempts to compress the data in incoming sectors, cursorily upon entry, and more exhaustively during its *continuous storage reclamation* background process.

Purity//FA applies pattern elimination, duplicate elimination, and compression techniques to data as it enters an array, as well as throughout the data's lifetime.

The following hypothetical example illustrates the cumulative effect of FlashArray data reduction on physical storage consumption.

**Figure 1.2: Data Reduction Example**



In the example, hosts have written data to a total of 1,000 unique sector addresses through:

- **Pattern elimination.** 100 blocks contain repeated patterns, for which Purity//FA stores metadata descriptors rather than the actual data.
- **Duplicate elimination.** 200 blocks are duplicates of blocks already stored in the array; Purity//FA stores references to these rather than duplicating stored data.
- **Compression.** The remaining 70% of blocks compress to half their host-written size; Purity//FA compresses them before storing, and during continuous storage reclamation.

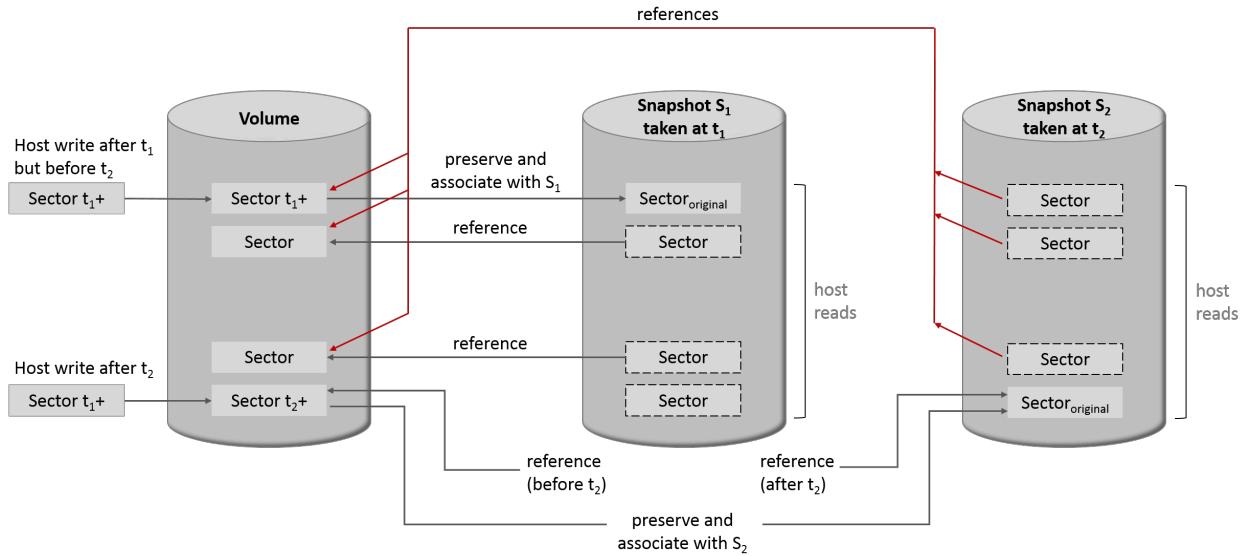
Therefore, the net physical storage consumed by host-written data in this example is 35% of the number of unique volume sector addresses to which hosts have written data.

The data reduction example is hypothetical; each data set reduces differently, and unrelated data stored in an array can influence reduction. Nevertheless, administrators can use the array and volume measures reported by Purity//FA to estimate the amount of physical storage likely to be consumed by data sets similar to those already stored in an array.

## Snapshots and Physical Storage

FlashArray snapshots occupy physical storage only in proportion to the number of sectors of their source volumes that are overwritten by hosts.

**Figure 1.3: Snapshot Space Consumption Example**



In the example, two snapshots of a volume, S<sub>1</sub> and S<sub>2</sub>, are taken at times t<sub>1</sub> and t<sub>2</sub> (t<sub>1</sub> prior to t<sub>2</sub>). If a host writes data to the volume after t<sub>1</sub> but before t<sub>2</sub>, Purity//FA preserves the overwritten sectors' original contents and associates them with S<sub>1</sub> (i.e., space accounting charges them to S<sub>1</sub>). If in the interval between t<sub>1</sub> and t<sub>2</sub> a host reads sectors from snapshot S<sub>1</sub>, Purity//FA delivers:

- For sectors not modified since t<sub>1</sub>, current sector contents associated with the volume.
- For sectors modified since t<sub>1</sub>, preserved volume sector contents associated with S<sub>1</sub>.

Similarly, if a host writes volume sectors after t<sub>2</sub>, Purity//FA preserves the overwritten sectors' previous contents and associates them with S<sub>2</sub> for space accounting purposes. If a host reads sectors from S<sub>2</sub>, Purity//FA delivers:

- For sectors not modified since t<sub>2</sub>, current sector contents associated with the volume.
- For sectors modified since t<sub>2</sub>, preserved volume sector contents associated with S<sub>2</sub>.

If, however, a host reads sectors from S<sub>1</sub> after t<sub>2</sub>, Purity//FA delivers:

- For sectors modified since t<sub>2</sub>, current sector contents associated with the volume.
- For sectors modified between t<sub>1</sub> and t<sub>2</sub>, preserved volume sector contents associated with S<sub>1</sub>.
- For sectors modified since t<sub>1</sub>, preserved volume sector contents associated with S<sub>2</sub>.

If S<sub>1</sub> is destroyed, storage associated with it is reclaimed because there is no longer a need to preserve pre-update content for updates made prior to t<sub>2</sub>.

If S<sub>2</sub> is destroyed, however, storage associated with it is preserved and associated with S<sub>1</sub> because the data in it represents pre-update content for sectors updated after t<sub>1</sub>.

To generalize, for volumes with two or more snapshots:

- **Destroying the oldest snapshot.** Space associated with the destroyed snapshot is reclaimed after the 24-hour eradication pending period has elapsed or after an administrator purposely eradicates the destroyed snapshot.
- **Destroying other snapshots.** Space associated with the destroyed snapshot is associated with the next older snapshot unless it is already reflected there because the same sector was written both after the next older snapshot and after the destroyed snapshot, in which case it is reclaimed.

## Reporting Volume and Snapshot Storage Consumption

Because data stored in a FlashArray is virtualized, thin-provisioned, and reduced, volume storage is monitored, managed, and displayed from two viewpoints:

- **Host view.** Displays the *virtual* storage capacity (size) and consumption as seen by the host storage administration tools.
- **Array view.** Displays the *physical* storage capacity occupied by data and the metadata that describes and protects it.

Volume size and physical storage consumption data is displayed through the GUI (**Storage > Volumes**) and CLI (`purevol list --space`).

For example (single output displayed over two rows),

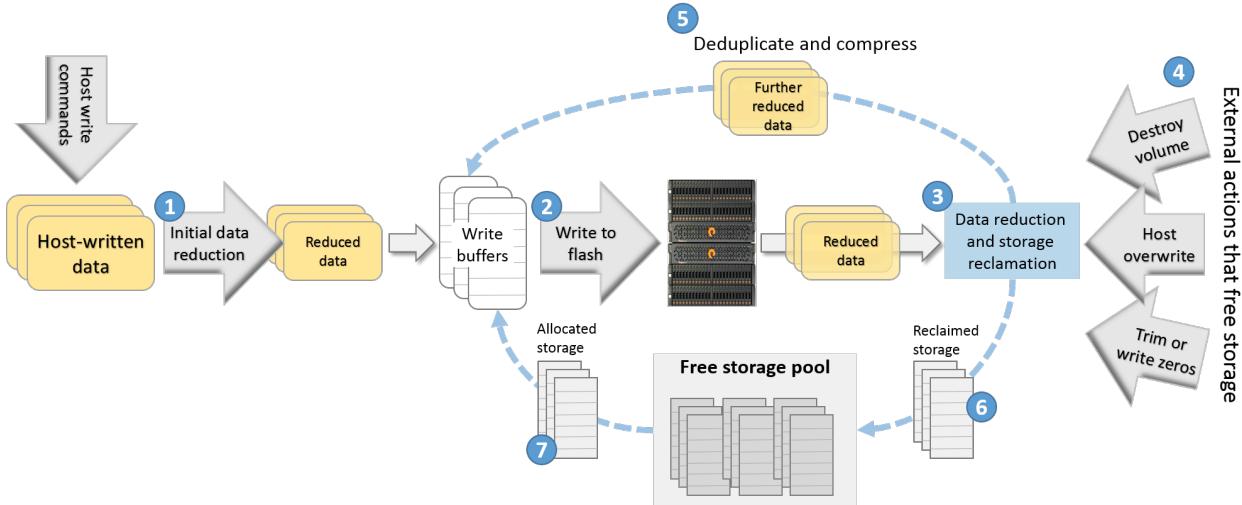
```
$ purevol list --space
Name  Size  Thin Provisioning  Data Reduction  Total Reduction ...  Total
VOL1  3T    50%                3.7 to 1       7.4 to 1        ...  664.64G
```

...

```
$ purevol list --space
Name  Size  ...  Volume   Snapshots  Shared Space  System  Total
VOL1  3T    ...  484.78G  31.85G    -           148.01G  664.64G
```

## FlashArray Data Lifecycle

Data stored in a FlashArray undergoes continuous reorganization to improve physical storage utilization and reclaim storage occupied by data that has been superseded by host overwrite or deletion.

**Figure 1.4: FlashArray Physical Storage Life Cycle**


The steps enumerated in Figure 1.4 are as follows:

#### Host Write Processing (1)

Data written by hosts undergoes initial processing as it enters an array. The result is data that has undergone initial reduction, and been placed in write buffers.

#### Writing to Persistent Storage (2)

As write buffers fill, they are written to segments of persistent flash storage.

#### Segment Selection (3)

A Purity//FA background process continually monitors storage segments for data that has been obsoleted by host overwrites, volume destruction or truncation, or trimming (4). Segments that contain a predominance of obsolete data become high-priority candidates for storage reclamation and further reduction of the live data in them.

#### Data Reduction (5)

As Purity//FA processes segments, it opportunistically deduplicates and compresses (5) the live data in them, using more exhaustive algorithms than those used during initial reduction (1). Reprocessed data is moved to write buffers that are being filled; thus, write buffers generally contain a combination of new data entering the array and data that has been moved from segments being vacated to improve utilization.

#### Storage Reclamation (6)

As live data is moved from segments, they are returned to the pool of storage available for allocating segments. Purity//FA treats all of an array's flash module storage as a single homogeneous pool.

#### Reallocation (7)

Purity//FA allocates segments of storage from the pool of available flash modules as they are required. Typically, the software fills write buffers for multiple segments concurrently. This allows the software to consolidate different types of data (e.g., highly-compressible, highly-duplicated, etc.) so that the most appropriate policies can be applied to them.

Occasionally, continuous data reduction can result in behavior unfamiliar to administrators experienced with conventional arrays. For example, as Purity//FA detects additional duplication and compresses block contents more efficiently, a volume's physical storage occupancy may decrease, even as hosts write more data to it.



## Chapter 2. FlashArray Concepts and Features

This chapter provides a brief introduction to the FlashArray hardware, networking, and storage components and describes where they are managed in Purity//FA.

Purity//FA is the operating environment that manages the FlashArray. Purity//FA, which comes bundled with the FlashArray, can be administered through a graphical user interface (Purity//FA GUI) or command line interface (Purity//FA CLI).

The FlashArray can also be managed through the Pure Storage® REpresentational State Transfer (REST) API, which uses HTTP requests to interact with resources within Pure Storage. For more information about the Pure Storage REST API, refer to the Pure Storage REST API Reference Guide in the Pure1 Knowledge site at <https://support.purestorage.com>.

### Arrays

A FlashArray controller contains the processor and memory complex that runs the Purity//FA software, buffers incoming data, and interfaces to storage shelves, other controllers, and hosts. FlashArray controllers are stateless, meaning that all metadata related to the data stored in a FlashArray is contained in storage shelf storage. Therefore, it is possible to replace the controller of an array at any time with no data loss.

The following array-specific tasks can be performed through the Purity//FA GUI:

- Display array health through the **Health > Hardware** page.
- Monitor capacity, storage consumption, and performance (latency, IOPS, bandwidth) metrics through the **Analysis** page.
- Change the array name through the **Storage > Array** page.

The same tasks can also be performed through the CLI **purearray** command.

### Connected Arrays

A connection must be established between two arrays in order for data transfer to occur.

For example, two arrays must be connected in order to perform asynchronous replications. When two arrays are connected to replicate data from one array to another, the array where data is being transferred from is called the source array, and the array where data is being transferred to is called the target array.

As another example, two arrays must be connected to perform synchronous replications.

Arrays are connected using a connection key, which is supplied from one array and entered into the other array.

For asynchronous replication, once two arrays are connected, optionally configure network bandwidth throttling to set maximum threshold values for outbound traffic.

Connected arrays are managed through the GUI (**Storage > Array**) and CLI (`purearray connect` command).

Network bandwidth throttling is configured through the GUI (**Storage > Array**) and CLI (`purearray throttle` command).

## Hardware Components

Purity//FA displays the operational status of most FlashArray hardware components. The display is primarily useful for diagnosing hardware-related problems.

Status information for each component includes the functioning status, index numbers, speed at which a component is operating, and reported temperature.

In addition to general hardware component operational status, Purity//FA also displays status information for each flash module and NVRAM module on the array. Status information includes module status, physical storage capacity, module health, and time at which a module became non-responsive.

FlashArray hardware names are fixed. When they are powered on, FlashArray controllers and storage shelves automatically discover each other and self-configure to optimize I/O performance, data integrity, availability, and fault recoverability, all without administrator intervention.

Purity//FA visually identifies certain hardware components through LED lights and numbers. Controllers, flash module bays, NVRAM bays, and storage shelves contain LED lights that can be turned on and off through Purity//FA. Furthermore, storage shelves contain LED integers to uniquely identify shelves in multi-shelf arrays.

Hardware components are displayed and administered through the GUI (**Health > Hardware**) and CLI (`purehw` command).

Flash modules and NVRAM modules are displayed through the GUI (**Health > Hardware**) and CLI (`puredrive` command).

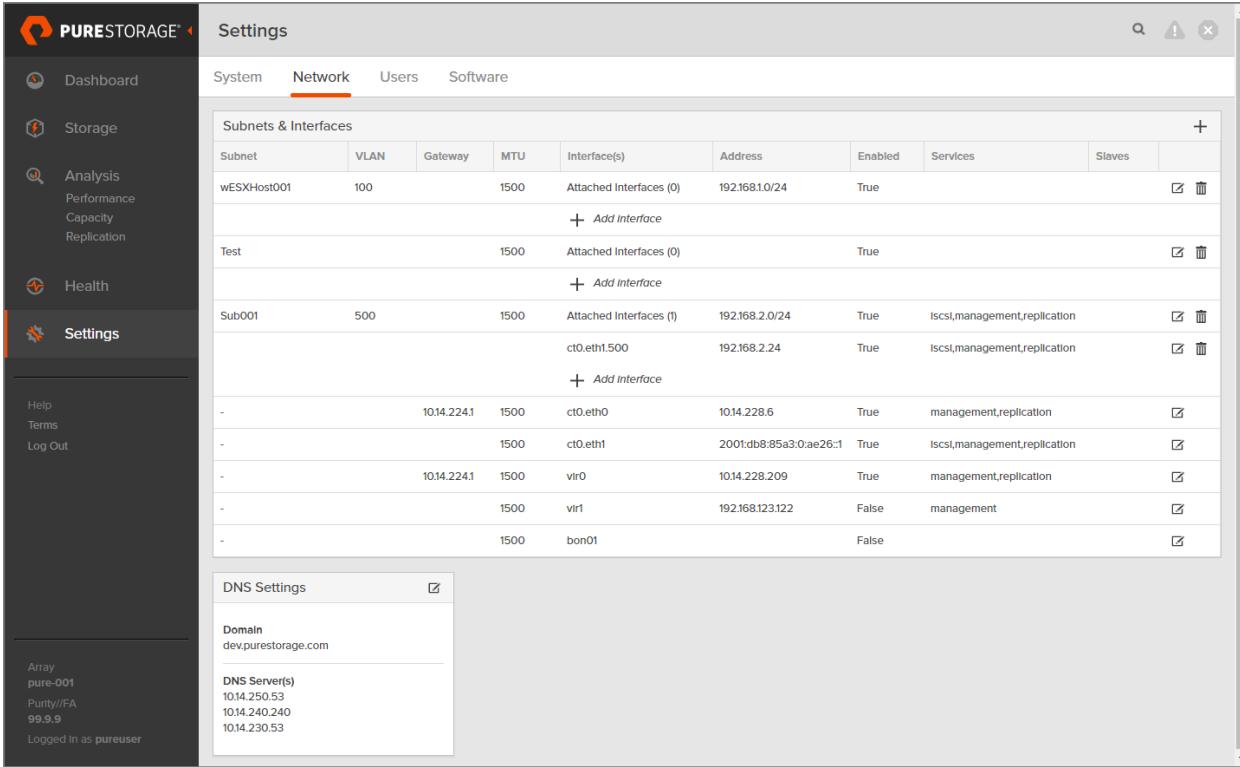
Each hardware component in a FlashArray has a unique name that identifies its location in the array for service purposes.

The hardware component names are used throughout Purity//FA, for instance in the GUI **Health > Hardware** page, and with CLI commands such as `puredrive` and `purehw`.

## Network Interface

View and configure network interface, subnet, and DNS attributes through Purity//FA.

The Purity//FA *network interfaces* manage the bond, Ethernet, virtual, and VLAN interfaces used to connect the array to an administrative network.



The screenshot shows the PureStorage FlashArray Network Settings interface. On the left, a sidebar includes links for Dashboard, Storage, Analysis (Performance, Capacity, Replication), Health, and Settings (selected). The main area has tabs for System, Network (selected), Users, and Software. Under Network, the 'Subnets & Interfaces' section lists subnets and their associated interfaces. A table shows subnets like wESXHost001 and Sub001 with their VLAN IDs, MTUs, and attached interfaces. Services like iSCSI, management, and replication are applied to specific interfaces. Below this is a 'DNS Settings' panel with fields for Domain (dev.purestorage.com) and DNS Server(s) (IP addresses 10.14.250.52, 10.14.240.240, 10.14.230.53).

Each FlashArray controller is equipped with two Ethernet interfaces that connect to a data center network for array administration.

A bond interface combines two or more similar Ethernet interfaces to form a single virtual "bonded" interface with optional slave devices. A bond interface provides higher data transfer rates, load balancing, and link redundancy. A default bond interface, named `replbond`, is created when Purity//FA starts for the first time.

Array administrators cannot create or delete bond interfaces. To create or delete a bond interface, contact Pure Storage Support.

Apply a service to an interface to specify the type of network traffic the device serves. Each interface must have at least one or more services applied. Supported services include 'iscsi,' 'management,' 'nvme-roce,' and 'replication'. For example, apply the replication service to the `replbond` bond interface to channel all replication traffic through that device.

View the network connection attributes, including interface, netmask, and gateway IP addresses, maximum transmission units (MTUs), and the network services (iSCSI, management, NVMe-RoCE, or replication) attached to each network interface.

Enable or disable an interface through Purity//FA at any time. Disabling an interface while an administrative session is being conducted causes the session to lose SSH connection and no longer be able to connect to the controller.

Configure the network connection attributes, including the interface, netmask, and gateway IP addresses, and the MTU. Ethernet and bond interface IP addresses and netmasks are set explicitly, along with the corresponding netmasks. DHCP mode is not supported.

Manage the *domain name system (DNS)* domains that are configured for the array. Each DNS domain can include up to three static DNS server IP addresses. DHCP mode is not supported.

Network interfaces and DNS settings are configured through the GUI (**Settings > Network**) and CLI (`purenetwork` command for network interfaces, and `puredns` for DNS settings).

# Storage

## Volumes

FlashArrays eliminate drive-oriented concepts such as RAID groups and spare drives that are common with disk arrays. Purity//FA treats the entire storage capacity of all flash modules in an array as a single homogeneous pool from which it allocates storage only when hosts write data to volumes created by administrators. Creating a FlashArray volume, therefore, only requires a volume name, to be used in administrative operations and displays, and a provisioned size.

FlashArray volumes are virtual, so creating, renaming, resizing, and destroying a volume has no meaning outside the array.

*Create* a single volume or multiple volumes at one time. Purity//FA administrative operations rely on volume names, so they must be unique within an array.

Creating a volume creates persistent data structures in the array, but does not allocate any physical storage. Purity//FA allocates physical storage only when hosts write data. Volume creation is therefore nearly instantaneous. Volumes do not consume physical storage until data is actually written to them, so volume creation has no immediate effect on an array's physical storage consumption.

*Rename* a volume to change the name by which Purity//FA identifies the volume in administrative operations and displays. The new volume name is effective immediately and the old name is no longer recognized in CLI, GUI, or REST interactions.

*Resize* an existing volume to change the virtual capacity of the volume as perceived by the hosts. The volume size changes are immediately visible to connected hosts. If you decrease (truncate) the volume size, Purity//FA automatically takes an undo snapshot of the volume. The undo snapshot enters a 24-hour *eradication pending* period, after which time the snapshot is destroyed. During the 24-hour pending period, the undo snapshot can be viewed, recovered, or permanently eradicated through the Destroyed Volumes folder. Increasing the size of a truncated volume will not restore any data that is lost when the volume was first truncated.

*Copy* a volume to create a new volume or overwrite an existing one. After you copy a volume, the source of the new or overwritten volume is set to the name of the originating volume.

*Destroy* a volume if it is no longer needed. When you destroy a volume, Purity//FA automatically takes an undo snapshot of the volume. The undo snapshot enters a 24-hour *eradication pending* period. During the 24-hour pending period, the undo snapshot can be viewed, recovered, or permanently eradicated through the Destroyed Volumes folder. Eradicating a volume completely obliterates the data within the volume, allowing Purity//FA to reclaim the storage space occupied by the data. After the 24-hour pending period, the undo snapshot is completely eradicated and can no longer be recovered.

## Volume Snapshots

Volume snapshots are immutable, point-in-time images of the contents of one or more volumes.

### Volume Snapshots vs. Protection Group Snapshots

There are two types of volume snapshots:

## Volume Snapshot

A volume snapshot is a snapshot that captures the contents of a single volume. Volume snapshot tasks include creating, renaming, destroying, restoring, and copying volume snapshots. Volume snapshot tasks are performed through the GUI (**Storage > Volumes**) or CLI (`purevol` command).

## Protection Group Volume Snapshot

A protection group volume snapshot is a volume snapshot that is created from a group of volumes that are part of the same protection group. All of the volume snapshots created from a protection group snapshot are point-in-time consistent with each other.

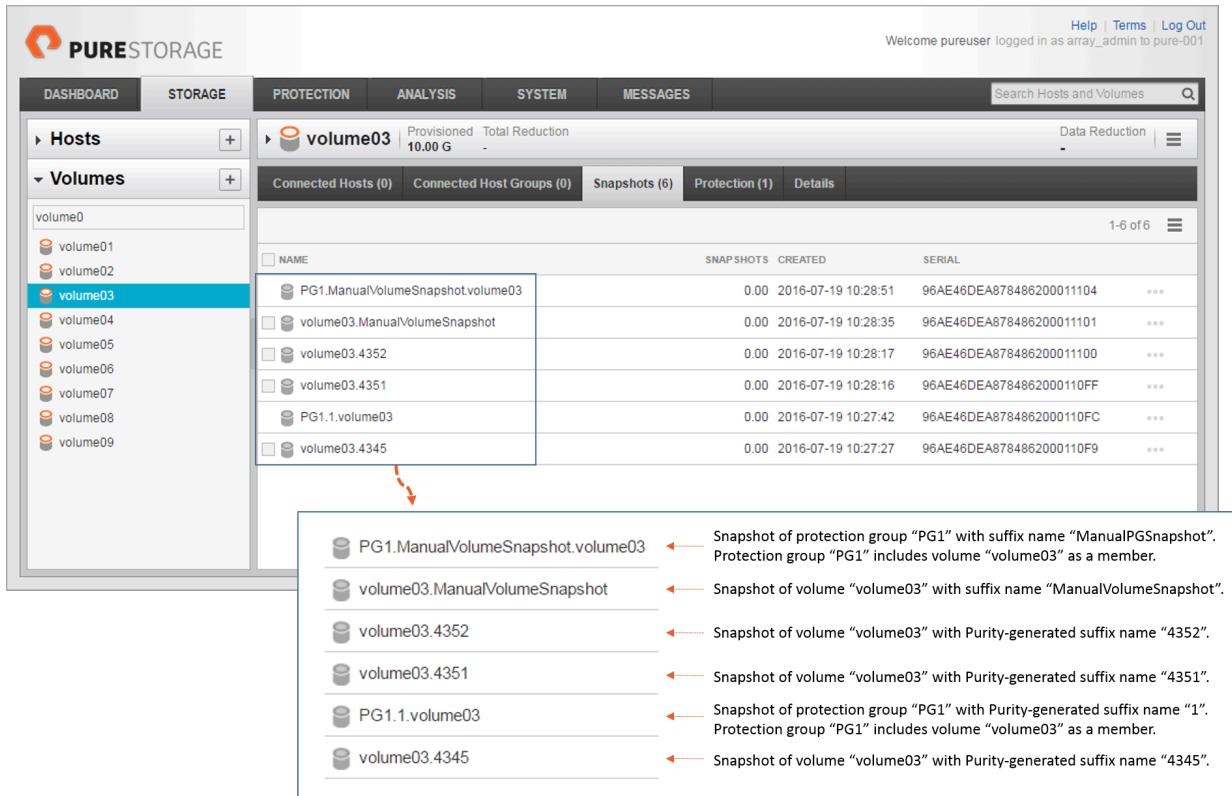
Protection group snapshots can be manually generated on demand or enabled to automatically generate at scheduled intervals. After a protection group snapshot has been taken, it is either stored on the local array or replicated over to a remote (target) array.

Protection group volume snapshot tasks performed through the **Storage > Volumes** page of the GUI or `purevol` command of the CLI are limited to copying snapshots. All other protection group snapshot tasks are performed through the **Storage > Protection Groups** page of the GUI or `purepgroup` command of the CLI.

For more information about protection groups and protection group snapshots, refer to the *Protection Groups and Protection Group Snapshots* section.

All volume snapshots are visible through the Storage > Volumes page.

**Figure 2.1: Storage - Details Pane - Volumes - Snapshots**



NAME	SNAP SHOTS	CREATED	SERIAL
PG1.ManualVolumeSnapshot.volume03	0.00	2016-07-19 10:28:51	96AE46DEA878486200011104
volume03.ManualVolumeSnapshot	0.00	2016-07-19 10:28:35	96AE46DEA878486200011101
volume03.4352	0.00	2016-07-19 10:28:17	96AE46DEA878486200011100
volume03.4351	0.00	2016-07-19 10:28:16	96AE46DEA8784862000110FF
PG1.1.volume03	0.00	2016-07-19 10:27:42	96AE46DEA8784862000110FC
volume03.4345	0.00	2016-07-19 10:27:27	96AE46DEA8784862000110F9

PG1.ManualVolumeSnapshot.volume03	Snapshot of protection group "PG1" with suffix name "ManualPGSnapshot". Protection group "PG1" includes volume "volume03" as a member.
volume03.ManualVolumeSnapshot	Snapshot of volume "volume03" with suffix name "ManualVolumeSnapshot".
volume03.4352	Snapshot of volume "volume03" with Purity-generated suffix name "4352".
volume03.4351	Snapshot of volume "volume03" with Purity-generated suffix name "4351".
PG1.1.volume03	Snapshot of protection group "PG1" with Purity-generated suffix name "1". Protection group "PG1" includes volume "volume03" as a member.
volume03.4345	Snapshot of volume "volume03" with Purity-generated suffix name "4345".

Create a volume snapshot to generate a point-in-time image of the contents of the specified volume(s). Volume snapshot names append a unique number assigned by Purity//FA to the name of the snapped volume. For example, vol01.4166. Optionally specify a suffix to replace the unique number.

The volume snapshot naming convention is **VOL.NNN**, where:

- **VOL** is the name of the volume.
- **NNN** is a unique monotonically increasing number or a manually-assigned volume snapshot suffix name.

Rename a volume snapshot suffix to change the name by which Purity//FA identifies the snapshot in administrative operations and displays. The new snapshot suffix name is effective immediately and the old name is no longer recognized in CLI, GUI, or REST interactions.

Destroy a volume snapshot if it is no longer needed. If you destroy a volume snapshot, Purity//FA automatically takes an undo snapshot. The undo snapshot enters a 24-hour *eradication pending* period, after which time the snapshot is eradicated. During the 24-hour pending period, the undo snapshot can be viewed, recovered, or permanently eradicated through the Destroyed Volumes folder.

Restore a volume from a volume snapshot to bring the volume back to the state it was when the snapshot was taken. When a volume is restored from a volume snapshot, Purity//FA overwrites the entire volume with the snapshot contents. After you restore a volume snapshot, the created date of

the overwritten volume is set to the snapshot created date. Purity//FA automatically takes an undo snapshot of the overwritten volume. The undo snapshot enters a 24-hour *eradication pending* period, after which time the snapshot is destroyed. During the 24-hour pending period, the undo snapshot can be viewed, recovered, or permanently eradicated through the Destroyed Volumes folder.

**Copy** a volume snapshot or protection group volume snapshot to create a new volume or overwrite an existing one. After you copy a snapshot, the source of the new or overwritten volume is set to the name of the originating volume, and the created date of the volume is set to the snapshot created date. If the copy overwrites an existing volume, Purity//FA automatically takes an undo snapshot of the existing volume. The undo snapshot enters a 24-hour *eradication pending* period, after which time the snapshot is destroyed. During the 24-hour pending period, the undo snapshot can be viewed, recovered, or permanently eradicated through the Destroyed Volumes folder.

## Hosts

The host organizes the storage network addresses - the iSCSI Qualified Names (IQNs), NVMe Qualified Names (NQNs), and Fibre Channel World Wide Names (WWNs) - that identify the host computer initiators. The host communicates with the array through the Ethernet or Fibre Channel ports. The array accepts and responds to commands received on any of its ports from any of the IQNs, NQNs, and WWNs associated with a host.

Purity//FA hosts are virtual, so creating, renaming, and deleting a host has no meaning outside the array.

**Create** hosts to access volumes on the array. A Purity//FA host is comprised of a host name and one or more IQNs, NQNs, or WWNs. Host names must be unique within an array.

**Associate** one or more IQNs, NQNs, or WWNs with the host after it has been created. The host cannot communicate with the array until at least one IQN, NQN, or WWN has been associated with it.

iSCSI Qualified Names (IQNs) follow the naming standards set by the Internet Engineering Task Force (see RFC 3720). For example, `iqn.2016-01.com.example:flasharray.491b30d0efd97f25`.

NVMe Qualified Names (NQNs) follow the naming standards set by NVM Express. For example, `nqn.2016-01.com.example:flasharray.491b30d0efd97f25`.

Fibre Channel World Wide Names (WWNs) follow the naming standards set by the IEEE Standards Association. WWNs are comprised of eight pairs of case-insensitive hexadecimal numbers, optionally separated by colons. For example, `21:00:00:24:FF:4C:C5:49`.

Like hosts, IQNs, NQNs, and WWNs must be unique in an array. A host can be associated with multiple storage network addresses, but a storage network address can only be associated with one host.

Host IQNs, NQNs, and WWNs can be added or removed at any time.

**Rename** a host to change the name by which Purity//FA identifies the host in administrative operations and displays. Host names are used solely for FlashArray administration and have no significance outside the array, so renaming a host does not change its relationship with host groups and volumes. The new host name is effective immediately and the old name is no longer recognized in CLI, GUI, or REST interactions.

Optionally, *configure the Challenge-Handshake Authentication Protocol (CHAP)* to verify the identity of the iSCSI initiators and targets to each other when they establish a connection. By default, the CHAP credentials are not set.

To ensure the array works optimally with the host, *set the host personality* to the name of the host operating or virtual memory system. The host personality setting determines how the Purity//FA system tunes the protocol used between the array and the initiator. For example, if the host is running the HP-UX operating system, set the host personality to HP-UX. By default, the host personality is not set. If your system is not listed as one of the valid host personalities, do not set it.

**Delete** a host if it is no longer required. Purity//FA will not delete a host while it has connections to volumes, either private or shared. You cannot recover a host after it has been deleted.

## Host Guidelines

Purity//FA will not create a host if:

- The specified name is already associated with another host in the array.
- Any of the specified IQNs, NQNs, or WWNs are already associated with an existing host in the array.
- The creation of the host would exceed the limit of concurrent hosts, or the creation of the IQN, NQN, or WWN would exceed the limit of concurrent initiators.

Purity//FA will not delete a host if:

- The host has private connections to one or more volumes.

Purity//FA will not associate an IQN, NQN, or WWN with a host if:

- The creation of the IQN, NQN, or WWN would exceed the maximum number of concurrent initiators.
- The specified IQN, NQN, or WWN is already associated with another host on the array.

Hosts are configured through the GUI (**Storage > Hosts**) and CLI (**purehost** command).

## Host Groups

A host group represents a collection of hosts with common connectivity to volumes.

Purity//FA host groups are virtual, so creating, renaming, and deleting a host group has no meaning outside the array.

Create a host group if several hosts share access to the same volume(s). Host group names must be unique within an array.

After you create a host group, add hosts to the host group and then establish connections between the volumes and the host group.

When a volume is connected to a host group, it is assigned a logical unit number (LUN), which all hosts in the group use to communicate with the volume. If a LUN is not manually specified when the connection is first established, Purity//FA automatically assigns a LUN to the connection.

Once a connection has been established between a host group and a volume, all of the hosts within the host group are able to access the volume through the connection. These connections are called shared connections because the connection is shared between all of the hosts within the host group.

*Rename* a host group to change the name by which Purity//FA identifies the host group in administrative operations and displays. Renaming a host group does not change its relationship with hosts and volumes. The new host group name is effective immediately and the old name is no longer recognized in CLI, GUI, or REST interactions.

*Delete* a host group if it is no longer required. You cannot recover a host group after it has been deleted.

## Host Group Guidelines

Purity//FA will not create a host group if:

- A host group with the specified name already exists in the array.
- The creation of the host group would exceed the limit of concurrent host groups.

Purity//FA will not delete a host group if:

- Any hosts are associated with the host group or any volumes are connected to it.

A host cannot be added to a host group if:

- The host is associated with another host group. A host can only be associated with one host group at a time.
- The host has a private connection to a volume associated with the host group.

Host groups are configured through the GUI (**Storage > Hosts**) and CLI ([purehgroup](#) command).

## Host-Volume Connections

For a host to read and write data on a FlashArray volume, the two must be connected. Purity//FA only responds to I/O commands from hosts to which the volume addressed by the command is connected; it ignores commands from unconnected hosts.

Hosts are connected to volumes through private or shared connections. Private and shared connections are functionally identical: both make it possible for hosts to read and write data on volumes. They differ in how administrators create and delete them.

### Private Connections

Connecting a volume to a host establishes a private connection between the volume and the host. You can connect multiple volumes to a host. Likewise, a volume can be connected to multiple hosts.

Disconnecting a volume from a host, or vice versa, breaks the private connection between the volume and host. Other shared and private connections are unaffected.

## Shared Connections

Connecting a volume to a host group establishes a shared connection between the volumes and all of the hosts within that host group. You can connect multiple volumes to a host group. Likewise, a volume can be connected to multiple host groups.

Disconnecting a volume-host group connection breaks the shared connection between the volume and all of the hosts within the host group. Other shared and private connections are unaffected.

## Breaking Private and Shared Connections

Breaking a connection between a host and volume causes the host to lose access to the volume. There are three ways in which a host-volume connection can be broken:

- Break the private connection between a volume and a host, which causes the host to lose access to the volume. Volume-host connections are broken when you disconnect a volume from its host, or disconnect a host from the volume.
- Break the shared connection between a volume and a host group, which disconnects the volume and all of the host group's member hosts. Other shared and private connections to the volume are unaffected. Volume-host group connections are broken when you disconnect a volume from its host group, or disconnect a host group from the volume.
- Remove a host from a host group, which breaks the connections between the host and all volumes with shared connections to the host group. The removed host's private connections are unaffected.

## Logical Unit Number (LUN) Guidelines

Each host-volume connection has three components: a host, a volume, and a logical unit number (LUN) used by the host to address the volume. Purity//FA supports LUNs in the [1...4095] range.

Hosts establish connections to volumes either through private or shared (via host groups) connections. A host can have only one connection, private or shared, to a given volume at a time.

Purity//FA follows these guidelines to automatically assign a LUN to the connection:

- For private connections, Purity//FA starts at LUN 1 and counts up to the maximum LUN 4095, assigning the first available LUN to the connection.
- For shared connections, Purity//FA starts at LUN 254 and counts down to the minimum LUN 1, assigning the first available LUN to the connection. If all LUNs in the [1...254] range are taken, Purity//FA starts at LUN 255 and counts up to the maximum LUN 4095, assigning the first available LUN to the connection.

A host cannot be associated with a host group if it has a private connection to a volume associated with the same host group.

The LUN can be changed after the connection has been created. If you change a LUN, the volume may become temporarily disconnected from the host to which it is connected.

## Connection Guidelines

Purity//FA will not establish a (private) connection between a volume and a host if:

- An unavailable LUN was specified.
- The volume is already connected to the host, either through a private or shared connection.

Purity//FA will not establish a (shared) connection between a volume and host group if:

- An unavailable LUN was specified.
- The volume is already connected to the host group.
- The volume is already connected to a host associated with the host group.

Host-volume connections are performed through the GUI (**Storage > Hosts** and **Storage > Volumes**) and CLI ([purehgroup connect](#), [purehost connect](#) and [purevol connect](#) commands).

## Connections

The Connections page displays connectivity details between the Purity//FA hosts and the array ports.

The Host Connections pane displays a list of hosts, the connectivity status of each host, and the number of initiator ports associated with each host. Connectivity statuses range from "None", where the host does not have any paths to any target ports, to "Redundant", where the host has the same number of paths from every initiator to every target port on both controllers.

The Target Ports pane displays the connection mappings between each array port and initiator port. Each array port includes the following connectivity details: associated iSCSI Qualified Name (IQN), NVMe Qualified Name (NQN), or Fibre Channel World Wide Name (WWN) address, failover status, and communication speed. A check mark in the Failover column indicates that the port has failed over to the corresponding port pair on the primary controller.

Host connections and target ports are displayed through the GUI (select **Health > Connections**) and CLI (`pureport list`, `purehost list --all`, and `purevol list --all` commands).

## Protection Groups and Protection Group Snapshots

Protection groups support several Purity//FA features:

- Purity//FA FlashRecover is a policy-based data protection and disaster recovery solution. Through FlashRecover, generate protection group snapshots and retain them on the array and/or asynchronously replicate them to target arrays.
- Purity//FA Snap to NFS and Purity//FA Snap to Cloud are policy-based solutions that manage portable snapshots through the offload of volume snapshots to a target, such as an Azure Blob container, a NAS/NFS device, an NFS storage system, an S3 bucket, or a generic Linux server, for long-term retention.

Protection groups and protection group snapshots are configured and managed through the GUI (**Storage > Protection Groups** and **Storage > Volumes**) and CLI (`purepggroup` and `purevol` commands). Offload targets are further configured and managed through the GUI (**Storage > Array > Offload Targets**) and CLI (`pureoffload` command).

### Protection Groups

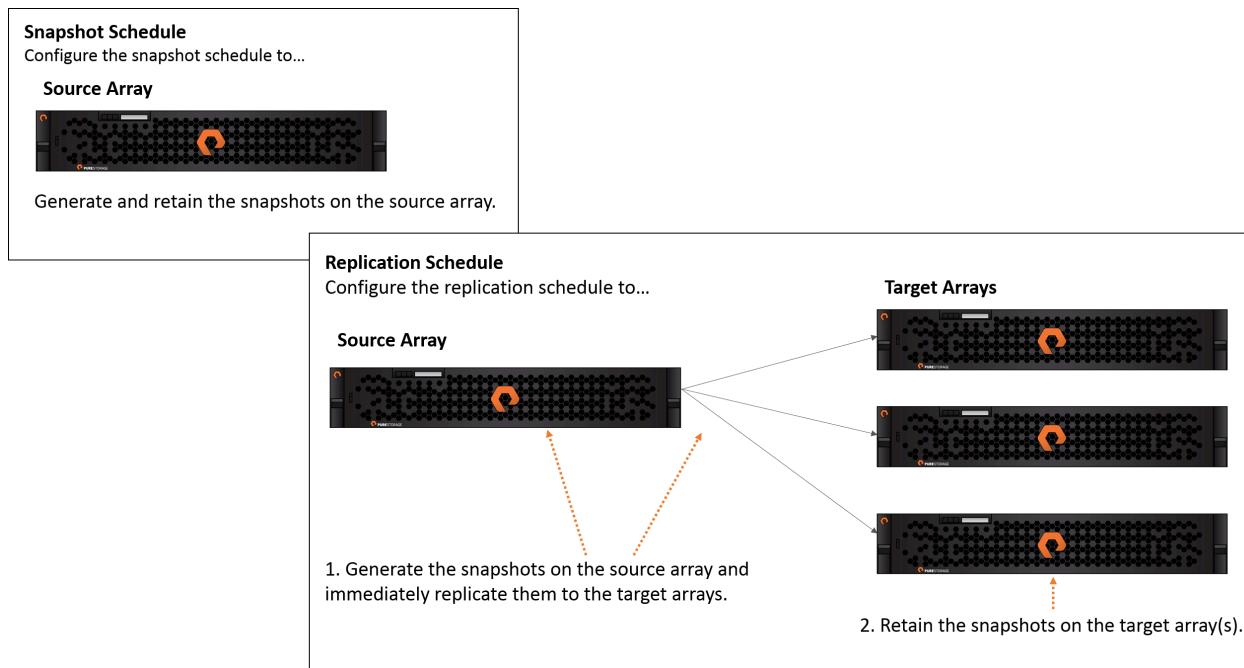
A protection group defines a set of volumes, hosts, or host groups (called members) that are protected together through snapshots with point-in-time consistency across the member volumes. The members within the protection group have common data protection requirements and the same snapshot, replication, and retention schedules.

Each protection group includes the following components:

- **Source array.** An array from which Purity//FA generates a point-in-time snapshot of its protection group volumes. Depending on the protection group schedule settings, the snapshot data is either retained on the source array or replicated over to and retained on one or more target arrays.
- **Targets.** One or more arrays or storage systems that receive snapshot data from the source array. Targets are only required if snapshot data needs to be replicated over to remote arrays or storage systems.

- Members.** Volumes, hosts, or host groups that have common data protection requirements and the same snapshot/replication frequency and retention policies. Only members of the same object type can belong to a protection group.  
 Replication to offload targets only supports volumes; hosts and host groups are not supported.  
 For asynchronous replication, a single protection group can consist of multiple hosts, host groups, and volumes. Likewise, hosts, host groups, and volumes can be associated with multiple protection groups. Protection groups can also contain overlapping volumes, hosts, and host groups. In such cases, Purity//FA counts the volume once and ignores all other occurrences of the same member.
- Schedules.** Each protection group includes a snapshot schedule and a replication schedule.  
 Configure and enable the snapshot schedule to generate snapshots and retain them on the source array.  
 Configure and enable the replication schedule to generate snapshots on the source array, immediately replicate the snapshots to the targets, and retain those snapshots on the targets. When replicating to an array or offload target, Purity//FA only transfers the incremental data between two snapshots. Furthermore, during the replication data transfer process, data deduplicated on the source array is not sent again if the same data was previously sent to the same target array.

**Figure 2.2: Protection Group Schedules**



Create a protection group to add members (volumes, hosts, or host groups) that have common data protection requirements. Pure Storage protection groups are virtual, so creating, renaming, and destroying a protection group has no meaning outside the array. Protection group names must be unique within an array.

*Copy* a protection group to restore the state of the volumes within a protection group to a previous protection group snapshot. The restored volumes are added as real volumes to a new or existing protection group. Note that restoring volumes from a protection group snapshot does not automatically expose the restored volumes to hosts and host groups.

*Rename* a protection group to change the name by which Purity//FA identifies the protection group in administrative operations and displays. When you rename a protection group, the name change is effective immediately and the old name is no longer recognized by Purity//FA.

*Destroy* a protection group if it is no longer needed.

Destroying a protection group implicitly destroys all of its snapshots. Once a protection group has been destroyed, all snapshot and replication processes for the protection group stop and the destroyed protection group begins its 24-hour eradication pending period.

When the 24-hour eradication pending period has lapsed, Purity//FA starts reclaiming the physical storage occupied by the protection group snapshots.

During the eradication pending period, you can recover the protection group to bring the group and its content back to its original state, or manually eradicate the destroyed protection group to reclaim physical storage space occupied by the destroyed protection group snapshots.

Once reclamation starts, either because you have manually eradicated the destroyed protection group, or because the eradication pending period has lapsed, the destroyed protection group and its snapshot data can no longer be recovered.

The Time Remaining column displays the 24-hour eradication pending period in `hh:mm` format, which begins at 24:00 and counts down to 00:00. When the eradication pending period reaches 00:00, Purity//FA starts the reclamation process. The Time Remaining number remains at 00:00 until the protection group or snapshot is completely eradicated.

## Space Consumption Considerations

Consider space consumption when you configure the snapshot, replication, and retention schedules.

The amount of space consumed on the source array depends on how many snapshots you want to generate, how frequently you want to generate the snapshots, how many snapshots you want to retain, and how long you want to retain the snapshots.

Likewise, the amount of space consumed on the target depends how many snapshots you want to replicate, how frequently you want to replicate the snapshots, how many replicated snapshots you want to retain, and how long you want to retain them.

## Protection Group Snapshots

Protection group snapshots capture the content of all volumes on the array for the specified protection group at a single point in time. The snapshot is an immutable image of the volume data at that instance in time. The volumes are either direct members of the protection group or connected to any of its hosts or host groups within a protection group.

Generate a protection group snapshot to create snapshots of the volumes within the protection group.

Protection group snapshots can be generated automatically (using schedules) or on-demand.

The volumes within a protection group snapshot can be copied as-needed to create live, host-accessible volumes.

The protection group snapshot naming convention is **PGROUP.NNN**, where:

- **PGROUP** is the name of the protection group.
- **NNN** is a unique monotonically increasing number or a manually-assigned protection group snapshot suffix name.

The protection group volume snapshot naming convention is **PGROUP.NNN.VOL**, where:

- **PGROUP** is the name of the protection group.
- **NNN** is a unique monotonically increasing number or a manually-assigned protection group snapshot suffix name.
- **VOL** is name of the volume member.

If you are viewing replicated snapshots on a target array, the snapshot name begins with the name of the source array from where the snapshot was taken.

Destroy a protection group snapshot if it is no longer required. Destroying a protection group snapshot destroys all of its protection group volume snapshots, thereby reclaiming the physical storage space occupied by its data.

Destroyed protection group snapshots follow the same eradication pending behavior as destroyed protection groups. If you destroy a protection group snapshot, Purity//FA automatically takes an undo snapshot. The undo snapshot enters a 24-hour eradication pending period, after which time the snapshot is eradicated. During the 24-hour pending period, the undo snapshot can be viewed, recovered, or permanently eradicated.

Protection group volume snapshots cannot be destroyed individually. A protection group volume snapshot can only be destroyed by destroying the protection group snapshot to which it belongs.

## Users and Security

Purity//FA release comes with a single local administrative account named **pureuser**. The account is password-protected, and may alternatively be accessed using a public-private key pair.

Users can be added to the array either locally by creating and configuring a local user directly on the array, or through Lightweight Directory Access Protocol (LDAP) by integrating the array with a directory service, such as Active Directory or OpenLDAP.

The Pure Storage REST API uses authentication tokens to create sessions. All Purity//FA users can generate their own API token and view only their own token.

Local user configuration and API token generation is performed through the GUI (select **Settings > Users**) and CLI (`pureadmin` command). Directory service configuration is performed through the GUI (**Settings > Users**) and CLI (`pureds` command).

## Directory Service

Additional Purity//FA accounts can be enabled by integrating the array with an existing directory service, such as Microsoft Active Directory or OpenLDAP, allowing multiple users to log in and use the array and providing role-based access control.

Configuring and enabling the Pure Storage directory service changes the array to use the directory when performing user account and permission level searches. If a user is not found locally, the directory servers are queried.

Directory service configuration is performed through the GUI (**Settings > Users**) and CLI (`pureds` command).

## SSL Certificate

Purity//FA creates a self-signed certificate and private key when the system is started for the first time.

SSL certificate configuration includes changing certificate attributes, creating new self-signed certificates to replace existing ones, constructing certificate signing requests, importing certificates and private keys, and exporting certificates.

SSL certificate configuration is performed through the GUI (**Settings > System**) and CLI (`purecert` command).

## Industry Standards

Purity//FA includes the Pure Storage Storage Management Initiative Specification (SMI-S) provider.

The SMI-S initiative was launched by the Storage Networking Industry Association (SNIA) to provide a unifying interface for storage management systems to administer multi-vendor resources in a storage area network. The SMI-S provider in Purity//FA allows FlashArray administrators to manage the array using an SMI-S client over HTTPS.

SMI-S client applications optionally use the Service Location Protocol (SLP) as a directory service to locate resources.

The SMI-S provider is optional and must be enabled before its first use. The SMI-S provider is enabled and disabled through the GUI (**Settings > System**) and CLI (`puresmis` command).

For detailed information on the Pure Storage SMI-S provider, refer to the *Pure Storage SMI-S Provider Guide* in the Pure Storage Knowledge Base at <https://support.purestorage.com>.

For general information on SMI-S, refer to the Storage Networking Industry Association (SNIA) website at <https://www.snia.org>.

## Troubleshooting and Logging

Purity//FA continuously logs a variety of array activities, including performance summaries, hardware and operating status reports, and administrative actions that modify the array. For certain array state changes and events that are potentially significant to array operation, Purity//FA immediately generates alert messages and transmits them to one or more user-specified destinations for immediate action.

The FlashArray troubleshooting mechanisms assume that Pure Storage Support can actively participate in helping organizations maintain “healthy” arrays. However, for organizations where operating procedures do not permit outside connections to equipment, troubleshooting reports can be directed to internal email addresses or displayed on a GUI or CLI console.

### Alerts

Alert, audit record, and user session messages are retrieved from a list of log entries that are stored on the array.

To conserve space, Purity//FA stores a reasonable number of log entries on the array. Older entries are deleted from the log as new entries are added. To access the complete list of messages, configure the Syslog Server feature to forward all messages to your remote server.

An alert is triggered when there is an unexpected change to the array or to one of the Purity//FA hardware or software components. Alerts are categorized by severity level as critical, warning, or informational.

Alerts are displayed in the GUI and CLI. Alerts are also logged and transmitted to Pure Storage Support via the phone home facility. Furthermore, alerts can be sent as messages to designated email addresses and as Simple Network Management Protocol-based (SNMP) traps and informs to SNMP managers.

#### Phone Home Facility

The phone home facility provides a secure direct link between the array and the Pure Storage Support team. The link is used to transmit log contents and alert messages to the Pure Storage Support team.

If the phone home facility is disabled, the log contents are delivered when the facility is next enabled or when the user manually sends the logs through the GUI or CLI.

Optionally configure the proxy host for HTTPS communication.

The phone home facility is managed through the GUI (**Settings > System**) and CLI ([purearray](#) command).

Proxies are configured through the GUI (**Settings > System**) and CLI ([purearray setattr --proxy](#) command).

#### Email

Alerts can be sent to designated email recipients. The list includes the built-in `flasharray-alerts@purestorage.com` address, which cannot be deleted. Individual email addresses can be added to and removed from the list, and transmission of alert messages to specific addresses can be temporarily enabled or disabled without removing them from the list.

The list of email alert recipients is managed through the GUI (**Settings > System** and CLI (`purealert` command)).

### SNMP Managers

If SNMP manager objects are configured on the array, each alert is transmitted to the SNMP managers.

The SNMP manager objects are configured through the GUI (**Settings > System** and CLI (`puresnmp` command)).

Alerts are displayed through the GUI (**Health > Alerts**) and the CLI (`puremessage` command).

### Audit Trail

The audit trail represents a chronological history of the Purity//FA GUI, Purity//FA CLI, or REST API operations that a user has performed to modify the configuration of the array. For example, changing the size of a volume, deleting a host, changing the replication frequency of a protection group, and associating a WWN to a host generates an audit record.

Audit trails are displayed through the GUI (**Settings > Users**) and the CLI (`pureaudit` command).

### User Session Logs

User session logs represent user login and authentication events performed in the Purity//FA GUI, Purity//FA CLI, and REST API. For example, logging in to and out of the Purity//FA GUI, attempting to log in to the Purity//FA CLI with an invalid password, or opening a Pure Storage REST API session generates a user session log entry.

User sessions are displayed through the GUI (**Settings > Users**) and the CLI (`puremessage` command).

### SNMP Agent and SNMP Managers

The Simple Network Management Protocol (SNMP) is used by SNMP agents and SNMP managers to send and retrieve information. FlashArray supports SNMP versions v2c and v3.

In the FlashArray, the built-in SNMP agent has local knowledge of the array. The agent collects and organizes this array information and translates it via SNMP to or from the SNMP managers. The agent, named `localhost`, cannot be deleted or renamed. The managers are defined by creating SNMP manager objects on the array. The managers communicate with the agent via the standard TCP port 161, and they receive notifications on port 162.

In the FlashArray, the `localhost` SNMP agent has two functions, namely, responding to GET-type SNMP requests and transmitting alert messages.

The SNMP agent generates and transmits messages to the SNMP manager as traps or inform requests (informs), depending on the notification type that is configured on the manager. An SNMP trap is an unacknowledged SNMP message, meaning the SNMP manager does not acknowledge receipt of the message. An SNMP inform is an acknowledged trap.

SNMPv2 uses a type of password called a community string to authenticate the messages that are passed between the agent and manager. The community string is sent in clear text, which is considered

an unsecured form of communication. SNMPv3, on the other hand, supports secure communication between the agent and manager through the use of authentication and privacy encryption methods.

The SNMP agent and list of SNMP managers are managed through the GUI (**Settings > System**) and CLI (`puresnmp` command).

Download the MIB through the GUI (**Settings > System**).

## Remote Assist Facility

In many cases, the most efficient way to service an array or diagnose problems is through direct intervention by a Pure Storage Support representative.

The Remote Assist facility enables Pure Storage Support to communicate with an array, effectively establishing an administrative session for service and diagnosis. Optionally configure the proxy host for HTTPS communication.

Remote assist sessions are controlled by the array administrator, who opens a secure channel between the array and Pure Storage Support, making it possible for a technician to log in to the array. The administrator can check session status and close the channel at any time.

Remote assist sessions are opened and closed through the GUI (**Settings > System**) and CLI (`purearray remoteassist` command).

Proxies are configured through the GUI (**Settings > System**) and CLI (`purearray setattr --proxy` command).

## Syslog Logging Facility

The Purity//FA syslog logging facility generates messages deemed major events within the FlashArray and forwards the messages to remote servers via TCP or UDP protocol. Purity//FA generates syslog messages for three types of events:

- Alerts (`purity.alert`)
- Audit Trails (`purity.audit`)
- Tests (`purity.test`)

The syslog server output location is configured through the GUI (**Settings > System**) and CLI (`purearray setattr` command).

## Chapter 3. Conventions

Purity//FA is the operating environment that queries and manages the FlashArray hardware, networking, and storage components. The Purity//FA software is distributed with the FlashArray.

Purity//FA provides two ways to administer the FlashArray: through the browser-based graphical user interface (Purity//FA GUI) and the command-driven interface (Purity//FA CLI).

Purity//FA follows certain naming and numbering conventions.

### Object Names

Valid characters are letters (A-Z and a-z), digits (0-9), and the hyphen (-) character. The first and last characters of the name must be alphanumeric, and the name must contain at least one letter or '-'.

Most objects in Purity//FA that can be named, including host groups, hosts, volumes, protection groups, volume and protection group suffixes, SNMP managers, and subnets, can be 1-63 characters in length.

Array names can be 1-56 characters in length. The array name length is limited to 56 characters so that the names of the individual controllers, which are assigned by Purity//FA based on the array name, do not exceed the maximum allowed by DNS.

Names are case-insensitive on input. For example, `vol1`, `Vol1`, and `VOL1` all represent the same volume. Purity//FA displays names in the case in which they were specified when created or renamed.

Pods and volume groups provide a namespace with unique naming conventions.

All objects in a pod have a fully qualified name that include the pod name and object name. The fully qualified name of a volume in a pod is `POD::VOLUME`, with double colons (::) separating the pod name and volume name. The fully qualified name of a protection group in a pod is `POD::PGROUP`, with double colons (::) separating the pod name and protection group name. For example, the fully qualified name of a volume named `vo101` in a pod named `pod01` is `pod01::vo101`, and the fully qualified name of a protection group named `pgroup01` in a pod named `pod01` is `pod01::pgroup01`.

If a protection group in a pod is configured to asynchronously replicate data to a target array, the fully qualified name of the protection group on the target array is `POD:PGROUP`, with single colons (:) separating the pod name and protection group name. For example, if protection group `pod01::pgroup01` on source array `array01` asynchronously replicates data to target array `array02`, the fully qualified name of the protection group on target array `array02` is `pod01:pgroup01`.

All objects in a volume group have a fully qualified name that includes the volume group name and the object name, separated by a forward slash (/). For example, the fully qualified name of a volume named `vo101` in a volume group named `vgroup01` is `vgroup01/vo101`.

### Volume Sizes

Volume sizes are specified as an integer, optionally followed by one of the suffix letters `K`, `M`, `G`, `T`, `P`, denoting 512-byte sectors, KiB, MiB, GiB, TiB, and PiB, respectively, where "Ki" denotes  $2^{10}$ , "Mi" denotes  $2^{20}$ , and so on. If a suffix letter is not specified, the size is expressed in sectors.

Volumes must be between one megabyte and four petabytes in size. If a volume size of less than one megabyte is specified, Purity//FA adjusts the volume size to one megabyte. If a volume size of more than four petabytes is specified, the Purity//FA command fails.

Volume sizes cannot contain digit separators. For example, **1000g** is valid, but **1,000g** is not.

## IP Addresses

FlashArray supports two versions of the Internet Protocol: IP Version 4 (IPv4) and IP Version 6 (IPv6). IPv4 and IPv6 addresses follow the addressing architecture set by the Internet Engineering Task Force.

An IPv4 address consists of 32 bits and is entered in the form `ddd.ddd.ddd.ddd`, where `ddd` is a number ranging from 0 to 255 representing a group of 8 bits. Here are some examples:

```
puredns setattr --domain mydomain.com --nameservers 192.0.2.10
purelog create --uri tcp://192.0.2.100:614 LOGSERVER2
purenetwork setattr --address 192.0.2.0/24 ct0.eth1
purenetwork setattr --address 192.0.2.0 --netmask 255.255.255.0 ct0.eth1
puresnmp create --host 192.0.2.255 --community SNMPMANAGER1
puresubnet create --prefix 192.0.2.0/24 --vlan 100 ESXHost001
```

An IPv6 address consists of 128 bits and is written in the form

`xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx`, where `xxxx` is a hexadecimal number representing a group of 16 bits. Colons separate each 16-bit field. Leading zeros can be omitted in each field. Furthermore, consecutive fields of zeros can be shortened by replacing the zeros with a double colon (`::`). For example, IPv6 address `2001:0db8:85a3:0000:0000:8a2e:0370:7334` becomes `2001:db8:85a3::8a2e:370:7334`.

To use an IPv6 address in a URL or URI, enclose the entire address in square brackets ([ ]). When specifying a URL or URI with a port number, append the port number after the end of the entire address. Here are some examples:

```
puredns setattr --domain mydomain.com --nameservers 2001:db8:85a3::8a2e:370:7334
purelog create --uri tcp://[2001:db8:85a3::8a2e:370:7334]:614 LOGSERVER2
purenetwork setattr --address 2001:db8:85a3::8a2e:370:7334/64 ct0.eth1
purenetwork setattr --address 2001:db8:85a3::8a2e:370:7334 --netmask 64 ct0.eth1
puresnmp create --host [2001:db8:85a3::8a2e:370:7334] --community SNMPMANAGER1
puresubnet create --prefix 2001:db8:85a3::/64 --vlan 100 ESXHost001
```

## Storage Network Addresses

A Purity//FA host is comprised of a host name and one or more IQNs, NQNs, or WWNs. The host cannot communicate with the array until at least one IQN, NQN, or WWN has been associated with it.

iSCSI Qualified Names (IQNs) follow the naming standards set by the Internet Engineering Task Force (see RFC 3720). For example, `iqn.2016-01.com.example:flasharray.491b30d0efd97f25`.

NVMe Qualified Names (NQNs) follow the naming standards set by NVMe Express. For example, `nqn.2016-01.com.example:flasharray.491b30d0efd97f25`.

Fibre Channel World Wide Names (WWNs) follow the naming standards set by the IEEE Standards Association. WWNs are comprised of eight pairs of case-insensitive hexadecimal numbers, optionally separated by colons. For example, `21:00:00:24:FF:4C:C5:49`.

Like hosts, IQNs, NQNs, and WWNs must be unique in an array. A host can be associated with multiple storage network addresses, but a storage network address can only be associated with one host.



## Part 2:

# Using the GUI to Administer a FlashArray



## Chapter 4. GUI Overview

The Purity//FA graphical user interface (GUI) is a browser-based system used to view and administer the FlashArray.

**Figure 4.1: Purity//FA GUI**



The Purity//FA GUI contains the following pages:

### Dashboard

Represents a graphical overview of the array, including storage capacity, recent alerts, hardware status, and performance metrics.

### Storage

Displays storage objects on the array, including hosts, host groups, volumes, protection groups, volume groups, and pods. View and manage the storage objects and the connections between them.

### Analysis

Displays historical array information, including storage capacity and I/O performance metrics, from various viewpoints.

### Health

Displays array health, including hardware status, parity, alerts, and connections.

### Settings

Displays array-wide system and network settings. Manage array-wide components, including network interfaces, system time, connectivity and connection configurations, and alert settings.

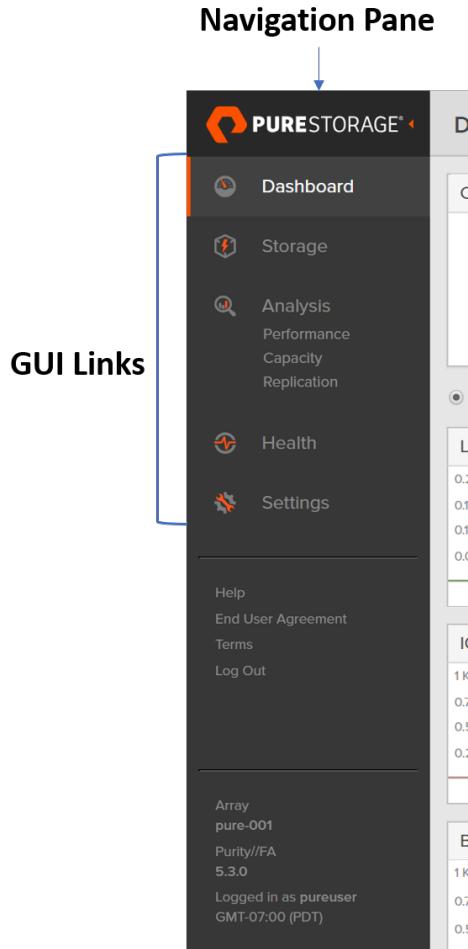
Also display user accounts, audit trails, user session logs, and software details.

The Purity//FA GUI pages are described in more detail in Part 2, “Using the GUI to Administer a FlashArray” [43].

## GUI Navigation

The dark gray navigation pane that appears along the left side of the Purity//FA GUI contains links to the GUI pages.

**Figure 4.2: Purity//FA GUI - Navigation Pane**



Click the Pure Storage® logo at the top of the navigation pane to toggle between the expanded and collapsed views of the pane.

Just below the Pure Storage logo are links to the GUI pages. Click a link to analyze or configure the information that appears in the page to its right. For example, click the Storage link to view information about the FlashArray storage objects, such as hosts, host group, volumes, protection group, volume groups, and pods.

The navigation pane includes links to the following external sites:

**Help**

Accesses the FlashArray user guides and launches the Pure1® community and support portals.

**End User Agreement**

Displays the terms of the Pure Storage End User Agreement (EULA).

For more information about the Pure Storage End User Agreement, refer to the section called “End User Agreement (EULA)” [49].

**Terms**

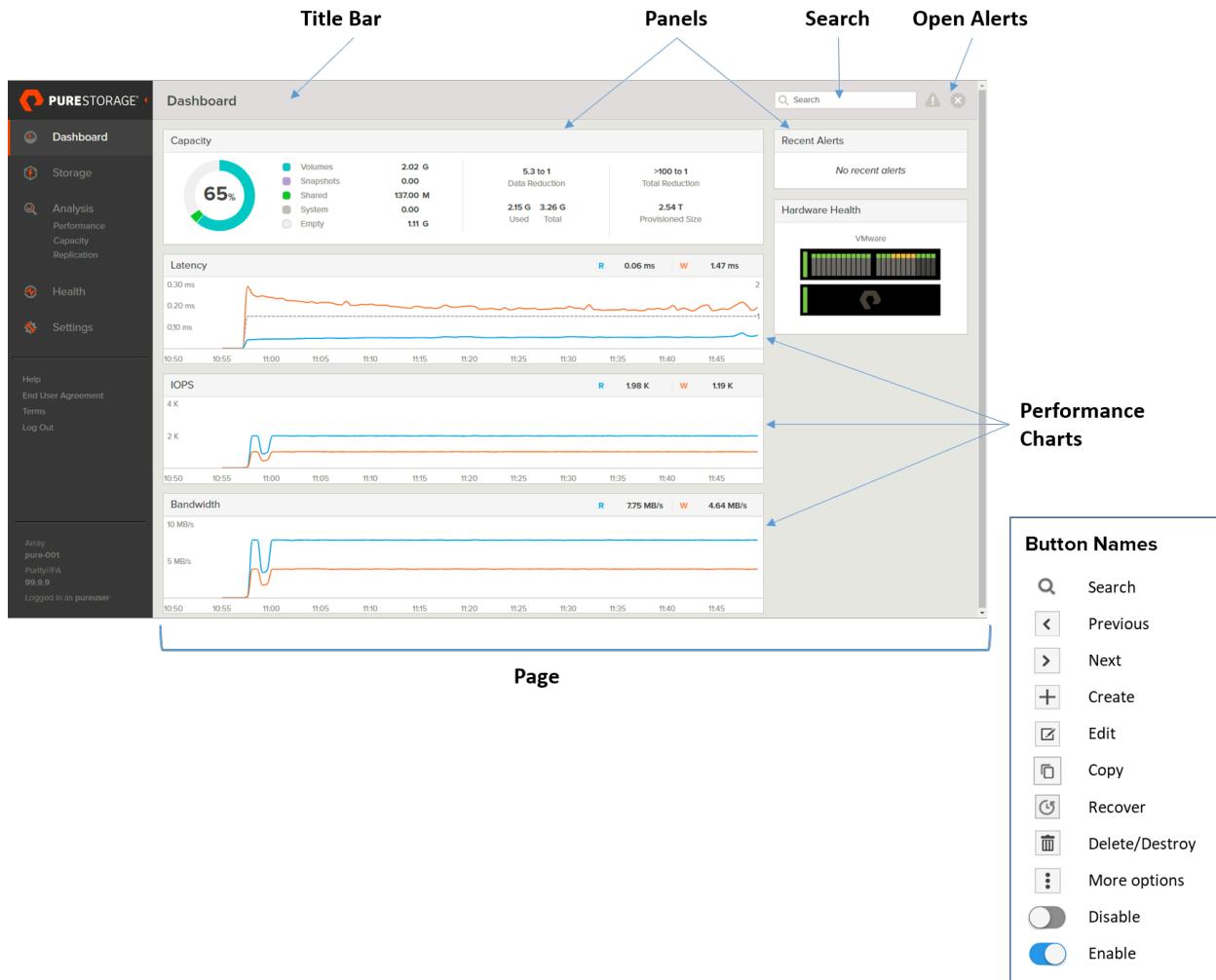
Launches the Pure Storage Product End User Information page, which includes a link to the Pure Storage end user agreement.

**Log Out**

Logs the current user out of the Purity//FA GUI.

The bottom of the navigation pane displays FlashArray and Purity//FA version information, and the name of the user who is currently logged into the Purity//FA GUI.

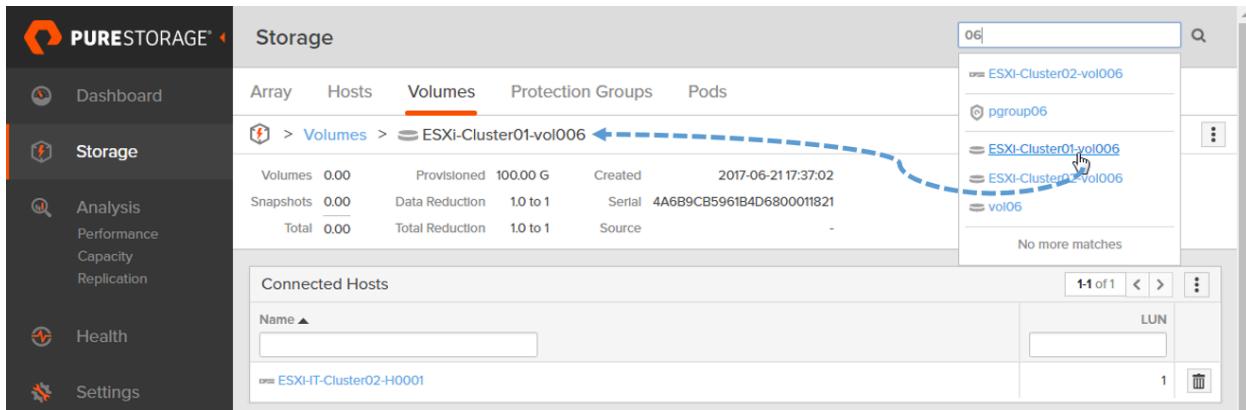
The page to the right of the navigation pane displays the information and configuration options for the selected GUI link. The information on each page is organized into panels, charts, and lists.

**Figure 4.3: Purity//FA GUI - Page and Buttons**


The alert icons that appear to the far right of the title bar indicate the number of recent **Warning** and **Critical** alerts, respectively. A recent alert represents one that Purity//FA saw within the past 24 hours and still considers an open issue that requires attention. Click anywhere in an alert row to display additional alert details. To analyze the alerts in more detail, select **Health > Alerts**.

The Search field (magnifying glass) in the upper-right corner of the screen allows you to quickly search for existing hosts, host groups, volumes, protection groups, volume groups, and pods on the array. Type any part of the name (case-insensitive) in the field to display all matches, and then click the name in the list of results to view its details in the Storage page.

**Figure 4.4: Purity//FA GUI - Navigation - Quick Search**



Various panels, such as **Storage > Volumes** and **Health > Alerts**, contain lists of information. The total number of rows in a list output is displayed in the upper-right corner of the list. Some lists can be very large, extending beyond hundreds of rows.

Volumes								1-10 of 51
Name	Source	#Hosts	Provisioned	Volumes	Snapshots	Reduction	Serial	Actions
ESXi-Cluster01-vol001		1	100.00 G	0.00	0.00	1.0 to 1	4A6B9CB5961B4D680001181A	<input type="checkbox"/> <input type="checkbox"/>
ESXi-Cluster01-vol002		1	100.00 G	0.00	0.00	1.0 to 1	4A6B9CB5961B4D6800011820	<input type="checkbox"/> <input type="checkbox"/>
ESXi-Cluster01-vol003		1	100.00 G	0.00	0.00	1.0 to 1	4A6B9CB5961B4D6800011823	<input type="checkbox"/> <input type="checkbox"/>

Pagination divides a large list output into discrete pages. Pagination is enabled by default and is only in effect if the number of lines in the list output exceeds 10 rows. To move through a paginated list, click **<** to go to the previous page, or click **>** to go to the next page.

## End User Agreement (EULA)

The Pure Storage End User Agreement (EULA) represents a contract between Pure Storage and users of Pure Storage software. The most recent version of the agreement governs use of the Purity//FA software and can be found at <http://www.purestorage.com/legal/productenduserinfo.html>.

To view the terms of the Pure Storage End User Agreement through the Purity//FA GUI, click **End User Agreement**. The name and title of the individual who accepted the terms of the agreement appear at the bottom of the End User Agreement pop-up window. Click **Download Agreement** to download a copy of the End User Agreement from the array to your local machine.

Accept the terms of the agreement by completing the fields at the bottom of the agreement and clicking **Accept**. Only array administrators (i.e., users with the Array Admin role) have the necessary permissions to complete the fields at the bottom of the agreement and click **Accept**.

Accepting the agreement requires the following information:

- **Name** - Full legal name of the individual at the company who has the authority to accept the terms of the agreement.
- **Title** - Individual's job title at the company.

- **Company** - Full legal name of the entity.

The name, title, and company name must each be between 1 and 64 characters in length.

If the agreement is not accepted, Purity//FA generates an alert notifying all Purity//FA alert watchers that the agreement is pending acceptance. A warning alert also appears in the Purity//FA GUI. Pure Storage is not notified of the alert. The alert remains open until the agreement is accepted. Furthermore, whenever a user logs in to Purity//FA GUI, the End User Agreement window pops up as a reminder that the agreement is pending acceptance.

Once the terms of the agreement have been accepted, Purity//FA closes the alert and stops generating the End User Agreement pop-up window.

## GUI Login

Logging in to the Purity//FA GUI requires a virtual IP address or fully-qualified domain name (FQDN) and an Purity//FA login username and password; this information is provided during the FlashArray installation.

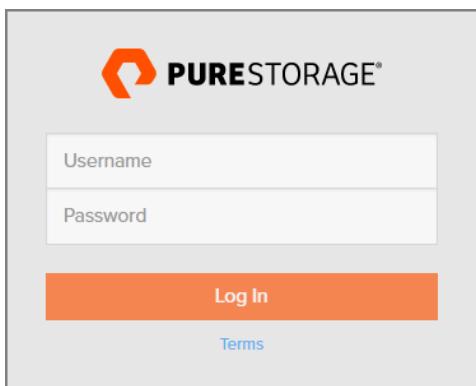
FlashArray is installed with one administrative account with the username `pureuser`. The initial password for the account is `pureuser`. The password can be changed any time through the `pureadmin` CLI command.

Pure Storage tests the Purity//FA GUI with the two most recent versions of the following web browsers:

- Apple Safari
- Google Chrome
- Microsoft Edge
- Microsoft Internet Explorer (IE)
- Mozilla Firefox

To launch the Purity//FA GUI login screen, open a browser and type the virtual IP address or FQDN into the address bar, and press **Enter**.

**Figure 4.5: Purity//FA GUI - Login Screen**



## Logging in to the Purity//FA GUI

Log in to the Purity//FA GUI to view and administer the FlashArray.

1. Open a web browser.
2. Type the virtual IP address or fully-qualified domain name of the FlashArray in the address bar and press **Enter**. The Purity//FA GUI login screen appears.
3. In the Username field, type the FlashArray user name. For example, **pureuser**.
4. In the Password field, type the password for the FlashArray user. For example, **pureuser**.
5. Click **Log In** to log in to the Purity//FA GUI.
6. If the Pure Storage End User Agreement (EULA) has not been accepted, the End User Agreement pop-up window appears. Accept the terms of the agreement by completing the fields at the bottom of the agreement and clicking **Accept**.

Note that only array administrators have the necessary permissions to complete the fields at the bottom of the agreement and click **Accept**. The agreement should be signed by individuals at the company who have the authority to accept the terms of the agreement.

For more information about the Pure Storage End User Agreement, refer to the section called “End User Agreement (EULA)” [49].

## Accepting the Terms of the End User Agreement (EULA)

Only array administrators have the necessary permissions to complete the fields at the bottom of the agreement and click **Accept**. The agreement should be signed by individuals at the company who have the authority to accept the terms of the agreement.

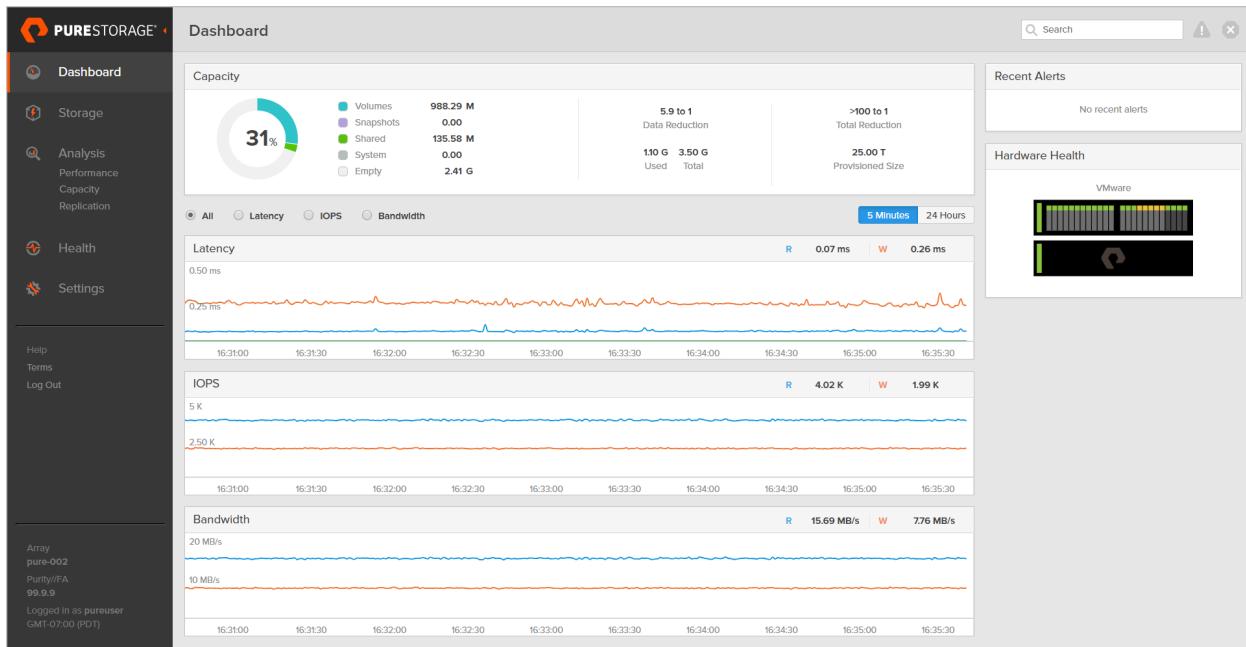
1. From the Purity//FA GUI, click **End User Agreement**.
2. Read the terms of the agreement.
3. At the bottom of the agreement, complete the following fields:
  - **Name** - Full legal name of the individual at the company who has the authority to accept the terms of the agreement.
  - **Title** - Individual's job title at the company.
  - **Company** - Full legal name of the entity.  
The name, title, and company name must each be between 1 and 64 characters in length.
4. Click **Accept** to accept the terms of the agreement.



## Chapter 5. Dashboard

The Dashboard page displays a running graphical overview of the array's storage capacity, performance, and hardware status.

**Figure 5.1:** Dashboard



The Dashboard page contains the following panels and charts:

- Capacity
- Recent Alerts
- Hardware Health
- Performance Charts

## Capacity

The Capacity panel displays array size and storage consumption details. All capacity values are rounded to two decimal places.

The capacity wheel displays the percentage of array space occupied by data and metadata. The percentage value in the center of the wheel is calculated as **Used/Total**.

The capacity wheel is broken down into the following components:

### Volumes

Physical space occupied by volume data that is not shared between volumes, excluding array metadata and snapshots.

### Snapshots

Physical space occupied by data unique to one or more snapshots.

### Shared

Physical space occupied by deduplicated data, meaning that the space is shared with other volumes and snapshots as a result of data deduplication.

### System

Physical space occupied by internal array metadata.

### Empty

Unused space available for allocation.

The capacity panel also displays the following information:

### Data Reduction

Ratio of mapped sectors within a volume versus the amount of physical space the data occupies after data compression and deduplication. The data reduction ratio does not include thin provisioning savings.

For example, a data reduction ratio of 5:1 means that for every 5 MB the host writes to the array, 1 MB is stored on the array's flash modules.

### Total Reduction

Ratio of provisioned sectors within a volume versus the amount of physical space the data occupies after reduction via data compression and deduplication *and* with thin provisioning savings. Total reduction is data reduction with thin provisioning savings.

For example, a total reduction ratio of 10:1 means that for every 10 MB of provisioned space, 1 MB is stored on the array's flash modules.

### Used

Physical storage space occupied by volume, snapshot, shared space, and system data.

### Total

Total physical usable space on the array.

Replacing a drive may result in a dip in usable space. This is intended behavior. RAID striping splits data across an array for redundancy purposes, spreading a write across multiple drives. A newly added drive cannot use its full capacity immediately but must stay in line with the available

space on the other drives as writes are spread across them. As a result, usable capacity on the new drive may initially be reported as less than the amount expected because the array will not be able to write to the unallocatable space. Over time, usable capacity fluctuations will occur, but as data is written to the drive and spreads across the array, usable capacity will eventually return to expected levels.

#### **Provisioned Size**

Total provisioned size of all volumes. Represents storage capacity reported to hosts.

## **Recent Alerts**

The Recent Alerts panel displays a list of alerts that Purity//FA saw within the past 24 hours and considers open issues that require attention. The list contains recent alerts of all severity levels.

To view the details of an alert, click the alert message.

To view a list of all alerts including ones that are in no longer open, go to the **Health > Alerts** page.

## **Hardware Health**

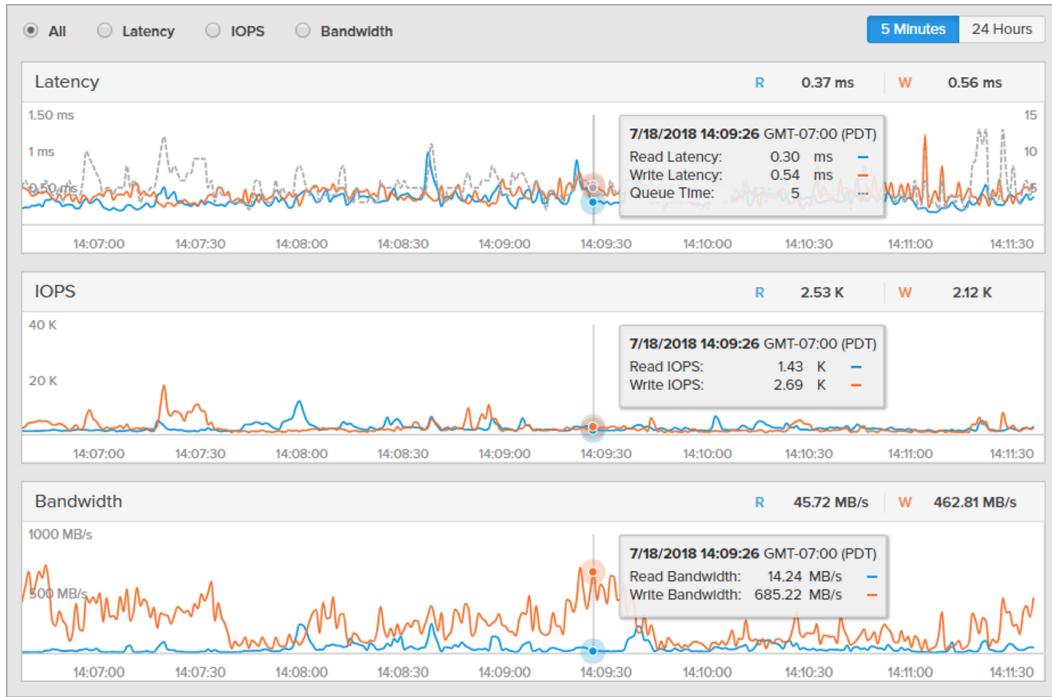
The Hardware Health panel displays the operational state of the array controllers, flash modules, and NVRAM modules.

To analyze the hardware components in more detail, click the image or go to the **Health > Hardware** page.

## **Performance Charts**

The performance charts display I/O performance metrics in real time.

**Figure 5.2: Dashboard - Performance Graphs**



The performance metrics are displayed along a scrolling graph; incoming data appear along the right side of each graph every few minutes as older numbers drop off the left side. Each performance chart includes Read (R), Write (W), and Mirrored Write (MW) values, representing the most recent data samples.

Hover over any of the charts to display metrics for a specific point in time. The values that appear in the point-in-time tooltips are rounded to two decimal places.

The performance panel includes Latency, IOPS, and Bandwidth charts.

### Latency

The Latency chart displays the average latency times for various operations.

- **Read Latency (R)** - Average arrival-to-completion time, measured in milliseconds, for a read operation.
- **Write Latency (W)** - Average arrival-to-completion time, measured in milliseconds, for a write operation.
- **Mirrored Write Latency (MW)** - Average arrival-to-completion time, measured in milliseconds, for a write operation. Represents the sum of writes from hosts into the volume's pod and from remote arrays that synchronously replicate into the volume's pod. The MW value only appears if there are writes through synchronous replication being processed.
- **Queue Time** - Average time, measured in microseconds, that an I/O request spends in the array waiting to be served. The time is averaged across all I/Os of the selected types.

## IOPS

The IOPS (Input/output Operations Per Second) chart displays I/O requests processed per second by the array. This metric counts requests per second, regardless of how much or how little data is transferred in each.

- **Read IOPS (R)** - Number of read requests processed per second.
- **Write IOPS (W)** - Number of write requests processed per second.
- **Mirrored Write IOPS (MW)** - Number of write requests processed per second. Represents the sum of writes from hosts into the volume's pod and from remote arrays that synchronously replicate into the volume's pod. The MW value only appears if there are writes through synchronous replication being processed.

## Bandwidth

The Bandwidth chart displays the number of bytes transferred per second to and from all file systems. The data is counted in its expanded form rather than the reduced form stored in the array to truly reflect what is transferred over the storage network. Metadata bandwidth is not included in these numbers.

- **Read Bandwidth (R)** - Number of bytes read per second.
- **Write Bandwidth (W)** - Number of bytes written per second.
- **Mirrored Write Bandwidth (MW)** - Number of bytes written into the volume's pod per second. Represents the sum of writes from hosts into the volume's pod and from remote arrays that synchronously replicate into the volume's pod.

By default, the performance charts display performance metrics for the past 1 minute. To display more than 1 minute of historical data, select **Analysis > Performance**.

## Note about the Performance Charts

The Dashboard and Analysis pages display the same latency, IOPS, and bandwidth performance charts, but the information is presented differently between the two pages.

In the Dashboard page:

- The performance charts are updated once every 30 seconds.
- The performance charts display up to 30 day's worth of historical data.
- The Latency charts displays only internal latency times. SAN times are not included.

In the Analysis page:

- The performance charts are updated once every minute.
- The performance charts display up to one year's worth of historical data.
- The performance charts can be further dissected by I/O type.
- The Latency charts displays both internal latency times and SAN times.

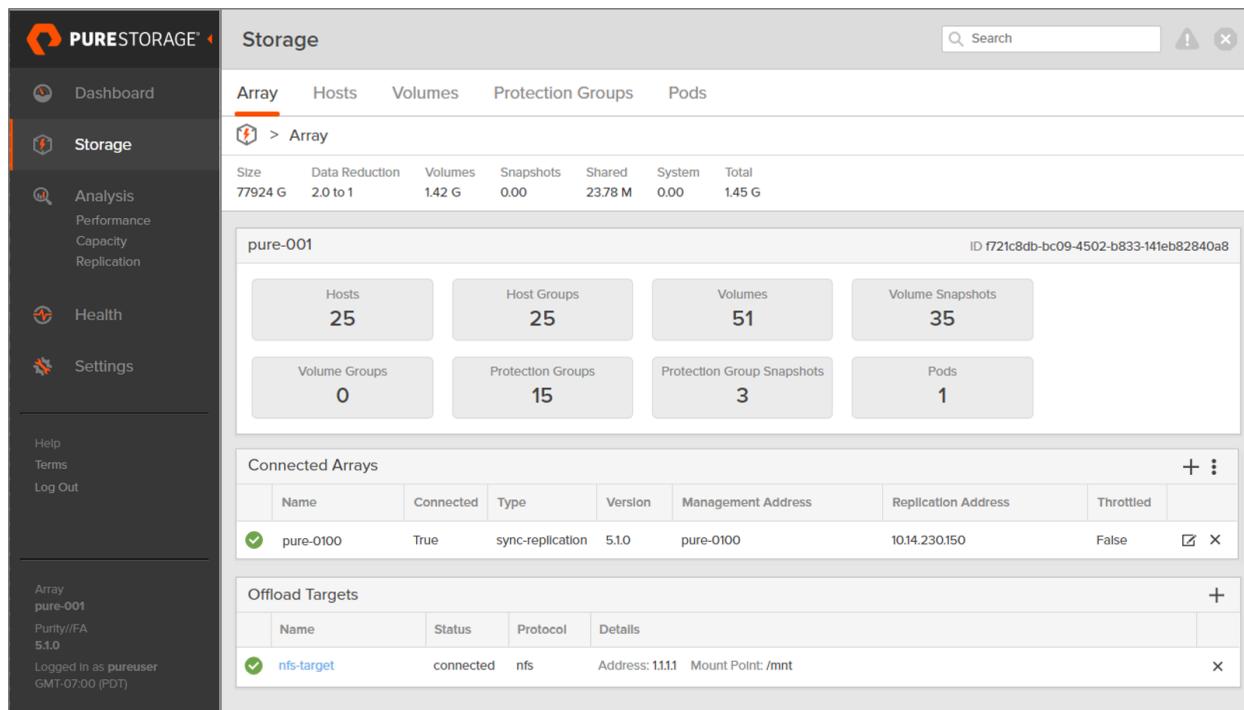


## Chapter 6. Storage

The Storage page displays configuration, space, and snapshot details for all types of FlashArray storage objects.

The metrics that appear near the top of each page represent the capacity and consumption details for the selected storage object. For example, the Storage > Array page displays array-wide capacity usage. Likewise, the Storage > Pods page displays the capacity and consumption details for all volumes within all pods in the array. Drill down to a specific storage object to view additional details. For example, drill down to a specific volume to see its creation date and unique serial number.

**Figure 6.1: Storage**



Name	Connected	Type	Version	Management Address	Replication Address	Throttled
pure-0100	True	sync-replication	5.1.0	pure-0100	10.14.230.150	False

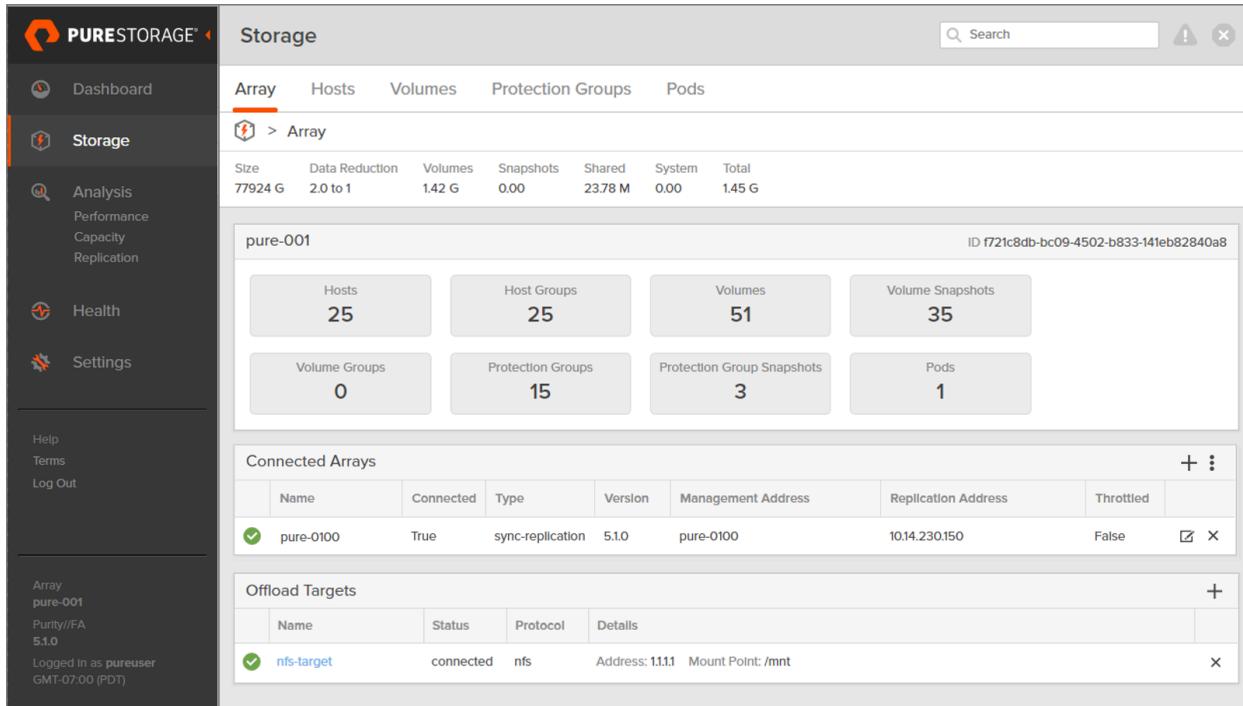
Name	Status	Protocol	Details
nfs-target	connected	nfs	Address: 1.1.1.1 Mount Point: /mnt

FlashArray storage objects include:

- Array
- Hosts
- Volumes
- Protection Groups
- Pods

## Array

The Storage > Array page displays a summary of all storage components on the array, a list of other arrays that are connected to this array, and a list of offload targets, such as Azure Blob containers, NFS devices, and S3 buckets, that are connected to this array.



The screenshot shows the PureStorage Storage > Array interface. The left sidebar includes links for Dashboard, Storage (which is selected), Analysis, Health, and Settings, along with Help, Terms, and Log Out options. The main content area has a header bar with tabs for Storage, Array, Hosts, Volumes, Protection Groups, and Pods. Below the tabs, it says "pure-001" and "ID f721c8db-bc09-4502-b833-141eb82840a8". It displays various storage metrics and counts for hosts, host groups, volumes, volume snapshots, volume groups, protection groups, and protection group snapshots. The "Connected Arrays" section lists one array: "pure-0100" (Connected: True, Type: sync-replication, Version: 5.1.0, Management Address: pure-0100, Replication Address: 10.14.230.150, Throttled: False). The "Offload Targets" section lists one target: "nfs-target" (Status: connected, Protocol: nfs, Address: 1.1.1, Mount Point: /mnt).

The array summary panel (with the array name in the header bar) contains a series of rectangles (technically known as hero images) representing the storage components of the array. The numbers inside each hero image represent the number of objects created for each of the respective components. Click a rectangle to jump to the page containing the details for that particular storage component.

Array attributes, such as array name and array time, are configured through the **Settings > System** page.

The Connected Arrays panel displays a list of arrays that are connected to the current array.

A connection must be established between two arrays in order for array-based data replication to occur. Purity//FA offers two types of replication: asynchronous replication and synchronous replication.

Asynchronous replication allows data to be replicated from one array to another. When two arrays are connected for asynchronous replication, the array where data is being transferred *from* is called the local (source) array, and the array where data is being transferred *to* is called the remote (target) array. Asynchronous replication is configured through protection groups. For more information about protection groups, refer to the Storage > Protection Groups section.

Synchronous replication allows I/O to be sent into either of two connected arrays and have it synced up on the other array. Synchronous replication is configured through pods. For more information about pods, refer to the Storage > Pods section.

Arrays are connected using a connection key, which is supplied from one array and entered into the other array.

The Connected Arrays panel displays a list of FlashArray arrays that are connected to the current array, and the attributes associated with each connection.

**Figure 6.2: Connected Arrays**

Connected Arrays							
	Name	Connected	Type	Version	Management Address	Replication Address	Throttled
<input checked="" type="checkbox"/>	pure-002	True	sync-replication	5.0.0	10.22.41.123	10.14.230.150	<input type="checkbox"/> <input checked="" type="checkbox"/>
<input checked="" type="checkbox"/>	pure-001	True	async-replication	5.0.0			<input type="checkbox"/> <input checked="" type="checkbox"/>

The Connected column displays the connectivity status between the current array and each remote array. A connected status of True means the current array is connected to the remote array. A connected status of False means the current array cannot establish a connection to the remote array due to network connection or firewall issues.

The Type column displays the type of connection that has been established between the two arrays for asynchronous replication (**async-replication**) and synchronous replication (**sync-replication**) purposes. Array connections set to **async-replication** support asynchronous replications only, while array connections set to **sync-replication** support both synchronous and asynchronous replications.

The Management Address column displays the virtual IP address or FQDN of the other array. The Replication Address column displays the IP address or FQDN of the interface(s) on the other array that have been configured with the replication service. The management and replication addresses only appear for the arrays from where an array connection was made. If the array connection was made from its peer array, the Management Address and Replication Address columns are empty.

The Connected Arrays panel also allows you to create new connections to other FlashArray arrays, view and copy the array connection key, and configure network bandwidth throttling limits for asynchronous replications.

The Network bandwidth throttling feature regulates when and how much data should be transferred between the arrays. Once two arrays are connected, optionally configure network bandwidth throttling to set maximum threshold values for outbound traffic.

In the Connected Arrays panel, the Throttled column indicates whether network bandwidth throttling has been enabled (**True**) or disabled (**False**).

Two different network bandwidth limits can be set:

- Set a *default* maximum network bandwidth threshold for outbound traffic.

and/or

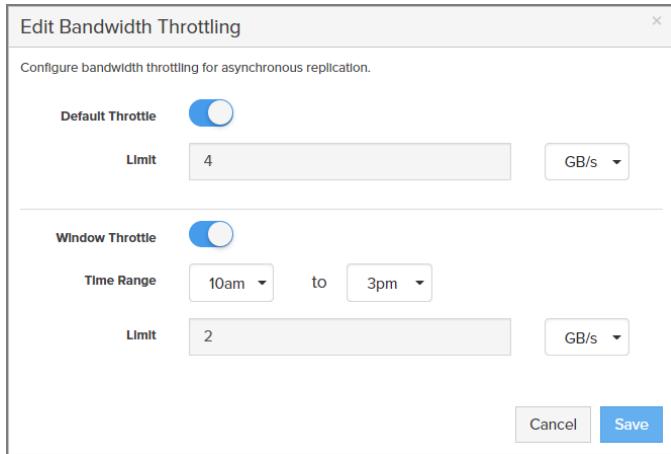
- Set a range (*window*) of time in which the maximum network bandwidth threshold is in effect.

If both thresholds are set, the “window” limit overrides the “default” limit.

The limit represents an average data rate, so actual data transfer rates can fluctuate slightly above the configured limit.

To completely stop the data transfer process, either by default or during a window of time, set the threshold to "0". During this time, all in-progress and scheduled data transfer processes are aborted.

In the following example, the current array has been configured to throttle whenever the rate of data being transferred to array vm-rep exceeds 4 GB/s, except between 10:00am and 3:00pm, when throttling will occur whenever the data transfer rate exceeds 2 GB/s.



## Offload Targets

The offload target feature enables system administrators to replicate point-in-time volume snapshots from the array to an external storage system. Each snapshot is an immutable image of the volume data at that instance in time. The data is transmitted securely and stored unencrypted on the storage system.

Before you can connect to, manage, and replicate to an offload target, the respective Purity//FA app must be installed. For example, to connect to an NFS offload target, the Snap to NFS app must be installed. To connect to an Azure Blob container or S3 bucket, the Snap to Cloud app must be installed. To determine if apps are installed on your array, run the `pureapp list` command. To install the Snap to NFS or Snap to Cloud app, contact Pure Storage Support.

The Offload Targets panel displays a list of all offload targets that are connected to the array.

Offload Targets					<a href="#">+</a>
Name	Status	Protocol	Details		<a href="#">X</a>
 nfs-target	connected	nfs	Address: 10.14.228.6	Mount Point: /mnt	<a href="#">X</a>

Each offload target represents an external storage system such as an Azure Blob container, NFS device, or S3 bucket to where Purity//FA volume snapshots (generated via protection group snapshots) can be replicated.

An array can be connected to one offload target at a time, while multiple arrays can be connected to the same offload target.

An offload target can have one of the following statuses:

- **Connected:** Array is connected to the offload target and is functioning properly.
- **Connecting:** Connection between the array and offload target is unhealthy, possibly due to network issues. Check the network connectivity between the interfaces, disconnect the array from the offload target, and then reconnect. If the issue persists, contact Pure Storage Support.
- **Not Connected:** Offload app is not running. Data cannot be replicated to offload targets. Contact Pure Storage Support.
- **Scanning:** A connection has been established between the array and offload target, and the system is determining the state of the offload target. Once the scan successfully completes, the status will change to Connected.

Offload targets that are disconnected from the array do not appear in the list of offload targets.

Whenever an array is disconnected from an offload target, any data transfer processes that are in progress are suspended. The processes resumes when the connection is re-established.

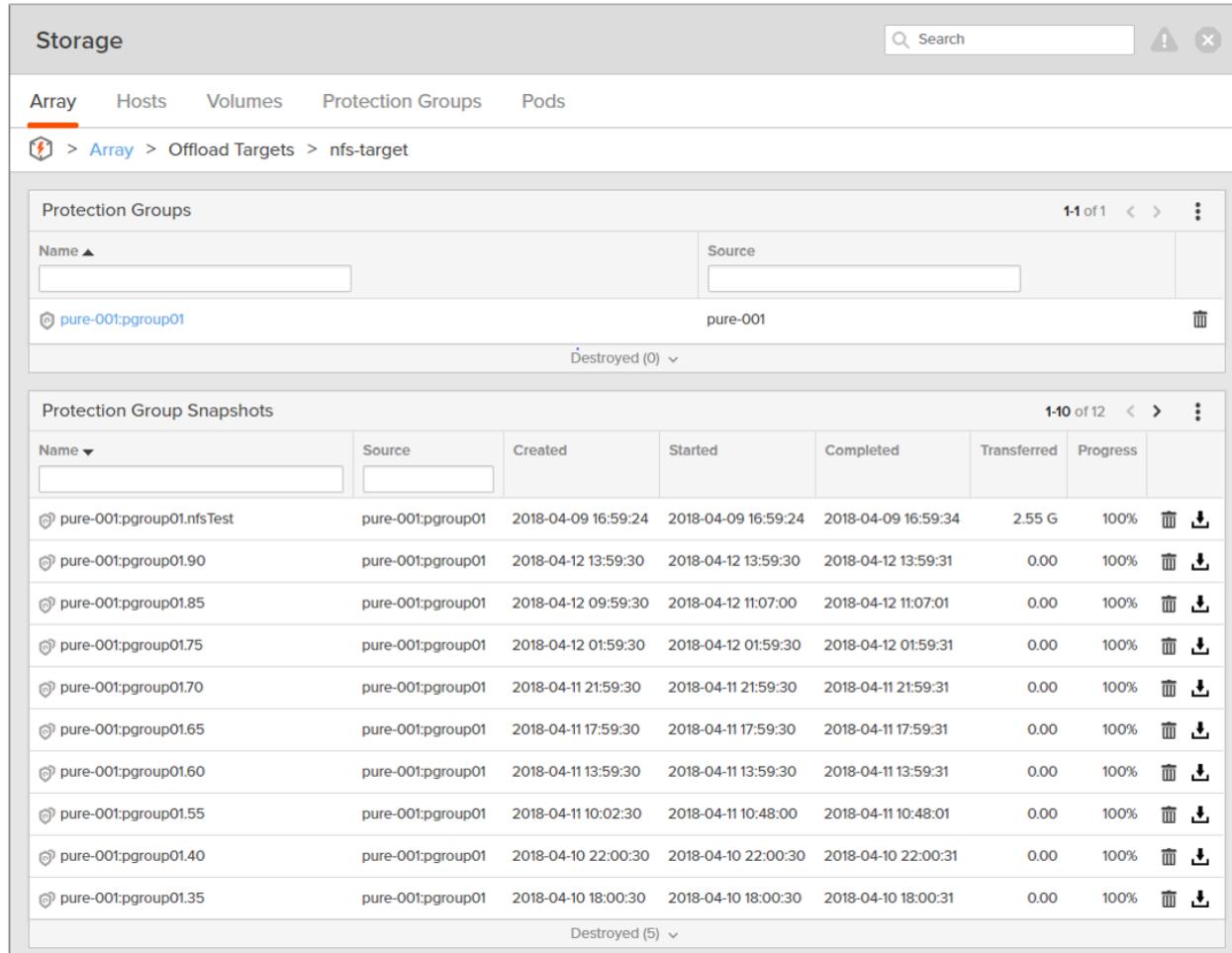
In the Offload Targets panel, click the name of the offload target to view its details.

The Offload Targets detailed view, which is accessed by clicking the name of the offload target from the Storage > Array > Offload Targets panel, displays a list of protection groups that are connected to the offload target and the protection group snapshots that have been replicated and retained on the offload target.

The Protection Groups panel displays a list of all protection groups both local and remote to the array that are connected to the offload target. If the protection group exists on local array, click the name of the protection group to drill down to its protection group details; otherwise, hover over the name of the protection group to view its snapshot retention details.

In the Protection Group Snapshots panel, the details for each snapshot include the snapshot name, source array and protection group, replication start and end times, amount of data transferred, and replication progress. The data transferred amount is calculated as the size difference between the current and previous snapshots after data reduction.

In the following example, an offload target named **nfs-target** is connected to array **pure-001** and is an offload target for protection group **pgroup01**. Twelve protection group snapshots have been replicated to offload target **nfs-target**.



The screenshot shows the PureStorage Management UI interface. The top navigation bar includes tabs for Storage, Array, Hosts, Volumes, Protection Groups, and Pods. The current path is Array > Offload Targets > nfs-target. The main content area is divided into two sections: Protection Groups and Protection Group Snapshots.

**Protection Groups:**

Name	Source
pure-001:pgroup01	pure-001

Count: 1-1 of 1 | Destroyed (0)

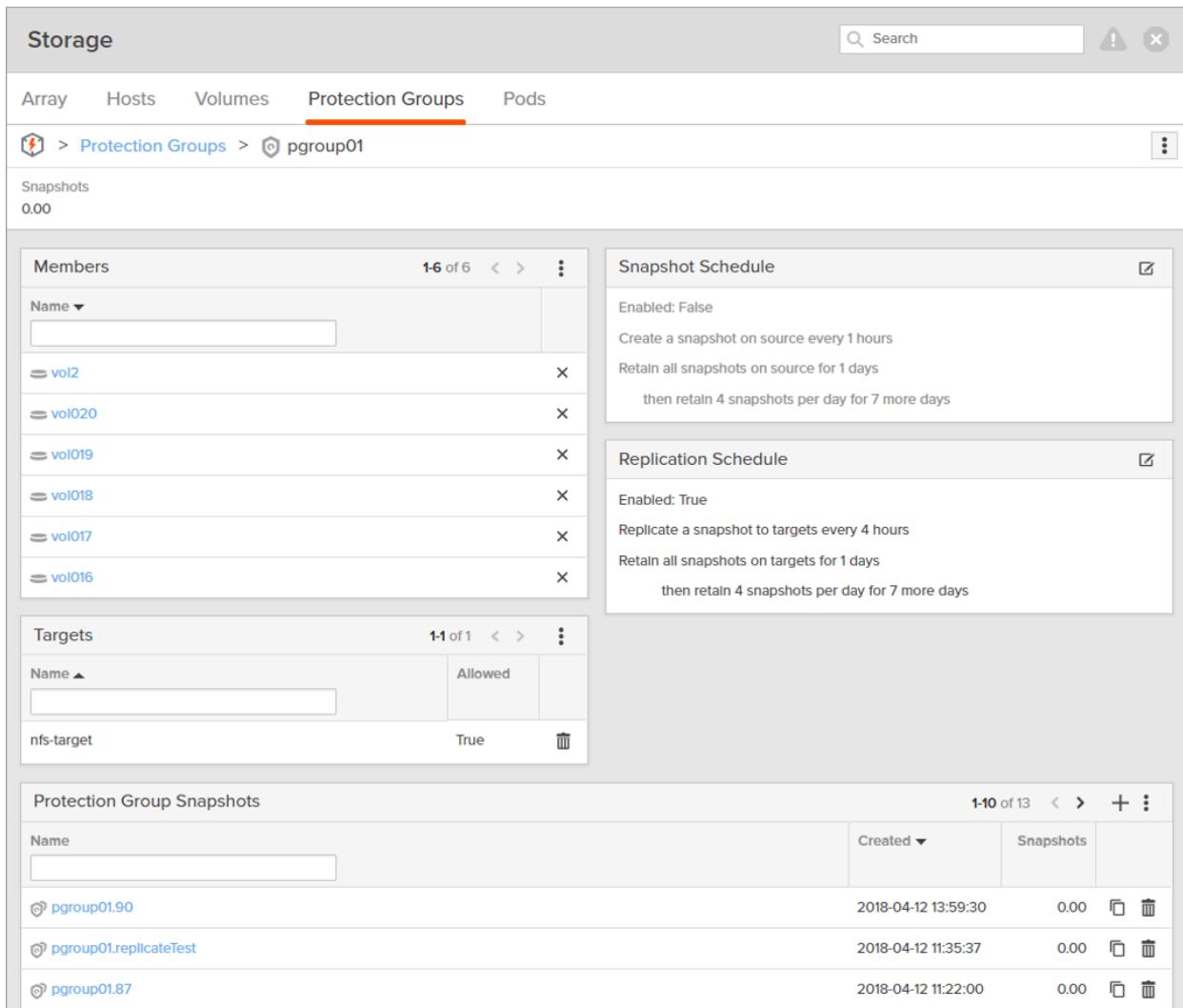
**Protection Group Snapshots:**

Name	Source	Created	Started	Completed	Transferred	Progress
pure-001:pgroup01.nfsTest	pure-001:pgroup01	2018-04-09 16:59:24	2018-04-09 16:59:24	2018-04-09 16:59:34	2.55 G	100%
pure-001:pgroup01.90	pure-001:pgroup01	2018-04-12 13:59:30	2018-04-12 13:59:30	2018-04-12 13:59:31	0.00	100%
pure-001:pgroup01.85	pure-001:pgroup01	2018-04-12 09:59:30	2018-04-12 11:07:00	2018-04-12 11:07:01	0.00	100%
pure-001:pgroup01.75	pure-001:pgroup01	2018-04-12 01:59:30	2018-04-12 01:59:30	2018-04-12 01:59:31	0.00	100%
pure-001:pgroup01.70	pure-001:pgroup01	2018-04-11 21:59:30	2018-04-11 21:59:30	2018-04-11 21:59:31	0.00	100%
pure-001:pgroup01.65	pure-001:pgroup01	2018-04-11 17:59:30	2018-04-11 17:59:30	2018-04-11 17:59:31	0.00	100%
pure-001:pgroup01.60	pure-001:pgroup01	2018-04-11 13:59:30	2018-04-11 13:59:30	2018-04-11 13:59:31	0.00	100%
pure-001:pgroup01.55	pure-001:pgroup01	2018-04-11 10:02:30	2018-04-11 10:48:00	2018-04-11 10:48:01	0.00	100%
pure-001:pgroup01.40	pure-001:pgroup01	2018-04-10 22:00:30	2018-04-10 22:00:30	2018-04-10 22:00:31	0.00	100%
pure-001:pgroup01.35	pure-001:pgroup01	2018-04-10 18:00:30	2018-04-10 18:00:30	2018-04-10 18:00:31	0.00	100%

Count: 1-10 of 12 | Destroyed (5)

Click a protection group to further drill down to its details, including the volumes that it protects, the snapshot and replication schedules, and the offload targets to where the protected volumes are replicated. In the Protection Group Snapshots panel, the protection group snapshots listed represent the snapshots that have been taken and retained on the current array in accordance with the snapshot schedule.

In the following example, protection group `pgroup01` has six volume members. The protection group is connected to offload target `nfs-target`. Thirteen protection group snapshots have been generated and retained on the array in accordance with the snapshot schedule. To view the protection group snapshots that have been replicated to the offload target, go to Storage > Array and click the name of the offload target. The protection group snapshots that have been replicated to the offload target appear in the Protection Group Snapshots panel.



Name	Created	Snapshots
pgroup01.90	2018-04-12 13:59:30	0.00
pgroup01.replicateTest	2018-04-12 11:35:37	0.00
pgroup01.87	2018-04-12 11:22:00	0.00

To replicate volume snapshots to an offload target, the array must be able to connect to and reach the external storage system. Before you configure an offload target on the array, perform the following steps to verify that the network is set up to support the offload process:

1. Verify that at least one interface with the replication service is configured on the array. Assign an IP address to the port; this will be interface that will be used to connect to the target device,

such as an Azure Blob container, a NAS/NFS device, an NFS storage system, an S3 bucket, or a generic Linux server. For optimum performance, an Ethernet interface of at least 10GbE is recommended.

2. Prepare the offload target.

- For Azure Blob, create a Microsoft Azure Blob container and set the storage account to the hot access tier. Grant basic read and write ACL permissions, and verify that the container contains no blobs. By default, server-side encryption is enabled for the container and cannot be disabled.
- For NFS, create the NFS export, granting read and write access to the array for all users.
- For S3, create an Amazon S3 bucket. Grant basic read and write ACL permissions, and enable default (server-side) encryption for the bucket. Also verify that the bucket is empty of all objects and does not have any lifecycle policies.

3. Verify that the array can reach the offload target.

After you have prepared the network connections on the array to support replication to an offload target, perform the following high-level steps to configure the offload target on the array:

1. Connect the array to the offload target.

- For Azure Blob, creating the connection to the Microsoft Azure Blob container requires the Azure Blob account name and the secret access key, both of which are created through the Microsoft Azure storage website.
- For NFS, creating the connection requires the host name or IP address of the server (such as the NFS server) and the mount point on the server.
- For S3, creating the connection to the Amazon S3 bucket requires the bucket's access key ID and secret access key, both of which are created through Amazon Web Services.

2. Define which volumes are to be replicated to the offload target.

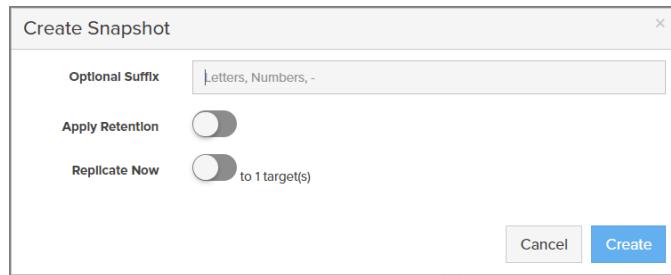
3. Create a protection group.

4. Add the volumes to the protection group.

5. Add the offload target to the protection group.

6. To replicate data to the offload target on a scheduled basis, set the replication schedule for the protection group, and then enable the schedule to begin replicating the volume snapshot data to the offload target according to the defined schedule. Skip this step if you only want to replicate data on demand.

Snapshot data can also be replicated on demand. On-demand snapshots represent single snapshots that are manually generated and retained on the source array at any point in time. When generating an on-demand snapshot, optionally add a suffix to the snapshot name, apply the scheduled retention policy to the snapshot, and asynchronously replicate the on-demand snapshot to the offload target.



The "Optional Suffix" option allows you to add a unique suffix to the on-demand snapshot name. The suffix name, which can include letters, numbers and dashes (-), replaces the protection group snapshot number in the protection group snapshot name. Select the "Apply Retention" option to apply the scheduled snapshot retention policy to the on-demand snapshot. If you do not enable "Apply Retention", the on-demand snapshot is saved until you manually destroy it.

Select the "Replicate Now" option to replicate the snapshot to the target arrays. The snapshot is retained on both the source and target arrays. If you also enable "Apply Retention", the snapshots inherit the retention policies set in the respective snapshot and retention schedules.

Restoring a volume brings the volume back to the state it was when the snapshot was taken. Restoring a volume from an offload target involves getting the volume snapshot from the offload target, and then copying the restored volume snapshot to create a new volume or overwrite an existing one. Volumes snapshots that have been replicated to an offload target can only be restored through the Purity//FA system.

Any array that is connected to the offload target can get the volume snapshots. However, only the array that configured the offload target can modify its protection group replication schedule and destroy, recover, and eradicate the protection group snapshots on the offload target.

Destroying a protection group implicitly destroys all of its protection group snapshots. Destroying a protection group snapshot destroys all of its protection group volume snapshots, thereby reclaiming the physical storage space occupied by its data.

Protection groups and protection group snapshots created for offload targets follow the same eradication pending behavior for most other FlashArray storage objects.

## Connecting arrays

Connect two arrays to perform asynchronous and synchronous replication.

To connect two arrays:

1. Log in to one of the arrays.
2. Select **Storage > Array**.
3. In the Connected Arrays panel, click the menu icon and select **Get Connection Key**. The Connection Key pop-up window appears.
4. Copy the connection key string.
5. Log in to the other array.
6. Select **Storage > Array**.

7. Click the menu icon and select **Connect Array**. The Connect Array pop-up window appears.
8. Set the following connection details:
  - In the Management Address field, enter the virtual IP address or FQDN of the other array.
  - In the Type field, select the replication type. Valid replication types include **async-replication** for asynchronous replication, and **sync-replication** for synchronous replication.  
Note that array connections set to **async-replication** support asynchronous replications only, while array connections set to **sync-replication** support both synchronous and asynchronous replications.
  - In the Connection Key field, paste the connection key string that you copied from the other array.
  - In the Replication Address field, enter the IP address or FQDN of the interface on the other array.
9. Click **Connect**. The array appears in the list of connected arrays and a green check mark appears in the row, indicating that the two arrays are successfully connected.

## Configuring network bandwidth throttling

Once two arrays are connected, optionally configure network bandwidth throttling to set maximum threshold values for outbound traffic.

To configure network bandwidth throttling:

1. Log in to the array in which you want to set threshold values for outbound traffic.
2. Select **Storage > Array**.
3. In the Connected Arrays panel, click the edit icon for the array you want to configure network bandwidth throttling. The Edit Bandwidth Throttling dialog box appears.
4. Configure the following options:
  - To specify a *default* bandwidth limit, enable (blue) the Default Throttle toggle button and specify a bandwidth limit for the amount of data transferred to the remote array per second. The bandwidth limit must be between 1 MB/s and 4 GB/s. To completely stop the data transfer process, set the limit to 0.  
and/or
  - To specify a *window* of time during which network bandwidth throttling takes effect, enable (blue) the Window Throttle toggle button, select the start and end times, and specify a bandwidth limit for the amount of data transferred to the remote array during the time range. The bandwidth limit must be between 1 MB/s and 4 GB/s. To completely stop the data transfer process during the specified window of time, set the limit to 0.
- If you set both limits, the “window” limit overrides the “default” limit.
5. Click **Save**. Bandwidth limit changes take effect immediately.

## Getting the array connection key

To get the array connection key:

1. Log in to the array.
2. Select **Storage > Array**.
3. In the Connected Arrays panel, click the menu icon and select **Get Connection Key**. The Connection Key pop-up window appears.
4. Copy the connection key string.

## Disconnecting arrays

For asynchronous replication, disconnecting two arrays suspends any in-progress data transfer processes. The process resumes when the arrays are reconnected.

For synchronous replication, you cannot disconnect the arrays if any pods are stretched between the two arrays.

To disconnect two arrays:

1. Log in to one of the arrays.
2. Select **Storage > Array**.
3. In the Connected Arrays panel, click the disconnect icon (X) for the array you want to disconnect.
4. Click **Disconnect**.

## Displaying offload targets connected to the array

1. Select **Storage > Array**. The Offload Targets panel displays a list of offload targets that are connected to the array. Offload targets that are disconnected from the array do not appear in the list.

## Displaying protection group and volume snapshot details for an offload target

1. Select **Storage > Array**.
2. Click the name of the offload target.

The Protection Groups panel displays a list of all protection groups both local and remote to the array that are connected to the offload target. If the protection group exists on local array, click the name of the protection group to drill down to its protection group details; otherwise, hover over the name of the protection group to view its snapshot retention details.

The Protection Group Snapshots panel displays a list of protection group snapshots that have been replicated to the offload target. To further drill down to see the volume snapshots for a protection group snapshot, click the corresponding **Get snapshots from offload targets** (download) icon.

## Connecting the array to an Azure Blob container

Before you connect the array to the Azure Blob container, ensure that you have the Azure Blob account name, the Azure Blob container name, and the secret access key. If this is the first time a FlashArray array is connecting to the Azure Blob container, verify that the container is empty.

1. Select **Storage > Array**.
2. In the Offload Targets panel, click the connect icon. The Connect Offload Target pop-up window appears.
3. In the Connect Offload Target pop-up window, specify the following details:
  - **Protocol:** Select **azure** from the drop-down list.
  - **Name:** Type a name for the Azure Blob offload target on the array.
  - **Account:** Type the Microsoft Azure Blob account. The account name is between 3 and 24 characters in length.
  - **Secret Access Key:** Type the secret access key of the Azure Blob account to authenticate requests between the array and Azure Blob container.
  - **Container:** Type the name of the Microsoft Azure Blob container. If not specified, the default is **offload**.
  - If this is the first time a FlashArray array is connecting to this container, select the check box next to Initialize container as offload target to prepare the Azure Blob container as an offload target. The array will only initialize the Azure Blob container if it is empty.  
If other FlashArray arrays have already connected to this container, do not select the check box.
4. Click **Connect**.

## Connecting the array to an NFS offload target

Before you connect the array to the NFS offload target, verify you have the hostname or IP address of the NFS server and the location of the mount point on the NFS server.

1. Select **Storage > Array**.
2. In the Offload Targets panel, click the connect icon. The Connect Offload Target pop-up window appears.
3. In the Connect NFS Target pop-up window, specify the following details:
  - **Protocol:** Select **nfs** from the drop-down list.
  - **Name:** Type a name for the NFS offload target on the array.
  - **Address:** Type the hostname or IP address of the NFS server.
  - **Mount Point:** Type the NFS export on the NFS server

- **Mount Options:** Specify mount options, as applicable. List mount options in comma-separated value (CSV) format. Supported mount options include `port`, `rsize`, `wsize`, `nfsvers`, and `tcp` or `udp`, and are common options available to all NFS file systems.
4. Click **Connect**.

## Connecting the array to an S3 bucket

Before you connect the array to the S3 bucket, verify you have the name of the S3 bucket and its access key ID and secret access key. If this is the first time a FlashArray array is connecting to the S3 bucket, verify that the bucket is empty.

1. Select **Storage > Array**.
2. In the Offload Targets panel, click the connect icon. The Connect Offload Target pop-up window appears.
3. In the Connect S3 Target pop-up window, specify the following details:
  - **Protocol:** Select `s3` from the drop-down list.
  - **Name:** Type a name for the S3 offload target on the array.
  - **Access Key ID:** Type the access key ID of the AWS account. The access key is 20 characters in length.
  - **Bucket:** Type the name of the Amazon S3 bucket.
  - **Secret Access Key:** Type the secret access key of the AWS account to authenticate requests between the array and S3 bucket. The secret access key is 40 characters in length.
  - If this is the first time a FlashArray array is connecting to this bucket, select the check box next to Initialize bucket as offload target to prepare the S3 bucket as an offload target. The array will only initialize the S3 bucket if it is empty.  
If other FlashArray arrays have already connected to this bucket, do not select the check box.
4. Click **Connect**.

## Disconnecting the array from an offload target

1. Select **Storage > Array**.
2. In the Offload Targets panel, click the **X** disconnect icon next to the offload target you want to disconnect.  
The Disconnect Target pop-up window appears.
3. Click **Disconnect**.

## Restoring a volume snapshot from an offload target to the array

1. Select **Storage > Array**.
2. Click the offload target from where you want to restore the volume snapshot.

3. In the Protection Group Snapshots panel, click the **Get snapshots from offload target** (download) icon.  
The Get Volume Snapshots pop-up window appears.
4. In the Existing Snapshots column, select the volume snapshot(s) you want to restore from the offload target.
5. Click **Get** to immediately restore the selected volume snapshot.  
The Summary panel appears with the list of restored volume snapshots. Click **OK**. Optionally click the **Go to Volumes page** link to view the restored snapshots in the Volume Snapshots panel.

Once a volume snapshot has been restored, it can be copied to create a new volume or overwrite an existing one.

## Destroying an offloaded protection group snapshot

1. Select **Storage > Array**.
2. Click the offload target from where you want to destroy the offloaded protection group snapshot.
3. In the Protection Group Snapshots panel, click the **Destroy Snapshot** icon. The Destroy Protection Group Snapshots pop-up window appears.
4. Click **Destroy**.

The destroyed snapshot appears in the Destroyed Protection Group Snapshots panel and begins its 24-hour eradication pending period.

During the eradication pending period, you can recover the protection group snapshot and its volume snapshots to bring it back to its previous state, or manually eradicate the destroyed protection group snapshot to reclaim physical storage space occupied by its volume snapshots. When the 24-hour eradication pending period has lapsed, Purity//FA starts reclaiming the physical storage occupied by the volume snapshots. Once reclamation starts, either because you have manually eradicated the destroyed protection group snapshot, or because the eradication pending period has lapsed, the destroyed protection group snapshots and its volume snapshots can no longer be recovered.

## Recovering a destroyed offloaded protection group snapshot

1. Select **Storage > Array**.
2. Click the offload target from where you want to recover the destroyed protection group snapshot.
3. At the bottom of the Protection Group Snapshots panel, click **Destroyed** to expand the window. The Destroyed Protection Group Snapshots panel appears.
4. In the Destroyed Protection Group Snapshots panel, click the **Recover Protection Group Snapshot** icon. The Recover Protection Group Snapshot pop-up window appears.
5. Click **Recover**.

## Eradicating a destroyed offloaded protection group snapshot

1. Select **Storage > Array**.
2. Click the offload target from where you want to eradicate the destroyed protection group snapshot.
3. At the bottom of the Protection Group Snapshots panel, click **Destroyed** to expand the window. The Destroyed Protection Group Snapshots panel appears.
4. In the Destroyed Protection Group Snapshots panel, click the **Eradicate Protection Group Snapshot** icon. The Eradicate Protection Group Snapshot pop-up window appears.
5. Click **Eradicate**.

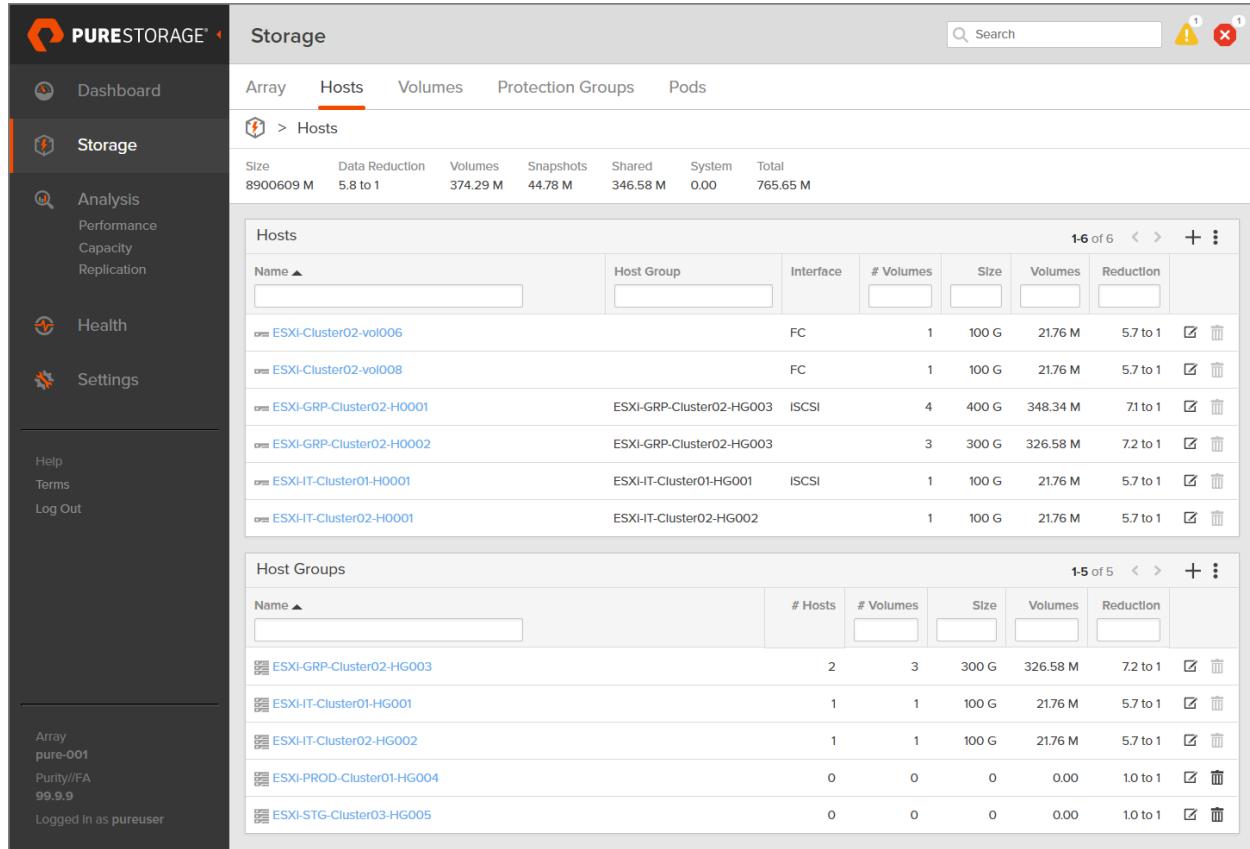
## Hosts and Host Groups

The Storage > Hosts page displays summary information for each host and host group on the array.

### Hosts

The Hosts panel displays summary information, including host group association, interface, connected volumes (both shared and private), provisioned size, and storage consumption for each host on the array.

Host names that begin with an @ symbol represent app hosts. For more information about app hosts, refer to the section called “Installed Apps” [190].



The screenshot shows the PureStorage Management UI with the following details:

- Left Sidebar:**
  - Dashboard
  - Storage** (selected)
  - Analysis
  - Performance
  - Capacity
  - Replication
  - Health
  - Settings
  - Help
  - Terms
  - Log Out
- Bottom Left:**
  - Array: pure-001
  - Purity//FA 99.9.9
  - Logged In as pureuser
- Top Header:** Storage Hosts and Host Groups
- Hosts Page:**
  - Summary table:

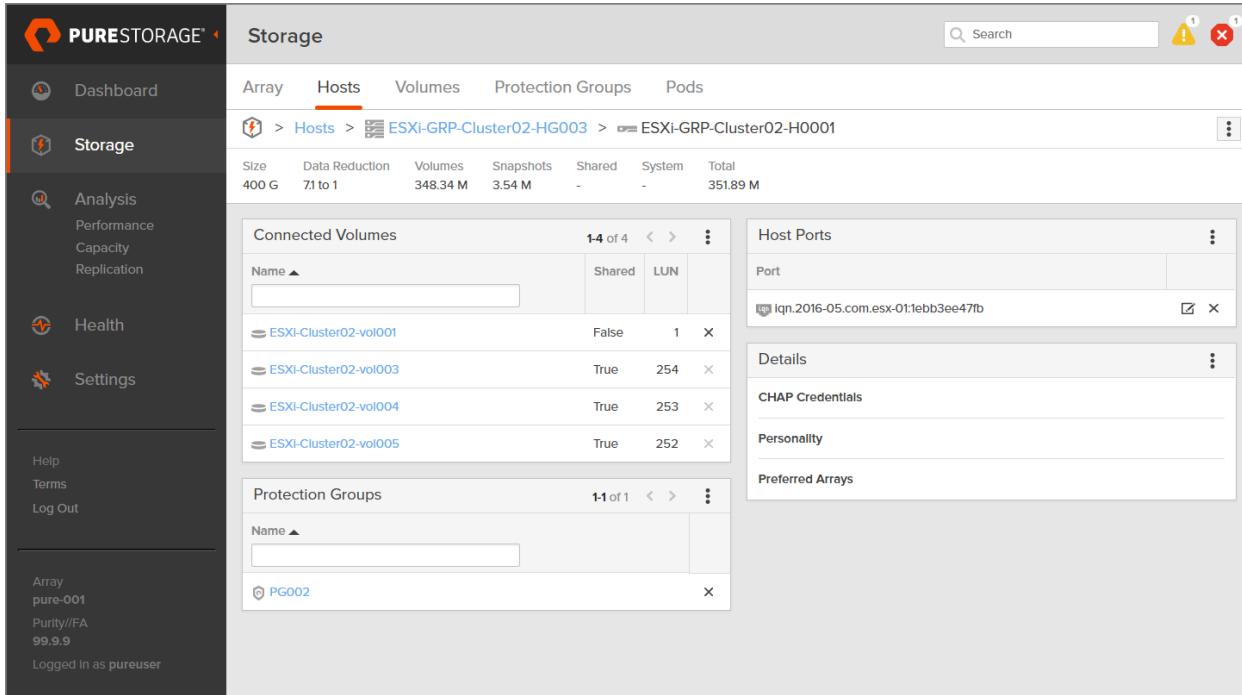
Size	Data Reduction	Volumes	Snapshots	Shared	System	Total
8900609 M	5.8 to 1	374.29 M	44.78 M	346.58 M	0.00	765.65 M

  - Hosts table (1-6 of 6):
 

Name	Host Group	Interface	# Volumes	Size	Volumes	Reduction
ESXI-Cluster02-vol006		FC	1	100 G	21.76 M	5.7 to 1
ESXI-Cluster02-vol008		FC	1	100 G	21.76 M	5.7 to 1
ESXI-GRP-Cluster02-HG001	ESXI-GRP-Cluster02-HG003	ISCSI	4	400 G	348.34 M	7.1 to 1
ESXI-GRP-Cluster02-HG002	ESXI-GRP-Cluster02-HG003		3	300 G	326.58 M	7.2 to 1
ESXI-IT-Cluster01-HG001	ESXI-IT-Cluster01-HG001	ISCSI	1	100 G	21.76 M	5.7 to 1
ESXI-IT-Cluster02-HG001	ESXI-IT-Cluster02-HG002		1	100 G	21.76 M	5.7 to 1
- Host Groups Page:**
  - Host Groups table (1-5 of 5):
 

Name	# Hosts	# Volumes	Size	Volumes	Reduction
ESXI-GRP-Cluster02-HG003	2	3	300 G	326.58 M	7.2 to 1
ESXI-IT-Cluster01-HG001	1	1	100 G	21.76 M	5.7 to 1
ESXI-IT-Cluster02-HG002	1	1	100 G	21.76 M	5.7 to 1
ESXI-PROD-Cluster01-HG004	0	0	0	0.00	1.0 to 1
ESXI-STG-Cluster03-HG005	0	0	0	0.00	1.0 to 1

From the Hosts page, click a host name to display its details. The following example displays the details for host **ESXi-GRP-Cluster02-H0001**, which is connected to one host group (**ESXi-GRP-Cluster02-HG003**) and four volumes, and is a member of protection group **PG002**.



The screenshot shows the PureStorage Management UI interface. The left sidebar has navigation links: Dashboard, Storage (selected), Analysis, Health, Settings, Help, Terms, and Log Out. Below the sidebar, it shows the array information: Array pure-001, Purity/FA 99.9%, and Logged In as pureuser.

The main content area is titled "Storage" and has tabs: Array, Hosts (selected), Volumes, Protection Groups, and Pods. The breadcrumb navigation shows: Hosts > ESXi-GRP-Cluster02-HG003 > ESXi-GRP-Cluster02-H0001. Below the breadcrumb, there's a summary table:

Size	Data Reduction	Volumes	Snapshots	Shared	System	Total
400 G	7:1 to 1	348.34 M	3.54 M	-	-	351.89 M

Below the summary table are two panes:

- Connected Volumes:** Shows four volumes connected to the host. The table includes columns: Name, Shared, and LUN. The volumes are:
 

ESXi-Cluster02-vol001	False	1	X
ESXi-Cluster02-vol003	True	254	X
ESXi-Cluster02-vol004	True	253	X
ESXi-Cluster02-vol005	True	252	X
- Protection Groups:** Shows one protection group named PG002.

On the right side of the screen, there are several collapsed sections: Host Ports, Details, CHAP Credentials, Personality, and Preferred Arrays.

The Host details page contains the following panes:

### Connected Volumes

Displays a list of volumes that have private connections to the host or shared connections through a host group.

### Host Ports

Displays a list of the iSCSI Qualified Names (IQNs), NVMe Qualified Names (NQNs), or Fibre Channel World Wide Names (WWNs) of the ports associated with the host.

### Protection Groups

Displays any protection groups to which the host belongs.

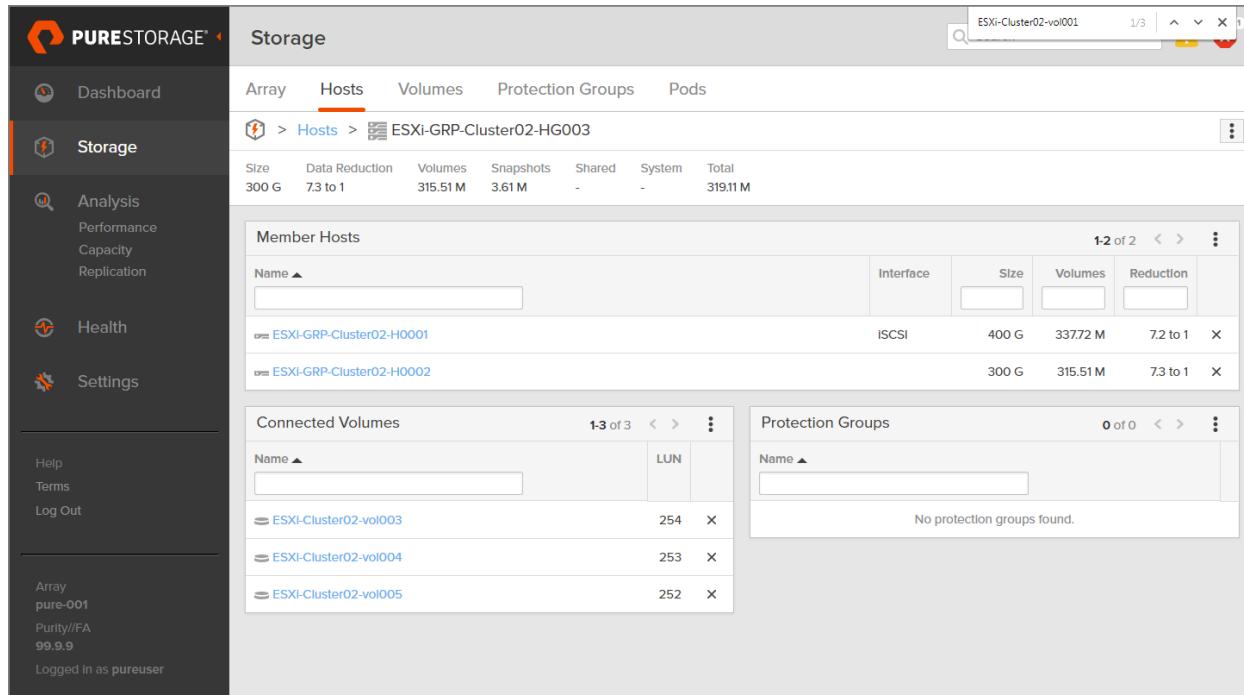
### Details

Displays additional details specific to the selected host, including CHAP credentials and host personality.

## Host Groups

In the Storage > Hosts page, the Host Groups panel displays summary information, including host associations, connected (shared) volumes, provisioned size, and storage consumption, for each host group on the array.

From the Hosts page, click a host group name to display its details. The following example displays the details for host group **ESXi-GRP-Cluster02-HG003**, which is connected to two hosts and three volumes.



Name	Interface	Size	Volumes	Reduction
ESXi-GRP-Cluster02-H0001	iSCSI	400 G	337.72 M	7.2 to 1
ESXi-GRP-Cluster02-H0002		300 G	315.51 M	7.3 to 1

Name	LUN	
ESXi-Cluster02-vol003	254	X
ESXi-Cluster02-vol004	253	X
ESXi-Cluster02-vol005	252	X

The Host Group details page contains the following panes:

### Member Hosts

Displays a list of hosts that have been added to the host group.

### Connected Volumes

Displays a list of volumes that have shared connections to the host group.

### Protection Groups

Displays any protection groups to which the host group belongs.

## Creating hosts

Create hosts to access volumes on the array. Create a single host or multiple hosts at one time.

To create a host:

1. Select **Storage > Hosts**.
2. In the Hosts panel, click the menu icon and select **Create...**. The Create Host dialog box appears.

3. In the Name field, type the name of the new host.
4. Click **Create**.
5. For the host you just created, perform the following steps to configure the host personality:
  - a. In the Hosts panel, click the name of the host you just created to drill down to its details.
  - b. In the Details panel, click the menu icon and select **Set Personality....** The Configure Personality dialog box appears.
  - c. Select the name of the host operating or virtual memory system. If your host personality does not appear in the list, select **None**.
  - d. Click **Save**.

To create multiple hosts:

1. Select **Storage > Hosts**.
2. In the Hosts panel, click the menu icon and select **Create...**. The Create Host dialog box appears.
3. Click **Create Multiple...**. The Create Multiple Hosts dialog box appears.
4. Complete the following fields:
  - **Name:** Specify the template used to create the host names. Host names cannot consist of all numeric values.  
Place the hash (#) symbol where the numeric part of the host name should appear. When Purity//FA creates the host names, the hash symbol is replaced with the host number, beginning with the start number specified.
  - **Start Number:** Enter the host number used to create the first host name.
  - **Count:** Enter the number of hosts to create.
  - **Number of Digits:** Enter the minimum number of numeric digits of the host number. If the number of digits is greater than the start number, the host number begins with leading zeros.
5. Click **Create**.
6. For each of the hosts you just created, perform the following steps to configure the host personality:
  - a. In the Hosts panel, click the name of the host you just created to drill down to its details.
  - b. In the Details panel, click the menu icon and select **Set Personality....** The Configure Personality dialog box appears.
  - c. Select the name of the host operating or virtual memory system. If your host personality does not appear in the list, select **None**.
  - d. Click **Save**.

## Creating host groups

Create a host group if several hosts share access to the same volume(s). After you create the host group, add the hosts to the host group and then establish a shared connection between the volumes and the host group. Once a shared connection is established, all of the hosts within the host group share access to the volumes.

Create a single host group or multiple host groups at one time.

To create a host group:

1. Select **Storage > Hosts**.
2. In the Host Groups panel, click the menu icon and select **Create...**. The Create Host Group dialog box appears.
3. In the Name field, type the name of the new host group.
4. Click **Create**.

To create multiple host groups:

1. Select **Storage > Hosts**.
2. In the Host Groups panel, click the menu icon and select **Create...**. The Create Host Groups dialog box appears.
3. Click **Create Multiple...**. The Create Multiple Host Groups dialog box appears.
4. Complete the following fields:
  - **Name:** Specify the template used to create the host group names. Host group names cannot consist of all numeric values.  
Place the hash (#) symbol where the numeric part of the host group name should appear. When Purity//FA creates the host group names, the hash symbol is replaced with the host group number, beginning with the start number specified.
  - **Start number:** Enter the host group number used to create the first host group name.
  - **Count:** Enter the number of host groups to create.
  - **Number of Digits:** Enter the minimum number of numeric digits of the host group number. If the number of digits is greater than the start number, the host group number begins with leading zeros.
5. Click **Create**.

## Configuring host ports

Hosts communicate with the array via one or more IQNs, NQNs, or WWNs. The initiators are either discovered by Purity//FA or manually assigned.

To associate iSCSI IQNs with a host:

1. Select **Storage > Hosts**.

2. In the Hosts panel, click the host name to drill down to its details.
3. In the Host Ports panel, click the menu icon and select **Configure IQNs....** The Configure iSCSI IQNs dialog box appears.
4. In the Port IQNs field, type the IQNs in comma-separated format.
5. Click **Add**.

To associate NVMe-oF NQNs with a host:

1. Select **Storage > Hosts**.
2. In the Hosts panel, click the host name to drill down to its details.
3. In the Host Ports panel, click the menu icon and select **Configure NQNs....** The Configure NVMe-oF NQNs dialog box appears.
4. In the Port NQNs field, type the NQNs in comma-separated format.
5. Click **Add**.

To associate Fibre Channel WWNs with a host:

1. Select **Storage > Hosts**.
2. In the Hosts panel, click the host name to drill down to its details.
3. In the Host Ports panel, click the menu icon and select **Configure Fibre Channel WWNs....** The Configure Fibre Channel WWNs dialog box appears.  
The WWNs in the Existing WWNs column of the dialog box represent the WWNs that have been discovered by Purity//FA (i.e., the WWNs of computers whose initiators have "logged in" to the array).
4. Click an existing WWN in the left column to add it to the Selected WWNs column.  
Alternatively, to manually add a WWN, click **Enter WWNs Manually** and type the WWNs, in comma-separated format, in the Port WWNs field.
5. Click **Add**.

## Adding hosts to host groups

Adding a host to a host group automatically establishes connections between it and all volumes with shared connections to the host group.

To add a host to a host group:

1. Select **Storage > Hosts**.
2. In the Volumes panel, click the volume to drill down to its details. In the Host Groups panel, click the host to drill down to its details.
3. In the Member Hosts panel, click the menu icon and select **Add....** The Add Hosts to Host Group dialog box appears. The hosts in the Existing Hosts column of the dialog box represent the hosts that are not associated with any host groups and are eligible to be connected to the host group.

4. Click an existing host in the left column to add it to the Selected Hosts column.
5. Click **Add**.

## Configuring CHAP authentication

To configure iSCSI CHAP authentication:

1. Select **Storage > Hosts**.
2. In the Hosts panel, click the host for which you want to configure CHAP authentication.
3. In the Details panel, click the menu icon and select **Configure CHAP...**. The Configure CHAP dialog box appears.
4. Complete the following fields:
  - **Host User**: Set the host user name for CHAP authentication.
  - **Host Password**: Enter the host password for CHAP authentication. The password must be between 12 and 255 characters (inclusive) and cannot be the same as the target password.
  - **Target User**: Set the target user name for CHAP authentication.
  - **Target Password**: Enter the target password for CHAP authentication. The host password cannot be the same as the target password. The password must be between 12 and 255 characters (inclusive) and cannot be the same as the host password.
5. Click **Save**. To disable CHAP, clear the fields in the Configure CHAP dialog box and click **Save**.

## Configuring host personalities

To configure host personalities:

1. Select **Storage > Hosts**.
2. In the Hosts panel, click the host for which you want to configure the host personality.
3. In the Details panel, click the menu icon and select **Set Personality...**. The Configure Personality dialog box appears.
4. Select the name of the host operating or virtual memory system. If your host personality does not appear in the list, select **None**.
5. Click **Save**.

## Adding preferred arrays

To add a preferred array:

1. Select **Storage > Hosts**.
2. In the Hosts panel, click the host for which you want to add a preferred array.

3. In the Details panel, click the menu icon and select **Add Preferred Arrays...**. The Add Preferred Arrays dialog box appears.
4. From the Available Arrays column, click the arrays you want to add as preferred arrays for the host.
5. Click **Add**.

## Removing preferred arrays

To remove a preferred array:

1. Select **Storage > Hosts**.
2. In the Hosts panel, click the host from which you want to remove a preferred array.
3. In the Details panel, click the **x** for the array you want to remove as a preferred array. The Remove Preferred Array dialog box appears.
4. Click **Remove**.

## Renaming a host

To rename a host:

1. Select **Storage > Hosts**.
2. In the Hosts panel, click the rename icon for the host you want to rename. The Rename Host dialog box appears.
3. In the Name field, enter the new name of the host.
4. Click **Rename**.

## Deleting a host

either through private or shared connections. You cannot delete a host if it is connected to volumes, Before deleting a host, disconnect all hosts and volumes from the host group.

To delete a host:

1. Select **Storage > Hosts**.
2. In the Hosts panel, click the delete icon for the host you want to delete. The Delete Host dialog box appears.
3. Click **Delete**. Any volumes that were connected to the host are disconnected, and the deleted host no longer appears in the Host panel.

## Renaming a host group

To rename a host group:

1. Select **Storage > Hosts**.

2. In the Host Groups panel, click the rename icon for the host group you want to rename. The Rename Host Group dialog box appears.
3. In the Name field, enter the new name of the host group.
4. Click **Rename**.

## Deleting a host group

You cannot delete a host if it is connected to volumes, either through private or shared connections. Before deleting a host, break all of its connections. You cannot delete a host group if it is connected to volumes or hosts. Before deleting a host group, disconnect all hosts and volumes from the host group.

To delete a host group:

1. Select **Storage > Hosts**.
2. In the Host Groups panel, click the delete icon for the host group you want to delete. The Delete Host Group dialog box appears.
3. Click **Delete**.

## Removing a host from a host group

Removing a host from a host group disconnects all volumes with shared connections from the removed host. The removed host's private connections are unaffected.

1. Select **Storage > Hosts**.
2. In the Host Groups panel, click the host group to drill down to its details.
3. From the Member Hosts panel, click the remove host (x) icon next to the host you want to disconnect. The Remove Host dialog box appears.
4. Click **Remove**.

## Removing a host port

Warning: Removing a host may disrupt connectivity to some volumes.

To disassociate an IQN, NQN, or WWN from a host:

1. Select **Storage > Hosts**.
2. In the Hosts panel, click the host from which you want to remove the host ports.
3. From the Host Ports panel, click the remove port (x) icon next to the IQN, NQN, or WWN you want to remove. The Remove Selected Port dialog box appears.
4. Click **Remove**. Purity//FA immediately breaks the association and any communication with the array via the initiator associated with the removed IQN, NQN, or WWN.

## Downloading host details

Downloading host details generates a comma-separated value text file containing host summary information.

To download host details:

1. Select **Storage > Hosts**.

In the Hosts panel, click the menu icon and select **Download CSV** to save the `hosts.csv` file to your local machine.

## Downloading host group details

Downloading host group details generates a comma-separated value text file containing host group summary information.

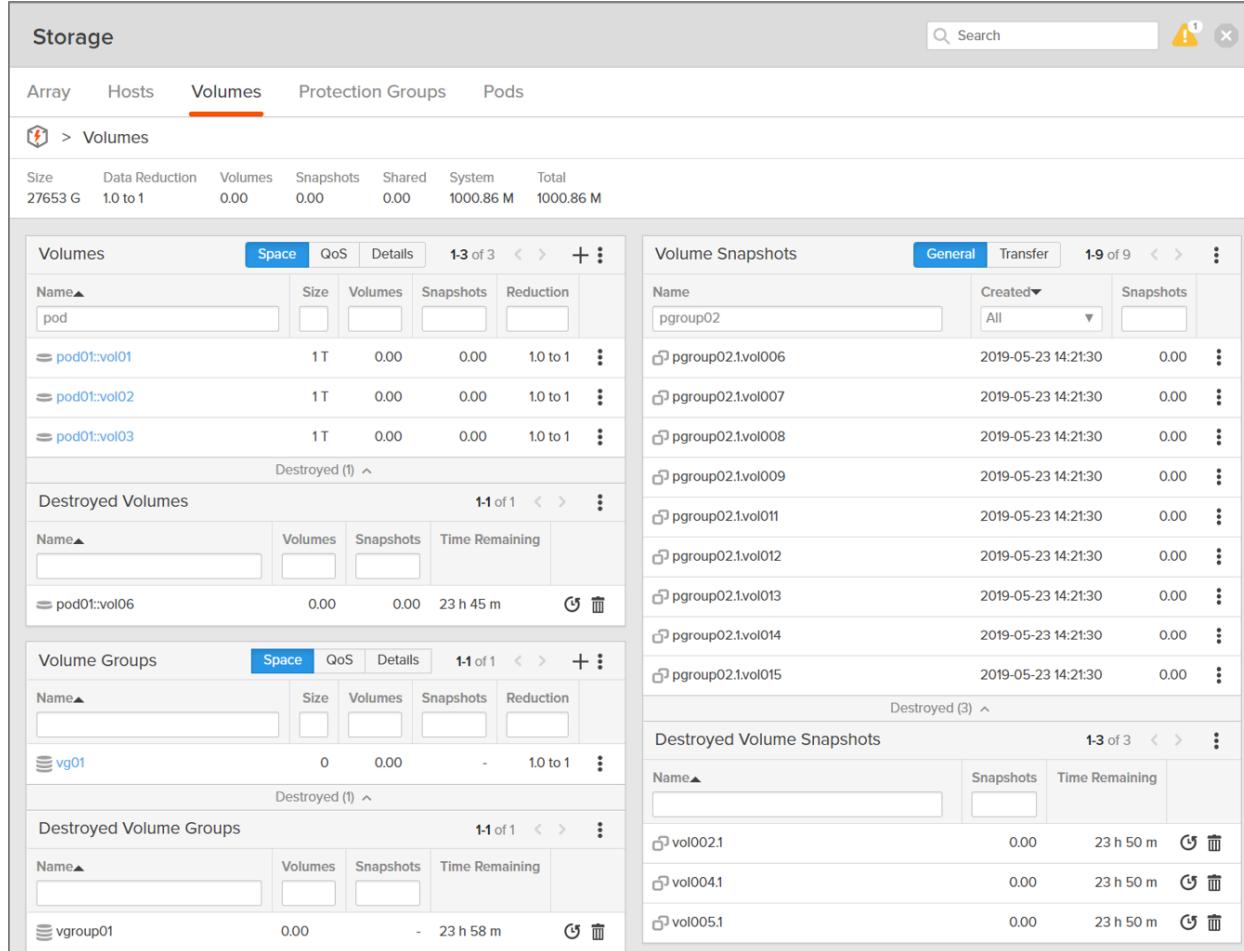
To download host group details:

1. Select **Storage > Hosts**.

In the Host Groups panel, click the menu icon and select **Download CSV** to save the `host_groups.csv` file to your local machine.

## Volumes

The Storage > Volumes page displays summary information for all volumes on the array.



Size	Data Reduction	Volumes	Snapshots	Shared	System	Total
27653 G	1.0 to 1	0.00	0.00	0.00	1000.86 M	1000.86 M

Name	Size	Volumes	Snapshots	Reduction
pod				
pod01::vol01	1T	0.00	0.00	1.0 to 1
pod01::vol02	1T	0.00	0.00	1.0 to 1
pod01::vol03	1T	0.00	0.00	1.0 to 1
Destroyed (1) ^				
Destroyed Volumes				
Name	Volumes	Snapshots	Time Remaining	
pod01::vol06	0.00	0.00	23 h 45 m	
Destroyed (1) ^				
Name	Size	Volumes	Snapshots	Reduction
vg01	0	0.00	-	1.0 to 1
Destroyed (1) ^				
Destroyed Volume Groups				
Name	Volumes	Snapshots	Time Remaining	
vgroup01	0.00	-	23 h 58 m	

Name	Created	Snapshots
pgroup02	All	
pgroup02.1.vol006	2019-05-23 14:21:30	0.00
pgroup02.1.vol007	2019-05-23 14:21:30	0.00
pgroup02.1.vol008	2019-05-23 14:21:30	0.00
pgroup02.1.vol009	2019-05-23 14:21:30	0.00
pgroup02.1.vol011	2019-05-23 14:21:30	0.00
pgroup02.1.vol012	2019-05-23 14:21:30	0.00
pgroup02.1.vol013	2019-05-23 14:21:30	0.00
pgroup02.1.vol014	2019-05-23 14:21:30	0.00
pgroup02.1.vol015	2019-05-23 14:21:30	0.00
Destroyed (3) ^		
Destroyed Volume Snapshots		
Name	Snapshots	Time Remaining
vol002.1	0.00	23 h 50 m
vol004.1	0.00	23 h 50 m
vol005.1	0.00	23 h 50 m

## Volumes

The Volumes panel displays a list of all volumes on the array.

Volume names that include a double colon (::) represent volumes inside pods. Volume names that include a forward slash (/) represent volumes inside volume groups.

Volume names that begin with an @ symbol represent app volumes. For more information about app volumes, refer to the section called “Installed Apps” [190].

The Volumes page also displays volumes, volume snapshots, and volume groups that have been destroyed and are pending eradication.

The Volumes panel is organized into the following three subpanels:

- **Details** - Displays general information about each volume, including the number of hosts to which the volume is connected either through private or shared connections, and the unique serial number of the volume.
- **Space** - Displays information about the provisioned (virtual) size, snapshots, and physical storage consumption for each volume.
- **QoS** - Displays the bandwidth limit and the IOPS limit of each volume. If the bandwidth limit is not set, the value appears as a dash (-), representing unlimited throughput. If the IOPS limit is not set, the value appears as a dash (-), representing unlimited IOPS.

## Storage Containers

A volume can reside in one of the following types of storage containers: root of the array (""), pod, or volume group. The most simple of array configurations is one that contains volumes at the root of the array. Each pod and volume group is a separate namespace for the volumes it contains.

Pods are created and configured to store volumes and protection groups that need to be fully synchronized with other arrays.

Each volume in a pod consists of the pod namespace identifier and the volume name, separated by a double colon (::). The naming convention for a volume inside a pod is **POD::VOL**, where:

- **POD** is the name of the container pod.
- **VOL** is the name of the volume inside the pod.

For example, the fully qualified name of a volume named **vol01** inside a pod named **pod01** is **pod01::vol01**.

For more information about pods, refer to the section called “Pods” [119].

Volume groups organize volumes into logical groupings. If virtual volumes are configured, a volume group is automatically created for each virtual machine that is created.

Each volume in a volume group consists of the volume group namespace identifier and the volume name, separated by a forward slash (/). The naming convention for a volume inside a volume group is **VGROUP/VOL**, where:

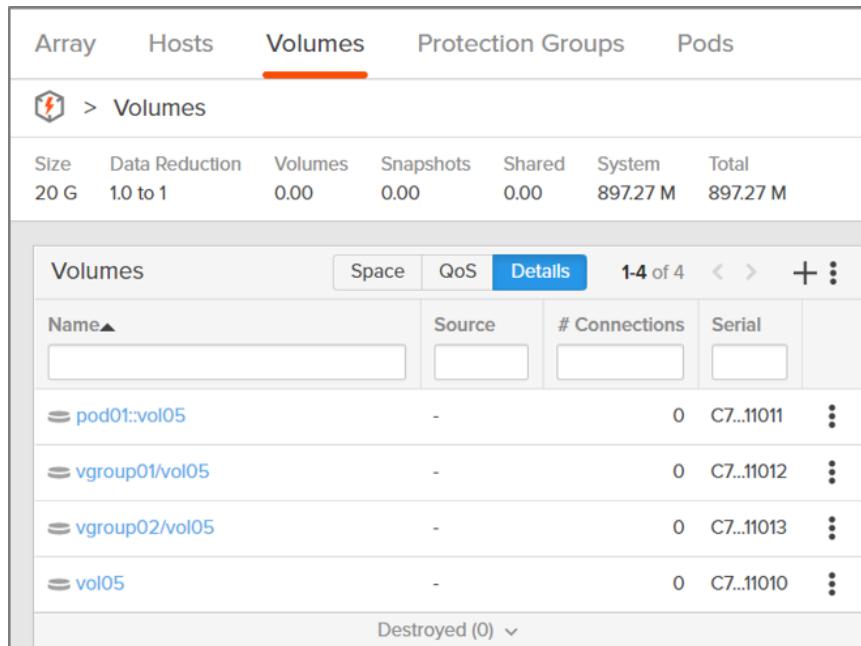
- **VGROUP** is the name of the container volume group.
- **VOL** is the name of the volume in the volume group.

For example, the fully qualified name of a volume named **vol01** inside a volume group named **vgroup01** is **vgroup01/vol01**.

For more information about volume groups, refer to the section called “Volume Groups” [89].

Volumes that reside in one storage container are independent of the volumes that reside in other containers. For example, a volume named **vol01** is completely unrelated to a volume named **vgroup01/vol01**.

In the following example, volume `vol05` represents a volume named vol05 that resides on the root of the array, volume `vgroup01/vol05` represents a volume named vol05 that resides in volume group vgroup01, volume `vgroup02/vol05` represents a volume named vol05 that resides in volume group vgroup02, and volume `pod01::vol05` represents a volume named vol05 that resides in pod pod01. Though all four volumes have "vol05" in their names, they are completely independent of one another.



Name	Source	# Connections	Serial
pod01::vol05	-	0	C7...11011
vgroup01/vol05	-	0	C7...11012
vgroup02/vol05	-	0	C7...11013
vol05	-	0	C7...11010

## Virtual Volumes

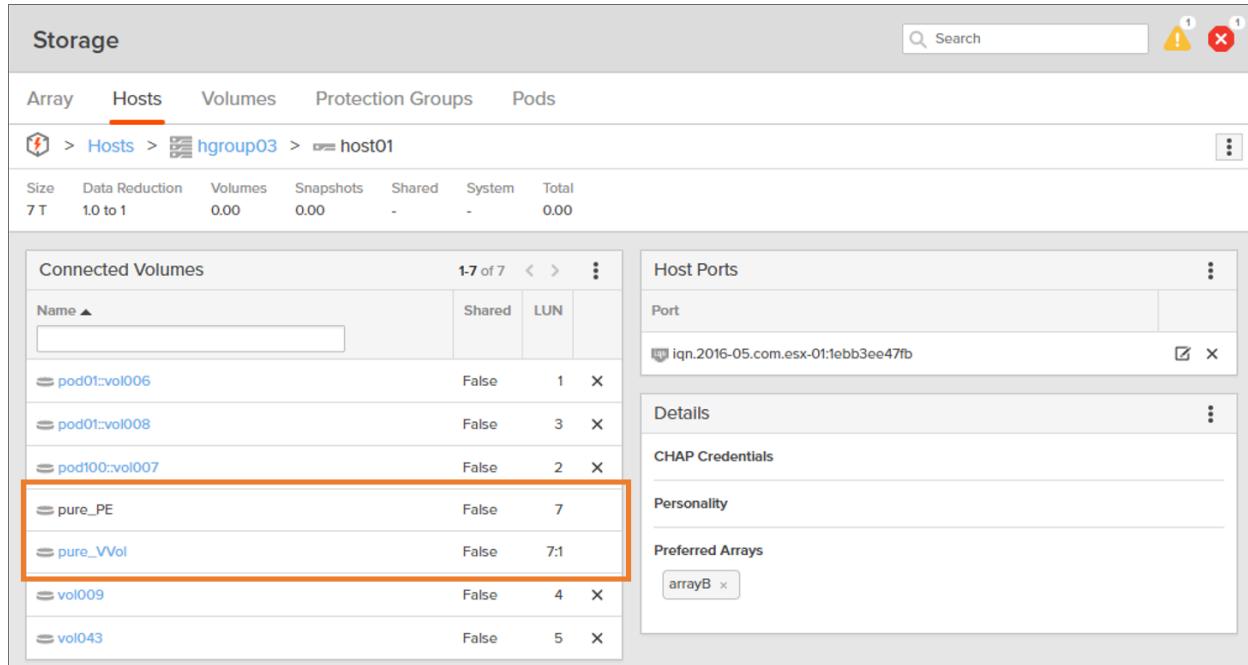
VMware's Virtual Volumes (VVols) storage architecture is designed to give VMware administrators the ability to perform volume operations and apply protection group snapshot and replication policies to FlashArray volumes directly through vSphere.

On the FlashArray side, virtual volumes are created and then connected to VMware ESXi hosts or host groups via a protocol endpoint (also known as a conglomerate volume). The protocol endpoint itself does not serve I/Os; instead, its job is to form connections between FlashArray volumes and ESXi hosts and host groups.

Each protocol endpoint can connect multiple virtual volumes to a single host or host group, and each host or host group can have multiple protocol-endpoint connections.

LUN IDs are automatically assigned to each protocol endpoint connection and each virtual volume connection. Specifically, each protocol endpoint connection to a host or host group creates a LUN (PE LUN), while each virtual volume connection to a host or host group creates a sub-LUN. The sub-LUN is in the format `x:y`, where `x` represents the LUN of the protocol endpoint through which the virtual volume is connected to the host or host group, and `y` represents the sub-LUN assigned to the virtual volume.

In the following example, one virtual volume named `pure_VVol` and one protocol endpoint named `pure_PE` are connected to host `host01`. The virtual volume is identified by the sub-LUN (`7:1`).



Name	Shared	LUN
<code>pod01:vol006</code>	False	1
<code>pod01:vol008</code>	False	3
<code>pod100:vol007</code>	False	2
<code>pure_PE</code>	False	7
<code>pure_VVol</code>	False	7:1
<code>vol009</code>	False	4
<code>vol043</code>	False	5

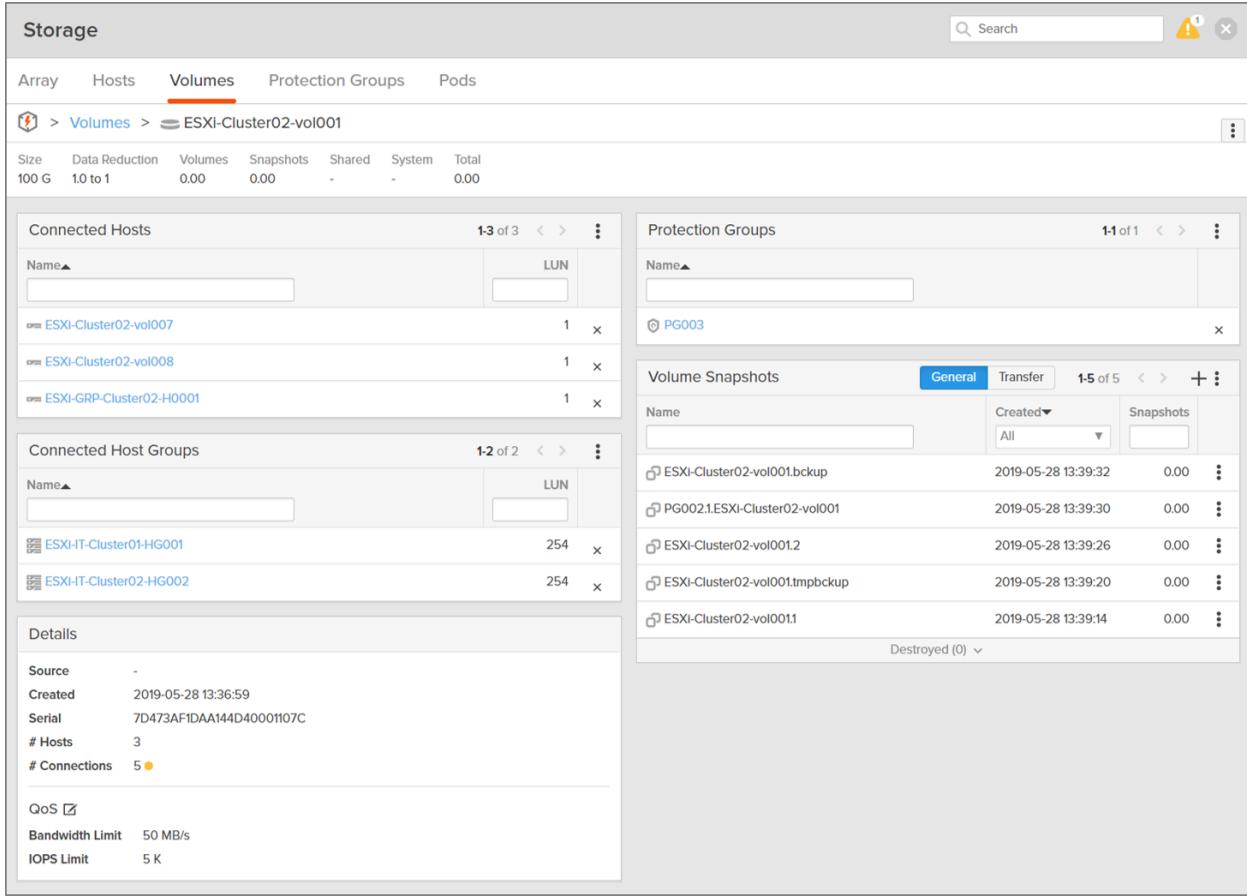
Note that virtual volumes are primarily configured through the vSphere Web Client plugin.

For more information about virtual volumes, including configuration steps, refer to the Pure Storage vSphere Web Client Plugin for vSphere User Guide in the Pure Storage Knowledge Base at <https://support.purestorage.com>.

## Volume Details

From the Volumes page, click a volume name to display details, including connected hosts (through both shared and private connections), provisioned size, storage consumption, and serial number, for the specified volume on the array.

The following example displays the details for volume `esxi-Cluster02-vol001`, which is connected to three hosts and two host groups, and is a member of protection group `PG003`.



The screenshot shows the PureStorage Storage Volumes interface. The top navigation bar has tabs for Array, Hosts, Volumes (which is selected), Protection Groups, and Pods. Below the navigation is a breadcrumb trail: Home > Volumes > ESXi-Cluster02-vol001. A search bar and a status indicator (yellow exclamation mark) are also at the top.

**Connected Hosts:** Shows 13 connections to the volume. Three hosts are listed: ESXi-Cluster02-vol007, ESXi-Cluster02-vol008, and ESXi-GRP-Cluster02-H001. Each host has a LUN assigned.

**Connected Host Groups:** Shows 2 host groups connected. Two groups are listed: ESXi-IT-Cluster01-HG001 and ESXi-IT-Cluster02-HG002, each with 254 LUNs assigned.

**Protection Groups:** Shows 1 protection group named PG003. It contains 5 snapshots:

Name	Created	Snapshots
ESXi-Cluster02-vol001.bckup	2019-05-28 13:39:32	0.00
PG002.1.ESXi-Cluster02-vol001	2019-05-28 13:39:30	0.00
ESXi-Cluster02-vol001.2	2019-05-28 13:39:26	0.00
ESXi-Cluster02-vol001.tmpbckup	2019-05-28 13:39:20	0.00
ESXi-Cluster02-vol001.1	2019-05-28 13:39:14	0.00

**Details:** Displays unique volume details:

- Source: -
- Created: 2019-05-28 13:36:59
- Serial: 7D473AF1DAA144D40001107C
- # Hosts: 3
- # Connections: 5
- QoS: [checkbox]
- Bandwidth Limit: 50 MB/s
- IOPS Limit: 5 K

The Volume details page contains the following panes:

### Connected Hosts

Displays a list of private host connections to the volume, and the LUNs they use to address the volume.

### Connected Host Groups

Displays a list of public host connections to the volume, and the LUNs they use to address the volume.

### Protection Groups

Displays any protection groups to which the volume belongs.

### Volume Snapshots

Displays a list of volume snapshots. A volume snapshot is a point-in-time image of the contents of a volume. There are various ways to create volume snapshots: as a single volume or multiple volumes at the same time (atomically) through the Storage > Volumes page, or as part of protection group snapshots through the Storage > Protection Groups page.

### Details

Displays the unique details for the volume, such as volume creation date, unique serial number, and QoS information including bandwidth limit and IOPS limit. If the volume was created from

another source, such as a volume snapshot, the Source field displays the name of the source from where the volume was created.

## Volume Groups

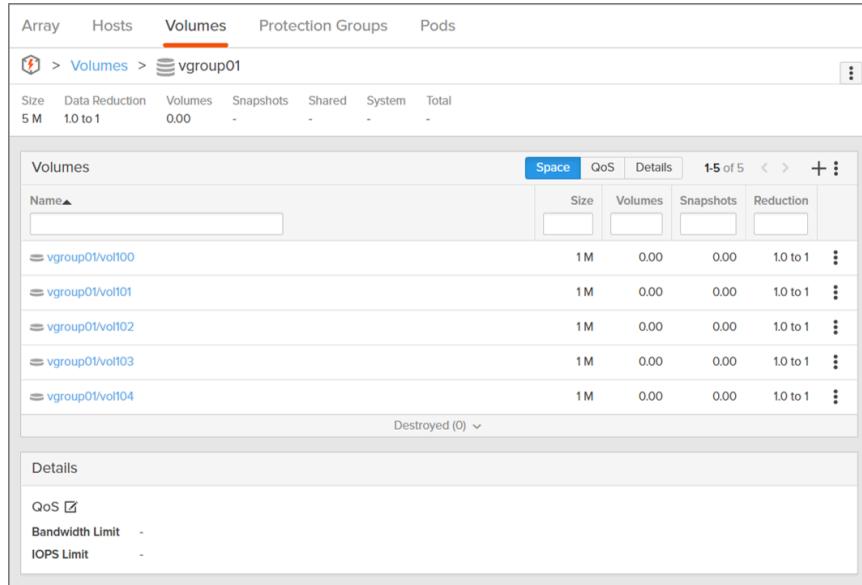
The Volumes Groups panel displays a list of volume groups that have been created on the array. Volume groups organize FlashArray volumes into logical groupings.

If virtual volumes are configured, each volume group on the array represents its associated virtual machine, and inside each of those volume groups are the FlashArray volumes that are assigned to the virtual machine. Volume groups that are associated with virtual machines have names that begin with "vvol1-" and end with the virtual machine name. For more information about virtual volumes, including configuration steps, refer to the Pure Storage vSphere Web Client Plugin for vSphere User Guide in the Pure Storage Knowledge Base at <https://support.purestorage.com>.

Volume groups can also be created through the Volumes page. Once a volume group has been created, create new volumes directly in the volume group or move existing ones into the volume group.

In the Volume Groups panel, click the name of a volume group to display its details, such as provisioned size, storage consumption, and a list of volumes that reside in the volume group.

The following example displays the details for volume group **vgroup01**, which contains five volumes.



Name	Size	Volumes	Snapshots	Shared	System	Total
vgroup01/vol100	1 M	0.00	0.00	-	-	-
vgroup01/vol101	1 M	0.00	0.00	1.0 to 1	-	-
vgroup01/vol102	1 M	0.00	0.00	1.0 to 1	-	-
vgroup01/vol103	1 M	0.00	0.00	1.0 to 1	-	-
vgroup01/vol104	1 M	0.00	0.00	1.0 to 1	-	-

## Quality of Service Limits

Quality of service (QoS) limits define the maximum level of throughput and the maximum number of I/O operations per second for a volume or volume group. The QoS limits of a volume can be set when you create or configure the volume, but the QoS limits of a volume group can be set only when you configure the volume group. Note that you cannot set the QoS limits on individual volumes in a volume group.

Quality of service limits include:

- **QoS Bandwidth Limit**

The bandwidth limit can be set on volumes or volume groups to enforce maximum allowable throughput. Whenever throughput exceeds the bandwidth limit, throttling occurs. If set, the bandwidth limit must be between 1 MB/s and 512 GB/s. By default, the QoS bandwidth limit is unlimited.

The bandwidth limit of a volume group represents the aggregate bandwidth for all the volumes in the volume group.

- **QoS IOPS Limit**

The IOPS limit can be set on volumes or volume groups to enforce maximum I/O operations processed per second. Whenever the number of I/O operations per second exceeds the IOPS limit, throttling occurs. If set, the IOPS limit must be between 100 and 100M. By default, the QoS IOPS limit is unlimited.

The IOPS limit of a volume group represents the aggregate IOPS for all the volumes in the volume group.

The QoS limits are not enforced on volumes or volume groups that do not have the bandwidth limit or the IOPS limit set.

## Creating a volume

Create a single volume or multiple volumes at one time.

To create a volume:

1. Select **Storage > Volumes**.
2. In the Volumes panel, click the menu icon and select **Create...**. The Create Volume dialog box appears.
3. In the Container field, select the root location, pod, or volume group to where the volume will be created.
4. In the Name field, type the name of the new volume.
5. In the Provisioned Size field, specify the provisioned (virtual) size number and size unit. The volume size must be between one megabyte and four petabytes. The provisioned size is reported to hosts.
6. Optionally click **QoS Configuration (Optional)** to set quality of service (QoS) limits.
  - In the Bandwidth Limit field, set the maximum QoS bandwidth limit for the volume. Whenever throughput exceeds the bandwidth limit, throttling occurs. If set, bandwidth limit must be between 1 MB/s and 512 GB/s.
  - In the IOPS Limit field, set the maximum QoS IOPS limit for the volume. Whenever the number of I/O operations per second exceeds the IOPS limit, throttling occurs. If set, IOPS limit must be between 100 and 100M.
7. Click **Create**.

To create multiple volumes:

1. Select **Storage > Volumes**.
2. In the Volumes panel, click the menu icon and select **Create...**. The Create Volumes dialog box appears.
3. Click **Create Multiple...**. The Create Multiple Volumes dialog box appears.
4. Complete the following fields:
  - In the Container field, select the root location, pod, or volume group to where the volumes will be created.
  - **Name:** Specify the template used to create the volume names. Volume names cannot consist of all numeric values.  
Place the hash (#) symbol where the numeric part of the volume name should appear. When Purity//FA creates the volume names, the hash symbol is replaced with the volume number, beginning with the start number specified.
  - **Provisioned Size:** Specify the provisioned (virtual) size of the volume. The volume size must be between one megabyte and four petabytes. The provisioned size is reported to hosts.
  - **Start Number:** Enter the volume number used to create the first volume name.
  - **Count:** Enter the number of volumes to create.
  - **Number of Digits:** Enter the minimum number of numeric digits of the volume number. If the number of digits is greater than the start number, the volume number begins with leading zeros.
  - **Bandwidth Limit:** Optionally set the maximum QoS bandwidth limit. The bandwidth limit applies to each of the volumes created in this set of volumes. Whenever throughput exceeds the bandwidth limit, throttling occurs. If set, bandwidth limit must be between 1 MB/s and 512 GB/s.
  - **IOPS Limit:** Optionally set the maximum QoS IOPS limit. The IOPS limit applies to each of the volumes created in this set of volumes. Whenever the number of I/O operations per second exceeds the IOPS limit, throttling occurs. If set, the IOPS limit must be between 100 and 100M.
5. Click **Create**.

## Creating a volume snapshot

Volume snapshots are created through the Storage > Volumes page. Protection group snapshots are created through the Storage > Protection Groups page.

To create a volume snapshot:

1. Select **Storage > Volumes**.
2. In the Volumes panel, click the volume to drill down to its details.

3. In the details panel, click the **Snapshots** sub-heading to display a list of volume and protection group snapshots that have been created for the selected volume.
4. In the Snapshots panel, click the menu icon and select **Create....** The Create Snapshot dialog box appears.
5. Optionally, specify a suffix to replace the unique number that Purity//FA creates for the volume snapshot. The suffix cannot consist of all numeric values.
6. Click **Create**.

## Renaming a volume snapshot suffix

To rename a volume snapshot suffix:

1. Select **Storage > Volumes**.
2. In the Snapshots panel, click the menu icon for to the snapshot you want to rename and select **Rename....** The Rename Snapshot dialog box appears.
3. In the Name field, enter the new name of the volume snapshot suffix.
4. Click **Rename**.

## Destroying a volume snapshot

You can only destroy a volume snapshot from its originating container. For example, to destroy a volume snapshot that was created on the root of the array, go to **Storage > Volumes**. Likewise, to destroy a volume snapshot that was created as part of a protection group snapshot, go to **Storage > Protection Group** to destroy the protection group snapshot. To destroy a volume snapshot that was created on the root of the array:

1. Select **Storage > Volumes**.
2. In the Snapshots panel, click the menu icon for to the snapshot you want to destroy and select **Destroy...** The Destroy Snapshot dialog box appears.
3. Click **Destroy**. The destroyed snapshot appears in the Destroyed Snapshots panel and begins its 24-hour eradication pending period.

During the eradication pending period, you can recover the volume snapshot to bring it back to its previous state, or manually eradicate the destroyed volume snapshot to reclaim physical storage space occupied by the snapshot.

When the 24-hour eradication pending period has lapsed, Purity//FA starts reclaiming the physical storage occupied by the volume snapshot.

Once reclamation starts, either because you have manually eradicated the destroyed volume snapshot, or because the eradication pending period has lapsed, the destroyed volume snapshot can no longer be recovered.

## Recovering a volume snapshot

To recover a destroyed volume snapshot:

1. Select **Storage > Volumes**.
2. In the Destroyed Snapshots panel, click the Recover Snapshot icon for to the snapshot you want to recover. The Recover Snapshot dialog box appears.
3. Click **Recover**. The snapshot is recovered to the volume from which it was destroyed.

## Eradicating a volume snapshot

During the eradication pending period, you can manually eradicate the destroyed volume snapshot to reclaim physical storage space occupied by the destroyed snapshot.

Once reclamation starts, the destroyed volume snapshot can no longer be recovered.

To eradicate a destroyed volume snapshot:

1. Select **Storage > Volumes**.
2. In the Destroyed Snapshots panel, click the Eradicate Snapshot icon for to the snapshot you want to permanently eradicate. The Eradicate Snapshot dialog box appears.
3. Click **Eradicate**.

## Restoring a volume from a volume snapshot

Restoring a volume from a snapshot overwrites the contents of the volume with data from the snapshot.

To restore a volume from a volume snapshot:

1. Select **Storage > Volumes**.
2. In the Snapshots panel, click the menu icon for to the snapshot you want to restore and select **Restore Volume**.... The Restore Volume from Snapshot dialog box appears.
3. Click **Restore**. Optionally view the volume **Details** to verify that the created date of the overwritten volume is set to the snapshot creation date, indicating that the volume snapshot has been successfully restored.

## Copying a volume snapshot

Copy a volume snapshot to create a new volume or overwrite an existing one.

To copy a volume snapshot:

1. Select **Storage > Volumes**.
2. In the Snapshots panel, click the menu icon for to the snapshot you want to copy and select **Create Volume**.... The Create Volume from Snapshot dialog box appears.
3. In the Container field, specify the root location, pod, or volume group to where the new volume will be created. The forward slash (/) represents the root location of the array.
4. In the Volume Name field, type the name of the new or existing volume.
5. To overwrite an existing volume, click the Overwrite toggle button to enable (blue) the overwrite feature.

6. Click **Copy**. Optionally click the **Details** sub-heading and verify that the created date of the new or overwritten volume is set to the snapshot created date.

## Moving a volume

Volumes can be moved into, out of, and between pods and volume groups. You cannot move volumes into or out of stretched pods. To move volumes into or out of a stretched pod, unstretch the pod before you move the volumes.

To move a volume:

1. Select **Storage > Volumes**.
2. In the Volumes panel, click the menu icon and select **Move....** The Move Volumes dialog box appears.
3. In the Existing Volumes column, select the volumes you want to move. All of the selected volumes will be moved to the same destination.
4. In the Container field, specify the root location, pod, or volume group to where the volumes will be moved. The forward slash (/) represents the root location of the array.
5. Click **Move**.

## Renaming a volume

To rename a volume:

1. Select **Storage > Volumes**.
2. In the Volumes panel, click the rename icon for the volume you want to rename. The Rename Volume dialog box appears.
3. In the Name field, enter the new name of the volume.
4. Click **Rename**.

## Resizing a volume

Resizing a volume changes its provisioned (virtual) size.

To change the provisioned size a volume:

1. Select **Storage > Volumes**.
2. From the Volumes panel, click the volume name to drill down to its details.
3. Click the menu icon and select **Resize....** The Resize Volume dialog box appears.
4. In the Provisioned Size field, enter the new volume size. The volume size must be between one megabyte and four petabytes.
5. Click **Resize**.

## Configuring the maximum QoS bandwidth and IOPS limits of a volume

To configure the maximum QoS bandwidth and IOPS limits of a volume:

1. Select **Storage > Volumes**.
2. Select one of the following options to edit the bandwidth and IOPS limits:
  - In the Volumes panel, click the menu icon for the volume you want to set the QoS bandwidth and IOPS limits, and then select **Configure QoS....**
  - In the Volumes panel, click the volume to drill down to its details, and then click the QoS edit icon in the Details panel.
3. The Configure QoS dialog box appears.

In the Bandwidth Limit field, set the maximum QoS bandwidth limit for the volume. Whenever throughput exceeds the bandwidth limit, throttling occurs. If set, the bandwidth limit must be between 1 MB/s and 512 GB/s.

To give the volume unlimited throughput, clear the Bandwidth Limit field.

In the IOPS Limit field, set the maximum QoS IOPS limit for the volume. Whenever the number of I/O operations per second exceeds the IOPS limit, throttling occurs. If set, the bandwidth limit must be between 100 and 100M.

To give the volume unlimited IOPS, clear the IOPS Limit field.
5. Click **Save**.

## Destroying a volume

Destroying a volume will also destroy its snapshots.

You cannot destroy a volume if it is connected to hosts, either through private or shared connections. Before deleting a volume, disconnect all hosts and host groups from the volume.

To destroy a volume:

1. Select **Storage > Volumes**.
2. From the Volumes panel, click the garbage icon for the volume you want to destroy. The Destroy Volume dialog box appears.
3. Click **Destroy**.

The destroyed volume(s) appear in the Destroyed Volumes folder and begins its 24-hour eradication pending period.

During the eradication pending period, you can recover the volume to bring it and its snapshots back to their previous states, or manually eradicate the destroyed volume to reclaim physical storage space occupied by the volume snapshots.

When the 24-hour eradication pending period has lapsed, Purity//FA starts reclaiming the physical storage occupied by the volume snapshots.

Once reclamation starts, either because you have manually eradicated the destroyed volume, or because the eradication pending period has lapsed, the destroyed volume and its snapshots can no longer be recovered.

## Recovering a destroyed volume

To recover a destroyed volume:

1. Select **Storage > Volumes**.
2. From the Destroyed Volumes panel, click the recover icon for the volume you want to recover. The Recover Volume dialog box appears.
3. Click **Recover**. The recovered volume(s) and its snapshots appear in the list of volumes.

## Eradicating a destroyed volume

Eradicating a volume destroys the volume and its snapshots. During the eradication pending period, you can manually eradicate the destroyed volume to reclaim physical storage space occupied by the destroyed volume's snapshots.

Once reclamation starts, the destroyed volume and its snapshots can no longer be recovered.

To eradicate destroyed volumes:

1. Select **Storage > Volumes**.
2. From the Destroyed Volumes panel, click the eradicate (garbage) icon for the volume you want to eradicate. The Eradicate Volume dialog box appears.
3. Click **Eradicate**. The volume and its snapshots are completely eradicated from the array.

## Copying a volume

Copy a volume to create a new volume or overwrite an existing one.

To copy a volume:

1. Select **Storage > Volumes**.
2. In the Volumes panel, click the volume that you want to copy.
3. Click the menu icon and select **Copy Volume**. The Copy Volume dialog box appears.
4. In the Container field, specify the root location, pod, or volume group to where the new volume will be created. The forward slash (/) represents the root location of the array.
5. In the Name field, type the name of the new or existing volume.
6. To overwrite an existing volume, click the Overwrite toggle button to enable (blue) the overwrite feature.
7. Click **Copy**.

## Establishing private volume-host connections

Creating a volume-host connection establishes a private connection between the volume and host. There are two ways to establish private volume-host connections: 1) establish a private connection from a volume to a host, or 2) establish a private connection from a host to a volume.

To establish a private connection from a volume to a host:

1. Select **Storage > Hosts**.
2. In the Hosts panel, click the host to drill down to its details.
3. In the Connected Volumes panel, click the menu icon and select **Connect....** The Connect Volumes to Host dialog box appears.  
The volumes in the Existing Volumes column represent the volumes that are eligible to be connected to the host.
4. Click an existing volume in the left column to add it to the Selected Volumes column. If the volume does not exist, click **Create New Volume** to create a new volume and connect it to the host.
5. Click **Connect**.

To establish a private connection from a host to a volume:

1. Select **Storage > Volumes**.
2. In the Volumes panel, click the volume to drill down to its details.
3. In the Connected Hosts panel, click the menu icon and select **Connect....** The Connect Hosts dialog box appears.  
The hosts in the Available Hosts column represent the hosts that are eligible to be connected to the volume.
4. Click an existing volume in the left column to add it to the Selected Volumes column. If the volume does not exist, click **Create New Volume** to create a new volume and connect it to the host.
5. Optionally assign a LUN to the connection. If the field is left blank, Purity//FA automatically assigns the next available LUN to the connection.
6. Click **Connect**.

## Establishing shared volume-host group connections

Creating a shared volume-host group connection automatically establishes connections between the volume and all hosts affiliated with the host group. There are two ways to establish shared volume-host connections: 1) establish a shared connection from a volume to a host group, or 2) establish a shared connection from a host group to a volume.

To establish a shared connection from a volume to a host group:

1. Select **Storage > Hosts**.

2. In the Host Groups panel, click the host group to drill down to its details.
3. In the Connected Volumes panel, click the menu icon and select **Connect....** The Connect Shared Volumes to Host Group dialog box appears.  
The volumes in the Existing Volumes column represent the volumes that are eligible to be connected to the host group.
4. Click an existing volume in the left column to add it to the Selected Volumes column.
5. Click **Connect**.

To establish a shared connection from a host group to a volume:

1. Select **Storage > Volumes**.
2. In the Volumes panel, click the volume to drill down to its details.
3. In the Connected Host Groups panel, click the menu icon and select **Connect....** The Connect Host Groups dialog box appears.  
The host groups in the Available Host Groups column represent the host groups that are eligible to be connected to the volume.
4. Click an existing host group in the left column to add it to the Selected Host Groups column. If the host group does not exist, click **Create New Host Group** to create a new host group and connect it to the volume.
5. Click **Connect**.

## Breaking volume-host connections

Break a volume-host connection when there is no longer a need for the two to communicate.

Breaking a private volume-host connection causes the host to lose access to the volume. Other shared and private connections to the volume are unaffected.

There are two ways to break private volume-host connections: 1) disconnect a volume from its host, or 2) disconnect a host from the volume.

To disconnect a volume from its host:

1. Select **Storage > Hosts**.
2. In the Hosts panel, click the host name to drill down to its details.
3. In the Connected Volumes panel, click the disconnect volume (x) icon next to the volume you want to disconnect. The Disconnect Volume dialog box appears.
4. Click **Disconnect**.

To disconnect a host from a volume:

1. Select **Storage > Volumes**.
2. In the Volumes panel, click the volume to drill down to its details.

3. In the Connected Hosts panel, click the disconnect host (x) icon next to the host you want to disconnect. The Disconnect Host dialog box appears.
4. Click **Disconnect**.

## Breaking volume-host group connections

Break a volume-host group connection when there is no longer a need for the volume to communicate to the hosts within the host group.

Breaking a volume-host group connection breaks all connections between the volume and all hosts affiliated with the host group. Other shared and private connections to the volume are unaffected.

There are two ways to break shared volume-host group connections: 1) disconnect a volume from its host group, or 2) disconnect a host group from the volume.

To disconnect a volume from its host group:

1. Select **Storage > Hosts**.
2. In the Host Groups panel, click the host group name to drill down to its details.
3. In the Connected Volumes panel, click the disconnect volume (x) icon next to the volume you want to disconnect. The Disconnect Volume dialog box appears.
4. Click **Disconnect**.

To disconnect a host group from a volume:

1. Select **Storage > Volumes**.
2. In the Volumes panel, click the volume to drill down to its details.
3. In the Connected Host Groups panel, click the disconnect host group (x) icon next to the host group you want to disconnect. The Disconnect Host Group dialog box appears.
4. Click **Disconnect**.

## Downloading volume details

Downloading volume details generates a comma-separated value text file containing volume summary information.

To download volume details:

1. Select **Storage > Volumes**.  
In the Volumes panel, click the menu icon and select **Download CSV** to save the `volumes.csv` file to your local machine.

## Creating a volume group

Create a single volume group or multiple volume groups at one time.

To create a volume group:

1. Select **Storage > Volumes**.

2. In the Volume Groups panel, click the menu icon and select **Create...**. The Create Volume Group dialog box appears.
3. In the Name field, type the name of the new volume group.
4. Click **Create**.

To create multiple volume groups:

1. Select **Storage > Volumes**.
2. In the Volume Groups panel, click the menu icon and select **Create...**. The Create Volume Group dialog box appears.
3. Click **Create Multiple...**. The Create Multiple Volume Groups dialog box appears.
4. Complete the following fields:
  - **Name:** Specify the template used to create the volume group names. Volume group names cannot consist of all numeric values.  
Place the hash (#) symbol where the numeric part of the volume group name should appear. When Purity//FA creates the volume group names, the hash symbol is replaced with the volume group number, beginning with the start number specified.
  - **Start Number:** Enter the volume number used to create the first volume group name.
  - **Count:** Enter the number of volume groups to create.
  - **Number of Digits:** Enter the minimum number of numeric digits of the volume group number. If the number of digits is greater than the start number, the volume number begins with leading zeros.
5. Click **Create**.

## Configuring the maximum QoS bandwidth and IOPS limits of a volume group

To configure the maximum QoS bandwidth and IOPS limits of a volume group:

1. Select **Storage > Volumes**.
2. Select one of the following options to edit the bandwidth and IOPS limits:
  - In the Volume Groups panel, click the menu icon for the volume group you want to set the QoS bandwidth and IOPS limits, and then select **Configure QoS...**
  - In the Volumes Groups panel, click the volume group to drill down to its details, and then click the QoS edit icon in the Details panel.  
The Configure QoS dialog box appears.
3. In the Bandwidth Limit field, set the maximum QoS bandwidth limit for the volume group. Whenever throughput exceeds the bandwidth limit, throttling occurs. If set, the bandwidth limit must be between 1 MB/s and 512 GB/s.  
To give the volume group unlimited throughput, clear the Bandwidth Limit field.

4. In the IOPS Limit field, set the maximum QoS IOPS limit for the volume group. Whenever the number of I/O operations per second exceeds the IOPS limit, throttling occurs. If set, the IOPS limit must be between 100 and 100M.

To give the volume group unlimited IOPS, clear the IOPS Limit field.

5. Click **Save**.

## Renaming a volume group

To rename a volume group:

1. Select **Storage > Volumes**.
2. In the Volume Groups panel, click the rename icon for the volume group you want to rename. The Rename Volume Group dialog box appears.
3. In the Name field, enter the new name of the volume group.
4. Click **Rename**.

## Destroying a volume group

A volume group can only be destroyed if it is empty, so before destroying a volume group, ensure all volumes inside the volume group have been either moved out of the volume group or destroyed.

To destroy a volume group:

1. Select **Storage > Volumes**.
2. From the Volume Groups panel, click the garbage icon for the volume group you want to destroy. The Destroy Volume Group dialog box appears.
3. Click **Destroy**.

The destroyed volume group appears in the Destroyed Volume Groups folder and begins its 24-hour eradication pending period.

During the eradication pending period, you can recover the volume group or manually eradicate the destroyed volume group.

When the 24-hour eradication pending period has lapsed, Purity//FA eradicates the destroyed volume group. Once a volume group has been eradicated, it can no longer be recovered.

## Recovering a destroyed volume group

To recover a destroyed volume group:

1. Select **Storage > Volumes**.
2. From the Destroyed Volume Groups panel, click the recover icon for the volume group you want to recover. The Recover Volume Group dialog box appears.
3. Click **Recover**. The recovered volume group appears in the list of volume groups.

## Eradicating a destroyed volume group

Once a volume group has been eradicated, it can no longer be recovered.

To eradicate a destroyed volume group:

1. Select **Storage > Volumes**.
2. From the Destroyed Volume Groups panel, click the eradicate (garbage) icon for the volume group you want to eradicate. The Eradicate Volume dialog box appears.
3. Click **Eradicate**. The volume group is permanently eradicated and can no longer be recovered.

## Protection Groups

The Storage > Protection Groups page displays summary information for all protection groups on the array.

A protection group represents a collection of members (volumes, hosts, or host groups) on the FlashArray that are protected together through snapshots. The members within the protection group have common data protection requirements and the same snapshot, replication, and retention schedules.

Creating a protection group snapshot creates snapshots of the volumes within the protection group, which are then retained on the current array. Protection group snapshots can also be asynchronously replicated to other arrays and external storage systems, such as Azure Blob containers, NFS devices, and S3 buckets. When replicating, the array from where a snapshot is created is called the source array, while the array to where the snapshot is replicated is called the target.

The Storage > Protection Groups page displays a list of protection groups, protection group snapshots, and destroyed protection groups and protection group snapshots on the array.

A local protection group represents a protection group that has been created on the current array to generate and retain snapshots. In the Protection Groups and Snapshots panels, local protection groups are identified by the protection group name.

A remote protection group represents a protection group that has been created on another (remote) array and set the current array as one of its replication targets. In the Protection Groups and Snapshots panels, remote protection groups are identified by the remote array named, followed by a colon (:) and then protection group name. For example, a protection group with the name `c14-d1-w11:pgroup001` represents a protection group named `pgroup001` that has been created on array `c14-d1-w11`. Protection group `pgroup001` has added the current array as a target array.

In the following example, array `pure-001` has four protection groups. Three of the protection groups (`ESXi-Cluster01-vol003`, `PG001`, and `PG002`) have been created on the current array. A fourth protection group named `pgroup001` has been created on remote array `c14-d1-w11`. The Snapshots panel displays various protection group snapshots; some of the snapshots are taken from the current array, while others are taken from remote array `c14-d1-w11` and asynchronously replicated over to the current array.

The Destroyed Groups and Destroyed Snapshots panels display a list of destroyed protection groups and protection snapshots, respectively, that are in the eradication pending period

Dashboard | Storage | Analysis | Health | Settings | Help | Terms | Log Out

**Storage**

- [Array](#)
- [Hosts](#)
- [Volumes](#)
- [Protection Groups](#) ●
- [Pods](#)

**Snapshots**  
0.00

Protection Groups		Schemas		Targets	
Name	Size	Snapshots	Targets	Actions	
c14-d1-wft:pgroup001	0.00	Allowed	-	<input type="checkbox"/> <span style="color: red;">●</span> <span style="color: red;">●</span>	
ESXI-Cluster01-vol003	0.00	-	-	<input type="checkbox"/> <span style="color: red;">●</span>	
PG001	0.00	Allowed on 1 of 1 replication targets	-	<input type="checkbox"/> <span style="color: red;">●</span> <span style="color: red;">●</span>	
PG002	0.00	-	-	<input type="checkbox"/> <span style="color: red;">●</span> <span style="color: red;">●</span>	

**Snapshots**  
1-5 of 5

Name	Created	Started	Completed	Transferred	Progress	Snapshots	Actions
PG001.5	2017-10-26 12:37:30	-	-	-	-	0.00	<span style="color: red;">●</span> <span style="color: red;">●</span>
c14-d1-wft:pgroup001.backup01	2017-10-26 12:35:51	2017-10-26 12:35:51	2017-10-26 12:35:55	305.00 B	100%	0.00	<span style="color: red;">●</span> <span style="color: red;">●</span>
c14-d1-wft:pgroup001.2	2017-10-26 12:34:00	2017-10-26 12:34:00	2017-10-26 12:34:03	305.00 B	100%	0.00	<span style="color: red;">●</span> <span style="color: red;">●</span>
PG002.2	2017-10-26 12:29:30	-	-	-	-	0.00	<span style="color: red;">●</span> <span style="color: red;">●</span>
PG001.Backup001	2017-10-26 12:24:21	-	-	-	-	0.00	<span style="color: red;">●</span> <span style="color: red;">●</span>

**Destroyed Groups**  
1-2 of 2

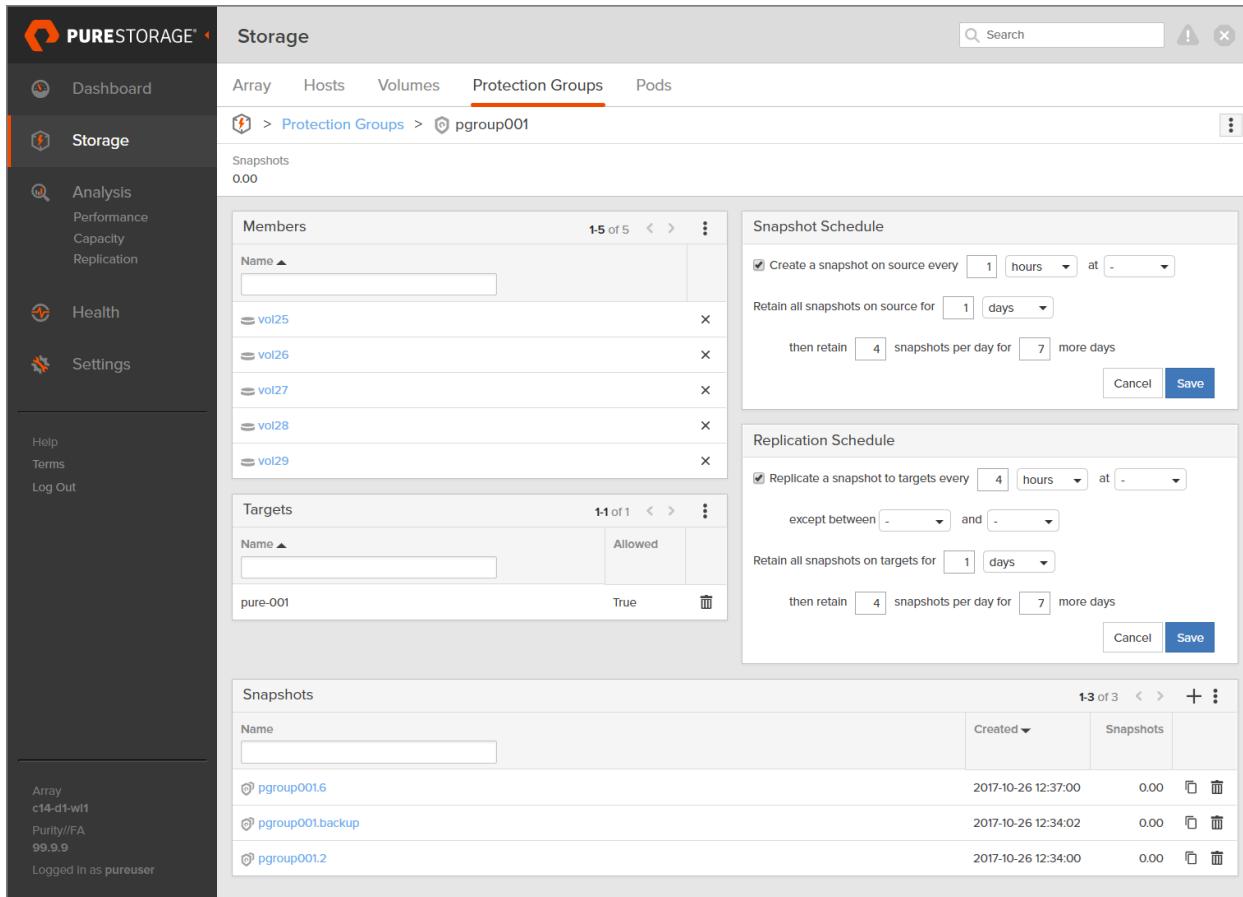
Name	Snapshots	Time Remaining	Actions
ESXI-Cluster01-vol008	0.00	23:51	<span style="color: red;">●</span> <span style="color: red;">●</span>
ESXI-GRP-Cluster02-H0002	0.00	23:51	<span style="color: red;">●</span> <span style="color: red;">●</span>

**Destroyed Snapshots**  
1-2 of 2

Name	Snapshots	Time Remaining	Actions
PG002.1	0.00	23:51	<span style="color: red;">●</span> <span style="color: red;">●</span>
PG003.1	0.00	23:53	<span style="color: red;">●</span> <span style="color: red;">●</span>

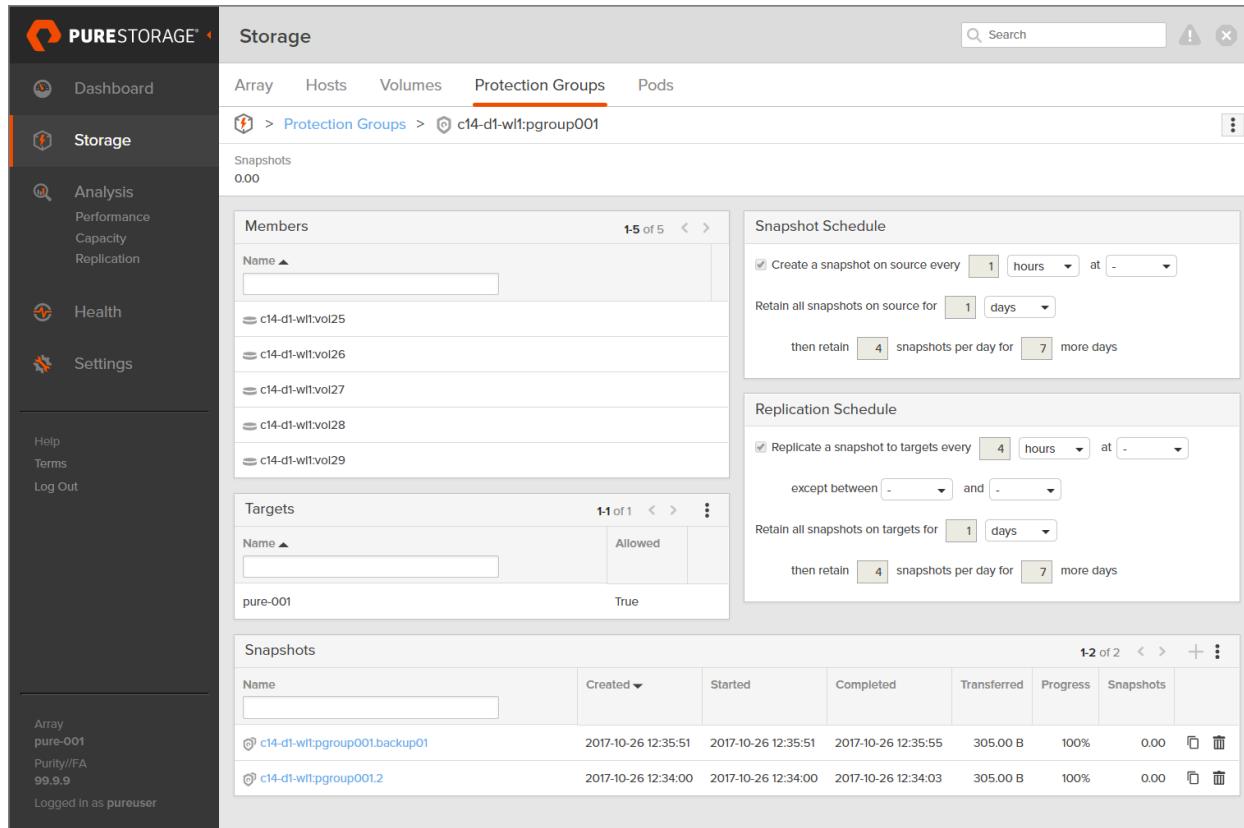
Click a protection group name to display a detailed view of the protection group.

Here is a view of protection groups from array **c14-d1-w11**. Protection group **pgroup001** was created on array **c14-d1-w11** and has five volume members and one target array named **pure001**. Three protection group snapshots have been created. The snapshot schedule has been set to create a protection group snapshot once every hour, while the replication schedule has been set to take a protection group snapshot every four hours and immediately replicate the snapshot to the specified target array (**pure-001**).



The screenshot shows the PureStorage Management UI interface. The left sidebar includes links for Dashboard, Storage (selected), Analysis, Health, and Settings, along with Help, Terms, and Log Out options. The bottom left also displays the Array name (c14-d1-w11) and the user (pureuser). The main content area is titled 'Storage' and shows tabs for Array, Hosts, Volumes, Protection Groups (selected), and Pods. Under 'Protection Groups', it shows the path 'Protection Groups > pgroup001'. Below this, it displays 'Snapshots' (0.00) and the 'Members' section, which lists five volumes: vol25, vol26, vol27, vol28, and vol29. To the right of the members, there are 'Snapshot Schedule' and 'Replication Schedule' configuration panels. The 'Snapshot Schedule' panel shows a schedule to create a snapshot once every hour and retain it for 4 days. The 'Replication Schedule' panel shows a schedule to replicate snapshots to the target array 'pure-001' every 4 hours. At the bottom, the 'Snapshots' section lists three existing snapshots: pgroup001.6 (Created: 2017-10-26 12:37:00), pgroup001.backup (Created: 2017-10-26 12:34:02), and pgroup001.2 (Created: 2017-10-26 12:34:00).

Here is the view of the same protection group, but from array **pure-001**. Since the protection group is created on array **c14-d1-w11**, the attributes of protection group **pgroup001** can only be changed from array **c14-d1-w11**.



Name	Created	Started	Completed	Transferred	Progress	Snapshots
c14-d1-w11:pgroup001.backup01	2017-10-26 12:35:51	2017-10-26 12:35:51	2017-10-26 12:35:55	305.00 B	100%	0.00
c14-d1-w11:pgroup001	2017-10-26 12:34:00	2017-10-26 12:34:00	2017-10-26 12:34:03	305.00 B	100%	0.00

## Members

The Members panel displays a list of all storage objects (volumes, hosts, or host groups) that have been added to the source array. Only members of the same object type can belong to a protection group. Hosts and host groups are not supported for replication to offload targets. Replication to offload targets only supports volumes; hosts and host groups are not supported.

If you are viewing member details for a target group, the member name is made up of the array name and the protection group name.

If you added volumes to the source array, Purity//FA generates snapshots of those specific volumes. If you added hosts or host groups, Purity//FA generates snapshots of the volumes within those hosts or host groups. If the same volume appears in multiple hosts or host groups, only one copy of the volume is kept. Note: Volumes, hosts, and host groups are managed through the Storage tab.

## Targets

The Targets panel lists the target arrays and offload targets that have been added to the source array. You only need to add targets if you plan to asynchronously replicate snapshots to another array or to an external storage system. The Allowed column indicates whether a target array has allowed (true) or disallowed (false) asynchronous replication. By default, targets arrays allow protection group snapshots to be asynchronously replicated to it from the source array.

## Source Arrays

The Source Arrays panel lists the source arrays of the protection group. The Source Arrays panel only appears if the protection group is in a pod on a remote array, and the protection group has added the current array as a target for asynchronous replication.

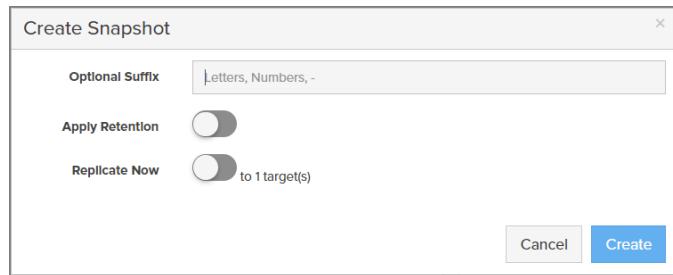
If the protection group is in a stretched pod, both arrays of the stretched pod should be connected to the target array for high availability and therefore be listed in the Source Arrays panel. If only one of the arrays is connected to the target array, Purity//FA generates an alert notifying users of this misconfiguration.

## Protection Group Snapshots

The Protection Group Snapshots panel displays a list of all protection group snapshots, both scheduled and on-demand, that have been taken and retained on the source array or taken on another array and asynchronously replicated to this array. The list includes the snapshot creation date and time and the physical storage space occupied by snapshot data. If the snapshot was replicated to this array, the list also displays the replication start and end times, amount of data transferred, and replication progress. The data transferred amount is calculated as the size difference between the current and previous snapshots after data reduction.

### On-Demand Snapshots

On-demand snapshots represent single snapshots that are manually generated and retained on the source array at any point in time. When you generate an on-demand snapshot, you can also add a suffix to the snapshot name, apply the scheduled retention policy to the on-demand snapshot, and asynchronously replicate the on-demand snapshot to the targets.



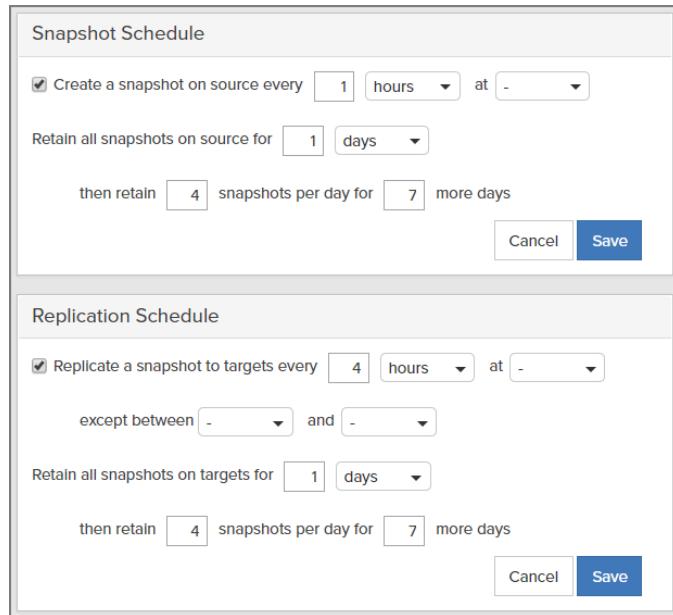
The "Optional Suffix" option allows you to add a unique suffix to the on-demand snapshot name. The suffix name, which can include letters, numbers and dashes (-), replaces the protection group snapshot number in the protection group snapshot name.

Select the "Apply Retention" option to apply the scheduled snapshot retention policy to the on-demand snapshot. If you do not enable "Apply Retention", the on-demand snapshot is saved until you manually destroy it.

Select the "Replicate Now" option to replicate the snapshot to the targets. The snapshot is retained on both the source and targets. If you also enable "Apply Retention", the snapshots inherit the retention policies set in the respective snapshot and retention schedules.

## Snapshot and Replication Schedules

The Snapshot Schedule and Replication Schedule screens display the scheduling details for the protection group.



Each protection group includes two schedules:

- Snapshot Schedule
- Replication Schedule

The snapshot schedule is independent of the replication schedule, meaning that you can enable one schedule without enabling the other. You can also enable or disable both schedules.

## Snapshot Schedule

The snapshot schedule displays the snapshot and retention schedule.

Configure the snapshot schedule to determine how often Purity//FA should generate protection group snapshots and how long Purity//FA should retain the generated snapshots.

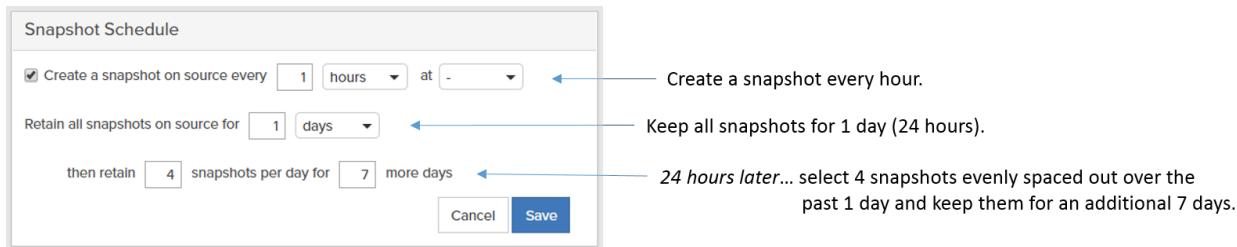
For example, a new protection group snapshot schedule may be set to:

- Create a snapshot every 1 hour.
- Retain all snapshots for 1 day, and then retain 4 snapshots per day for 7 more days.

This means that Purity//FA generates a snapshot every hour and keeps each generated snapshot for 24 hours.

For example, a snapshot that is generated on Saturday at 1:00 pm is kept until Sunday 1:00 pm.

**Figure 6.3: Snapshot Schedule**



After the one-day retention period, Purity//FA keeps four of the snapshots for an additional seven days. To determine which four snapshots are retained per day, Purity//FA takes all of the snapshots generated in the past day and selects the four snapshots that are most evenly spaced out throughout the day. As the seven-day period for each snapshot elapses, the snapshot is eradicated.

If the retention schedule is configured to retain one snapshot per day, Purity//FA will retain the very first snapshot taken after the snapshot schedule is enabled, and then retain the next snapshot taken approximately 24 hours thereafter, and so on.

Once you enable the snapshot schedule, Purity//FA immediately starts the snapshot process.

## Replication Schedule

The replication schedule section displays the asynchronous replication and retention schedules.

Configure the replication schedule to determine how often Purity//FA should replicate the protection group snapshots to the targets and how long Purity//FA should retain the replicated snapshots. You can configure a blackout period to specify when replication should not occur.

For example, a new protection group replication schedule may be set to the following:

- Replicate the snapshot every 4 hours, except between 8:00 am and 5:00 pm.
- Retain all replicated snapshots for 1 day, and then retain 4 snapshots per day for 7 more days.

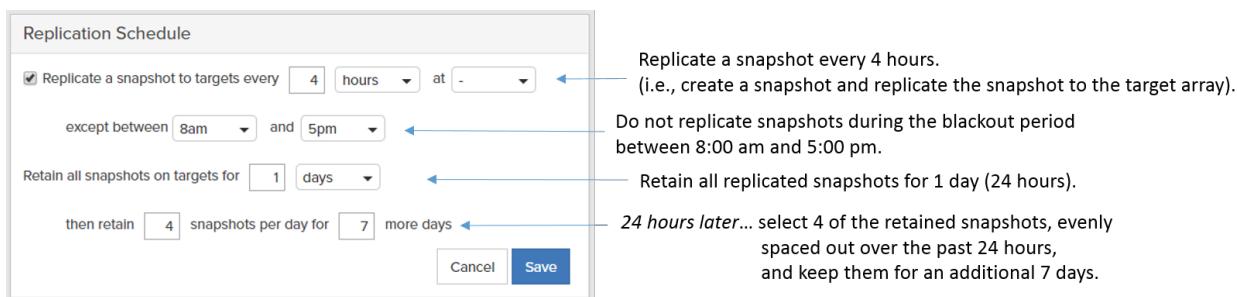
This means that Purity//FA generates a snapshot on the source array every four hours and immediately replicates each snapshot to the targets. Purity//FA retains each replicated snapshot for one day (24 hours).

For example, a snapshot that is generated on the source array and replicated to the targets on Friday 2:00 am is kept until Saturday 2:00 am.

The asynchronous replication process stops during the blackout period between 8:00 am and 5:00 pm. The start of a blackout period will not impact any snapshot replication sessions that are already in progress. Instead, Purity//FA will wait until the in-progress snapshot replication is complete before it observes the blackout period.

Blackout periods only apply to scheduled asynchronous replications. Asynchronous replications generated by on-demand snapshots (via Protection > Snapshots > Create Snapshot > Replication Now) do not observe the blackout period.

**Figure 6.4: Replication Schedule**



After the one-day retention period, Purity//FA keeps four of the replicated snapshots for an additional seven days. The other replicated snapshots are eradicated. To determine which four replicated snapshots are retained per day, Purity//FA takes all of the replicated snapshots generated in the past day and selects the four that are most evenly spaced out throughout the day. Purity//FA destroys each replicated snapshot as its seven-day period elapses.

If the retention schedule is configured to retain one replicated snapshot per day, Purity//FA will retain the very first snapshot taken after the replication schedule is enabled, and then retain the next snapshot taken approximately 24 hours thereafter, and so on.

Once you enable the replication schedule, Purity//FA immediately starts the asynchronous replication process, with the following exceptions:

- If you are enabling the replication schedule during the blackout period, Purity//FA waits for the blackout period to end before it begins the replication process.
- If you are enabling the replication schedule and the 'at' time is specified, Purity//FA starts the replication process at the specified 'at' time.

## Protection Group Configuration

Configure the snapshot schedule to determine how often Purity//FA should generate protection group snapshots and how long Purity//FA should retain the generated snapshots.

Configure the replication schedule to determine how often Purity//FA should asynchronously replicate the protection group snapshots to the target arrays and how long Purity//FA should retain the replicated snapshots.

Since the snapshot schedule is independent of the replication schedule, you can configure either or both schedules.

### Snapshot Schedule Configuration

To configure the snapshot schedule, create and configure the protection group and set its snapshot and retention schedule.

#### Figure 6.5: Snapshot Schedule Configuration

##### Snapshot Schedule Configuration

1. **Configure** the protection groups.
  - a. Create the protection group.<sup>\*</sup>
  - b. \* Add members to the protection group.
2. **Schedule** the snapshots and set the retention policy.
  - a. \* Set the snapshot schedule to specify how often you want to generate the snapshots.
  - b. \* Set the retention policy to specify how long you want to keep the generated snapshots.
  - c. Enable the snapshot schedule.

\* You can perform these steps in any order after you create the protection group and before you enable the schedule.

### Create and Configure the Protection Group

Protection groups are created and configured on the source array. After you create the protection group, you must add members (volumes, hosts, or host groups) and set the snapshot schedule. You can perform these steps in any order before you enable the schedule.

You can only add members of the same object type. For example, you cannot add hosts or host groups to a protection group that contains volumes. If you add hosts or host groups, it is the volumes within those hosts or host groups that are protected. The members within the protection group have common data protection requirements and the same snapshot, replication, and retention schedules.

### Set the Snapshot and Retention Schedule

Set the snapshot schedule to specify how often Purity//FA should generate protection group snapshots and how long Purity//FA should retain the generated snapshots. If the snapshot frequency is set to one or more days, optionally specify the preferred time of day for the snapshot to occur.

After you have added members and set the snapshot schedule, enable the schedule to start the snapshot and retention process. You can enable and disable the schedule at any time to manually start and stop, respectively, the process.

## Replication Schedule Configuration

To configure the replication schedule, connect the source and targets, configure the protection group, and set its replication and retention schedule.

**Figure 6.6: Replication Schedule Configuration**

### Replication Schedule Configuration

**1. Connect** the source and targets.

For asynchronous replication to another array:

- a. Log in to the target array and generate a connection key.
- b. Log in to the source array and use the connection key to connect the current array to the target array.

For replication to an offload target:

- a. Connect the current array to the offload target.

**2. Configure** the protection groups.

- a. Create the protection group.
- b. \* Add members to the protection group.
- c. \* For asynchronous replication, add target arrays to the protection group.

For replication to an offload target, add offload targets to the protection group.

**3. Schedule** the replication and set the retention policy.

- a. \* Set the replication schedule to specify how often you want to generate and replicate the snapshots.
- b. \* Set the retention policy to specify how long you want to keep the replicas on the targets.
- c. Enable the schedule.

\* You can perform these steps in any order after you create the protection group and before you enable the schedule.

## Connect the Source and Targets

The first step in the configuration process is to establish connections between the source array and its target, whether it be another array for asynchronous replication or an external storage system for replication to an offload target.

To connect the source array to a target array for asynchronous replication, log in to the target array to obtain a connection key, and then log in to the source array and use the connection key to connect to the target array. These steps are performed through the **Storage > Array > Connected Arrays** panel. Once a source and target array are connected, optionally configure network bandwidth throttling on the source array to set maximum threshold values for outbound traffic. Network bandwidth throttling is configured through the **Storage > Array > Connected Arrays** panel.

To connect the source array to an external storage system, such as an Azure Blob container, NFS device, or S3 bucket, for replication to an offload target, add the offload target to the array through the **Storage > Array > Offload Targets** panel.

## Create and Configure the Protection Group

Protection groups are created and configured on the source array. After you create the protection group, you must add members (volumes, hosts, or host groups), targets, and set the replication schedule. The members within the protection group must have the same snapshot, replication, and retention schedules.

For asynchronous replication, add hosts, host groups, or volumes, respectively, as members to the protection group. Snapshot data will only be transferred to a target array that has allowed replication. If you add hosts or host groups, it is the volumes within those hosts or host groups that are protected. Only members of the same object type can belong to a protection group. For example, you cannot add hosts or host groups to a protection group that contains volumes.

For replication to an offload target, add volumes as members to the protection group. Hosts and host groups are not supported for replication to offload targets.

The protection group must include at least one target to where the replicated data is written.

For asynchronous replication, the target is another array. By default, target arrays allow protection group snapshots to be asynchronously replicated to it from the source array. Administrators on target arrays can allow and disallow asynchronous replication at any time. Allowing and disallowing replication on a target array will not impact the replication process between the source array and other target arrays. If you disallow asynchronous replication while a replication session is in progress, Purity//FA will wait until the session is complete and then stop any new replication sessions from being created.

For replication to an offload target, the target is an external storage system, such as an Azure Blob container, NFS device, or S3 bucket.

#### Set the Replication and Retention Schedule

Set the replication schedule to specify how often Purity//FA should asynchronously replicate the protection group snapshots to the targets, and how long Purity//FA should retain the replicated snapshots. You can configure a blackout period to specify when replication should not occur.

If the replication frequency is set to one or more days, optionally specify the preferred time of day for the replication to occur.

After you have added the targets and members and set the replication schedule, enable the schedule to start the replication process. You can enable and disable the schedule at any time to manually start and stop, respectively, the process.

### Creating a protection group

1. Log in to the source array.
2. Select **Storage > Protection Groups**.
3. In the Protection Groups panel, click the menu icon and select **Create...**. The Create Protection Group dialog box appears.
4. In the Container field, select the root location, pod, or volume group to where the protection group will be created.
5. In the Name field, type the name of the new protection group.
6. Click **Create**.

### Adding a member (volume, host, or host group) to a protection group

For asynchronous replication, you can either add volumes *directly* to a protection group or add volumes *indirectly* to a protection group via the hosts or host groups in which they belong. A protection group

can only include members of one type. A volume, host, or host group can belong to any number of protection groups.

For replication to an offload target, add volumes as members to the protection group. Hosts and host groups are not supported for replication to offload targets.

1. Log in to the source array.
2. Select **Storage > Protection Groups**.
3. In the Protection Groups panel, click the protection group name to drill down to its details.
4. In the Members panel, click the menu icon and select **Add...** The Add Members dialog box appears.
5. In the Available Members column, click the member you want to add. The member appears in the Selected Members column.
6. Click **Add** to confirm the addition of the selected member.

## Adding a target to a protection group

Before you add a target array or offload target to a protection group, verify that its respective target array or external storage system is connected to the array.

1. Log in to the source array.
2. Select **Storage > Protection Groups**.
3. In the Protection Groups panel, click the protection group name to drill down to its details.
4. In the Targets panel, click the menu icon and select **Add....** The Add Targets dialog box appears.
5. In the Available Targets column, click the target array or offload target you want to add. The array appears in the Selected Targets column.
6. Click **Add** to confirm the addition of the selected target.

## Configuring the snapshot and retention schedule for a protection group

1. Log in to the source array.
2. Select **Storage > Protection Groups**.
3. In the Protection Groups panel, click the protection group name to drill down to its details.
4. In the Snapshot Schedule panel, click the edit icon. The Edit Snapshot Schedule pop-up window appears.
5. In the Edit Snapshot Schedule panel, click the **Enable** toggle icon to enable (blue) the snapshot schedule.
6. Set the following snapshot and retention details:

- Set the frequency of the snapshot creation. If the snapshot frequency is set to one or more days, optionally set the 'at' time to specify the preferred hour of each day when Purity//FA creates the snapshot. For example, if the snapshot schedule is set to "Create a snapshot every 2 days at 6pm," Purity//FA creates the snapshots every 2 days at or around 6:00 pm. If the 'at' option is set to dash (-), Purity//FA chooses the time of day to create the snapshot.
  - Set the snapshot retention schedule to keep the specified number of snapshots for the specified length of time (as minutes, hours, or days) and then to keep the specified number of snapshots for the specified additional number of days.
7. Click **Save** to save the snapshot and retention schedule. If the snapshot schedule is enabled, the array automatically starts generating and retaining snapshots according to the configured schedule.

## Configuring the replication and retention schedule for a protection group

Configure the replication and retention schedule to perform asynchronous replication to a target array or to perform replication to an offload target.

1. Log in to the source array.
2. Select **Storage > Protection Groups**.
3. In the Protection Groups panel, click the protection group name to drill down to its details.
4. In the Replication Schedule panel, click the edit icon. The Edit Replication Schedule pop-up window appears.
5. In the Edit Replication Schedule panel, click the **Enable** toggle icon to enable (blue) the replication schedule.
6. Set the following replication and retention details:
  - Set the asynchronous replication frequency. If the replication frequency is set to one or more days, optionally set the 'at' time to specify the preferred hour of each day when Purity//FA replicates the snapshot. For example, if the replication schedule is set to "Replicate every 4 days at 6pm," Purity//FA replicates the snapshots every four days at or around 6:00 pm. If the 'at' option is set to dash (-), Purity//FA chooses the time of day to replicate the snapshot.
  - Set the blackout period, if any. The asynchronous replication process stops during the blackout period. When the blackout period starts, replication processes that are still in progress will not be interrupted. Instead, Purity//FA will wait until the in-progress snapshot replication is complete before it observes the blackout period.
  - Set the retention schedule to keep the specified number of replicated snapshots for the specified length of time (as minutes, hours, or days) and then to keep the specified number of snapshots for the specified additional number of days.
7. Click **Save** to save the replication and retention schedule. If the replication schedule is enabled, the array automatically starts replicating snapshots and retaining the replicated snapshots according to the configured schedule.

## Enabling the snapshot and replication schedules

1. Log in to the source array.
2. Select **Storage > Protection Groups**.
3. In the Protection Groups panel, click the protection group name to drill down to its details.
4. Select one or both of the following options:
  - To enable the snapshot and retention schedule, in the Snapshot Schedule panel, click the edit icon, and then click the **Enable** toggle icon to enable (blue) the schedule.
  - To enable the replication and retention schedule, in the Replication Schedule panel, click the edit icon, and then click the **Enable** toggle icon to enable (blue) the schedule.

## Generating an on-demand snapshot

1. Select **Storage > Protection Group**.
2. In the Protection Groups panel, click the protection group name to drill down to its details.
3. In the Protection Group Snapshots panel, click the **Create Snapshots** (plus) icon.  
The Create Snapshot pop-up window appears.
4. Optionally type a unique suffix for the on-demand snapshot. The suffix name can include letters, numbers and dashes (-).  
The Create Snapshot pop-up window appears.
5. Optionally enable the Apply Retention option to apply the current retention policies to this snapshot.  
Optionally enable the Replicate Now option to asynchronously replicate the snapshot to the protection group's targets.
6. Click **Create**.

## Disabling the snapshot and asynchronous replication schedules

1. Log in to the source array.
2. In the Protection Groups panel, click the protection group name to drill down to its details.
3. Select one or both of the following options:
  - To disable the snapshot and retention schedule, in the Snapshot Schedule panel, click the edit icon, and then click the **Enable** toggle icon to disable (gray) the schedule.
  - To disable the replication and retention schedule, in the Replication Schedule panel, click the edit icon, and then click the **Enable** toggle icon to disable (gray) the schedule.

## Copying a volume snapshot on the current array to create a new volume

1. Log in to the source or target array.

2. Select **Storage > Protection Groups**.
3. In the Protection Groups panel, click the protection group name to drill down to its details.
4. In the Snapshots panel, click the name of the protection group snapshot containing the volume you want to copy. The Volume Snapshots pop-up window appears.
5. Click the copy icon for the volume you want to copy. The Copy Snapshot pop-up window appears.
6. In the Container field, select the root location, pod, or volume group to where the volume will be created.
7. In the Name field, type the name of the new volume.
8. To overwrite an existing volume, enable (blue) the Overwrite toggle icon.
9. Click **Copy**.

## Renaming a protection group

1. Log in to the source array.
2. Select **Storage > Protection Groups**.
3. In the Protection Groups panel, click the protection group name to drill down to its details.
4. Click the menu icon and select **Rename...**. The Rename Protection Group dialog box appears.
5. In the Name field, type the new name of the protection group.
6. Click **Rename**.

## Destroying a protection group

1. You can only destroy protection groups from the source array.
2. Log in to the source array.
3. Select **Storage > Protection Groups**.
4. In the Protection Groups panel, click the protection group name to drill down to its details.
5. Click the menu icon and select **Destroy...**. The Destroy Protection Group dialog box appears.
6. Click **Destroy**. The destroyed protection group appears in the **Destroyed Groups** panel and begins its 24-hour eradication pending period.  
  
During the eradication pending period, you can recover the protection group to bring the group and its content back to its previous state, or manually eradicate the destroyed protection group to reclaim physical storage space occupied by the destroyed protection group snapshots.  
  
When the 24-hour eradication pending period has lapsed, Purity//FA starts reclaiming the physical storage occupied by the protection group snapshots.

Once reclamation starts, either because you have manually eradicated the destroyed protection group, or because the eradication pending period has lapsed, the destroyed protection group and its snapshot data can no longer be recovered.

## Recovering a destroyed protection group

Recovering a destroyed protection group brings the group and its content back to its previous state. You can recover a destroyed protection group during the eradication pending period. Once the eradication pending period has lapsed, the protection group and its contents are no longer recoverable.

1. Log in to the source array.
2. Select **Storage > Protection Groups**.
3. In the Destroyed Groups panel, click the menu icon and select **Recover....** The Recover Protection Groups dialog box appears.
4. In the Destroyed Protection Groups column, select the protection group you want to recover.
5. Click **Recover**. The recovered protection group appears in the Protection Groups panel.

## Eradicating a destroyed protection group

During the eradication pending period, you can manually eradicate the destroyed protection group to reclaim physical storage space occupied by the destroyed protection group snapshots.

Once reclamation starts, the destroyed protection group and its snapshot data can no longer be recovered.

1. Log in to the source array.
2. Select **Storage > Protection Groups**.
3. In the Destroyed Groups panel, click the menu icon and select **Eradicate....** The Eradicate Protection Groups dialog box appears.
4. In the Destroyed Protection Groups column, select the protection group you want to permanently eradicate.
5. Click **Eradicate**. Purity//FA immediately starts reclaiming the physical storage occupied by the protection group snapshots.

## Allowing protection group replication

To allow protection group replication:

1. Log in to the target array.
2. Select **Storage > Protection Groups**.
3. In the Protection Groups panel, click the menu icon and select **Allow....** The Allow Protection Groups dialog box appears.
4. In the Target Protection Groups column, click the target protection group you want to allow.

- 
5. Click **Allow**.

## Disallowing protection group replication

By default, target arrays allow protection group snapshots to be asynchronously replicated to it from the source array.

Allowing and disallowing protection group replication on a target array will not impact the replication process between the source array and other target arrays.

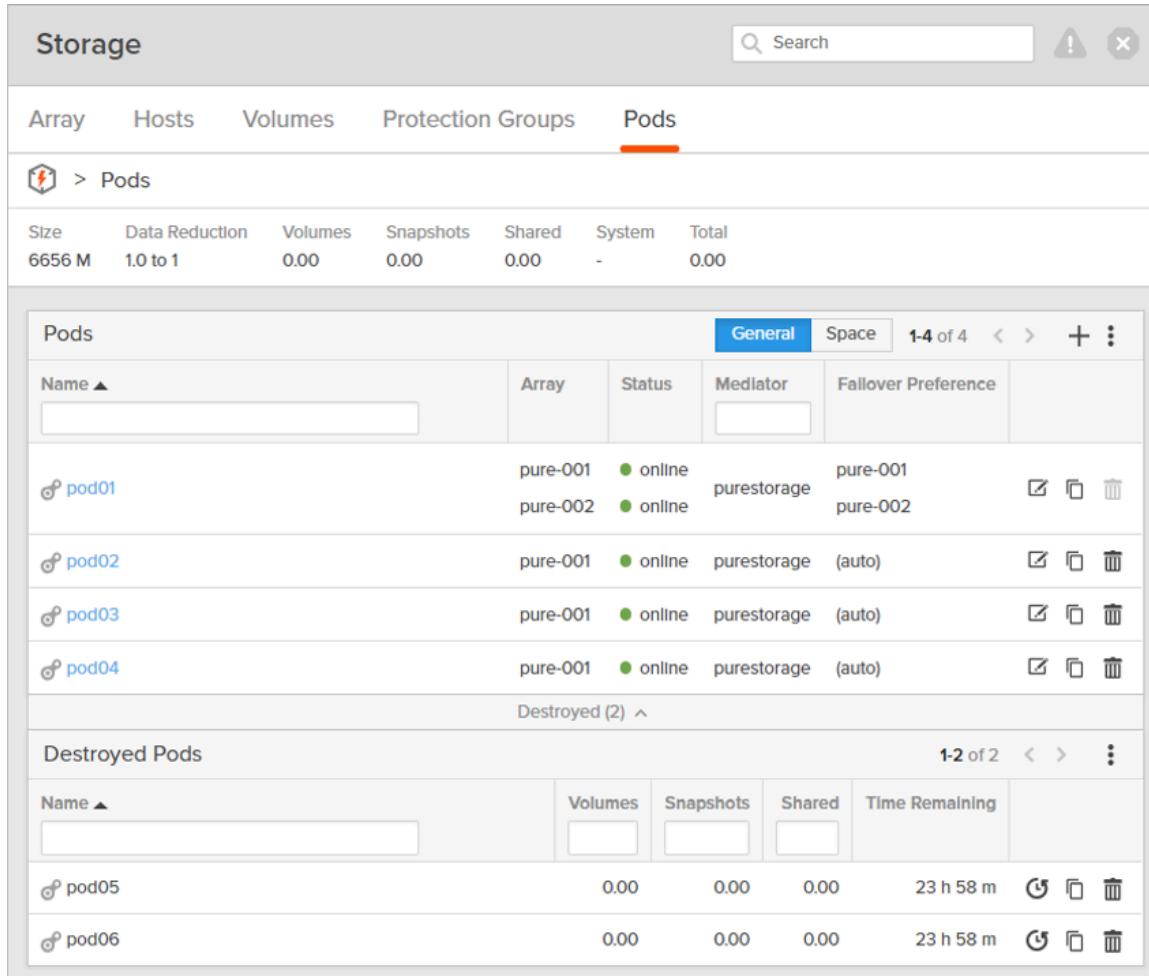
If you disallow protection group replication while a replication session is already in progress, Purity//FA will wait until the session is complete and then stop any new replication sessions from being created.

To disallow protection group replication:

1. Log in to the target array.
2. Select **Storage > Protection Groups**.
3. In the Protection Groups panel, click the menu icon and select **Disallow...**. The Disallow Protection Groups dialog box appears.
4. In the Target Protection Groups column, click the target protection group you want to disallow.
5. Click **Disallow**.

## Pods

The Storage > Pods page displays summary information for each pod on the array.



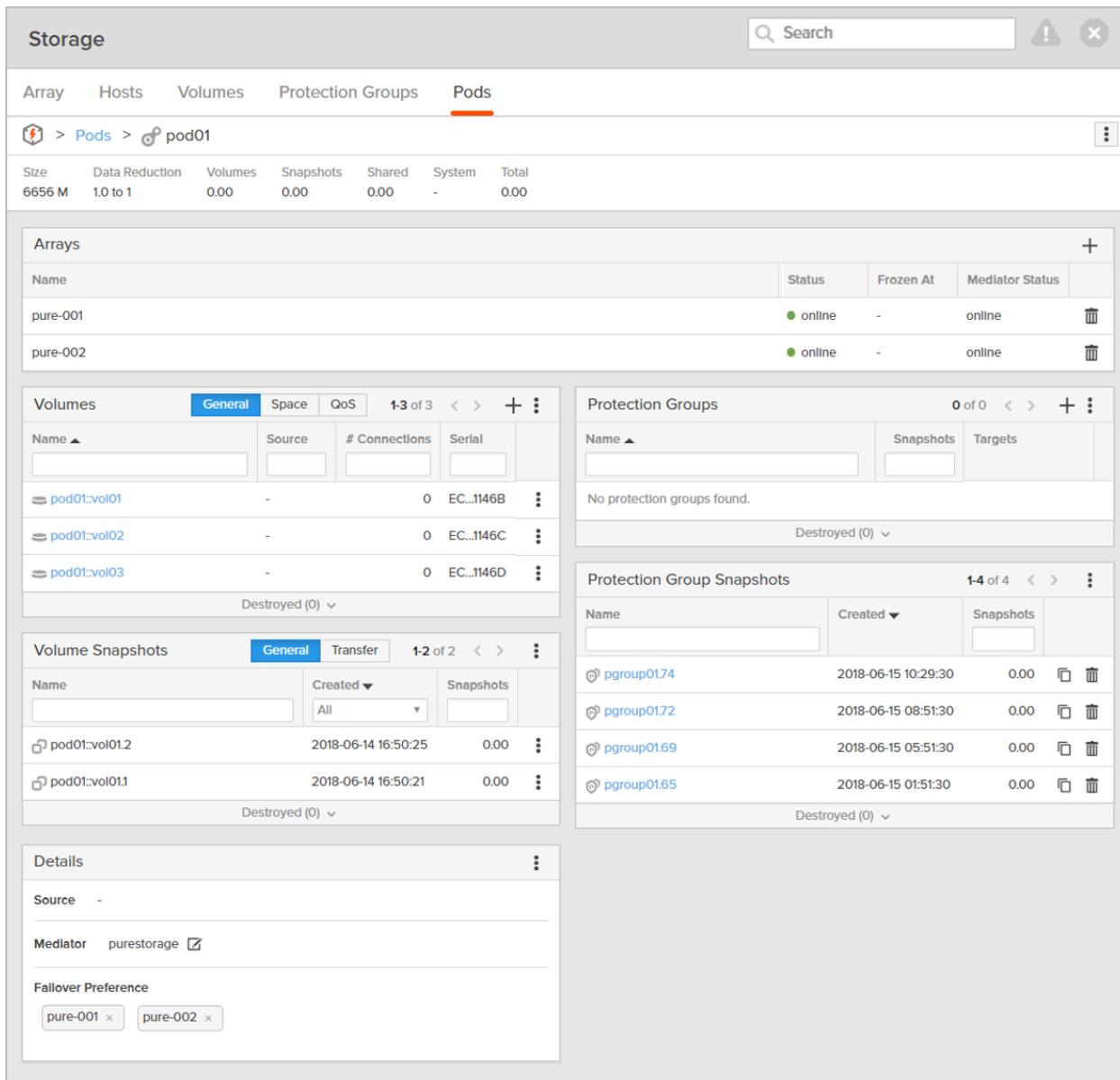
Name	Array	Status	Mediator	Fallover Preference
pod01	pure-001 pure-002	online online	purestorage	pure-001 pure-002
pod02	pure-001	online	purestorage	(auto)
pod03	pure-001	online	purestorage	(auto)
pod04	pure-001	online	purestorage	(auto)
Destroyed (2) ^				
pod05		0.00	0.00	0.00 23 h 58 m
pod06		0.00	0.00	0.00 23 h 58 m

A pod is a management container containing a group of volumes that can be stretched between two arrays. A pod serves as a consistency group that is created for truly active/active synchronous replication purposes. When a pod is stretched over two arrays, any time there is a failover between the two arrays, anything that was contained in that pod will be write-order consistent.

An array can have multiple pods, and each pod can be stretched and unstretched. Volumes can be moved into and out of pods. Pods can also contain protection groups with volume members. Pods cannot contain protection groups with host or host group members.

A pod provides a private namespace, so the names of volume and protection groups in pods will not conflict with any volumes or protection groups with the same name on the root of the array. The fully qualified name of a volume in a pod is **POD::VOLUME**, with double colons (::) separating the pod name and volume name. The fully qualified name of a protection group in a pod is **POD::PGROUP**, with double colons (::) separating the pod name and protection group name.

For example, a volume named **vol002** in a pod named **pod001** will be named **pod001::vol002**. A protection group named **pgroup001** in a pod named **pod001** will be named **pod001::pgroup001**.



The screenshot shows the PureStorage Management UI interface. The top navigation bar includes tabs for Array, Hosts, Volumes, Protection Groups, and Pods. The 'Pods' tab is currently selected, indicated by a red underline. Below the navigation is a breadcrumb trail: Array > Pods > pod001. The main content area is divided into several sections:

- Arrays:** Shows two arrays: pure-001 (online) and pure-002 (online).
- Volumes:** Shows three volumes under the pod001:: namespace: pod001::vol01, pod001::vol02, and pod001::vol03. All are EC..1146B and have 0 connections.
- Protection Groups:** Shows no protection groups found.
- Protection Group Snapshots:** Shows four snapshots under the pgroup001 namespace, all created on 2018-06-15 at different times. They are all 0.00 bytes and have 0 snapshots.
- Volume Snapshots:** Shows two snapshots under the pod001:: namespace, both created on 2018-06-14 at different times. They are both 0.00 bytes and have 0 snapshots.
- Details:** A sidebar section containing fields for Source (set to -), Mediator (purestorage checked), and Fallback Preference (pure-001 and pure-002 listed).

If a protection group in a pod is configured to asynchronously replicate data to a target array, the fully qualified name of the protection group on the target array is **POD:PGROUP**, with single colons (:) separating the pod name and protection group name. For example, if protection group **pod001::pgroup001** on source array **array001** asynchronously replicates data to target

array **array002**, the fully qualified name of the protection group on target array **array002** is **pod001:pgroup001**.

In addition to passive mediation and failover preference, Purity provides the pre-election behavior to further ensure a stretched pod remains online. With pre-election, an array within a stretched pod is chosen by Purity to keep a pod online when other failures occur in the environment.

The pre-election behavior elects one array of the stretched pod to remain online in the rare event that:

- The mediator is inaccessible on both arrays within the stretched pod, preventing the arrays from racing to the mediator to determine which one keeps the pod online.

...and then later...

- The arrays within the stretched pod become disconnected from each other.

When the mediator becomes inaccessible on both arrays, Purity pre-elects an array per pod to keep the pod online. Then, if the arrays lose contact with each other, the pre-elected array for that pod takes over to keep the pod online while its peer array takes the pod offline.

If either array reconnects to the mediator before they lose contact with each other, the pre-election result is cancelled. The array with access to the mediator will race to the mediator and keep the pod online if its peer array fails or the arrays become disconnected from each other.

The pre-election status appears in the form of a heart symbol in the Mediator status column of the **Storage > Pods > Arrays** panel; a gray heart means the array was pre-elected by Purity to keep the pod online, while an empty heart means the array was pre-elected by Purity to take the pod offline. If a heart does not appear, this means the array is connected to its peer array within the stretched pod and at least one array in the pod has access to the mediator.

One and only one array within each pod is pre-elected at a given point in time, so while a pre-elected array is keeping the pod online, the pod on its non-elected peer array remains offline during the communication failure.

Users cannot pre-elect arrays. Purity uses various factors, including the following ones (listed in order of precedence), to determine which array is pre-elected:

- If a pod has a failover preference set, then the array that is preferred will be pre-elected.
- If one of the arrays has no hosts connected to volumes in the pod, then the other array will be pre-elected.
- If neither of the above factors applies, one of the arrays is selected by Purity.

If the pre-elected array goes down while pre-election is in effect, the non-elected peer array will not bring the pod online.

If the non-elected array reconnects to the mediator while it is still disconnected from the pre-elected array, it is ignored and will still keep the pod offline. If the data in the non-elected pod must be accessed, clone it to create a point-in-time consistent copy of the pod and its contents, including its volumes and snapshot history. After the pod has been cloned, disconnect the hosts from the original volumes and reconnect the hosts to the volumes within the cloned pod.

If the arrays re-establish contact with each other but the mediator is still inaccessible, the array that was online throughout the outage starts replicating pod data to its peer array until the pod is once again in sync and both arrays serve I/O. One array will still be pre-elected (with the appropriate heart status still displayed) in case both arrays lose contact with each other again.

When the peer arrays re-establish contact with each other *and* can access the mediator, the array that was online throughout the outage starts replicating pod data to its peer array until the pod is once again in sync and both arrays serve I/O, at which time pod activity returns to normal.

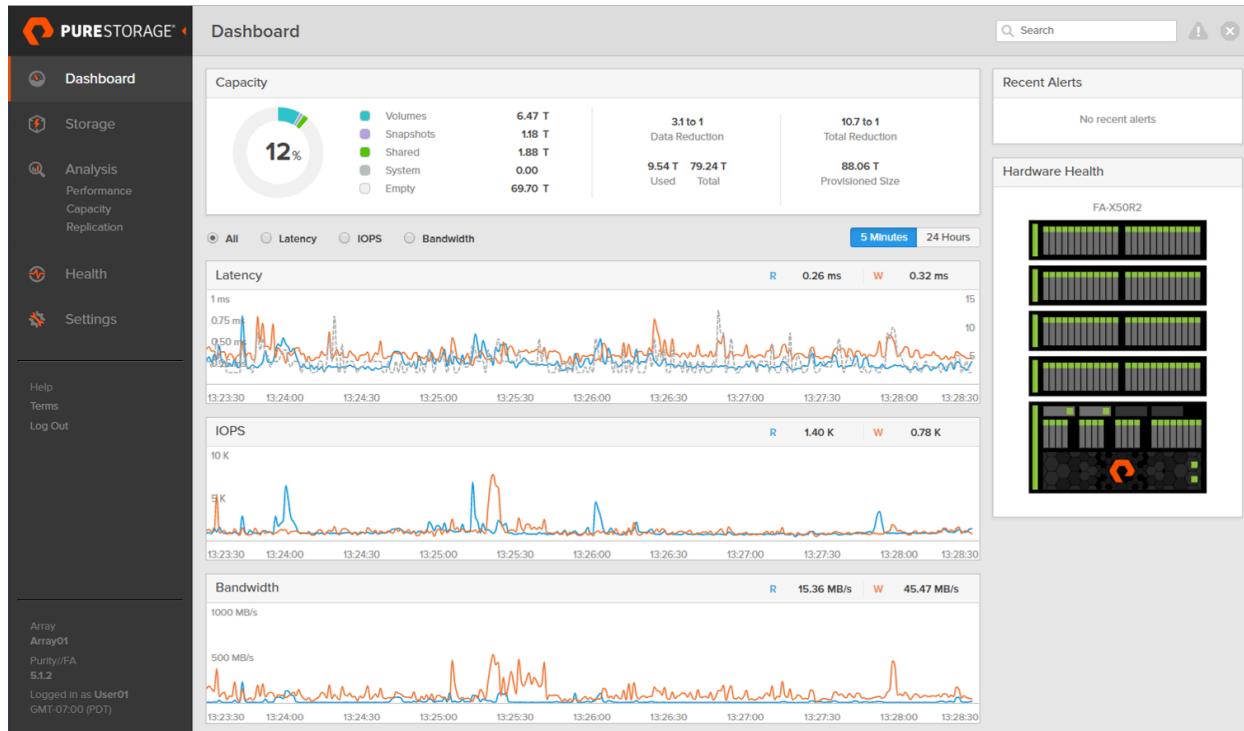
## Configuring Failover Preference

1. Log in to the target array.
2. Select **Storage > Pods**.
3. In the Pods panel, click the name of the pod to drill down to its details.
4. In the Details panel, click the menu icon and select **Add Arrays to Failover Preference....** The Add Arrays to Failover Preference dialog box appears.
5. Select the arrays you want to add for failover preference.
6. Click **Add**.

## Chapter 7. Analysis

The Analysis page displays historical array data, including storage capacity, consumption, and I/O performance trends across all volumes, and replication bandwidth activity across all source and target groups on the array.

**Figure 7.1:** Analysis



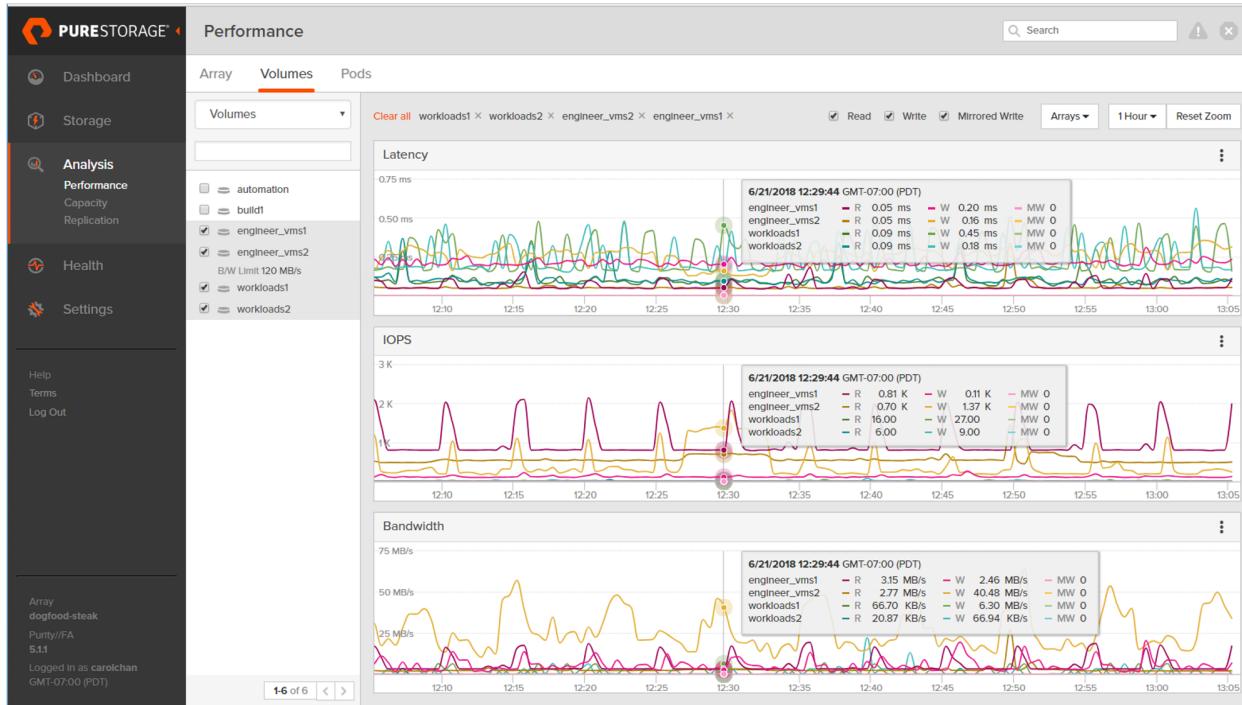
The Analysis page displays a series of rolling graphs consisting of real-time capacity, performance, and replication metrics; the incoming data appear along the right side of each graph as older numbers drop off the left side.

The curves in each graph are comprised of a series of individual data points. Hover over any part of a graph to display values for a specific point in time. The values that appear in the point-in-time pop-ups are rounded to two decimal places.

Different graphs display different metrics. Furthermore, specifying all or individual volumes, volume groups, pods, or protection groups determine the metrics that appear within a graph.

The FlashArray maintains a rolling one-year history of data. The granularity of the historical data increases with age; older data points are spaced further apart in time than more recent ones.

The following example displays the performance statistics for the five selected volumes on the array on **6/21/2018 at 12:29:44**.



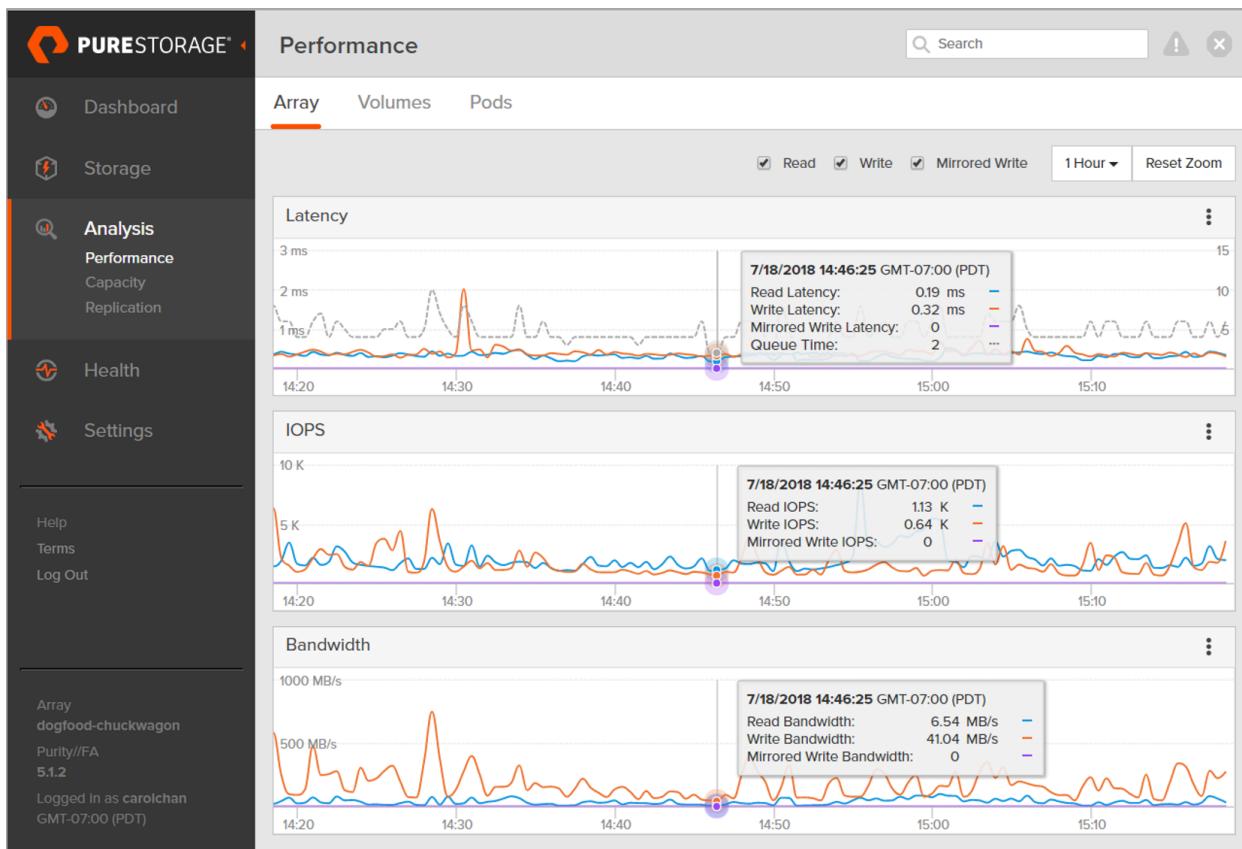
By default, the Analysis charts display data for the past 1 hour. To view historical data over a different time range, click the **1 Hour** range button and select the desired time range. To further zoom into a time range, from anywhere inside the chart, click and drag from the desired start time to the desired end time. Click Reset Zoom to zoom back to the time range specified.

The charts in the Analysis page are grouped into the following areas: Performance, Capacity, and Replication.

## Performance

The Performance charts display I/O performance metrics in real time.

**Figure 7.2: Analysis - Performance**



By default, Purity//FA displays the performance details for the entire array.

To analyze the performance details of specific volumes, click the Volumes sub-tab along the top of the Performance page, select Volumes from the drop-down list, and select the volumes you want to analyze. If a bandwidth limit has been set for the volume, the limit appears when the volume is selected. You can analyze up to five volumes at one time. Click **Clear All** to clear the volume selections.

If a volume is in a pod that is stretched across another array, optionally click the **Arrays** button to filter the performance details by array. If none of the arrays are selected (default), the chart displays the overall performance trends for each selected volume. If one or more arrays are selected, the chart displays the performance trends by array for each selected volume.

To analyze the performance details of volumes within specific volume groups, click the Volumes sub-tab along the top of the Performance page, select Volume Groups from the drop-down list, and select the volume groups you want to analyze. You can analyze up to five volume groups at one time. Click **Clear All** to clear the volume group selections.

To analyze the performance details of volumes within specific pods, click the Pods sub-tab along the top of the Performance page and select the pods you want to analyze. You can analyze up to five pods at one time. Click **Clear All** to clear the pod selections.

If a pod is stretched across another array, optionally click the **Arrays** button to filter the performance details by array. If none of the arrays are selected (default), the chart displays the overall performance trends of all volumes in the selected pod. If one or more arrays are selected, the chart displays the performance trends by array for each selected volume.

The Performance page includes Latency, IOPS, and Bandwidth charts. The point-in-time pop-ups in each of the performance charts display the following values:

### Latency

The Latency chart displays the average latency times for various operations.

- **SAN Time** - Average time, measured in milliseconds, required to transfer data between the initiator and the array. SAN times are only displayed in graphs of one I/O type, such as Read, Write, or Mirrored Write.
- **QoS Rate Limit Time**: Average time, measured in microseconds, that all I/O requests spend in queue as a result of bandwidth limits reached on one or more volumes. QoS rate limit times are only displayed in graphs of one I/O type, such as Read, Write, or Mirrored Write.
- **Queue Time** - Average time, measured in microseconds, that an I/O request spends in the array waiting to be served. The time is averaged across all I/Os of the selected types.
- **Read Latency (R)** - Average arrival-to-completion time, measured in milliseconds, for a read operation.
- **Write Latency (W)** - Average arrival-to-completion time, measured in milliseconds, for a write operation.
- **Mirrored Write Latency (MW)** - Average arrival-to-completion time, measured in milliseconds, for a write operation. Represents the sum of writes from hosts into the volume's pod and from remote arrays that synchronously replicate into the volume's pod.

### IOPS

The IOPS (Input/output Operations Per Second) chart displays I/O requests processed per second by the array. This metric counts requests per second, regardless of how much or how little data is transferred in each.

- **Read IOPS (R)** - Number of read requests processed per second.
- **Read Average IO Size (R IO Size)** - Average read I/O size per request processed. Calculated as  $(\text{read bandwidth}) / (\text{read IOPS})$ .
- **Write IOPS (W)** - Number of write requests processed per second.
- **Write Average IO Size (W IO Size)** - Average write I/O size per request processed. Calculated as  $(\text{write bandwidth}) / (\text{write IOPS})$ .

- **Mirrored Write IOPS (MW)** - Number of write requests processed per second. Represents the sum of writes from hosts into the volume's pod and from remote arrays that synchronously replicate into the volume's pod.
- **Mirrored Write Average IO Size (MW IO Size)** - Average mirrored write I/O size per request processed. Calculated as `(mirrored write bandwidth) / (mirrored write IOPS)`.

### Bandwidth

The Bandwidth chart displays the number of bytes transferred per second to and from all file systems. The data is counted in its expanded form rather than the reduced form stored in the array to truly reflect what is transferred over the storage network. Metadata bandwidth is not included in these numbers.

- **Read Bandwidth (R)** - Number of bytes read per second.
- **Write Bandwidth (W)** - Number of bytes written per second.
- **Mirrored Write Bandwidth (MW)** - Number of bytes written into the volume's pod per second. Represents the sum of writes from hosts into the volume's pod and from remote arrays that synchronously replicate into the volume's pod.

### Note about the Performance Charts

The Dashboard and Analysis pages display the same latency, IOPS, and bandwidth performance charts, but the information is presented differently between the two pages.

In the Dashboard page:

- The performance charts are updated once every 30 seconds.
- The performance charts display up to 30 day's worth of historical data.
- The Latency charts displays only internal latency times. SAN times are not included.

In the Analysis page:

- The performance charts are updated once every minute.
- The performance charts display up to one year's worth of historical data.
- The performance charts can be further dissected by I/O type.
- The Latency charts displays both internal latency times and SAN times.

### Exporting Array-Wide Performance Metrics

To export array-wide performance metrics:

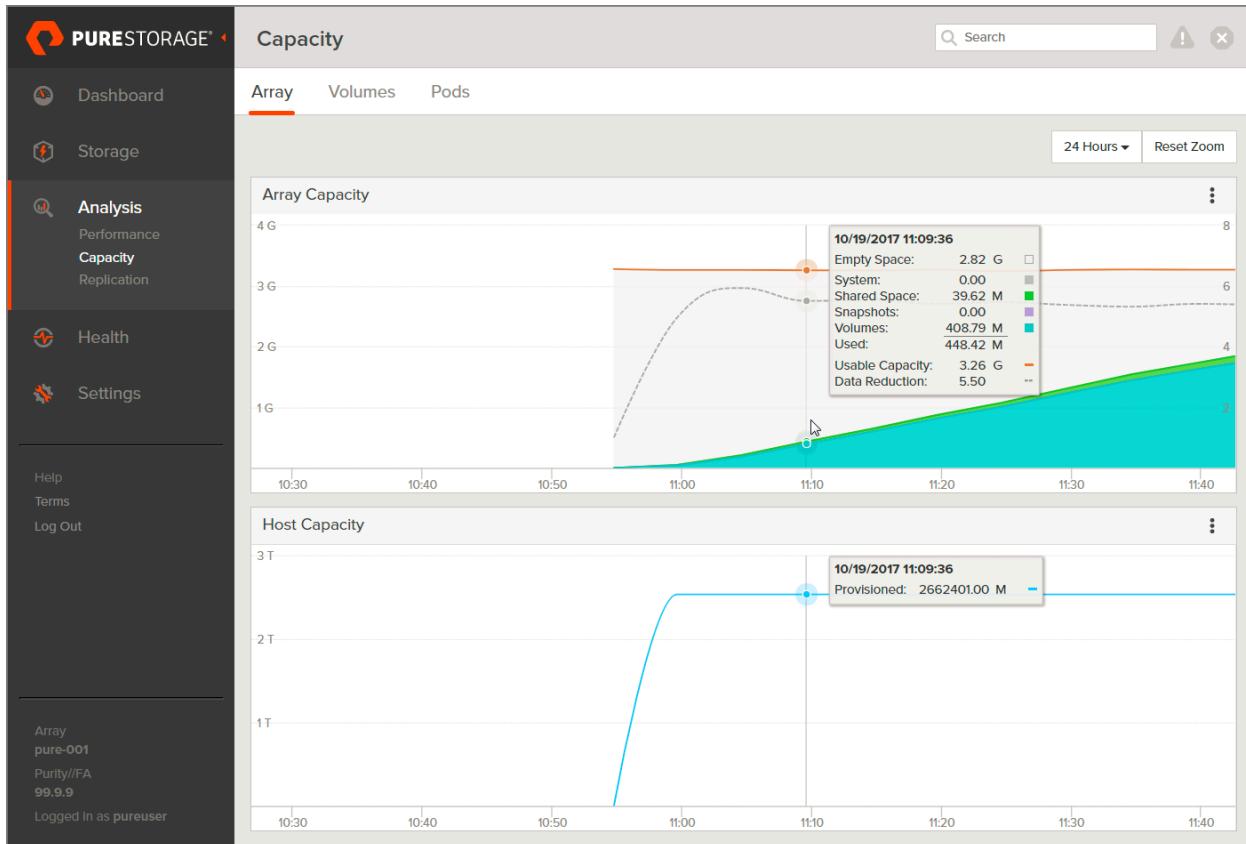
1. Select **Analysis > Performance > Array**.
2. Click the menu icon in the upper-right corner of the chart containing the performance metrics you want to export, and select one of the following options:
  - Select **Export to PNG** to export an image of the chart in PNG format to your local machine.

- Select **Export to CSV** to export the performance data to a comma-separated value (CSV) file on your local machine.

## Capacity

The Capacity charts display array-wide space consumption information, including physical storage capacity and the amount of storage occupied by data and metadata.

**Figure 7.3: Analysis - Capacity**



The Array Capacity chart displays the amount of usable physical storage on the array and the amount of storage occupied by data and metadata. The data point fluctuations represent changes in physical storage consumed by a volume.

For example, a volume may experience a spike in storage consumption when more data is being written to it or when other volumes with shared data are eradicated. Conversely, a volume may experience a dip in storage consumption from trimming or from an increased sharing of deduplicated data with other volumes.

By default, Purity//FA displays the capacity details for the entire array.

To analyze the capacity details of specific volumes, click the Volumes sub-tab along the top of the Capacity page, select Volumes from the drop-down list, and select the volumes you want to analyze. To analyze the performance details of volumes within specific volume groups, click the Volumes sub-tab along the top of the Capacity page, select Volume Groups from the drop-down list, and select the volume groups you want to analyze. You can analyze up to five volumes and volume groups at one time. Click **Clear All** to clear the volume or volume group selections.

To analyze the capacity details of volumes within specific pods, click the Pods sub-tab along the top of the Capacity page and select the pods you want to analyze. You can analyze up to five pods at one time. Click **Clear All** to clear the pod selections.

In the Capacity chart, the point-in-time pop-up displays the following metrics:

- **Empty Space**

Unused space available for allocation.

- **System**

Physical space occupied by internal array metadata.

- **Shared Space**

Physical space occupied by deduplicated data, meaning that the space is shared with other volumes and snapshots as a result of data deduplication.

- **Snapshots**

Physical space occupied by data unique to one or more snapshots.

- **Volumes**

Physical space occupied by volume data that is not shared between volumes, excluding array metadata and snapshots.

- **Used**

Physical storage space occupied by volume, snapshot, shared space, and system data.

- **Usable Capacity**

Total physical usable space on the array. Replacing a drive may result in a dip in usable capacity. This is intended behavior. RAID striping splits data across an array for redundancy purposes, spreading a write across multiple drives. A newly added drive cannot use its full capacity immediately but must stay in line with the available space on the other drives as writes are spread across them. As a result, usable capacity on the new drive may initially be reported as less than the amount expected because the array will not be able to write to the unallocatable space. Over time, usable capacity fluctuations will occur, but as data is written to the drive and spreads across the array, usable capacity will eventually return to expected levels.

- **Data Reduction**

Ratio of mapped sectors within a volume versus the amount of physical space the data occupies after data compression and deduplication. The data reduction ratio does not include thin provisioning savings.

The Host Capacity chart displays the provisioned size of all selected volumes. In the Host Capacity chart, the point-in-time pop-up displays the following metrics:

- **Provisioned**

Total provisioned size of all volumes. Represents storage capacity reported to hosts.

## Exporting Array-Wide Capacity Metrics

To export array-wide capacity metrics:

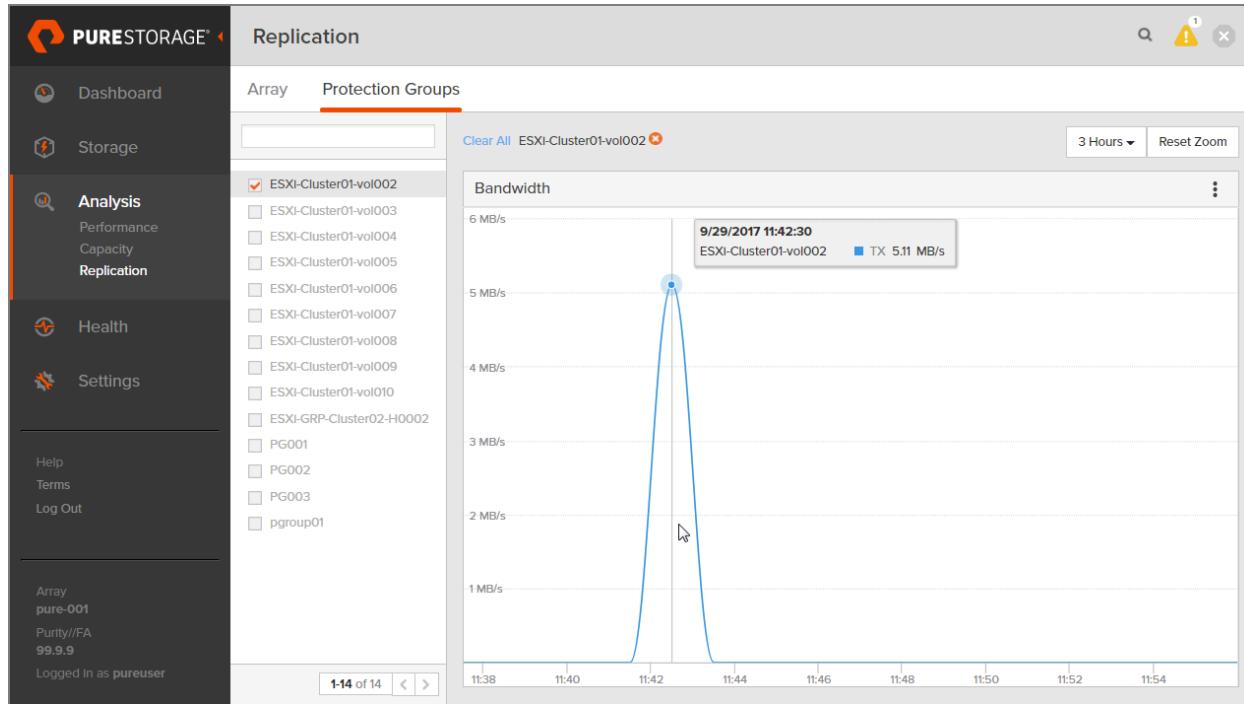
1. Select **Analysis > Capacity > Array**.
2. Click the menu icon in the upper-right corner of the chart containing the capacity metrics you want to export, and select one of the following options:
  - Select **Export to PNG** to export an image of the chart in PNG format to your local machine.
  - Select **Export to CSV** to export the capacity data to a comma-separated value (CSV) file on your local machine.

## Replication

The Replication charts display historical bandwidth information for asynchronous and synchronous replication activity on the array.

The Bandwidth chart (not to be confused with the performance Bandwidth chart) displays the number of bytes of replication snapshot data transferred over the storage network per second between this array and its source arrays, target arrays, and external storage systems (such as Azure Blob containers, NFS devices, and S3 buckets), at certain points in time.

**Figure 7.4: Analysis - Replication**



By default, Purity//FA displays bandwidth details for the entire array.

In the replication Bandwidth chart for the array, the point-in-time pop-up displays the following metrics:

- **Resync (RX + TX)**

Number of bytes of replication data transmitted and receive per second as the array actively gets the latest pod data so that it becomes fully synchronized with its peer arrays. This can be due to an initial pod stretch or due to an array coming back online after an extended offline event.

- **Sync (RX + TX)**

Number of bytes of synchronous replication data transmitted and received per second across all pods.

- **Async (RX + TX)**

Number of bytes of asynchronous replication snapshot data transmitted and received per second across all protection groups.

- **Total**

Total number bytes of replication data transmitted and received per second across all protection groups.

To analyze the details for a specific protection group, click the Protection Groups sub-tab along the top of the Replication page, and select the protection groups you want to analyze. You can select up to five protection groups at one time. The names of the selected protection groups appear at the top of the details pane. Click **Clear All** to clear the protection group selection.

In the replication Bandwidth chart for protection groups, the point-in-time pop-up displays the following metrics:

- **RX + TX**

Number of bytes of replication snapshot data transmitted and received per second across all protection groups.

- **RX**

Number of bytes of replication snapshot data received per second by the targets for the selected protection groups.

- **TX**

Number of bytes of replication snapshot data transmitted per second from the source array for the selected protection groups.

## Exporting Array-Wide Replication Metrics

To export array-wide replication metrics:

1. Select **Analysis > Replication > Array**.

2. Click the menu icon in the upper-right corner of the chart containing the replication metrics you want to export, and select one of the following options:

- Select **Export to PNG** to export an image of the chart in PNG format to your local machine.
- Select **Export to CSV** to export the replication data to a comma-separated value (CSV) file on your local machine.

## Chapter 8. Health

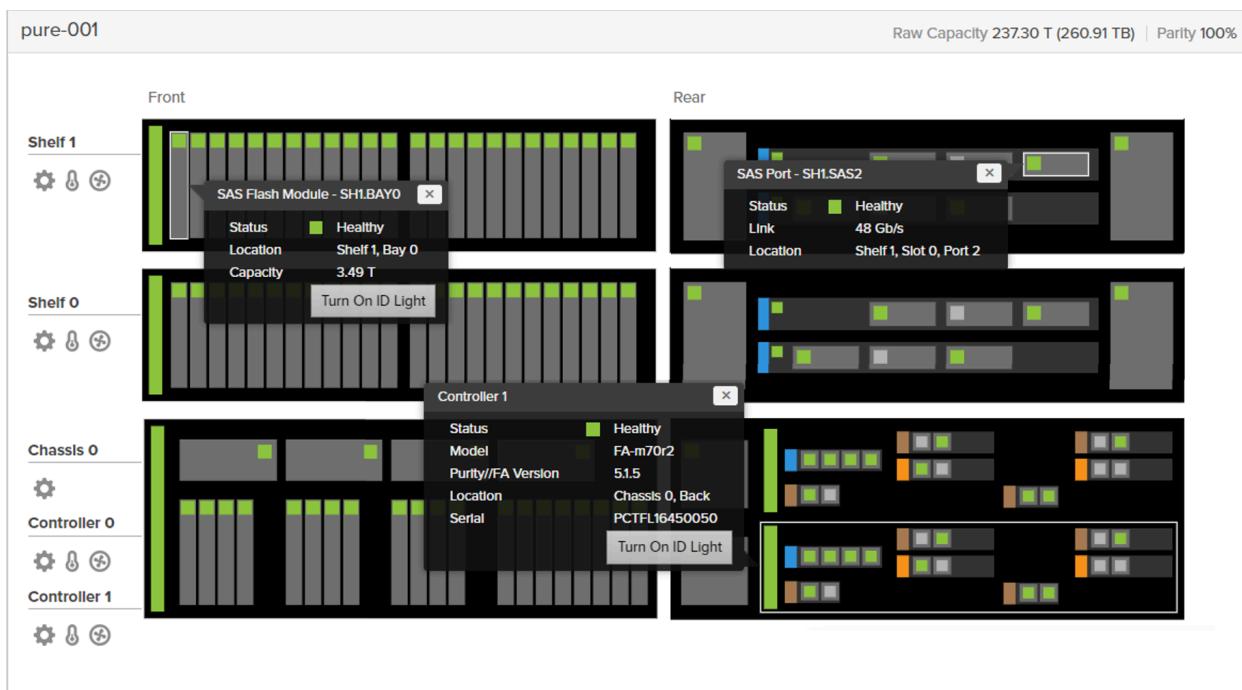
The Health page displays and manages the state of the array.

### Hardware

The **Hardware** panel graphically displays the status of the array hardware components.

**Figure 8.1: Hardware - FlashArray//M Series**

The following figure is a schematic representation of a FlashArray//M array with several component pop-ups displayed.



The title bar of the Hardware panel includes the array name, the raw capacity value, and parity information. The raw capacity value represents the total usable capacity of the array, displayed in bytes in both base 2 (e.g., 98.50 T for 98.50 tebibytes), and base 10 (e.g., 108.30 TB for 108.30 terabytes) formats. The parity value represents the percentage of data that is fully protected. The parity value will drop below 100% if the data isn't fully protected, such as when a module is pulled and the array is rebuilding the data to bring it back to full parity.

The image is a schematic representation of the array with colored indicators of each component's status. The colored squares within each hardware component represent the component status:

- **Green:** Healthy and functioning properly at full capacity.
- **Yellow:** At risk, outside of normal operating range, or unrecognized.
- **Red:** Failed, installed but not functioning, or not installed (but required).

- **Black:** Not installed. With FlashArray//M, used for NVRAM bays and storage bays that are allowed to be empty.
- **Gray:** Disconnected. Also used for components that are temporarily offline while undergoing a firmware update.

Hover the mouse over a hardware component to display its status and details. For example, hover over the Temperature component to display the following details: name of the shelf or controller that is being monitored, physical location of the temperature sensors, and current temperature readings.

Hardware components that can be actively managed from the Purity//FA GUI include buttons that perform certain functions, such as turning ID lights on and off, and changing shelf ID numbers. For example, hover over the Shelf component to display its health status and shelf ID number. Click the **Turn On ID Light** button to turn on the LED light on the physical shelf for easy identification. Click the **Change ID** button to change the ID number that appears on the physical shelf.

Hover over a flash module component to display its health status, physical location in the shelf, and capacity. If the module has been added to the array and is waiting to be admitted, click the **Admit all unadmitted drives** button to admit all of the unadmitted modules, including the current one.

## FlashArray Hardware Components

Hardware components and their naming vary by FlashArray series. To see the hardware technical specifications for each FlashArray model, refer to the Products page at <https://www.purestorage.com>.

### Hardware Components in the FA-400 Series

In the FA-400 series, the controller and storage shelf names have the form **xxm**. The names of other components have the form **xxm.YYn** or **xxm.YYYn**, where:

**xx**

Denotes the type of chassis. Controllers use **ct**. Storage shelves use **sh**.

**m**

Identifies the specific controller or storage shelf.

- For controllers, **m** has a value of 0 or 1. For example, **ct0**, **ct1**.
- For storage shelves, **m** represents the shelf number, starting at 0. For example, **sh0**, **sh1**. The assigned number can be changed on the shelf front panel or by running the **purehw setattr --id** command.

**YY or YYY**

Denotes the type of component. For example, **FAN** for cooling device, **FC** for Fibre Channel port.

**n**

Identifies the specific component by its index (its relative position within the array), starting at 0.

The following tables list the hardware components in the FA-400 series that report status, grouped by their location on the array.

The Identify Light column shows which components have an LED light on the physical component that can be turned on and off.

### **Controller (CTm)**

<b>Component Name</b>	<b>Identify Light</b>	<b>Component Type</b>
CTm	Yes	Controller
CTm.ETHn	--	Ethernet port
CTm.FANn	--	Cooling fan
CTm.FCn	--	Fibre Channel port
CTm.IBn	--	InfiniBand port
CTm.PWRn	--	Power module
CTm.SASN	--	SAS port
CTm.TMPn	--	Temperature sensor

### **Storage Shelf (SHm)**

<b>Component Name</b>	<b>Identify Light</b>	<b>Component Type</b>
SHn	Yes	Storage shelf
SHn.BAYn	Yes	Storage bay
SHn.FANn	--	Cooling fan
SHn.IOMn	--	I/O module
SHn.PWRn	--	Power module
SHn.SASN	--	SAS port
SHn.TMPn	--	Temperature sensor

## Hardware Components in FlashArray//X and FlashArray//M

The //X and //M chassis, controller and storage shelf names have the form **xxm**. The names of other components have the form **xxm.yyn** or **xxm.yyyyn**, where:

### **xx**

Denotes the type of component:

- **CH** - //X or //M chassis.
- **CT** - Controller.
- **SH** - Storage shelf.

### **m**

Identifies the specific controller or storage shelf:

- For an //X or //M chassis, **m** has a value of 0. For example, **CH0**.
- For controllers, **m** has a value of 0 or 1. For example, **CT0**, **CT1**.
- For storage shelves, **m** represents the shelf number, starting at 0. For example, **SH0**, **SH1**.  
The assigned number can be changed on the shelf front panel or by running the **purehw setattr --id** command.

### **YY or YYY**

Denotes the type of component. For example, **FAN** for cooling device, **FC** for Fibre Channel port.

### **n**

Identifies the specific component by its index (its relative position within the //X or //M chassis, controller, or storage shelf), starting at 0.

The following tables list the //X and //M hardware components that report status, grouped by their location on the array. The hardware component names are used throughout Purity//FA, for instance in the GUI **Health > Hardware** page, and with CLI commands such as **puredrive** and **purehw**.

The Identify Light column shows which components have an LED light on the physical component that can be turned on and off.

### **Chassis (CH0)**

Component Name	Identify Light	Component Type
CH0	Yes	Chassis
CH0.BAYn	Yes	Storage bay
CH0.NVBn	Yes	NVRAM bay
CH0.PWRn	--	Power module

### **Controller (CTm)**

Component Name	Identify Light	Component Type
CTm	Yes	Controller
CTm.ETHn	--	Ethernet port
CTm.FANn	--	Cooling fan
CTm.FCn	--	Fibre Channel port
CTm.IBn	--	InfiniBand port (included only with certain upgrade kits)
CTm.SASN	--	SAS port
CTm.TMPn	--	Temperature sensor

### **Storage Shelf (SHm)**

Component Name	Identify Light	Component Type
SHm	Yes	Storage shelf
SHm.BAYn	Yes	Storage bay
SHm.FANn	--	Cooling fan
SHm.IOMn	--	I/O module
SHm.PWRn	--	Power module
SHm.SASN	--	SAS port
SHm.TMPn	--	Temperature sensor

## **Capacity Upgrade and Drive Admission**

Increase storage capacity by adding data packs or individual drives to a shelf.

Data packs can be added to any shelf that has open space.

Individual drives can be added to any unused slot within a shelf. The drives must be DirectFlash modules, and the shelf must already contain similar-sized DirectFlash modules. The array will not admit individual drives of other types or sizes. Add each drive to the unused slots from left to right, starting with the lowest open slot.

Before you begin the upgrade, ensure your shelf has enough open spaces to hold the new packs or individual drives.

Performing a capacity upgrade is a two-step process: first, add the modules to the array; and second, admit the newly added modules.

When a module has been added to the array, its status changes from **unused** to **identifying** as Purity works to identify the module. The module transitions to its final **unadmitted** status when it has been successfully added to the array.

After all of the modules have been added (connected) to the array, they must be admitted. Admit all of the newly added modules at once by hovering over any one of the unadmitted modules and clicking **Admit all unadmitted modules**. Once a drive has been successfully admitted to the array, its module status changes from **unadmitted** to **healthy**. This completes the drive admission process.

If issues arise during the drive admission process, the module status changes to **unrecognized** or **failed**, or reverts to **unadmitted**. If the module status changes to **unrecognized** or **failed**, contact Pure Storage Support. If the module returns to the **unadmitted** status, try again to admit the modules. If the subsequent “admit” attempt is not successful, contact Pure Storage Support.

## Upgrading array capacity

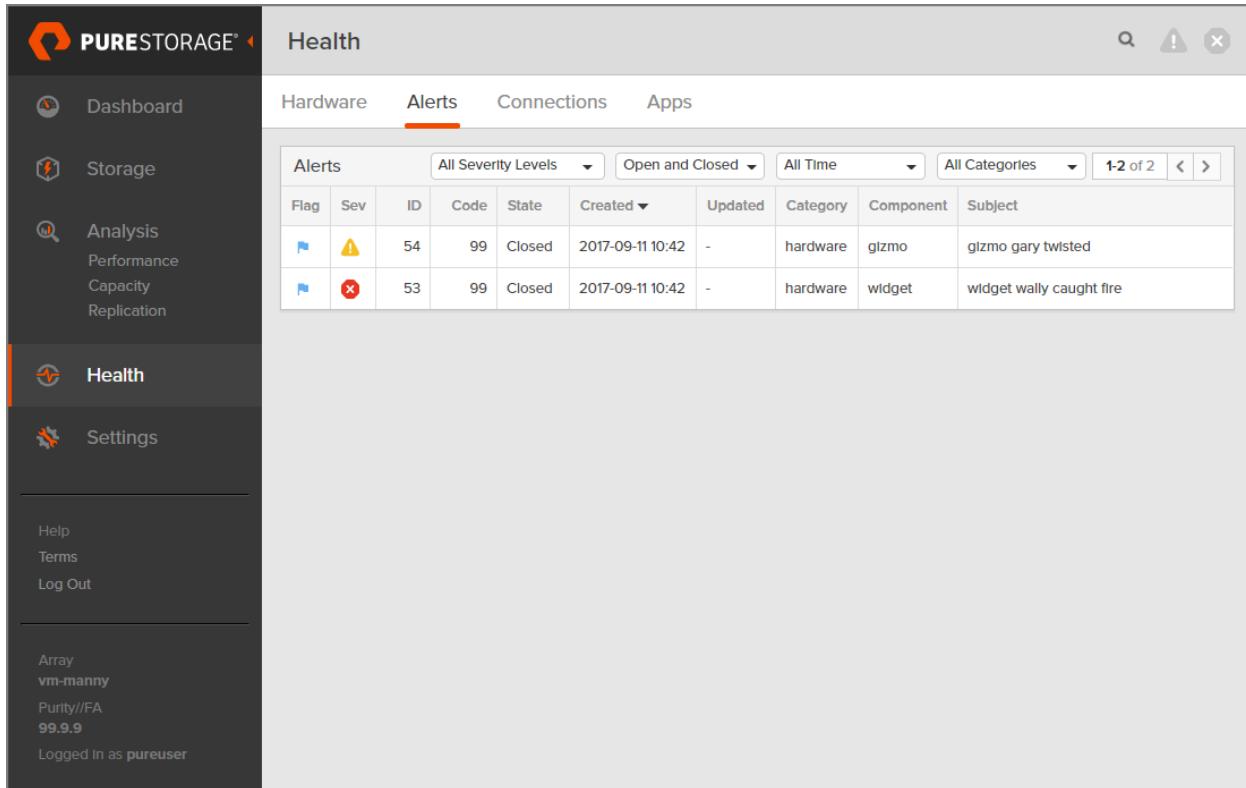
1. Verify the shelf has enough space to occupy the new data packs or individual drives. If you are adding individual drives, also verify the shelf is at least half full of DirectFlash modules that are similar in size to the drives being added.
2. Add the data packs or individual drives to the array. New drives should be added to the unused slots starting at the lowest numbered slot and working up slot by slot.
3. Hover over the newly added modules to verify that they are in **unadmitted** status, indicating that the modules have been successfully connected but not yet admitted to the array.
4. Hover over any one of the unadmitted modules and click **Admit all unadmitted modules** to admit all modules that have been added (connected) but not yet admitted to the array.
5. Hover over the newly admitted modules to verify that all of the shelves and drives are in **healthy** status, indicating that the modules have been successfully admitted and are in use by the system. This completes the drive admission process.

## Alerts

Purity//FA generates an alert when there is a change to the array or to one of its hardware or software components.

The Alerts panel displays the list of alerts that have been generated on the array.

**Figure 8.2:** Alerts



Alerts									
		All Severity Levels		Open and Closed		All Time		All Categories	
Flag	Sev	ID	Code	State	Created	Updated	Category	Component	Subject
!	!	54	99	Closed	2017-09-11 10:42	-	hardware	glizmo	glizmo gary twisted
!	!	53	99	Closed	2017-09-11 10:42	-	hardware	wlidget	wlidget wally caught fire

To conserve space, Purity//FA stores a reasonable number of alert records on the array. Older entries are deleted from the log as new entries are added. To access the complete list of messages, contact Pure Storage Support.

Purity//FA assigns a unique numeric ID to each alert as it is created. By default, alerts are sorted in chronological descending order by "Last Seen" date.

The icons that appear along the left side of each alert in the list output represent the alert severity level:

- **Blue (INFO)** icons represent informational messages generated due to a change in state. INFO messages can be used for reporting and analysis purposes. No action is required.
- **Yellow (WARNING)** icons represent important messages warning of an impending error if action is not taken.
- **Red (CRITICAL)** icons represent urgent messages that require immediate attention.

Click any of the column headings in the Alerts panel to change the sort order, and click anywhere in an alert row to display additional alert details.

Each alert in the list output includes the following information:

- **Flag:** Alert that has been flagged by Purity//FA or the user. Purity//FA automatically flags all warning and critical alerts. An alert remains flagged until you have manually cleared the flag to indicate that the alert has been addressed. If there are further changes to the condition that caused the alert (for example, a temperature of a controller or shelf has changed), Purity//FA will set the flag again.
- **Sev:** Alert severity, categorized as `critical`, `warning`, or `info`.  
Critical (red) alerts are typically triggered by service interruptions, major performance issues, or risk of data loss, and require immediate attention. For example, the array triggers a critical alert if a module has been removed from the chassis.  
Warning (yellow) alerts are of low to medium severity and require attention, though not as urgently as critical alerts. For example, the array triggers a warning alert if it detects an unhealthy module.  
Informational (blue) alerts inform users of a general behavior change and require no action. For example, the array triggers an informational alert if the NFS service is unhealthy.  
By default, alerts of all severity levels are displayed. To filter the list to display only alerts of a certain minimum severity level, click the All Severity Levels drop-down button and select the desired minimum severity level from the list.
- **ID:** Unique number assigned by the array to the alert. ID numbers are assigned to alerts in chronological ascending order.
- **Code:** Alert code number that Pure Storage uses to identify the type of alert event.
- **State:** Current state of the alert. Possible states include: `open` and `closed`.  
An alert goes from `open` state to `closed` state when the issue is completely resolved.  
By default, both open and closed alerts are displays. To filter the list to display only open alerts, click the Open and Closed drop-down button and select Open Only.
- **Created:** Date and time the alert was first generated and initial alert email notifications were sent to alert watchers.  
By default, all alert records on the array are displayed. To display a list of alerts that were created within a certain time range, click the All Time drop-down button and select the desired time range from the list.
- **Updated:** Most recent date and time the array saw the issue that generated the alert. Note that alerts that have been updated within the last 24 hours and are still open also appear in the Dashboard > Recent Alerts panel.
- **Category:** Group to which the alert belongs. Categories include Array Alerts, Hardware Alerts, Software Alerts.  
By default, alerts from all categories are displayed. To filter the list to display only alerts from a certain category, click the All Categories drop-down button and select the category from the list.
- **Component:** Specific array, software, or hardware component that triggered the alert.

- **Subject:** Alert details.

Alerting also appears in other sections of the Purity//GA GUI. From any page of the Purity//FA GUI, the alert icons that appear in the upper-right corner of the page display the number of open alerts for the respective alert severity. For example, a "1" next to a yellow warning icon indicates one open warning alert.

In the Dashboard page, the Recent Alerts pane displays a list of open alerts that have been updated within the last 24 hours.

## Flagging an alert message

To flag an alert message:

1. Select **Health > Alerts**.
2. In the Alerts panel, click the gray flag next to the alert message you want to flag. The flag turns blue, indicating that it is flagged.

## Clearing an alert flag

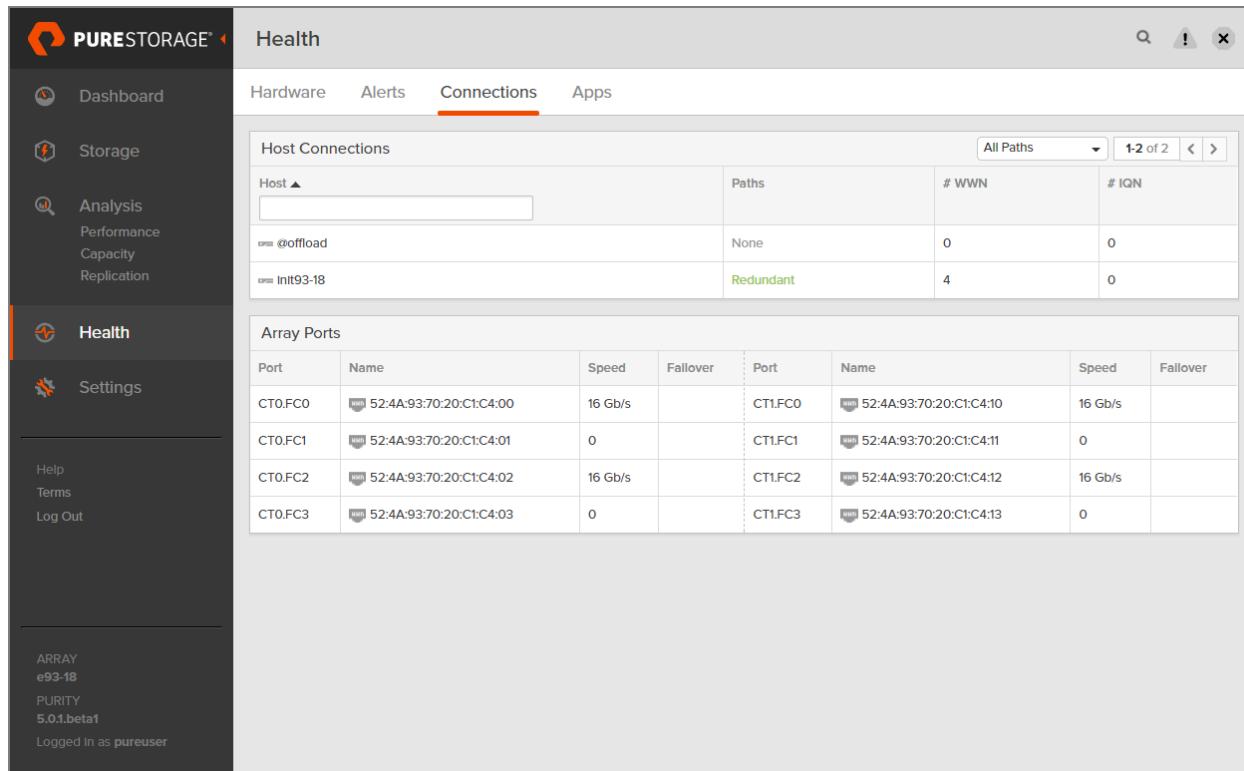
To clear an alert message flag:

1. Select **Health > Alerts**.
2. In the Alerts panel, click the blue flag next to the alert message you want to unflag. The flag turns gray, indicating that it is no longer flagged.

## Connections

The Connections page displays connectivity details between the Purity//FA hosts and the array ports.

The Host Connections panel displays a list of hosts, the connectivity status of each host, and the number of initiator ports associated with each host.

**Figure 8.3: Connections**


The screenshot shows the PureStorage Health interface. The left sidebar has a dark theme with white icons and text. It includes links for Dashboard, Storage, Analysis (Performance, Capacity, Replication), Health (selected), Settings, Help, Terms, and Log Out. Below the sidebar, it shows the array details: ARRAY e93-18, PURITY 5.0.1.beta1, and the user is Logged In as pureuser.

The main content area has a light gray background. At the top, there's a navigation bar with tabs: Hardware, Alerts, **Connections**, and Apps. Below that is a search bar with a magnifying glass icon, a warning icon, and a close button.

**Host Connections**

Host ▲	Paths	# WWN	# IQN
em@offload	None	0	0
em Init93-18	Redundant	4	0

**Array Ports**

Port	Name	Speed	Fallover	Port	Name	Speed	Fallover
CT0.FC0	52:4A:93:70:20:C1:C4:00	16 Gb/s		CT1.FC0	52:4A:93:70:20:C1:C4:10	16 Gb/s	
CT0.FC1	52:4A:93:70:20:C1:C4:01	0		CT1.FC1	52:4A:93:70:20:C1:C4:11	0	
CT0.FC2	52:4A:93:70:20:C1:C4:02	16 Gb/s		CT1.FC2	52:4A:93:70:20:C1:C4:12	16 Gb/s	
CT0.FC3	52:4A:93:70:20:C1:C4:03	0		CT1.FC3	52:4A:93:70:20:C1:C4:13	0	

The Paths column displays the connectivity status between the host and controllers in a highly available environment, where the colored value indicates one of the following connection health statuses:

- **Green:** Fully redundant and highly available. No issues detected.
- **Yellow:** Not fully redundant. Issues detected that may impact high availability.
- **Red:** Single controller connectivity only.
- **Gray:** No connectivity.

Possible connection statuses include:

#### Redundant

All paths between the host and each of the controllers in a highly available array are connected.

#### Uneven

The number of paths between the host and each controller is uneven. This may impact high availability. Make sure that there are the same number of paths from the host to each controller.

#### Unused Port

The host has unused initiators. This may impact high availability. Make sure that all of the initiators have at least one path to the array.

### **Single Controller**

The host has paths to only one of the controllers. No paths exist to the other controller. This impacts high availability. Make sure that there are redundant paths from the host to both controllers.

### **Single Controller - Failover**

The host has paths to one controller, but one or more of those paths has failed over.

### **None**

The host is not connected to any of the controllers.

Select the check boxes along the top of the Host Connections list to filter the hosts by connection status.

The Array Ports panel displays the connection mappings between each array port and initiator port. Each array port includes the following connectivity details: associated iSCSI Qualified Name (IQN), NVMe Qualified Name (NQN), or Fibre Channel World Wide Name (WWN) address, communication speed, and failover status. A check mark in the Failover column mean the port has failed over to the corresponding port pair on the primary controller.

## **Viewing host connection details**

The connection map displays connectivity details for all hosts.

To view the host connection details:

1. Select **Health > Connections**.

- To view the connection map details for a specific host-volume connection, in the Host Connections panel, click the host to drill down to its connection map.
- To view the connectivity status between the hosts and the array controller ports, in the Host Connections pane, click the menu icon, select up to 10 hosts that you want to analyze, and select **Compare**

## **Viewing array port details**

To view the array port details:

1. Select **Health > Connections**. The Array Ports pane displays the connection mappings between each array port and initiator port.

Optionally click the menu icon and select **Download CSV** to save the `ports.csv` file to your local machine.



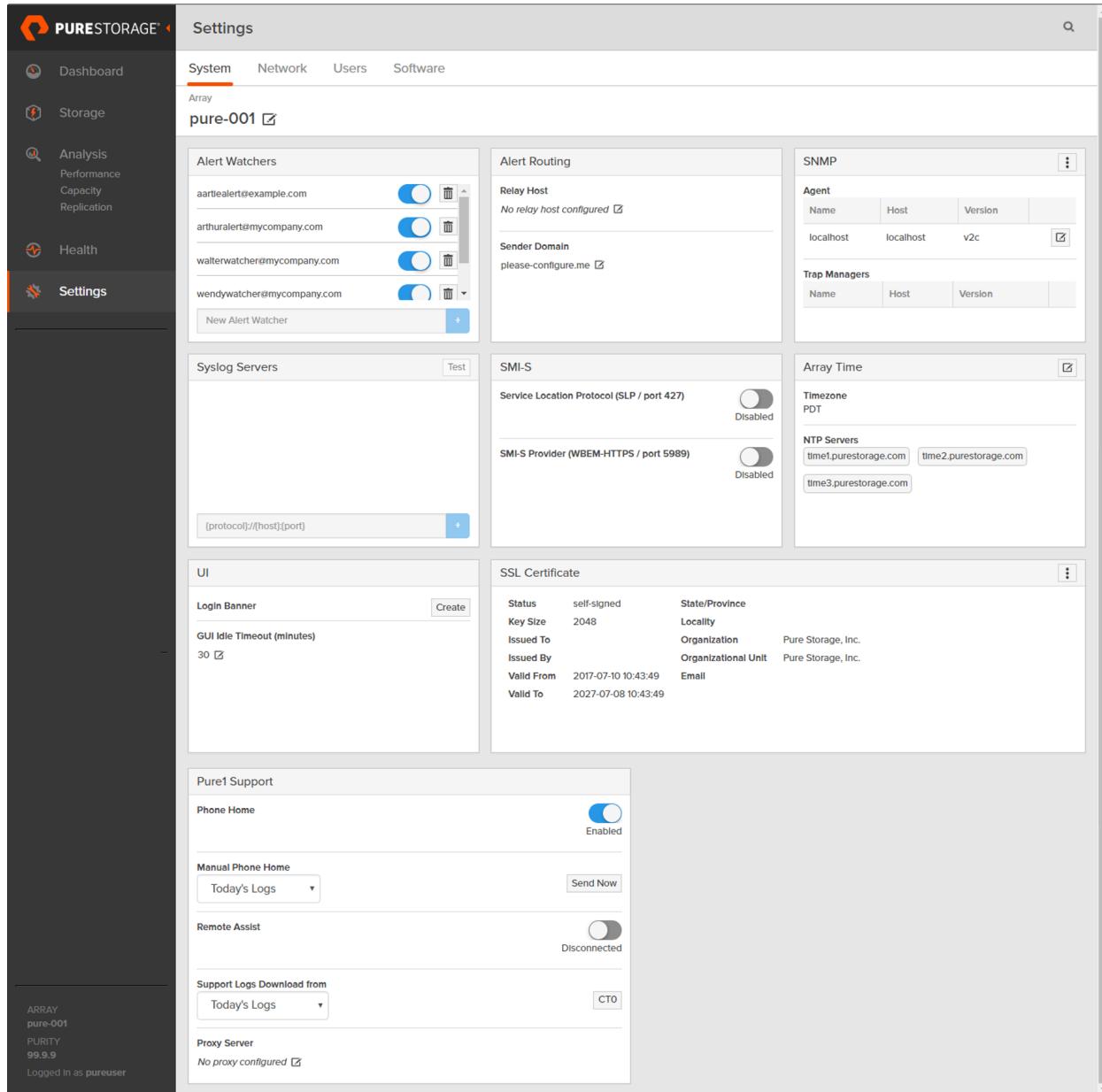
## Chapter 9. Settings

The Settings page displays and manages the general attributes and network settings of an array.

### System

The **Settings > System** page displays and manages the general attributes of the FlashArray array.

**Figure 9.1: Settings - System Page**



The screenshot shows the PureStorage Settings - System page for array pure-001. The left sidebar includes links for Dashboard, Storage, Analysis, Capacity, Replication, Health, and Settings (which is selected). The main content area is divided into several sections:

- Alert Watchers:** Lists four email addresses (aartaleal@example.com, arthuralert@mycompany.com, walterwatcher@mycompany.com, wendywatcher@mycompany.com) with toggle switches and delete icons.
- Alert Routing:** Shows 'Relay Host' status as 'No relay host configured' and 'Sender Domain' as 'please-configure.me'.
- SNMP:** Displays 'Agent' information for localhost (Host: localhost, Version: v2c) and 'Trap Managers' table.
- SMI-S:** Shows 'Service Location Protocol (SLP / port 427)' and 'SMI-S Provider (WBEM-HTTPS / port 5989)' both set to 'Disabled'.
- Array Time:** Shows 'Timezone' as PDT and 'NTP Servers' with entries for time1.purestorage.com, time2.purestorage.com, and time3.purestorage.com.
- SSL Certificate:** Details for a self-signed certificate issued to Pure Storage, Inc., valid from 2017-07-10 to 2027-07-08.
- Pure1 Support:** Includes sections for 'Phone Home' (Enabled), 'Manual Phone Home' (Today's Logs, Send Now button), 'Remote Assist' (Disconnected), 'Support Logs Download from' (Today's Logs, CTO button), and 'Proxy Server' (No proxy configured).

The bottom left corner of the sidebar shows the array name (pure-001), purity version (99.9.9), and user (Logged In as pureuser).

## Array Name

At the top of the System page is the name of the array. The array name is used for various administration and configuration purposes.

The array name appears in audit and alert messages. The array name also represents the sending account name for Purity//FA email alert messages.

The name is used to identify the array when connecting it to other arrays. For asynchronous replication, the array name appears as part of the snapshot name when viewing replicated snapshots on a target array. For synchronous replication, the array name is used to identify the arrays over which pods are stretched and unstretched.

The array can be renamed at any time, and the name change takes effect immediately. Note that Purity//FA does not register array names with the DNS, so if you change the array name, you must re-register the name before the array can be addressed by name in browser address bars, ICMP ping commands, and so on.

### Renaming the array

1. Select **Settings > System**.
2. Click the edit icon next to the current array name. The array name becomes an editable box.
3. In the editable box, type the new array name.
4. Click the check mark icon to confirm the change.

## Alert Watchers

Purity//FA generates an alert whenever the health of a component degrades or a capacity threshold is reached. Alerts can also be sent as email notifications to designated alert watchers.

The Alert Watchers panel displays the email addresses of designated alert watchers and the alert status of each watcher. The sending account name for Purity//FA alert email notifications is the array name at the configured sender domain.

Up to 20 alert watchers can be designated. The list of alert watchers includes the built-in `flasharray-alerts@purestorage.com` address, which cannot be deleted.

Once added, an alert watcher will start receiving alert email notifications.

Alert watchers can be in enabled or disabled status. Alert watchers who are in enabled status receive alert email notifications. When an alert watcher is created, its watcher status is automatically set to enabled status. Alert watchers who are in disabled status do not receive alert email notifications.

Disabling an alert watcher does not delete the recipient's email address - it only stops the watcher from receiving alert notifications. Alert watchers can be enabled and disabled at any time. The current alert watcher status is determined by the color of the toggle button that appears next to the alert watcher email address, where blue represents an enabled alert watcher and gray represents a disabled alert watcher.

Deleting an alert watcher completely removes the watcher from the list. Once an email address has been deleted, the corresponding alert watcher will no longer receive alert notifications.

## Adding an alert watcher

You can designate up to 19 alert recipients.

1. Select **Settings > System**.
2. In the Alert Watchers panel, type the email address of the alert watcher.
3. Click the Add button to add the email address to the list of alert watchers. Once added, the alert watcher immediately starts receiving alert email messages.

## Enabling and disabling an alert watcher

You cannot disable built-in alert recipient flasharray-alerts@purestorage.com.

1. Select **Settings > System**.
2. In the Alert Watchers panel, click the toggle button to enable (blue) and disable (gray) an alert watcher. Once enabled, an alert watcher starts receiving alert email notifications.

## Deleting an alert watcher

You cannot delete built-in alert recipient flasharray-alerts@purestorage.com.

1. Select **Settings > System**.
2. In the Alert Watchers panel, click the delete icon next to the alert watcher you want to delete.

## Alert Routing

The Alert Routing panel displays the ways in which alerts and logs are managed.

### Relay Host

The relay host represents the hostname or IP address of the email relay server currently being used as a forwarding point for alert email notifications generated by the array.

For SMTP servers that require authentication, also specify the username and password. The username represents the SMTP account name used to authenticate into the relay host SMTP server. The password represents the SMTP password used to authenticate into the relay host SMTP server.

If a relay host is not configured, Purity//FA sends all alert email notifications directly to the recipient addresses rather than route them via the relay (mail forwarding) server.

#### Configuring the SMTP relay host

1. Select **Settings > System**.
  2. In the Alert Routing panel, click the edit icon. The Edit SMTP dialog box appears.
  3. In the Relay Host field, type the host name or IP address of the email relay server that is to be used as the forwarding point for alert email notifications generated by the array. If specifying an IP address, enter the IPv4 or IPv6 address.
- For IPv4, specify the IP address in the form **ddd.ddd.ddd.ddd**, where **ddd** is a number ranging from 0 to 255 representing a group of 8 bits. If a port number is also specified, append

it to the end of the address in the form `ddd.ddd.ddd.ddd:PORT`, where `PORT` represents the port number.

For IPv6, specify the IP address in the form

`xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx`, where `xxxx` is a hexadecimal number representing a group of 16 bits. Consecutive fields of zeros can be shortened by replacing the zeros with a double colon (`::`). If a port number is also specified, enclose the entire address in square brackets (`[ ]`) and append the port number to the end of the address. For example, `[xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx] :PORT`, where `PORT` represents the port number.

4. In the Username field, type the SMTP account name used to authenticate into the relay host SMTP server. Only specify the username and password if the SMTP server requires authentication.
5. In the Password field, type the SMTP password used to authenticate into the relay host SMTP server. Only specify the username and password if the SMTP server requires authentication.
6. Click **Save**.

#### Deleting the SMTP relay host

1. Select **Settings > System**.
2. In the Alert Routing panel, click the edit icon. The Edit SMTP dialog box appears.
3. Delete the relay host name or IP address. Optionally delete the SMTP user name and password.
4. Click **Save**.

#### Sender Domain

The sender domain determines how logs are parsed and treated by Pure Storage Support. The domain name is also used in the "from" address of outgoing alert email notifications. By default, the sender domain is set to the domain name `please-configure.me`.

It is crucial that you set the sender domain to the correct domain name. If the array is not a Pure Storage test array, set the sender domain to the actual customer domain name. For example, `mycompany.com`.

The email address that Purity//FA uses to send alert messages includes the sender domain name and is comprised of the following components:

`<Array_Name>-<Controller_Name>@<Sender_Domain_Name>.com`

For example, `purearray-ct0@mycompany.com`.

#### Configuring the sender domain

1. Select **Settings > System**.
2. In the Sender Domain section of the Alert Routing panel, click the edit icon. The field becomes an editable box.
3. In the editable box, type the sender domain name.

The default domain name is `please-configure.me`. If this is not a Pure Storage test array, the domain name must be set to your company's domain name. For example, `mycompany.com`.

**Important Note:** The sender domain determines how Purity//FA logs are parsed and treated by Pure Storage Support, so it is crucial that you set the sender domain to the correct domain name.

4. Click the check mark icon to confirm the change.

## UI

The UI panel displays and manages general user interface details, including banner text and idle timeout.

### Login Banner

The Login Banner section enables you to create a message that Purity//FA users see in the Purity//FA GUI login screen when logging into the Purity//FA GUI, and before the password prompt when logging into the CLI.

Creating a banner message

To create a banner message for all Purity//FA users to see:

1. Go to the **Settings > System > UI** panel.
2. In the Login Banner section, click **Create**. The Edit Login Banner dialog box appears.
3. Click inside the text box and type the banner message. The message can be up to 2000 characters long and accepts ASCII characters.
4. Click **Save**, and then click **Close**. Notice that the **Create** button becomes a **View** button, allowing you to view and edit the existing banner message at any time.
5. To verify that the banner message appears, log out of the Purity//FA GUI. The banner message appears in the Purity//FA GUI login screen, just above the Pure Storage logo.

### GUI Idle Timeout

The GUI Idle Timeout feature displays the length of time, measured in minutes, that the Purity//FA GUI can be idle before the user is logged out of the session.

The default idle time is 30 minutes.

Setting the idle timeout value

To set the idle timeout value:

1. Go to the **Settings > System > UI** panel.
2. In the GUI Idle Timeout section, click the edit icon next to the current idle timeout value (in minutes), and then enter the amount of time in minutes that a Purity//FA GUI session can be idle before the user is logged out. The idle time can be any length between 5 and 180 minutes. To disable the idle timeout setting, set the idle time to 0 minutes.

3. Click the check mark to confirm the change. The idle timeout setting takes effect the next time you log in to the Purity//FA GUI.

Disabling the idle timeout setting

To disable the idle timeout setting:

1. Go to the **Settings > System > UI** panel.
2. In the GUI Idle Timeout section, click the edit icon next to the current idle timeout value (in minutes), and then enter 0.
3. Click the check mark to confirm the change. The idle timeout setting becomes disabled the next time you log in to the Purity//FA GUI.

## Syslog Servers

The Syslog Servers feature enables you to forward syslog messages to remote servers.

The Purity//FA syslog logging facility generates messages of major events within the FlashArray and forwards the messages to remote servers. Purity//FA generates syslog messages for three types of events:

- Alerts (purity.alert)
- Audit Trails (purity.audit)
- Tests (purity.test)

Purity//FA generates alerts when there is a change to the array or to one of the Purity//FA hardware or software components. There are three alert severity levels:

- **INFO:** Informational messages that are generated due to a change in state. INFO messages can be used for reporting and analysis purposes. No action is required.
- **WARNING:** Important messages that warn of an impending error if action is not taken.
- **CRITICAL:** Urgent messages that require immediate attention.

Syslog alerts are broken down into the following format:

```
<Event Timestamp> <Array IP Address> purity.alert <Alert Severity> <Alert Details>
```

In the following example, Purity//FA generated a WARNING alert because space consumption on the array exceeded 90%:

## Figure 9.2: Syslog Server - Alerts

```

Jun 16 18:51:49 10.124.190.231 purity.alert:
  · WARNING · Reminder: Storage consumption has reached 91.00% of usable capacity [25]#012I/O
  · performance may degrade when consumed storage is above the array's rated capacity.#012Replication
  · of protection group snapshots is stopped when array reaches full capacity#012#012
  · Severity: Warning #012UTC Time: 2014 Jun 28 23:28:19#012
  · Array Time: 2014 Jun 28 16:28:19 PDT#012Array Name: array1#012
  · Domain: purestorage.com#012Suggested Action: Consider increasing capacity by adding a storage
  · shelf, or an additional FlashArray to increase both capacity and performance.#012#012Purity
  · Version: 99.9.9#012 Array ID: bc11aa90-8698-d02d-93ad-b261c0ca1d73#012Controller Name: ct0#012
  · Controller Serial: SNO#012UUIDs: ['952f620088874e02b920c647f802fa52']#012 Variables:
  · (below)#012Used: 91#012Total: 100#012Percent Usage: 91.00%
  
```

Alerts are also sent via the phone home facility to the Pure Storage Support team. If configured, alerts can also be sent to designated email recipients and SNMP trap managers.

You can also view alerts through the GUI (Health > Alerts) and CLI (`puremessage list` command).

An audit trail represents a chronological history of the GUI, CLI, or REST API operations that a user has performed to modify the configuration of the array. Each message within an audit trail includes the name of the Purity//FA user who performed the operation and the Purity//FA operation that was performed.

Syslog audit trail messages are broken down into the following format:

```
<Event Timestamp> <Array IP Address> <purity.audit> <Purity//FA Username>
<Purity//FA Command> <Audit Trail Message Details>
```

In the following example, `pureuser` performed various GUI, CLI, or REST API operations:

## Figure 9.3: Syslog Server - Audit Trails

```

Jun 30 10:54:16 10.124.132.12 purity.audit: [pureuser] purearray setattr --syslogserver
tcp://pure01.example.com:514.
  · Message ID: 75 UTC Time: 2014-06-30T17:54:16Z Array Name: trlt12
Jun 30 11:11:39 10.124.191.254 purity.audit: [pureuser] purevol create vol2 --size 10G.
  · Message ID: 3302 UTC Time: 2014-06-30T18:11:39Z Array Name: array1
Jun 30 13:34:59 10.124.190.231 purity.audit: [pureuser] purevol destroy test-vol.
  · Message ID: 982 UTC Time: 2014-06-30T20:34:59Z Array Name: array1
Jun 30 13:34:59 10.124.190.231 purity.audit: [pureuser] purevol eradicate test-vol.
  · Message ID: 983 UTC Time: 2014-06-30T20:34:59Z Array Name: array1
Jun 30 13:35:00 10.124.190.231 purity.audit: [pureuser] purearray disconnect array1.
  · Message ID: 984 UTC Time: 2014-06-30T20:35:00Z Array Name: array1
  
```

You can also view audit messages through the GUI (Settings > Users) and CLI (`pureaudit list` command).

Test messages represent a history of all tests generated by users to verify that the array can send messages to email recipients. The message does not indicate whether or not the test message successfully reached the recipients.

Syslog test messages are broken down into the following format:

```
<Event Timestamp> <Array IP Address> <purity.test> <Purity//FA Username>
<Test Message Details>
```

In the following example, `pureuser` performed a test to determine if the array could send messages to email addresses:

#### Figure 9.4: Syslog Server - Tests

```
Jun 30 13:25:03 10.124.191.254 purity.test:  
INFO [pureuser] This is a test message generated by Pure Storage FlashArray.  
UTC Time: 2014-Jun 30 20:25:03 Array Name: array 1
```

#### Setting the syslog server output location

To set the syslog server output location:

1. Select **Settings > System**.
2. In the **Syslog Server** panel, enter the URI of the remote syslog server. For example, `tcp://MyHost.com`.

Specify the URI in the format `PROTOCOL://HOSTNAME:PORT`.

`PROTOCOL` is "tcp", "tls", or "udp".

`HOSTNAME` is the syslog server hostname or IP address. If specifying an IP address, for IPv4, specify the IP address in the form `ddd.ddd.ddd.ddd`, where `ddd` is a number ranging from 0 to 255 representing a group of 8 bits.

For IPv6, specify the IP address in the form

`[xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx]`, where `xxxx` is a hexadecimal number representing a group of 16 bits. Enclose the entire address in square brackets (`[]`). Consecutive fields of zeros can be shortened by replacing the zeros with a double colon (`::`).

`PORT` is the port at which the server is listening. Append the port number after the end of the entire address. If the port is not specified, it defaults to 514.

3. Click the Add button to add the URI to the list of syslog server output locations.
4. Optionally, click **Test** to test the setting.

#### SMI-S

The SMI-S panel manages the Pure Storage Storage Management Initiative Specification (SMI-S) provider.

Enable the SMI-S provider to administer the array through an SMI-S client. The SMI-S provider is optional and must be enabled before its first use.

For more information about the SMI-S provider, refer to the Pure Storage SMI-S Provider Guide in the Pure1 Knowledge site at <https://support.purestorage.com>.

#### Array Time

The Array Time panel displays the array's current time, and the IP addresses or fully qualified hostnames of the Network Time Protocol (NTP) servers with which array time is synchronized.

Pure Storage technicians set the array time zone during installation. By default, the array time is synchronized to an NTP server operated by Pure Storage. Alternate NTP servers can be designated.

## Time

The displayed time is based on the time zone of the array, which is set during the FlashArray installation.

## NTP Servers

The NTP Servers section displays the hostnames or IP addresses of the Network Time Protocol (NTP) servers that are currently being used by the array to maintain reference time. The installation technician sets the proper time zone for an array when it is installed. During operation, arrays maintain time synchronization by interacting with the NTP server.

Designating an alternate NTP server

The array maintains time synchronization by interacting with the NTP server.

To designate an alternate NTP server:

1. Select **Settings > System**.
2. In the NTP Servers section of the Time panel, perform one of the following tasks:
  - To add an NTP server, in the New NTP Server(s) text box, type the hostname or IP address of the NTP server used by the array to maintain reference time, and then click the Add button. You can add up to four NTP servers. Enter multiple servers as comma-separated values.  
If specifying an IP address, for IPv4, specify the IP address in the form **ddd.ddd.ddd.ddd**, where **ddd** is a number ranging from 0 to 255 representing a group of 8 bits. For IPv6, specify the IP address in the form **xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx**, where **xxxx** is a hexadecimal number representing a group of 16 bits. When specifying an IPv6 address, consecutive fields of zeros can be shortened by replacing the zeros with a double colon (**::**).
  - To remove an NTP server, select the check box of the server you want to remove, and then click the delete icon.

## Pure1 Support

The Pure1 Support panel displays and manages the features used to communicate with Pure Storage Support.

## Phone Home

The phone home facility provides a secure direct link between the array and Pure Storage Support to transmit log and diagnostic information.

This information provides Pure Storage Support with complete recent history about array performance and significant events in case diagnosis or remedial actions are required. Alerts are reported immediately when they occur so that timely action can be taken.

The phone home facility can be enabled (blue) or disabled (gray) at any time. By default, the phone home facility is enabled. Log and diagnostic information is only transmitted when the feature is enabled. If the phone home facility is disabled, historical log contents are delivered when the facility is next enabled; Purity will continue to send alerts to designated email recipients and SNMP trap managers if those features are configured.

#### Enabling and disabling phone home

Enable the phone home facility to automatically transmit log files on an hourly basis to Pure Storage Support via the phone home channel.

Note: If a proxy host is required by https, configure the proxy server.

1. Select **Settings > System**.
2. In the Phone Home section of the Pure1 Support panel, click the toggle button to switch between enabled (blue) and disabled (gray) status.

#### Manual Phone Home

Phone home logs can be sent to Pure Storage Support on demand, with options including Today's Logs, Yesterday's Logs, or All Logs.

#### Sending phone home logs to Pure Storage Support

Note: If a proxy host is required by https, configure the proxy server.

To manually send array log files to Pure Storage Support via the phone home channel:

1. Select **Settings > System**.
2. In the Manual Phone Home section of the Pure1 Support panel, In the Manual Phone Home section, select one of the following options from the drop-down list:
  - **Today's Logs**: Sends log information from the current day (in the array's time zone)
  - **Yesterday's Logs**: Sends log information from the previous day (in the array's time zone)
  - **All Log History**: Sends log information from the previous day (in the array's time zone)
3. Click **Send Now** to send the log files to Pure Storage Support.

#### Remote Assist

In some cases, the most efficient way for Pure Storage Support to service a FlashArray array or diagnose problems is through direct access to the array. A remote assistance (RA) session grants Pure Storage Support direct and secure access to the array through a reverse tunnel which you, the administrator, open. This is a two-way communication.

Opening an RA session gives Pure Storage Support the ability to log into the array, effectively establishing an administrative session. Once the RA session is successfully established, the array returns connection details, including the date and time when the session was opened, the date and time when the session expires, and the proxy status (true, if configured).

After the Pure Storage Support team has performed all of the necessary diagnostic or maintenance functions, close the RA session to terminate the connection.

RA sessions can be opened/connected (blue) and closed/disconnected (gray) at any time. By default, the RA session is closed/disconnected.

Opening and closing a remote assist session does not affect the current administrative session.

An open RA session automatically terminates (disconnects) after two days have lapsed.

Opening and closing a remote assistance (RA) session

Opening and closing a remote assist session does not affect the current administrative session.

1. Select **Settings > System**.
2. In the Remote Assistance section of the Pure1 Support panel, click the toggle button to open (blue) and close (gray) an RA session. Opening an RA session gives Pure Storage Support direct and secure access to the array. After the Pure Storage Support team has performed all of the necessary diagnostic functions, close the RA session.

## Support Logs

Purity//FA continuously logs a variety of array activity, including performance metrics, hardware and software operations, and administrative actions. Array activity is time stamped and organized in chronological order. The Support Logs panel allows you to download the Purity//FA log contents of the specified controller to the current administrative workstation.

If Phone Home is enabled, the logs are periodically transmitted to Pure Storage. The logs are also saved to the array, available for manual download.

When the support logs are manually downloaded, the array generates a password-protected **.zip** file containing all of the logs and saves it to your local machine.

### Downloading Support Logs

1. Select **Settings > System**.
2. In the Support Logs section of the Pure1 Support panel, select the time range representing the approximate array time the activity of interest occurred.
3. In the "Download from" section, click the button corresponding to the controller from which you want to download the support logs. For example, click **CT0** to download the logs for the primary controller. The password-encrypted **.zip** file is saved to your local machine. The file can only be opened by Pure Storage Support.

## Proxy Server

The Proxy section manages the proxy hostname for **https** log transmission. The proxy hostname, if set, represents the server to be used as the HTTP or HTTPS proxy. The format for the proxy host name is **http(s)://hostname:port**, where **hostname** is the name of the proxy host, and **port** is the TCP/IP port number used by the proxy host.

## Configuring the proxy host

To configure the proxy host for HTTPS communication for phone home and log transmission:

1. Select **Settings > System**.
2. In the Proxy Server section of the Pure1 Support panel, click the edit icon. The field becomes an editable box.
3. In the editable box, type the proxy host name.  
The format for the host name is `http(s)://hostname:port`, where `hostname` is the name of the proxy host, and `port` is the TCP/IP port number used by the proxy host.
4. Click the check mark icon to confirm the change.

## Deleting the proxy host

1. Select **Settings > System**.
2. In the Proxy Server section of the Pure1 Support panel, click the edit icon next to the current proxy host name. The proxy host name becomes an editable box.
3. Delete the proxy host name.
4. Click the check mark icon to confirm the deletion.

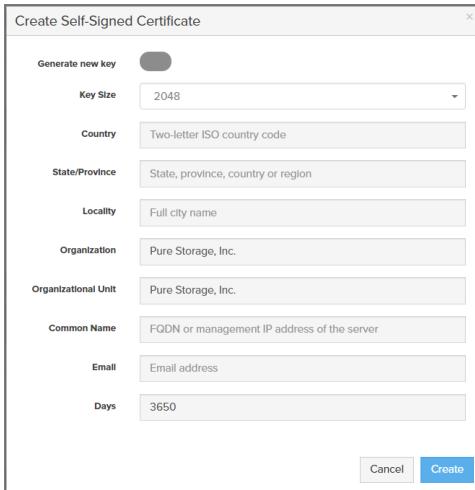
## SSL Certificate

Purity//FA creates a self-signed certificate and private key when you start the system for the first time. The SSL Certificate panel allows you to view and change certificate attributes, create a new self-signed certificate, construct certificate signing requests, import certificates and private keys, and export certificates.

### Self-Signed Certificate

Creating a self-signed certificate replaces the current certificate. When you create a self-signed certificate, include any attribute changes, specify the validity period of the new certificate, and optionally generate a new private key.

**Figure 9.5: SSL Certificate - Create Self-Signed Certificate**



When you create the self-signed certificate, you can generate a private key and specify a different key size. If you do not generate a private key, the new certificate uses the existing key.

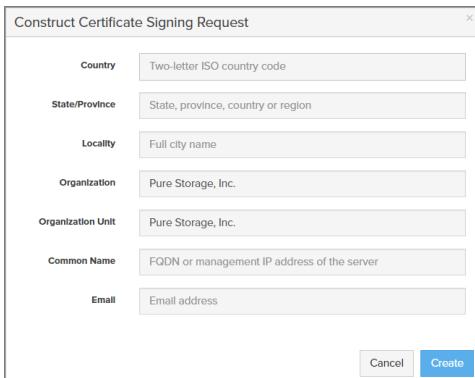
You can change the validity period of the new self-signed certificate. By default, self-signed certificates are valid for 3650 days.

### CA-Signed Certificate

Certificate authorities (CA) are third party entities outside the organization that issue certificates.

To obtain a CA certificate, you must first construct a certificate signing request (CSR) on the array.

**Figure 9.6: SSL Certificate - Construct Certificate Signing Request**

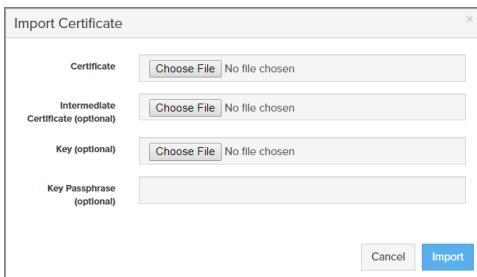


The CSR represents a block of encrypted data specific to your organization. You can change the certificate attributes when you construct the CSR; otherwise, Purity//FA will reuse the attributes of the current certificate (self-signed or imported) to construct the new one. Note that the certificate attribute changes will only be visible after you import the signed certificate from the CA.

Send the CSR to a certificate authority for signing. The certificate authority returns the SSL certificate for you to import. Verify that the signed certificate is PEM formatted (Base64 encoded), includes the "-----BEGIN CERTIFICATE-----" and "-----END CERTIFICATE-----" lines, and does not

exceed 3000 characters in total length. When you import the certificate, also import the intermediate certificate if it is not bundled with the CA certificate.

**Figure 9.7: SSL Certificate - Import CA Certificate**



If the certificate is signed with the CSR that was constructed on the current array and you did not change the private key, you do not need to import the key. However, if the CSR was not constructed on the current array or if the private key has changed since you constructed the CSR, you must import the private key. If the private key is encrypted, also specify the passphrase.

### Certificate Administration

The attributes of a self-signed certificate can only be changed by creating a new certificate. Certificate attributes include organization-specific information, such as country, state, locality, organization, organizational unit, common name, and email address.

The export feature allows you to view and export the certificate and intermediate certificates for backup purposes.

### Creating or Changing the Attributes of a Self-Signed Certificate

Note: When you change the certificate attributes, Purity//FA replaces the existing certificate with the new certificate and its specified attributes.

1. Select **Settings > System**.
2. In the SSL Certificate panel, click the menu icon and select **Create Self-Signed Certificate**. The Create Self-Signed Certificate pop-up window appears.
3. Complete or modify the following fields:
  - **Generate new key:** Click the toggle button to generate (blue) or not generate (gray) a new private key with the self-signed certificate. If you do not generate a new private key, the certificate uses the existing key.
  - **Key Size:** If you generate a new private key, specify the key size. The default key size is 2048 bits. A key size smaller than 2048 is considered insecure.
  - **Country:** Enter the two-letter ISO code for the country where your organization is located.
  - **State/Province:** Enter the full name of the state or province where your organization is located.
  - **Locality:** Enter the full name of the city where your organization is located.

- **Organization:** Enter the full and exact legal name of your organization. The organization name should not be abbreviated and should include suffixes such as Inc, Corp, or LLC.
  - **Organizational Unit:** Enter the department within your organization that is managing the certificate.
  - **Common Name:** Enter the fully qualified domain name (FQDN) of the current array. For example, the common name for <https://purearray.example.com> is purearray.example.com, or \*.example.com for a wildcard certificate. The common name can also be the management IP address of the array or the short name of the current array. Common names cannot have more than 64 characters.
  - **Email:** Enter the email address used to contact your organization.
  - **Days:** Specify the number of valid days for the self-signed certificate being generated. If not specified, the self-signed certificate expires after 3650 days.
4. Click **Create**. Purity//FA restarts the GUI and signs you in using the self-signed certificate.

### Constructing a Certificate Signing Request to Obtain a CA Certificate

Note: When you change the certificate attributes, Purity//FA replaces the existing certificate with the new certificate and its specified attributes.

1. Select **Settings > System**.
2. In the SSL Certificate panel, click the menu icon and select **Construct Certificate Signing Request**. The Construct Certificate Signing Request pop-up window appears.
3. Complete or modify the following fields:
  - **Country:** Enter the two-letter ISO code for the country where your organization is located.
  - **State/Province:** Enter the full name of the state or province where your organization is located.
  - **Locality:** Enter the full name of the city where your organization is located.
  - **Organization:** Enter the full and exact legal name of your organization. The organization name should not be abbreviated and should include suffixes such as Inc, Corp, or LLC.
  - **Organizational Unit:** Enter the department within your organization that is managing the certificate.
  - **Common Name:** Enter the fully qualified domain name (FQDN) of the current array. For example, the common name for <https://purearray.example.com> is purearray.example.com, or \*.example.com for a wildcard certificate. The common name can also be the management IP address of the array or the short name of the current array. Common names cannot have more than 64 characters.
  - **Email:** Enter the email address used to contact your organization.

4. Click **Create** to construct the CSR. The CSR pop-up window appears, displaying the CSR as a block of encrypted data.
5. Click **Download** to download the CSR, which you can send to a certificate authority (CA) for signing.

### Importing a CA Certificate

After you receive the signed certificate from the CA, you are ready to import it to replace the existing certificate.

1. Verify that the signed certificate is PEM formatted (Base64 encoded), includes the "-----BEGIN CERTIFICATE-----" and "-----END CERTIFICATE-----" lines, and does not exceed 3000 characters in length.
2. Select **Settings > System**.
3. In the SSL Certificate panel, click the menu icon and select **Import Certificate**. The Import Certificate pop-up window appears.
4. Complete or modify the following fields:
  - **Certificate:** Click **Choose File** and select the signed certificate you received from the CA.
  - **Intermediate Certificate:** If you also received an intermediate certificate from the CA, click **Choose File** and select the intermediate certificate.
  - **Key:** If the CSR was not constructed on the current array or the private key has changed since you constructed the CSR, click **Choose File** and select the private key.
  - **Key Passphrase:** If the private key is encrypted with a passphrase, enter the passphrase.
5. Click **Import**.

### Viewing and Exporting Certificate Details

1. Select **Settings > System**.
2. In the SSL Certificate panel, click the menu icon and select **Export Certificate** or **Export Intermediate Certificate**. The Export Certificate pop-up window appears.
3. Click **Download** to view and export the certificate and its details.

### Rapid Data Locking

The Rapid Data Locking panel displays the status of the Rapid Data Locking (RDL) feature as enabled or disabled. The RDL feature is a FlashArray option that adds external security tokens to enhance the data security of an array.

The RDL feature requires both hardware and software configuration and can only be enabled by a CLI command. For more information about the Rapid Data Locking (RDL) feature, refer to the FlashArray Enhanced Data Security Guide in the Pure1 Knowledge site at <https://support.purestorage.com>.

## SNMP

The Simple Network Management Protocol (SNMP) is used by SNMP agents and SNMP managers to send and retrieve information. FlashArray supports SNMP versions v2c and v3.

The SNMP panel displays the SNMP agent and the list of SNMP managers running in hosts with which the array communicates.

In the FlashArray, the built-in SNMP agent has local knowledge of the array. The agent collects and organizes this array information and translates it via SNMP to or from the SNMP managers. The agent, named **localhost**, cannot be deleted or renamed. The managers are defined by creating SNMP manager objects on the array. The managers communicate with the agent via the standard TCP port 161, and they receive notifications on port 162.

In the FlashArray, the **localhost** SNMP agent has two functions, namely, responding to GET-type SNMP requests and transmitting alert messages.

The agent responds to GET-type SNMP requests made by the SNMP managers, returning values for an information block, such as `purePerformance`, or individual variables within the block, depending on the type of request issued. The variables supported are:

<code>pureArrayReadBandwidth</code>	Current array-to-host data transfer rate
<code>pureArrayWriteBandwidth</code>	Current host-to-array data transfer rate
<code>pureArrayReadIOPS</code>	Current read request execution rate
<code>pureArrayWriteIOPS</code>	Current write request execution rate
<code>pureArrayReadLatency</code>	Current average read request latency
<code>pureArrayWriteLatency</code>	Current average write request latency

The FlashArray Management Information Base (MIB) describes the `purePerformance` variables and can be downloaded from the array to your local machine.

SNMP managers are added to the array through the creation of SNMP manager objects. When creating an SNMP manager object, enter the Host, which represents the DNS hostname or IP address of the computer that hosts the SNMP manager. Also specify the SNMP version from the Version drop-down list. Valid versions are **v2c** and **v3**.

The SNMP agent generates and transmits messages to the SNMP manager as traps or inform requests (informs), depending on the notification type that is configured on the manager. An SNMP trap is an unacknowledged SNMP message, meaning the SNMP manager does not acknowledge receipt of the message. An SNMP inform is an acknowledged trap.

If the SNMP manager notification type is set to **trap**, the agent sends the SNMP message (trap) without expecting a response. If the SNMP manager is set to **inform**, the agent sends the SNMP message (inform) and waits for a reply from the manager confirming message retrieval. If the agent does not receive a response within a certain timeframe, it will retry until the inform has passed through successfully. If the notification type is not set, the manager defaults to **trap**.

SNMPv2 uses a type of password called a community string to authenticate the messages that are passed between the agent and manager. The community string is sent in clear text, which is considered an unsecured form of communication. SNMPv3, on the other hand, supports secure communication

between the agent and manager through the use of authentication and privacy encryption methods. As such, SNMPv2c and SNMPv3 have different security attributes.

To configure the SNMPv2c agent and managers, set the Community field to the community string under which the agent is to communicate with the managers. The agent and manager must belong to the same community; otherwise, the agent will not accept requests from the manager. When setting the community, Purity prompts twice for the community string. To remove the agent or manager from the community, leave the field blank.

To configure the SNMPv3 agent and managers, in the User field, specify the user ID that Purity uses to communicate with the SNMP manager. Also set the authentication and privacy encryption security levels for the agent and managers. SNMPv3 supports the following security levels:

- **noAuthNoPriv.** Authentication and privacy encryption is not set. Similar to SNMPv2c, communication between the SNMP agent and managers is not authenticated and not encrypted. noAuthNPriv security requires no configuration.

- **authNoPriv.** Authentication is set, but privacy encryption is not set. Communication between the SNMP agent and managers is authenticated but not encrypted. Password authentication is based on **MD5** or **SHA** hash authentication.

To configure authNoPriv security, in the Auth Protocol field, set the authentication protocol to **MD5** or **SHA**, and in the Auth Passphrase field, enter an authentication passphrase.

- **authPriv.** Communication between the SNMP agent and managers is authenticated and encrypted. Password authentication is based on **MD5** or **SHA** hash authentication. Traffic between the FlashArray and SNMP manager is encrypted using encryption protocol **AES** or **DES**.

To configure authPriv security, in the Auth Protocol field, set the authentication protocol to **MD5** or **SHA**, and in the Auth Passphrase field, enter an authentication passphrase. Also, in the Privacy Protocol field, set the privacy protocol to **AES5** or **DES**, and in the Privacy Passphrase field, enter a privacy passphrase.

Note that privacy cannot be configured without authentication.

Once an SNMP manager object is created on the array, the FlashArray immediately starts transmitting SNMP messages and alerts to the manager.

### Downloading the management information base (MIB) file

To download the management information base (MIB) file:

1. Select **Settings > System**.
2. In the SNMP panel, click the menu icon and select **Download MIB** to download the MIB file to your local machine. The default filename is PURESTORAGE-MIB.

### Specifying the SNMP community string (applies to SNMPv2c only)

Specifying the community string adds the array to the SNMP community. You must specify the SNMP community string if the SNMP agent is configured to use the SNMPv2c protocol.

To specify the SNMP community string:

1. Select **Settings > System**.

2. Click the Edit (pencil) icon next to the built-in **localhost** SNMP agent. The Edit SNMP Agent dialog box appears.
3. In the Community field, enter the manager community ID under which Purity is to communicate with the managers.
4. Click **Save**.

### Creating an SNMP manager object

Once an SNMP manager object is created on the array, the FlashArray immediately starts transmitting SNMP messages and alerts to the manager.

To create an SNMP manager object:

1. Select **Settings > System**.
2. In the SNMP panel, click the menu icon and select **Add SNMP Manager**. The Add SNMP Manager dialog box appears.
3. Complete the following fields:
  - **Name**: Name of the SNMP manager.
  - **Host**: DNS hostname or IP address of a computer that hosts an SNMP manager to which Purity is to send messages when it generates alerts. If specifying an IP address, enter the IPv4 or IPv6 address.

For IPv4, specify the IP address in the form **ddd.ddd.ddd.ddd**, where **ddd** is a number ranging from 0 to 255 representing a group of 8 bits. If a port number is also specified, append it to the end of the address in the format **ddd.ddd.ddd.ddd:PORT**, where **PORT** represents the port number.

For IPv6, specify the IP address in the form **xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx**, where **xxxx** is a hexadecimal number representing a group of 16 bits. Consecutive fields of zeros can be shortened by replacing the zeros with a double colon (**::**). If a port number is also specified, enclose the entire address in square brackets (**[ ]**) and append the port number to the end of the address. For example, **[xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx]:PORT**, where **PORT** represents the port number.
  - **SNMP Version**: Version of the SNMP protocol to be used by Purity in communications with the specified managers. Valid values are v2c (default) and v3.
  - **Community**: SNMPv2c only. SNMP manager community ID under which Purity is to communicate with the specified managers.
  - **User**: User ID recognized by the specified SNMP managers that Purity is to use in communications with them.
  - **Auth Protocol**: SNMPv3 only. Hash algorithm used to validate the authentication passphrase. Valid values are MD5, SHA, or None.

- **Auth Passphrase:** SNMPv3 only. Passphrase used by Purity to authenticate the array with the specified managers. Required if the Auth Protocol option is set to MD5 or SHA.
- **Privacy Protocol:** SNMPv3 only. Encryption protocol for SNMP messages. Valid values are AES, DES, or None.
- **Privacy Passphrase:** SNMPv3 only. Passphrase used to encrypt SNMP messages. The passphrase must be between 8 and 63 non-spaced ASCII characters.
- **Notification:** Notification type that determines whether the recipient (remote host) acknowledges receipt of SNMP messages. Valid options are **trap** and **inform**. An SNMP trap is an unacknowledged (asynchronous) SNMP message, meaning the recipient does not acknowledge receipt of the message. An SNMP inform request is an acknowledged trap. If not specified, the notification type defaults to **trap**.

4. Click **Save**.

### Configuring the SNMP manager object

To configure the SNMP manager object:

1. Select **Settings > System**.
2. In the SNMP panel, click the menu icon for the SNMP manager object and select **Edit**. The Edit SNMP Manager dialog box appears.
3. Modify the following fields:
  - **Name:** Name of the SNMP manager.
  - **Host:** DNS hostname or IP address of a computer that hosts an SNMP manager to which Purity is to send messages when it generates alerts.
  - **SNMP Version:** Version of the SNMP protocol to be used by Purity in communications with the specified managers. Valid values are v2c (default) and v3.
  - **Community:** SNMPv2c only. SNMP manager community ID under which Purity is to communicate with the specified managers.
  - **User:** User ID recognized by the specified SNMP managers that Purity is to use in communications with them.
  - **Auth Protocol:** SNMPv3 only. Hash algorithm used to validate the authentication passphrase. Valid values are MD5, SHA, or None.
  - **Auth Passphrase:** SNMPv3 only. Passphrase used by Purity to authenticate the array with the specified managers. Required if the Auth Protocol option is set to MD5 or SHA.
  - **Privacy Protocol:** SNMPv3 only. Encryption protocol for SNMP messages. Valid values are AES, DES, or None.
  - **Privacy Passphrase:** SNMPv3 only. Passphrase used to encrypt SNMP messages. The passphrase must be between 8 and 63 non-spaced ASCII characters.

- **Notification:** Notification type that determines whether the recipient (remote host) acknowledges receipt of SNMP messages. Valid options are `trap` and `inform`. An SNMP trap is an unacknowledged (asynchronous) SNMP message, meaning the recipient does not acknowledge receipt of the message. An SNMP inform request is an acknowledged trap. If not specified, the notification type defaults to `trap`.

4. Click **Save**.

#### Deleting an SNMP manager object

Deleting an SNMP manager object stops communication with the specified SNMP manager and deletes the SNMP manager object from Purity.

To delete an SNMP manager:

1. Select **Settings > System**.
2. In the SNMP panel, click the menu icon for the SNMP manager object and select **Delete**. The Delete SNMP Manager dialog box appears.
3. Click **Delete**.

#### Sending a test SNMP message to a manager

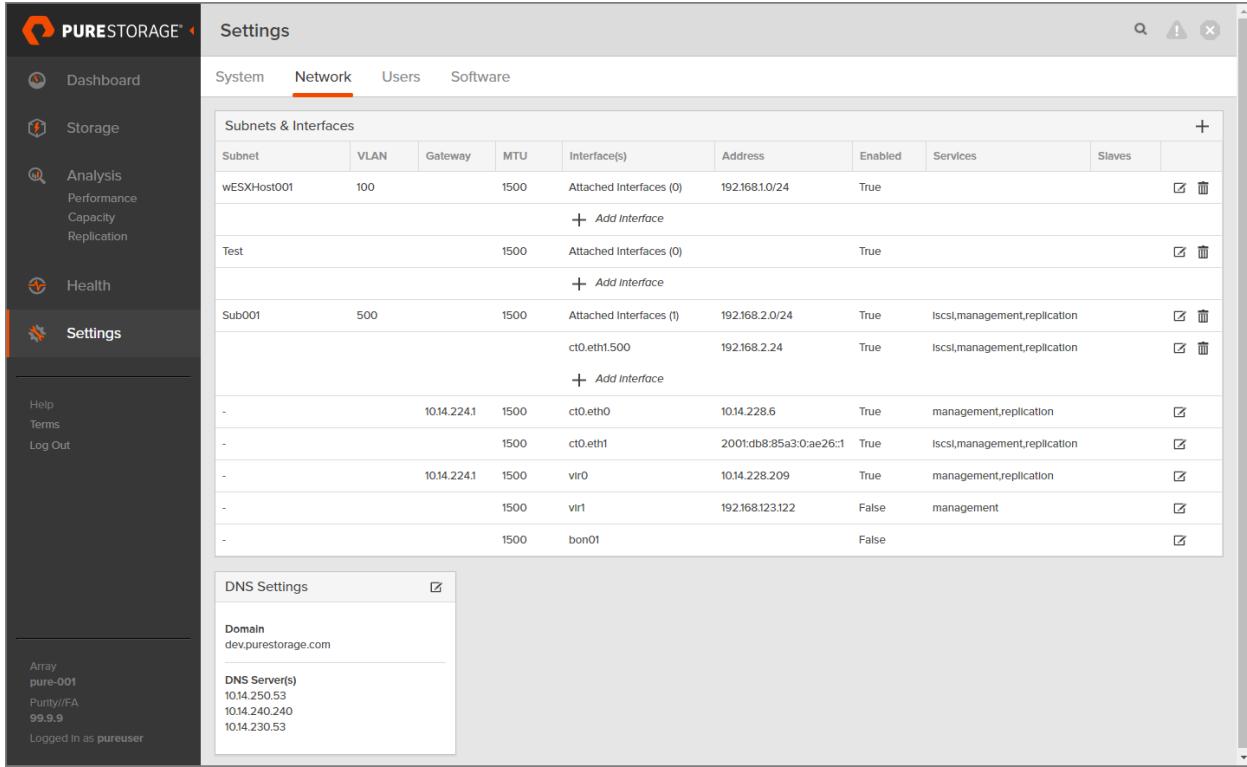
To send a test SNMP message to a manager:

1. Select **Settings > System**.
2. In the SNMP panel, click the menu icon for the SNMP manager object and select **Send Test Message**.

## Network

The Network page displays the network connection attributes of the array.

**Figure 9.8: Settings - Network Page**



The screenshot shows the PureStorage Settings - Network Page. The left sidebar includes links for Dashboard, Storage, Analysis, Health, and Settings. The Settings link is highlighted. The main content area has tabs for System, Network (which is selected), Users, and Software. The Subnets & Interfaces section lists subnets and their interfaces. The DNS Settings section shows domain and DNS server information.

Subnet	VLAN	Gateway	MTU	Interface(s)	Address	Enabled	Services	Slaves
wESXHost001	100		1500	Attached Interfaces (0)	192.168.1.0/24	True		
				+ Add Interface				
Test			1500	Attached Interfaces (0)		True		
				+ Add Interface				
Sub001	500		1500	Attached Interfaces (1)	192.168.2.0/24	True	Iscsi,management,replication	
				ct0.eth1.500	192.168.2.24	True	Iscsi,management,replication	
				+ Add Interface				
-		10.14.224.1	1500	ct0.eth0	10.14.228.6	True	management,replication	
-			1500	ct0.eth1	2001:db8:85a3:0:ae26:1	True	Iscsi,management,replication	
-		10.14.224.1	1500	vir0	10.14.228.209	True	management,replication	
-			1500	vir1	192.168.123.122	False	management	
-			1500	bon01		False		

**DNS Settings**

Domain: dev.purestorage.com

DNS Server(s):  
10.14.250.53  
10.14.240.240  
10.14.230.53

## Subnets & Interfaces

The Subnets & Interfaces panel manages the subnets and Ethernet (physical), virtual, bond, VLAN, and app interfaces used to connect the array to a network.

The Subnets & Interfaces panel displays a list of interfaces on the array, along with the following network connection attributes: interface status (enabled or disabled), interface IP address, netmask and gateway IP addresses, maximum transmission unit (MTU), and network service (iscsi, management, nvme-roce, or replication) that is attached to the interface.

If an interface belongs to a subnet, the subnet name appears in the Subnet column, and all of its interfaces are grouped with the subnet. A dash (-) in the Subnet column means the interface does not belong to a subnet. VLAN ID numbers are displayed for subnets that are configured with VLAN tagging.

A check mark in the Enabled column indicates that an interface or subnet is enabled. If a bond interface is disabled, all of its slave interfaces are also disabled. If a subnet is disabled, all of its interfaces, including ones that are individually enabled, are also disabled. If a subnet is enabled, only the enabled interfaces in the subnet are reachable; its disabled interfaces remain unreachable.

## Subnets

Interfaces with common attributes can be organized into subnetworks, or subnets, to enhance the efficiency of data (iSCSI or NVMe-RoCE), management, and replication traffic.

In Purity//FA, subnets can include physical, virtual, bond, and VLAN interfaces. Physical, virtual, and bond interfaces can belong to the same subnet. VLAN interfaces can only belong to subnets with other VLAN interfaces.

If the subnet is assigned a valid IP address, once it is created, all of its enabled interfaces are immediately available for connection. The subnet inherits the services from all of its interfaces. Likewise, the interfaces contained in the subnet inherit the netmask, gateway, MTU, and VLAN ID (if applicable) attributes from the subnet.

*Physical, virtual, and bond interfaces* in a subnet share common address, netmask, and MTU attributes. The subnet can contain a mix of physical, virtual, and bond interfaces, and the interface services can be of any type, such as iSCSI, management, NVMe-RoCE, or replication services.

Adding physical, virtual, and bond interfaces to a subnet involves the following steps:

1. Create a subnet.
2. Add the physical, virtual, and bond interfaces to the subnet.

A *VLAN interface* is a dedicated virtual network interface that is designed to be used with an organization's virtual local area network (VLAN). Through VLAN interfaces, Purity//FA employs VLAN tags to ensure the data passing between the array and VLANs is securely isolated and routed properly.

## VLAN Tagging

VLAN tagging allows customers to isolate traffic through multiple virtual local area networks (VLANs), ensuring data routes to and from the appropriate networks. The array performs the work of tagging and untagging the data that passes between the VLAN and array.

VLAN is only supported for the iSCSI service type, so before creating a VLAN interface, verify the iSCSI service is configured on the physical interface.

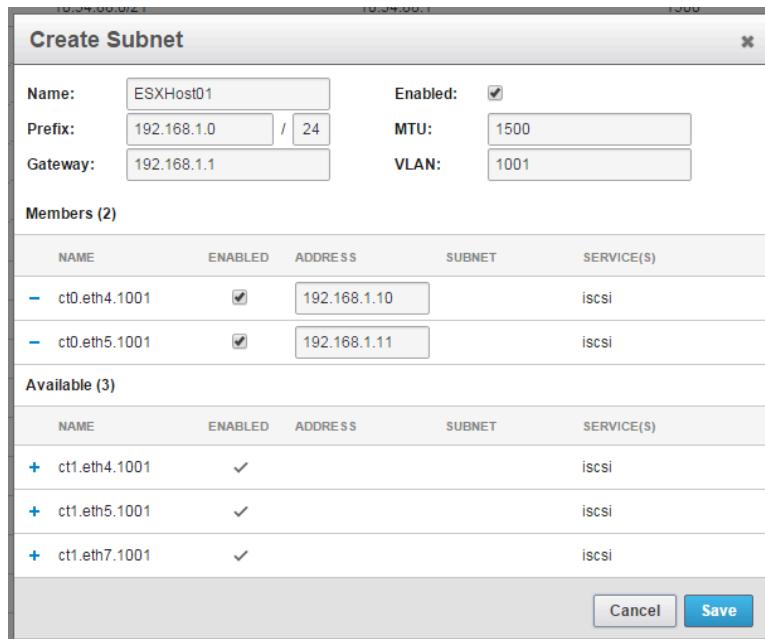
Creating and adding VLAN interfaces to a subnet involves the following steps:

1. Create a subnet, assigning a VLAN ID to the subnet.
2. Add one VLAN interface to the subnet for each corresponding physical network interface to be associated with the VLAN. All of the VLAN interfaces within the subnet must be in the same VLAN.

In Purity//FA, VLAN interfaces have the naming structure CTx.EThy.z, where x denotes the controller (0 or 1), y denotes the interface (0 or 1), and z denotes the VLAN ID number. For example, **ct0.eth1.500**.

### Figure 9.9: Networking - Creating a Subnet with VLAN Interfaces

In the following example, subnet **192.168.1.0/24** is being created. The subnet will be named **ESXHost01** and assigned VLAN ID **1001**. Physical interfaces **ct0.eth4** and **ct0.eth5** will then be added as VLAN interfaces to the subnet.

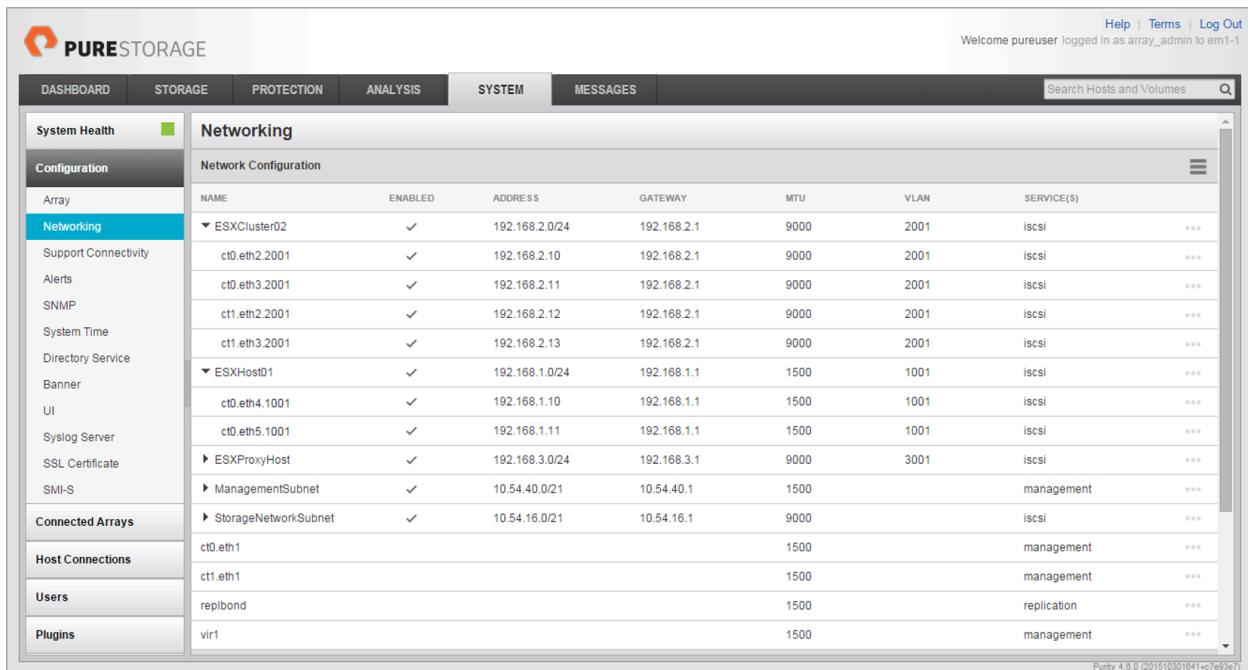


NAME	ENABLED	ADDRESS	SUBNET	SERVICE(S)
ct0.eth4.1001	<input checked="" type="checkbox"/>	192.168.1.10		iscsi
ct0.eth5.1001	<input checked="" type="checkbox"/>	192.168.1.11		iscsi

NAME	ENABLED	ADDRESS	SUBNET	SERVICE(S)
ct1.eth4.1001	<input checked="" type="checkbox"/>			iscsi
ct1.eth5.1001	<input checked="" type="checkbox"/>			iscsi
ct1.eth7.1001	<input checked="" type="checkbox"/>			iscsi

The new subnet details appear in the Subnets & Interfaces panel.



NAME	ENABLED	ADDRESS	GATEWAY	MTU	VLAN	SERVICE(S)
ESXCluster02	✓	192.168.2.0/24	192.168.2.1	9000	2001	iscsi
ct0.eth2.2001	✓	192.168.2.10	192.168.2.1	9000	2001	iscsi
ct0.eth3.2001	✓	192.168.2.11	192.168.2.1	9000	2001	iscsi
ct1.eth2.2001	✓	192.168.2.12	192.168.2.1	9000	2001	iscsi
ct1.eth3.2001	✓	192.168.2.13	192.168.2.1	9000	2001	iscsi
ESXHost01	✓	192.168.1.0/24	192.168.1.1	1500	1001	iscsi
ct0.eth4.1001	✓	192.168.1.10	192.168.1.1	1500	1001	iscsi
ct0.eth5.1001	✓	192.168.1.11	192.168.1.1	1500	1001	iscsi
ESXProxyHost	✓	192.168.3.0/24	192.168.3.1	9000	3001	iscsi
ManagementSubnet	✓	10.54.40.0/21	10.54.40.1	1500		management
StorageNetworkSubnet	✓	10.54.16.0/21	10.54.16.1	9000		iscsi
ct0.eth1				1500		management
ct1.eth1				1500		management
repbond				1500		replication
vir1				1500		management

## Changing the attributes of a network interface

You can change the IP address, netmask, gateway, and MTU attributes of physical, virtual, and bond interfaces. If the interface belongs to a subnet, you can only change the IP address. IPv4 and IPv6 addresses follow the addressing architecture set by the Internet Engineering Task Force.

To change the attributes of a physical, virtual, or bond interface:

1. Select **Settings > Network**.
2. In the Subnets & Interfaces panel, click the Edit Interface icon next the interface name. Select **Edit**. The Edit Network Interface dialog box appears.
3. Complete the following fields:
  - **Name:** Name of the network interface. The interface name cannot be changed.
  - **Enabled:** Indicates whether the network interface is enabled (blue) or disabled (gray).
  - **Address:** IP address to be associated with the specified Ethernet interface.
    - For IPv4, enter the address in CIDR notation **ddd.ddd.ddd.ddd/dd**. For example, **10.20.20.210/24**. Alternatively, specify the address **ddd.ddd.ddd.ddd**, and then specify the netmask in the Netmask field.
    - For IPv6, enter the address and prefix length in the form **xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx/xxx**.  
For example, **2620:125:9014:3224:14:227:196:0/64**. Consecutive fields of zeros can be shortened by replacing the zeros with a double colon (**::**). Alternatively, specify the address **xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx**, and then specify the prefix length in the Netmask field.
  - **Netmask:** Range of IP addresses that make up a group of IP addresses on the same network.
    - For IPv4, if the address entered is not in CIDR notation, enter the subnet mask in the form **ddd.ddd.ddd.ddd**. For example, **255.255.255.0**.
    - For IPv6, if the address entered did not include a prefix length, specify the prefix length. For example, **64**.
  - **Gateway:** IP address of the gateway through which the specified interface is to communicate with the network.
    - For IPv4, specify the gateway IP address in the form **ddd.ddd.ddd.ddd**.
    - For IPv6, specify the gateway IP address in the form **xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx**. Consecutive fields of zeros can be shortened by replacing the zeros with a double colon (**::**).
  - **MAC:** Unique media access control (MAC) address assigned to the network interface. This field cannot be modified.

- **MTU:** Maximum transmission unit (MTU) for the interface in bytes. If not specified, the MTU value defaults to 1500. If you are changing the MTU of a physical interface that is associated with a VLAN, verify the MTU of the physical interface is greater than or equal to (>=) the MTU of the VLAN interface. Note that the VLAN interface inherits the MTU value from its subnet.
  - **Service(s):** Services attached to the interface. For example, `iscsi`, `management`, `nvme-roce`, or `replication`. This field cannot be modified.
4. Click **Save**. Purity//FA restarts the GUI and signs you in using the self-signed certificate.

### Enabling or disabling a network interface

If a bond interface is disabled, all of its slave interfaces are also disabled.

To enable or disable a physical, virtual, or bond interface:

1. Select **Settings > Network**.
2. In the Subnets & Interfaces panel, click the Edit Interface icon for interface you want to enable or disable. The Edit Network Interface dialog box appears.
3. Click the Enabled toggle button to enable (blue) or disable (gray) the network interface.
4. Click **Save**.

### Creating a subnet

Creating the subnet involves setting the subnet attributes, and then adding the interfaces to the subnet.

A subnet can contain physical, virtual, and bond interfaces (for non-VLAN tagging purposes) or VLAN interfaces (for VLAN tagging purposes).

To create a subnet:

1. Select **Settings > Network**.
2. In the Subnets & Interfaces panel, click the Create Subnet icon in the upper-right corner of the panel. The Create Subnet dialog box appears.
3. Complete the following fields:
  - **Name:** Name of the subnet.
  - **Enabled:** Indicates whether the subnet is enabled (blue) or disabled (gray).
  - **Prefix:** IP address of the subnet prefix and prefix length (defaults to 24).
    - For IPv4, specify the prefix in the form `ddd.ddd.ddd.ddd/dd`.
    - For IPv6, specify the prefix in the form `xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx/xxxx`. Consecutive fields of zeros can be shortened by replacing the zeros with a double colon (`::`).
  - **VLAN:** For VLAN tagging, specify the VLAN ID, between 1 and 4094, to which the subnet is associated. If you specify the VLAN ID number, Purity//FA filters out all available physical

interfaces to only those set to iSCSI services. The physical interface name with the appended VLAN ID number becomes the VLAN interface name.

If the interface is not part of a VLAN, leave this field blank.

- **Gateway:** IP address of the gateway through which the specified interface is to communicate with the network.
  - For IPv4, specify the gateway IP address in the form **ddd.ddd.ddd.ddd**.
  - For IPv6, specify the gateway IP address in the form **xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx**. Consecutive fields of zeros can be shortened by replacing the zeros with a double colon (**::**).
- **MTU:** Maximum transmission unit (MTU) of the subnet. If not specified, the MTU value defaults to 1500. Interfaces inherit their MTU values from the subnet. Note that the MTU of a VLAN interface cannot exceed the MTU of the corresponding physical interface.

4. Click **Create**. After the subnet has been created, add interfaces to it.

### Enabling or disabling a subnet

If a subnet is disabled, all of its interfaces, including ones that are individually enabled, are also disabled. If a subnet is enabled, only the enabled interfaces in the subnet are reachable; its disabled interfaces remain unreachable.

To enable or disable a subnet:

1. Select **Settings > Network**.
2. In the Subnets & Interfaces panel, click the Edit Subnet icon for the subnet you want to enable or disable. The Edit Subnet dialog box appears.
3. Click the Enabled toggle button to enable (blue) or disable (gray) the network interface.
4. Click **Save**.

### Deleting a subnet

Deleting a subnet automatically removes all of the interfaces for the subnet and deletes the subnet. Any current connections through the subnet are disconnected.

To delete a subnet:

1. Select **Settings > Network**.
2. In the Subnets & Interfaces panel, click the Delete Subnet icon for the subnet you want to delete. The Delete Subnet dialog box appears notifying you that all interfaces in the subnet will be removed and the subnet will be deleted. When Purity//FA removes the interfaces, any current connections through the subnet will be disconnected.
3. Click **Save**. The interfaces appear in the subnet. If the subnet and added interfaces are enabled, they are immediately available for connection.

## DNS Settings

The DNS Settings panel manages the DNS domains that are configured for the array. Each DNS domain can include up to three static DNS server IP addresses. DHCP mode is not supported.

### Configuring the domain name system (DNS) server IP addresses

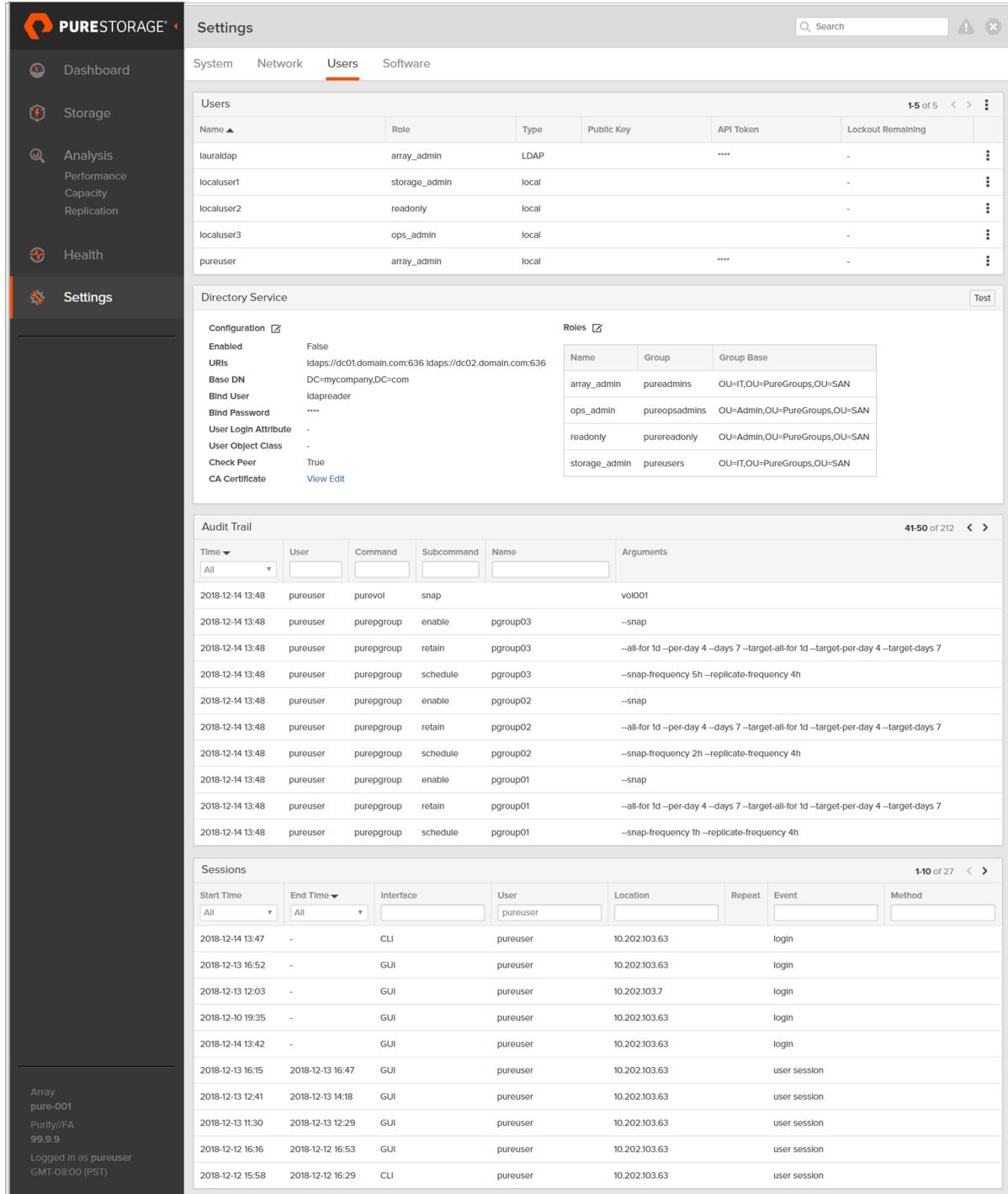
To configure the DNS server IP addresses:

1. Select **Settings > Network**.
2. In the DNS Settings panel, click the Edit DNS icon in the upper-right corner of the panel. The Edit DNS dialog box appears.
3. Complete the following fields:
  - **Domain**: Specify the domain suffix to be appended by the array when doing DNS lookups.
  - **DNS#**: Specify up to three DNS server IP addresses for Purity//FA to use to resolve hostnames to IP addresses. Enter one IP address in each DNS # field.
4. Click **Save**.

## Users

The Users page manages the Purity//FA user accounts and their attributes. The Users page also displays user details, such as audit trails and login activity.

**Figure 9.10: Settings - Users Page**



The screenshot shows the PureStorage Settings - Users page with the following sections:

- Users:** A table listing five users: lauraldap, localuser1, localuser2, localuser3, and pureuser, with columns for Name, Role, Type, Public Key, API Token, and Lockout Remaining.
- Directory Service:** Configuration settings and roles table.
 

Name	Group	Group Base
array_admin	pureadmins	OU=IT,OU=PureGroups,OU=SAN
ops_admin	pureopsadmins	OU=Admin,OU=PureGroups,OU=SAN
readonly	purereadonly	OU=Admin,OU=PureGroups,OU=SAN
storage_admin	pureusers	OU=IT,OU=PureGroups,OU=SAN
- Audit Trail:** A table showing a list of audit events for user pureuser, including Time, User, Command, Subcommand, Name, and Arguments.
- Sessions:** A table showing a list of sessions for user pureuser, including Start Time, End Time, Interface, User, Location, Repeat, Event, and Method.

At the bottom left, there is a sidebar with navigation links: Dashboard, Storage, Analysis, Performance, Capacity, Replication, Health, and Settings. The Settings link is highlighted with a red box. At the bottom right, there is a footer with the text "Logged in as pureuser" and "GMT-08:00 (PST)".

## Users

The Users panel displays a list of Purity//FA user accounts and their attributes. The Users panel displays the following types of users:

- **pureuser** administrative account.
- Local users that have been created on the array.
- LDAP users with a public key and/or API token. LDAP users that do not have a public key or API token do not appear in the list.

The FlashArray array is delivered with a single administrative account named **pureuser**. The account is password protected and may alternatively be accessed using a public-private key pair.

The **pureuser** account is set to the array administrator role, which has array-wide permissions. The **pureuser** account cannot be renamed or deleted.

Users can be added to the array either locally by creating and configuring a local user directly on the array, or through Lightweight Directory Access Protocol (LDAP) by integrating the array with a directory service, such as Active Directory or OpenLDAP. For more information about integrating the array with a directory service, refer to the Settings > Users > Directory Service section.

Local users can only be created by array administrators. The name of the local user must be unique. The local user name cannot be the same name as an LDAP user. If an LDAP user appears with the same name as a local user, the local user always has priority. The Type column of the Users panel identifies the way in which a user is added to the array as Local or LDAP.

Role-based access control (RBAC) restricts system access and capabilities to each user based on their assigned role in the array.

All users in the array, whether created locally or added to the array through LDAP integration, are assigned one of the following roles in the array:

- **Read-Only**. Users with the Read-Only (**readonly**) role can perform operations that convey the state of the array. Read Only users cannot alter the state of the array.
- **Ops Admin**. Users with the Ops Admin (**ops\_admin**) role can perform the same operations as Read Only users plus enable and disable remote assistance sessions. Ops Admin users cannot alter the state of the array.
- **Storage Admin**. Users with the Storage Admin (**storage\_admin**) role can perform the same operations as Read Only users plus storage related operations, such as administering volumes, hosts, and host groups. Storage Admin users cannot perform operations that deal with global and system configurations.
- **Array Admin**. Users with the Array Admin (**array\_admin**) role can perform the same operations as Storage Admin users plus array-wide changes dealing with global and system configurations. In other words, Array Admin users can perform all operations.

For local users, the role is set during user creation. For LDAP users, the role is set by configuring groups in the directory that correspond to the FlashArray user roles.

Each local user account on the array is password protected. The password is assigned during user creation and can be modified by array administrators. All local users can manage their own passwords, but only array administrators can manage the passwords of other users. Changing a local user's password requires knowledge of the current password. If the password of a local user is unknown, delete the account and recreate it with the desired password. Note that deleting a local user's account means deleting any public key associated with the user. If the password of the `pureuser` account is unknown, contact Pure Storage Support to reset the account to the default `pureuser` password. Passwords of LDAP users are managed in the directory service.

If a public key has been created for the user, it appears masked in the Public Key column. All users can manage their own public keys, but only array administrators can manage the public keys associated with other users.

If an API token has been created for the user, it appears masked in the API Token column. API tokens are used to securely create REST API sessions. After creating an API token, users can create REST API sessions and start sending requests. For more information about the Pure Storage REST API, refer to the REST API Reference Guide in the Pure1 Knowledge site at <https://support.purestorage.com>.

An API token is unique to the Purity//FA user for whom it was created. Once created, an API token is valid until it is deleted or recreated.

API token management does not affect Purity//FA user names and passwords. For example, deleting an API token does not invalidate the Purity//FA user name or password that was used to create the token. Likewise, changing the Purity//FA password does not affect the API token.

Single sign-on (SSO) gives LDAP users the ability to navigate seamlessly from Pure1 Manage to the current array through a single login. If single sign-on is not enabled on an array, users must manually log in with their credentials each time they navigate from Pure1 Manage to the array. Enabling and disabling single sign-on takes effect immediately. By default, single sign-on is not enabled.

Enabling single sign-on is a two-step process: first, configure single sign-on and LDAP integration through Pure1 Manage, and second, enable single sign-on on the array through Purity//FA. For more information about SSO and LDAP integration with Pure1 Manage, refer to the Pure1 Manage - SSO Integration article in the Pure1 Knowledge site at <https://support.purestorage.com>.

## Creating a local user

1. Select **Settings > Users**.
2. In the Users panel, click the Edit icon in the upper-right corner of the panel and select **Create User...** The Create User pop-up window appears.
3. In the User field, type the name of the new user. The name must be between 1 and 32 characters (alphanumeric and '-') in length and begin and end with a letter or number. The name must include at least one letter or '-'. All letters must be in lowercase.
4. In the Role field, select the role for the new user. Options include:
  - **Read-Only:** Users with the Read-Only (`readonly`) role can perform operations that convey the state of the array. Read Only users cannot alter the state of the array.

- **Ops Admin.** Users with the Ops Admin (`ops_admin`) role can perform the same operations as Read Only users plus enable and disable remote assistance sessions. Ops Admin users cannot alter the state of the array.
  - **Storage Admin.** Users with the Storage Admin (`storage_admin`) role can perform the same operations as Read Only users plus storage related operations, such as administering volumes, hosts, and host groups. Storage Admin users cannot perform operations that deal with global and system configurations.
  - **Array Admin.** Users with the Array Admin (`array_admin`) role can perform the same operations as Storage Admin users plus array-wide changes dealing with global and system configurations. In other words, Array Admin users can perform all operations.
5. In the **Password** field, type a password for the new user. The password must be between 1 and 100 characters in length, and can include any character that can be entered from a US keyboard.
  6. In the **Confirm Password** field, type the password again.
  7. Click **Create**.

#### Changing the login password of a local user

1. Select **Settings > Users**.
2. In the Users panel, click the Edit icon for the user you want to modify and select **Edit User...**.  
The Edit User pop-up window appears.
3. In the **Current Password** field, type the user's current password.
4. In the **New Password** field, type the user's new password. The password must be between 1 and 100 characters in length, and can include any character that can be entered from a US keyboard.
5. In the **Confirm New Password** field, type the new password again.
6. Click **Save**. The new password is required the next time the user logs in to Purity//FA.

#### Changing the role of a local user

1. Select **Settings > Users**.
2. In the Users panel, click the Edit icon for the user you want to modify and select **Edit User...**.  
The Edit User pop-up window appears.
3. In the **Role** field, select the role.
4. Click **Save**.

#### Deleting a local user

1. Select **Settings > Users**.

2. In the Users panel, click the Edit icon for the user you want to modify and select **Delete User...**.  
The Delete User pop-up window appears.
3. Click **Delete**.

#### Adding a public key

1. Select **Settings > Users**.
2. In the Users panel, click the Edit icon in the panel heading and select **Update Public Key....** The Update Public Key pop-up window appears.
3. In the User field, type the name of the local or LDAP user for which you want to create the public key.
4. If the user does not have an existing public key, enter the public key in the Public Key field. If the user already has a public key, select **Overwrite** and enter the public key.
5. Click **Save**.

#### Updating a public key

1. Select **Settings > Users**.
2. In the Users panel, click the Edit icon for the user you want to modify and select **Edit User....**  
The Edit User pop-up window appears.
3. In the Public Key field, select **Overwrite** and enter the public key.
4. Click **Save**.

#### Deleting a public key

1. Select **Settings > Users**.
2. In the Users panel, click the Edit icon for the user you want to modify and select **Edit User....**  
The Edit User pop-up window appears.
3. In the Public Key field, select **Remove**.
4. Click **Save**.
5. Click **Remove**.

#### Creating an API token

1. Select **Settings > Users**.
2. In the Users panel, click the Edit icon in the panel heading and select **Create API Token....** The Create API Token pop-up window appears.
3. In the User field, type the name of the local or LDAP user for which you want to create the API token.
4. To set an expiry date for the API token, in the Expires In field, specify the validity period of the API token. When an API token expires and is therefore no longer valid, the user cannot access

the REST API until the token is recreated. To create the API token without an expiry date, leave the Expires In field blank.

5. Click **Create**.

#### Recreating an API token

1. Select **Settings > Users**.
2. In the Users panel, click the Edit icon for the user you want to modify and select **Recreate API Token**.... The Recreate API Token pop-up window appears.
3. Click **Recreate**.

#### Removing an API token

1. Select **Settings > Users**.
2. In the Users panel, click the Edit icon for the user you want to modify and select **Remove API Token**.... The Remove API Token pop-up window appears.
3. Click **Remove**. Once the API token has been deleted, the user can no longer access the REST API.

#### Displaying the details of an API token

1. Select **Settings > Users**.
2. In the Users panel, click the Edit icon for the user you want to modify and select **Show API Token**.... The details for the API token, including token string, token creation date, and token expiry date, if any, appear.

#### Enabling single sign-on (SSO)

1. Verify that single sign-on and LDAP have been configured on Pure1 Manage. For more information about SSO and LDAP integration with Pure1 Manage, refer to the Pure1 Manage - SSO Integration article in the Pure1 Knowledge site <https://support.purestorage.com>.
2. Select **Settings > Users**.
3. In the Users panel, click the Edit icon in the panel heading and select **Enable Single Sign-on**.... The Enable Single Sign-on pop-up window appears.
4. Click **Enable**.

#### Disabling single sign-on (SSO)

1. Select **Settings > Users**.
2. In the Users panel, click the Edit icon in the panel heading and select **Disable Single Sign-on**.... The Disable Single Sign-on pop-up window appears.
3. Click **Disable**.

## Directory Service

The Directory Service panel manages the integration of FlashArray arrays with an existing directory service.

The Purity//FA release comes with a single local administrative account named "pureuser" with array-wide (`array_admin`) permissions. The account is password protected and may alternatively be accessed using a public-private key pair.

Additional users can be added to the array by creating and configuring local users directly on the array. For more information about local users, refer to the Settings > Users > Users section.

Users can also be added to the array through Lightweight Directory Access Protocol (LDAP) by integrating the array with an existing directory service. If a user is not found locally, the directory servers are queried. OpenLDAP and Microsoft's Active Directory (AD) are two implementations of LDAP that Purity//FA supports.

With LDAP integration, the array leverages the directory for authentication (validate user's password) and authorization (determine user's role in the array).

The Directory Service panel displays the settings for the directory service to be used for role-based access control.

The Configuration section of the Directory Service panel displays the details for the base configuration of the directory service, including its URLs, base DN, bind user name, and bind password. Configuring and then enabling the directory service allows users in the LDAP directory to log in to the array. If Check Peer is enabled, server authenticity using the CA certificate is enforced during the bind and query test. Note that you must set the CA certificate before you can enable Check Peer.

The Roles section of the Directory Service panel displays the current role-to-group configurations for the directory service. In order to log in to the array, a user must belong to a configured group in the LDAP directory, and that group must be mapped to an RBAC role in the array. The Group field represents the common name (CN) of the configured group that maps to the role in the array. The group name excludes the "CN=" specifier. For example, `purereadonly`. The Group Base field represents the common organizational unit (OU) under which to search for the group. The order of OUs gets smaller in scope from right to left. Multiple OUs are listed in comma-separated format.

The **Test** button in the upper-right corner of the Directory Service panel, when clicked, runs a series of tests to verify that the URLs can be resolved and that the array can bind and query the tree using the bind user credentials. The test also verifies that the array can find all the configured groups to ensure the common names and group base are correctly configured. The test can be run at any time.

## Users

An LDAP user is an individual in the LDAP directory.

LDAP users log in to the array via `ssh` by entering the following information, where `<user_name>` represents the sAMAccountName for Active Directory or uid for OpenLDAP, and `<array_name>` represents the name or the IPv4 or IPv6 address of the FlashArray array:

```
<user_name>@<array_name>
```

For IPv4, specify the IP address in the form `ddd.ddd.ddd.ddd`, where `ddd` is a number ranging from 0 to 255 representing a group of 8 bits.

For IPv6, specify the IP address in the form `[xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx]`, where `xxxx` is a hexadecimal number representing a group of 16 bits. Enclose the entire address in square brackets (`[]`). Consecutive fields of zeros can be shortened by replacing the zeros with a double colon (`::`).

For directory service enabled accounts, user passwords to the array are managed through the directory service, while public keys are configured through Purity//FA.

Accounts with user names that conflict with local accounts will not be authenticated against the directory. These account names include, but are not limited to: `pureuser`, `os76`, `root`, `daemon`, `sys`, `man`, `mail`, `news`, `proxy`, `backup`, `nobody`, `syslog`, `mysql`, `ntp`, `avahi`, `postfix`, `sshd`, `snmp`.

If an LDAP user has the same name as a locally created user, the locally created user always has priority.

Users with disabled accounts will not have access to the array.

## Groups

A group in the LDAP directory consists of users who share a common purpose.

Each configured group in the directory has a unique distinguished name (DN) representing the entire path of the object's location in the directory tree. The DN is comprised of the following attribute-value pairs:

- DC - Domain component base of the DN. For example, `DC=mycompany,DC=com`.
- OU - Organizational unit base of the group. For example, `OU=PureGroups,OU=SAN,OU=IT`.
- CN - Common name of the groups themselves. For example, `CN=purereadonly`.

For example, `CN=purereadonly,OU=PureGroups,OU=SAN,OU=IT,DC=mycompany,DC=com` is the DN for configured group `purereadonly` at group base `OU=PureGroups,OU=SAN,OU=IT` and with base DN `DC=mycompany,DC=com`.

The DN can contain multiple DC and OU attributes.

OUs are nested, getting more specific in purpose with each nested OU.

For OpenLDAP, for group configurations based on the `shadowAccount` class, groups must have the full DN of members in the `member` attribute. For group configurations based on the `posixAccount` class, groups must have the `uid` of members in the `memberUid` attribute.

When a user who is a member of a configured group logs in to the array, only the CLI actions that the user has permission to execute will be visible. Similarly, in the GUI, actions the user does not have permission to execute will be grayed out or disabled.

For Active Directory, two types of groups are supported: security groups and distribution groups. Distribution groups are used only with email applications to distribute messages to collections of users. Distribution groups are not security enabled. Security groups assign access to resources on your network. All groups configured on the array must be security groups.

## Role-Based Access Control

Role-based access control (RBAC) restricts the system access and capabilities of each user based on their assigned role in the array.

All users in the array, whether created locally or added to the array through LDAP integration, are assigned one of the following roles in the array:

- **Read Only.** Users with the Read-Only (`readonly`) role can perform operations that convey the state of the array. Read Only users cannot alter the state of the array.
- **Ops Admin.** Users with the Ops Admin (`ops_admin`) role can perform the same operations as Read Only users plus enable and disable remote assistance sessions. Ops Admin users cannot alter the state of the array.
- **Storage Admin.** Users with the Storage Admin (`storage_admin`) role can perform the same operations as Read Only users plus storage related operations, such as administering volumes, hosts, and host groups. Storage Admin users cannot perform operations that deal with global and system configurations.
- **Array Admin.** Users with the Array Admin (`array_admin`) role can perform the same operations as Storage Admin users plus array-wide changes dealing with global and system configurations. In other words, Array Admin users can perform all operations.

For LDAP users, role-based access control is achieved by configuring the groups in the LDAP directory to correspond to the different roles in the array. For example, a group named "purereadonly" in the directory might correspond to the `readonly` role in the array.

For security purposes, each user should be assigned to only one role in the array. If a user belongs to multiple configured groups that map to different roles in the array, modify the LDAP directory to ensure that the user belongs to only one group. If a user has multiple roles, one of which includes the `ops_admin` role, the user will be locked out of the system and an alert will be sent to all alert recipients. Modify the LDAP directory to ensure that the user has only one user role assigned. If a user has multiple roles, none of which include the `ops_admin` role, the user will have privileges corresponding to the least privileged group. For example, a user who has both the `readonly` and `array_admin` roles will have read-only privileges.

## Directory Service Configuration

Configuring the Pure Storage directory service requires a URI, a base DN, a bind user, a bind password, and at least one group within the LDAP directory that corresponds to a role in the array.

Before you start the configuration process, note the DN of each group within the directory server. Each component of the DN will be used to configure Pure Storage directory service. If you plan to enable Check Peer, also have the CA certificate available.

When you configure the array to integrate with a directory service, consider the following:

- If the directory service contains multiple groups, each group must have a common name (CN).
- All uniform resource identifiers (URIs) must be in the same, single domain.

To configure the Pure Storage directory service:

1. Configure the CA certificate. This is only required if Check Peer is going to be enabled.
2. Configure the base directory service settings, including the URIs, base DN, bind user name, and bind password. Optionally enable Check Peer.
3. Configure the directory service roles to map each role in the array to the appropriate LDAP group in the directory tree.
4. Test the directory service settings.
5. Enable the directory service. This allows users in the LDAP directory to log in to the array.

Disable the directory service at any time to stop all users in the directory server from logging in to the array.

## Configuring the directory service

1. Select **Settings > Users**.
2. In the Directory Service panel, click the Configuration edit icon. The Edit Directory Service Configuration dialog box appears.
3. In the Edit Directory Service Configuration dialog box, complete or modify the following fields:

### **Enabled:**

Click the toggle button to enable (blue) the directory service. Enable the directory service after you have configured the directory service, configured the roles, and tested the directory service configuration.

### **URI:**

Enter the comma-separated list of up to 30 URIs of the directory servers.

Each URI must include the scheme **ldap://** or **ldaps://** (for LDAP over SSL), a hostname, and a domain name or IP address. For example, **ldap://ad.company.com** configures the directory service with the hostname "ad" in the domain "company.com" while specifying the unencrypted LDAP protocol.

If specifying a domain name, it should be resolvable by the configured DNS servers.

If specifying an IP address, for IPv4, specify the IP address in the form **ddd.ddd.ddd.ddd**, where **ddd** is a number ranging from 0 to 255 representing a group of 8 bits.

For IPv6, specify the IP address in the form

**[xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx]**, where **xxxx** is a hexadecimal number representing a group of 16 bits. Enclose the entire address in square brackets (**[]**). Consecutive fields of zeros can be shortened by replacing the zeros with a double colon (**::**).

If the scheme of the URIs is **ldaps://**, SSL is enabled. SSL is either enabled or disabled globally, so the scheme of all supplied URIs must be the same. They must also all have the same domain.

If base DN is not configured and a URI is provided, the base DN will automatically default to the domain components of the URIs.



Optionally specify a port. Append the port number after the end of the entire address. Default ports are 389 for ldap, and 636 for ldaps. Non-standard ports can be specified in the URI if they are in use.

**Base DN:**

Enter the base distinguished name (DN) of the directory service. The Base DN is built from the domain and should consist only of domain components (DCs). For example, for `ldap://ad.storage.company.com`, the Base DN would be: "DC=storage,DC=company,DC=com"

**Bind User:**

Enter the username for the account that is used to perform directory lookups.

For Active Directory, enter the user name - often referred to as sAMAccountName or User Logon Name - for the account that is used to perform directory lookups. The user name cannot contain the characters " [ ] : ; | = + \* ? < > / \, and cannot exceed 20 characters in length.

For OpenLDAP, enter the full DN of the user. For example, "CN=John,OU=Users,DC=example,DC=com".

The bind account must be configured to allow the array to read the directory. It is good practice for this account to not be tied to any actual person and to have different password restrictions, such as "password never expires". The bind account should also not be a privileged account, since only read access to the directory is required.

**Bind Password:**

Enter the password for the bind user account. The password appears in masked form.

**Check Peer:**

Optionally click the toggle button to enable (blue) Check Peer. If Check Peer is enabled, Purity//FA validates the authenticity of the directory servers using the CA Certificate. If you enable Check Peer, you must provide a CA Certificate.

4. Click **Save**.

## Configuring the CA Certificate

1. Select **Settings > Users**.
2. In the Directory Service panel, click **Edit** next to CA Certificate. The Edit CA Certificate dialog box appears.
3. In the Edit CA Certificate dialog box, enter the certificate of the issuing certificate authority. Only one certificate can be configured at a time, so the same certificate authority should be the issuer of all directory server certificates.

The certificate must be PEM formatted (Base64 encoded) and include the "-----BEGIN CERTIFICATE-----" and "-----END CERTIFICATE-----" lines. The certificate cannot exceed 3000 characters in total length.

4. Click **Save**.

## Configuring the directory service roles

1. Select **Settings > Users**.
2. In the Directory Service panel, click the Roles edit icon. The Edit Directory Service Roles dialog box appears.
3. In the Edit Directory Service Roles dialog box, complete or modify the following fields:

### **Group:**

Enter the common name (CN) of the configured group that maps to the role in the array. The group name should be just the common name of the group without the "CN=" specifier. For example, `purereadonly`.

### **Group Base:**

Enter the common organizational unit (OU) under which to search for the group. Specify "OU=" for each organizational unit. The order of OUs should get smaller in scope from right to left. List multiple OUs in comma-separated format.

4. Click **Save**.

## Testing the directory service settings

1. Select **Settings > Users**.
2. In the Directory Service panel, click **Test**. The Test <Directory Service> Configuration pop-up window appears, displaying the output of the test.

During the directory service test, Purity//FA tests the directory service configuration to verify that the URLs can be resolved and that the directory service can successfully bind and query the tree using the bind user credentials.

If the test passes, enable the directory service.

## Audit Trail

The audit trail represents a chronological history of the Purity//FA GUI, Purity//FA CLI, or REST API operations that a user has performed to modify the configuration of the array. Each record within an audit trail includes the date and time the operation was performed, the name of the Purity//FA user who performed the operation and the Purity//FA operation that was performed.

Purity//FA creates audit records for operations that modify the configuration of the array (e.g., creating, modifying, deleting, or connecting volumes).

By default, all audit records on the array are displayed. To display a list of audit records that were created within a certain time range, click the All Time drop-down button and select the desired time range from the list.

Purity//FA does not flag audit records. Users can, however, manually flag audit records for internal tracking purposes. Each audit record includes the following information: the UTC time of the operation, the Purity//FA user who performed the operation, the Purity//FA command and subcommand that was performed, the object name against which the command was performed, and the arguments that were included in the command.

In addition to the Audit Trail panel, audit records are also logged and transmitted to Pure Storage Support via the phonehome facility. If SNMP managers are configured, Purity also sends alert messages as SNMP traps or informs to designated SNMP managers and as syslog messages to remote servers.

## Sessions

The Sessions panel displays a list of user session events performed through the Pure Storage Purity//FA GUI, Purity//FA CLI, and Pure Storage REST API interfaces.

User session events are divided into two main categories: session login and logout actions, and session authentication actions.

Login and logout actions include:

- Logging in to the Purity//FA GUI or Purity//FA CLI. This includes remote logins to the Purity//FA CLI via SSH.
- Logging out of the Purity//FA GUI or Purity//FA CLI
- Opening a Pure Storage REST API session
- Pure Storage REST API session timeouts

Authentication actions include:

- Generating an API token through the REST API
- Submitting a REST API request in a closed REST session
- Attempting to log in to the Purity//FA GUI or Purity//FA CLI using an invalid password
- Attempting to open a Pure Storage REST API session using an invalid API token
- Attempting to obtain a REST API token using an invalid user name and/or password

The Location column displays the IP address of the user client connecting to the array.

The Method column displays the authentication method by which the user attempted to log in, log out, or authenticate. Authentication methods include **API token**, **password**, and **public key**.

By default, all user session events on the array are displayed. To display a list of user session events that were performed within a certain time range, click the All Time drop-down button and select the desired time range from the list.

In addition to the Sessions panel, user session messages are also logged and transmitted to Pure Storage Support via the phonehome facility. If configured, Purity//FA can also send user session messages as syslog messages to remote servers.

### Login and Logout Events

When a user logs in to the Purity//FA GUI or Purity//FA CLI, the event appears as an 'login' event in the Sessions panel, where the Start Time represents the user login time. The End Time value appears as a dash ("") symbol until the user logs out.

When the user logs out of the Purity//FA interface, the same login event becomes a closed session, where the End Time represents the user logout time. The login event message ID is replaced with the logout event message ID. To conserve space, Purity//FA stores a reasonable number of log entries. Older entries are deleted from the log as new entries are added. If the matching start time log entry is no longer stored in the log when the user logs out of the Purity//FA interface, the Start Time value appears as a dash ("–") symbol.

#### Login and Logout Event Examples

The following examples represent common login and logout events:

##### Example 1

**Figure 9.11: User Session Logs - Login and Logout Example**

`pureuser` logs in to the Purity//FA GUI with a valid password

START TIME	END TIME	INTERFACE	USER	LOCATION	REPEATS	EVENT	METHOD
2014-09-17 12:54:49	–	GUI	pureuser	10.123.10.196		login	password

##### Example 2

**Figure 9.12: User Session Logs - Login and Logout Example**

The 'root' user logs in to the Purity//FA CLI with a valid public key and then logs out

START TIME	END TIME	INTERFACE	USER	LOCATION	REPEATS	EVENT	METHOD
2014-09-09 15:52:13	2014-09-09 15:55:54	CLI	root	10.123.10.103		user session	public key

##### Example 3

**Figure 9.13: User Session Logs - Login and Logout Example**

`pureuser` opens a REST API session with a valid API token and then the session times out.

START TIME	END TIME	INTERFACE	USER	LOCATION	REPEATS	EVENT	METHOD
2014-08-14 16:09:24	2014-08-14 16:09:24	REST	pureuser	10.124.190.231		user session	API token

## Authentication Events

Users can log into Purity//FA through various authentication methods, including passwords, public keys, and API tokens.

Purity//FA creates a “failed authentication” event when a user performs any of the following actions: log in to the Purity//FA GUI with an incorrect password, log in to the Purity//FA CLI with an invalid password or public key, or open a REST API session with an invalid API token.

Purity//FA creates an “API token obtained” event when a user attempts to create an API token via any of the Purity//FA interfaces.

Purity//FA creates a “request without session” event when a user attempts to submit a REST API request as an unauthenticated user.

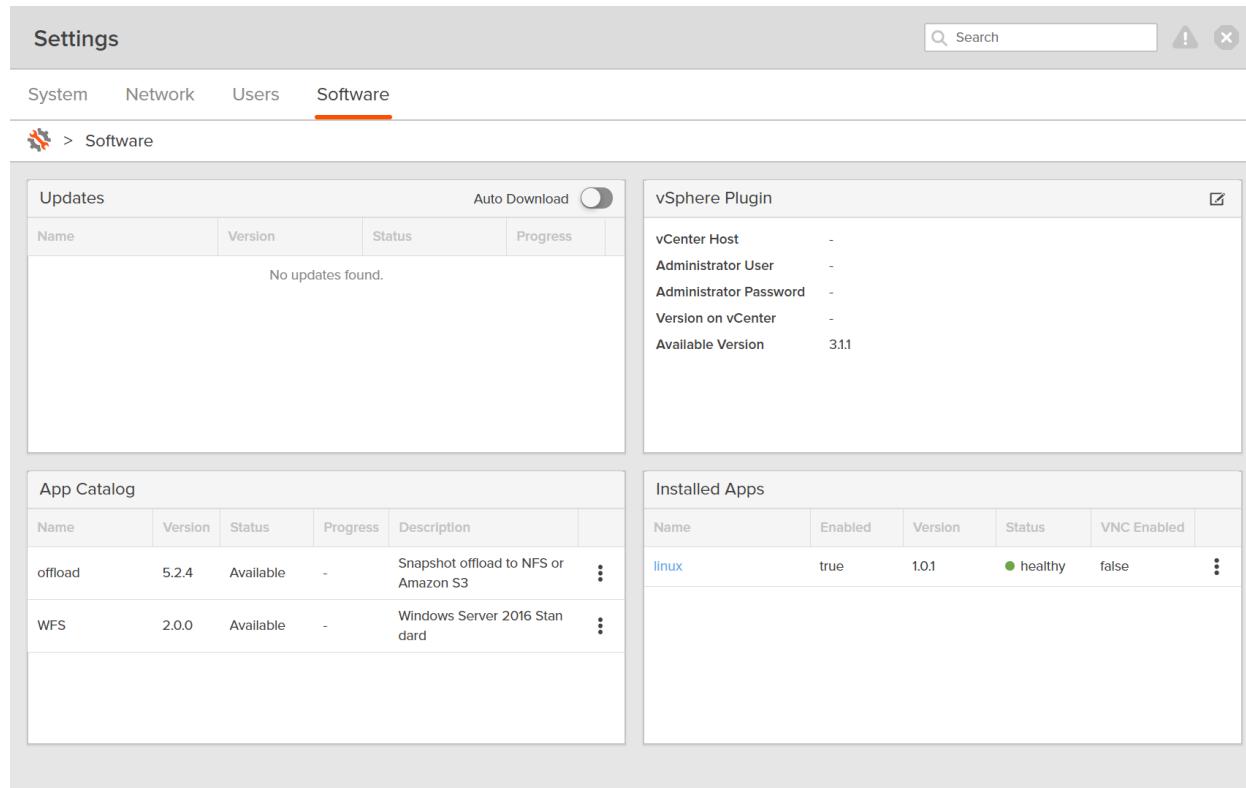
In the Sessions panel, repeated failed authentication attempts are displayed in pre-configured time periods. By default, failed authentication attempts are displayed in 15-minute time periods.

The Repeat value represents the number of attempts *in addition* to an initial attempt that a user has performed an authentication action within the 15-minute time period.

## Software

The Software page manages software, apps and third party plug-ins associated with the array.

**Figure 9.14: Settings - Software Page**



The Software page interface includes a navigation bar with tabs for System, Network, Users, and Software (which is selected). A search bar and a help icon are also present.

- Updates:** Displays a table for software updates. The table columns are Name, Version, Status, Progress, and Description. It shows "No updates found."
- vSphere Plugin:** Displays configuration for the vSphere Host. It includes fields for vCenter Host, Administrator User, Administrator Password, Version on vCenter, and Available Version (3.1.1).
- App Catalog:** Displays a table of available applications. The table columns are Name, Version, Status, Progress, Description, and three vertical ellipsis menu icons. Applications listed are offload (Version 5.2.4) and WFS (Version 2.0.0).
- Installed Apps:** Displays a table of installed applications. The table columns are Name, Enabled, Version, Status, VNC Enabled, and three vertical ellipsis menu icons. An application named "linux" is listed with status "healthy".

## Updates

The Updates panel displays a list of software updates. Software updates add or enhance Purity features and functionality. Perform periodic software updates to get the most out of your Purity system.

To perform a software update, contact Pure Storage Support to obtain the software installation files required for the update.

The Auto Download toggle icon enables (blue) or disables (gray) the Auto Download feature. If Auto Download is enabled, any software installation files that Pure Storage Support sends to the array will be automatically downloaded and ready to install. If Auto Download is disabled, the software installation files that Pure Storage Support sends to the array will only be downloaded during the software update process. Auto Download is disabled by default. Note that the Auto Download feature impacts software updates only. The Auto Download feature does not impact Purity apps or third party plug-ins.

A software version that is available for update will have one of the following statuses:

- **Available:** A software update for this version is available, but the installation files have not been downloaded to the array. Instead, the files will be downloaded to the array during the installation process. When scheduling the software update, make sure to factor enough time for the download process.
- **Downloaded:** The installation files for this software version have been successfully downloaded to the array.

Click **Install** to start the software update process. As the software update process progresses, the following statuses will appear:

- **Downloading:** Purity is downloading the installation files to the array for this software version. If Auto Download is enabled, any software installation files that Pure Storage Support sends to the array will be automatically downloaded and ready to install. If Auto Download is disabled, the software installation files that Pure Storage Support sends to the array will only be downloaded during the software update process. Auto Download is disabled by default.
- **Installing:** Purity is updating the software. Be prepared to be logged out of the software during the update process.

During the update process, you will be logged out of the software. After you have been logged out, log back in to continue monitoring the process. The software update process is complete when the software update no longer appears in the Updates panel.

If the software update fails, the software reverts to the previous version. If you encounter any problems during the update process, contact Pure Storage Support.

### Enabling and disabling Auto Download

You cannot change the Auto Download status while a software update is in progress.

To enable or disable Auto Download, select one of the following options:

- Click the Auto Download toggle button to enable (blue) automatic download. When the software update files are available, they will be automatically downloaded to the array.
- To disable Auto Download, click the Auto Download toggle button (gray).

### Performing a software update

1. Select **Settings > Software**.
2. In the Updates panel, click **Install** to start the software update process.

You will be logged out of the software during the update process. After you have been logged out, log back in to continue monitoring the update process.

The update is complete once the progress bar has reached 100%.

## vSphere Plugin

The Pure Storage Management Plugin for vSphere extends the vSphere Web Client, enabling users to manage Pure Storage FlashArray volumes and snapshots in a vCenter context.

The vSphere Plugin panel displays the connection details for the vSphere Web Client. Once a connection has been established, users can open Purity//FA GUI sessions via the vSphere Web Client.

For more information about the vSphere plugin, refer to the Pure Storage Management Plugin for vSphere User Guide in the Pure1 Knowledge site at <https://support.purestorage.com>.

## App Catalog

The Purity Run platform extends array functionality by integrating add-on services into the Purity//FA operating system. Each service that runs on the platform is provided by an app.

The App Catalog panel displays a list of apps that are available to be installed on the array, along with the following attributes for each app:

- **Name:** App name. The app name is pre-assigned and cannot be changed.
- **Version:** App version that is ready to be installed on the array.
- **Status:** Status of the app installation. Possible app statuses include:
  - **Available:** App (new or upgraded version of an existing one) is available to be installed.
  - **Downloading:** App installation files are being downloaded in preparation for an installation.
  - **Downloaded:** App installation files have been successfully downloaded. Installation will begin shortly.
  - **Installing:** App is currently being installed.
  - **Uninstalling:** App is currently being uninstalled.
  - **Aborted:** App installation process has encountered issues. The installation is rolling back. If the app is no longer available, it will not reappear in the list; otherwise, it will eventually return to its previous "Available" status. If the app is available, try the installation again. If you continue to encounter issues, contact Pure Storage Support.
- **Progress:** Download progress during the installation process.
- **Description:** Description of the app.

Apps require CPU, memory, network, and storage resources. For this reason, apps by default are not installed.

To install an app, click the menu icon next to the app and select **Install**. After an app has been installed, it appears in the Installed Apps panel.

### Installing an app

1. Select **Settings > Software**.
2. In the App Catalog panel, click the menu icon and select **Install**. The Install App dialog box appears.  
Click **Install**.

## Installed Apps

The Installed Apps panel displays a list of apps that are installed on the array, along with the following attributes for each app:

- **Name:** App name. The app name is pre-assigned and cannot be changed.
- **Enabled:** App enable/disable status. An app must be enabled so the array can reach the app service. Apps are disabled by default.
- **Version:** App version that is currently installed on the array.
- **Status:** App status. A status of **healthy** means the app is running. A status of **unhealthy** means the app is not running.

There are various factors that contribute to an unhealthy app. In most cases, the unhealthy status is temporary, such as when the app is being restarted; upon successful restart, the app returns to healthy status. The app might also be unhealthy if, upon enabling the app, Purity//FA determines that there are insufficient resources to run it. An accompanying message appears in the Details column stating that there are insufficient resources to operate the app. Disable any apps that are currently not in use to free up some resources and try to enable the app again.

If the app is in an unhealthy status for a longer than expected period of time, contact Pure Storage Support.

- **VNC Enabled:** Indicates whether VNC access is enabled (**true**) or disabled (**false**) for each installed app. The default is **false**. When VNC Enabled is **true**, a port is open to allow VNC connections.

Note that if an app migrates between controllers, it briefly stops and restarts.

## App Volumes

For each app that is installed, a boot volume is created. For some apps, a data volume is also created. Boot and data volumes are known as app volumes.

Select **Storage > Volumes** to see a list of volumes, including app volumes.

Boot and data app volume names begin with a distinctive @ symbol. The naming convention for app volumes is **@APP\_boot** for boot volumes and **@APP\_data** for data volumes, where **APP** denotes the app name.

App volumes are connected to their associated app host. For example, the **linux** boot and data volumes are connected to the **linux** app host. From the list of volumes, click an app volume to see its associated app host.

The boot volume represents a copy of the boot drive of the app. Do not modify or save data to the boot volume. When an app is upgraded, the boot volume is overwritten, completely destroying its contents including any other data that is saved to it. The data volume is used by the app to store data.

The following example shows that the drives were correctly mounted inside the **linux** app.

```
pureuser@linux:~$ df
Filesystem      1K-blocks      Used      Available Use%  Mounted on
```

<code>udev</code>	<code>8198768</code>	<code>0</code>	<code>8198768</code>	<code>0%</code>	<code>/dev</code>
<code>tmpfs</code>	<code>1643272</code>	<code>8756</code>	<code>1634516</code>	<code>1%</code>	<code>/run</code>
<code>/dev/sda1</code>	<code>15348720</code>	<code>1721392</code>	<code>12824616</code>	<code>12%</code>	<code>/</code>
<code>/dev/sdb</code>	<code>17177782208</code>	<code>33608</code>	<code>17177748600</code>	<code>1%</code>	<code>/data</code>

Disk device `/dev/sdb`, which corresponds to the app data volume, is mounted on `/data`, meaning the data will be saved to the data volume (and not the boot volume), and disk device `/dev/sda1`, which corresponds to the app boot volume, is mounted on `/`.

## App Hosts

Each app has a dedicated host, known as an app host. The app host is connected to the associated boot and data volumes. The app host is also used to connect FlashArray volumes to the app.

Select **Storage > Hosts** to see a list of hosts, including app hosts.

Unlike regular FlashArray hosts, app hosts cannot be deleted, renamed, or modified in any way. Furthermore, app hosts cannot be added to host groups or protection groups.

App host names begin with a distinctive @ symbol. The naming convention for app hosts is `@APP`, where `APP` denotes the app name.

## Connecting FlashArray Volumes to an App

FlashArray volumes are connected to apps via the app host. The volumes are connected to the app hosts in the same way that they are connected to regular FlashArray hosts.

A FlashArray volume can only be connected to one app host at a time. Furthermore, the FlashArray volume cannot be connected to other hosts or host groups while it is connected to an app host.

After a FlashArray volume has been connected to an app host, rescan the SCSI bus to ensure the newly-connected volumes are visible from inside the app.

The following example displays five FlashArray volumes (and their target LUNs) as SCSI devices from inside the `linux` app, ready to be mounted.

```
pureuser@linux:~$ cat /proc/scsi/scsi
Attached devices:
Host: scsi2 Channel: 00 Id: 01 Lun: 03
        Vendor: PURE      Model: FlashArray      Rev: 9999
        Type: Direct-Access      ANSI  SCSI revision: 06
Host: scsi2 Channel: 00 Id: 01 Lun: 04
        Vendor: PURE      Model: FlashArray      Rev: 9999
        Type: Direct-Access      ANSI  SCSI revision: 06
Host: scsi2 Channel: 00 Id: 01 Lun: 05
        Vendor: PURE      Model: FlashArray      Rev: 9999
        Type: Direct-Access      ANSI  SCSI revision: 06
Host: scsi2 Channel: 00 Id: 01 Lun: 06
        Vendor: PURE      Model: FlashArray      Rev: 9999
        Type: Direct-Access      ANSI  SCSI revision: 06
Host: scsi2 Channel: 00 Id: 01 Lun: 07
        Vendor: PURE      Model: FlashArray      Rev: 9999
```

Type: Direct-Access

ANSI SCSI revision: 06

## App Interfaces

For each app that is installed, one app management interface is created per array management interface. An app data interface may also be created for high-speed data transfers.

Select **Settings > Network** to view and configure app interfaces.

The naming convention for app interfaces is **APP.data** for the app data interface, and **APP.mgmt** for the app management interface, where **APP** denotes the app name, and **y** denotes the interface.

Configure an app interface to give **pureuser** the ability to log into the app or transfer data through a separate interface. Configuring an app interface involves assigning an IP address to the interface and then enabling the interface.

Optionally set the gateway. Note that only one of the app interfaces of a particular app can have a gateway set.

Before you configure an app interface, make sure the corresponding external interface is physically connected.

Configure one or more of the following app interfaces:

- **App Management Interface**

Configure the app management interface to give **pureuser** the ability to log into the app with the same Purity//FA login credentials. If a public key has been created for the user, it can be used to log into the app. Purity//FA password changes are automatically applied to the app.

To configure the app management interface, assign an IP address to one of the app management interfaces, and then enable the interface.

- **App Data Interface**

Configure the app data interface to use a separate interface for high-speed data transfers.

To configure the app data interface, assign an IP address to the app data interface, and then enable the interface.

## VNC Access for Apps

VNC (Virtual Network Computing) enables you to remotely access an installed app on the array in graphical mode from anywhere over the network. If an app supports VNC access, you can access the app through VNC when VNC access is enabled. Enabling VNC access opens a TCP port; the array management IP address and the TCP port create an endpoint on which the VNC server listens for connections.

## Nodes of an App

A node of an app is a dedicated instance running the app. Some apps are made up of multiple nodes. For easy identification, nodes are indexed starting at 0.

## Uninstalling an app

1. Select **Settings > Software**.

2. In the Installed Apps panel, verify the app is disabled.
3. Click the menu icon and select **Uninstall**. The Uninstall App dialog box appears.  
Click **Uninstall**.

#### Enabling an app

1. Select **Settings > Software**.
2. In the Installed Apps panel, click the menu icon and select **Enable**.

#### Disabling an app

1. Select **Settings > Software**.
2. In the Installed Apps panel, click the menu icon and select **Disable**.

#### Enabling VNC access for an app

1. Select **Settings > Software**.
2. In the Installed Apps panel, click the menu icon and select **Enable VNC**.  
The VNC Enabled column of the app changes to **true**.

#### Disabling VNC access for an app

1. Select **Settings > Software**.
2. In the Installed Apps panel, click the menu icon and select **Disable VNC**.  
The VNC Enabled column of the app changes to **false**.

#### Displaying the node details of an app

1. Select **Settings > Software**.
2. In the Installed Apps panel, click the app hyperlink in the Name column.  
The Nodes and Details information panels appear. The Nodes information panel shows the name, node indexes, version, status, and VNC endpoints in the format **IP address:port**, where **IP address** represents the array management IP address and **port** represents the VNC port. The Details information panel shows the app details.

#### Establishing connections between FlashArray volumes and apps

FlashArray volumes are connected to apps via the app host. To connect a FlashArray volume to an app:

1. Select the **Storage > Hosts**.
2. In the Hosts panel, click the app host associated with the app to which you want to connect the volumes.
3. In the Connected Volumes panel, click an existing volume in the left column to add it to the Selected Volumes column.
4. Click **Connect**.

5. Rescan the SCSI bus to ensure that all newly-added FlashArray volumes are visible from inside the app.

After the SCSI bus rescan, the FlashArray volumes (and their target LUNs) are visible as SCSI devices from inside the app, ready to be mounted.

# Part 3:

## Using the CLI to Administer a FlashArray



## Chapter 10. CLI Overview

The Purity//FA command line interface (CLI) is a non-graphical, command-driven interface used to query and administer the FlashArray. The Purity//FA CLI is comprised of built-in commands specific to the Purity//FA operating environment.

This chapter covers general Purity//FA CLI concepts and conventions.

## CLI Command Syntax and Conventions

Purity//FA CLI commands have the general form:

```
command subcommand --options OBJECT-LIST
```

The parts of a command are:

### COMMAND

Type of FlashArray object to be acted upon, prefixed by "pure". For example, the **purevol** command acts on Purity//FA virtual storage volumes.

Run the **pureman** command to see a list of Purity//FA CLI commands.

### SUBCOMMAND

Action to be performed on the specified object. Most CLI subcommands are common to some or all object types.

For example, **purehost list** lists all hosts on the array, while **purevol list** lists all volumes on the array.

The following subcommands are common to most or all object types:

#### **create**

Creates and names one or more FlashArray objects.

#### **delete**

Deletes one or more specified objects.

#### **list**

Lists information about one or more objects. To list information for all objects, do not specify the object in the command.

#### **listobj**

Creates whitespace-separated lists of objects or attributes related to one or more objects.

For example, **purevol listobj --type host** creates a list of the hosts to which volumes have connections. The **listobj** subcommand is primarily used to create lists of object and attribute names for scripting purposes.

#### **rename**

Changes the name of the specified object. Purity//FA identifies the object name in administrative operations and displays. The new name is effective immediately and the old

name is no longer recognized in Purity//FA GUI and CLI interactions. In the Purity//FA GUI, the new name appears upon page refresh. Hardware object names cannot be changed.

**setattr**

Changes the attribute values of the specified objects.

**OPTIONS**

Options that specify attribute values or modify the action performed by the subcommand.

For example, in the following command, the **--addvollist** option adds volumes **VOL1**, **VOL2**, and **VOL3** to protection group **PGROUP1**:

```
purepgroup setattr --addvollist VOL1,VOL2,VOL3 PGROUP1
```

Some options, such as **--hostlist**, inherently apply only to a single object. Other options, such as **--size**, can be set for multiple objects in a single command.

Some options can be multi-valued. For example, in the following command, the **--hostlist** option associates multiple hosts (**HOST1**, **HOST2**, and **HOST3**) with the host group **HGROUP1**.

```
purehgroup setattr --hostlist HOST1,HOST2,HOST3 HGROUP1
```

**OBJECT-LIST**

Object or list of objects upon which the command is to be operated.

If a subcommand changes the object state, then at least one object must be specified. Examples of subcommands that change the object state include **create**, **delete**, and **setattr**. For example, **purehost create HOST1** creates host **HOST1**. In the command synopses, OBJECT specifications that are not enclosed in square brackets (for example, "**HOST**") represent ones that are required.

Passive subcommands, such as **list**, which do not change object state, do not require object specification. Leaving out the object is equivalent to specifying all objects of the type. For example, **purevol list** with no volumes specified displays information about all volumes in an array. In the command synopses, OBJECT specifications enclosed in square brackets (for example, "[**HOST**]"") represent ones that are optional.

Most subcommands act on a single object. For example, in the following command, the **setattr** subcommand can only be run on a single host group to change the attributes of that host group.

```
purehgroup setattr --addhostlist HOST1 HGROUP1
```

Certain subcommands can operate on multiple objects. For example, the following command connects volume **VOL1** to host groups **HGROUP1** and **HGROUP2**.

```
purehgroup connect --vol VOL1 HGROUP1 HGROUP2
```

In the command synopses, OBJECT specifications that are followed by ellipses (for example, **HGROUP...**) indicate that multiple objects can be entered.

The following list describes the conventions used in CLI help documentation:

- Text in fixed-width (Courier) font must be entered exactly as shown. For example, `purehost list`.
- Text not enclosed in brackets represents mandatory text.
- Text inside square brackets ("[ ]") represents optional items. Do not type the brackets.
- Text inside curly braces ("{ }") represents text, where one (and only one) item must be specified. Do not type the braces.
- The vertical bar (|) separates mutually exclusive items.
- Uppercase italic text represents entered text whose value is based on the nature of the subcommand or option. For example, `--size SIZE`, where SIZE represents the value to be entered, such as `100g`.

## CLI Output

Various Purity//FA CLI subcommands, including `list` and `monitor`, generate list outputs. With the ability to create and manage hundreds of volumes and snapshots, list outputs can become very long.

The Purity//FA CLI includes options to control the way a list output is displayed and formatted. The CLI also includes sort, filter, and limit options to control the results you see and the order in which you see them.

### Interactive Paging

Pagination divides a large output into discrete pages. Pagination is disabled by default and is only in effect if the `--page` option is specified and the number of lines in the list output exceeds the size of the window.

To interactively move through a paginated list output:

- Press the `[Right Arrow]` key or space bar to move to the next page.
- Press the `[Left Arrow]` key to move to the previous page.

To quit interactive paging and exit the list view, press `q`.

With pagination, each page of the CLI list output begins with the column titles. To suppress the column titles, run the command with the `--notitle` option.

### Using Wildcards

The asterisk (\*) symbol denotes a wildcard character used in list or monitor operations to represent zero or more characters in an object name. The object name can include multiple asterisks.

When performing a list or monitor operation, include the asterisk in the name argument to expand the list results. For example, the `purehost list *cat*` command displays a list of all hosts that contain "cat", such as `cat`, `catnap`, `happycats`, and `lolcat`. If the asterisks were not included, only the host named `cat` would be returned. As another example, the `purevol list *vol*` command displays a list of all volumes that contain "vol", including ones that begin or end with "vol".

```
$ purevol list *vol*
```

Name	Size	Source	Created
Myvol	100G	-	2016-04-11 10:18:07 PDT
MyVol-01	100G	-	2016-04-11 10:18:07 PDT
vol	1G	-	2016-04-11 11:19:19 PDT
vol01	100G	-	2016-04-11 10:17:23 PDT
Volume	100G	-	2016-04-11 10:17:23 PDT

If Purity//FA cannot find matches for the wildcard argument specified, an error message appears.

Asterisks are also allowed in Purity//FA CLI options that take a list of object names. For example, the `purevol list --snap --pgrouplist *pg*` command displays a list of all volume snapshots that are created as part of a protection group snapshot with "pg" in the protection group name. As another example, the following command displays a list of volume snapshots for all volumes that end with "01" and are created as part of a protection group snapshot with "pg" in the protection group name.

```
$ purevol list --snap --pgrouplist *pg* *01
Name          Size  Source  Created
pgroup01.001.vol01  100G  vol01  2016-04-13 11:44:46 PDT
pgroup01.123.vol01  100G  vol01  2016-04-13 11:44:46 PDT
pgroup01.abc.vol01   100G  vol01  2016-04-13 11:44:46 PDT
pgroup01.backup.vol01  100G  vol01  2016-04-13 11:44:45 PDT
pgroup01.snap001.vol01  100G  vol01  2016-04-13 11:44:45 PDT
pgroup01.snap002.vol01  100G  vol01  2016-04-13 11:44:42 PDT
pgroup01.suffix.vol01   100G  vol01  2016-04-13 11:44:46 PDT
```

## Formatting

Format options control the way list outputs are displayed. The following format options are common to most `list` and `monitor` subcommands:

### --cli

Displays output in the form of CLI commands that can be issued to reproduce the current configuration. The `--cli` output is not meaningful when combined with immutable attributes.

### --csv

Lists information in comma-separated value (CSV) format. The `--csv` output can be used for scripting purposes and imported into spreadsheet programs.

### --notitle

Lists information without column titles.

### --nvp

Lists information in name-value pair (NVP) format, in the form `ITEMNAME=VALUE`. Argument names and information items are displayed flush left. The `--nvp` output is designed both for convenient viewing of what might otherwise be wide listings, and for parsing individual items for scripting purposes.

### --page

Turns on interactive paging.

**--raw**

Displays the unformatted version of column titles and data. For example, in the `purearray monitor` output, the unformatted version of column title `us/op (read)` is `usec_per_read_op`. The `--raw` output is used to sort and filter list results.

Here is a comparison between the `purehost list` output with formatted column titles and data, and the same output with unformatted column titles and data:

```
$ purehost list --space
Name  Size  Thin Provisioning  Data Reduction  ...
H001  60T   57%                1.5 to 1       ...

$ purehost list --space --raw
name  size          thin_provisioning  data_reduction  ...
H001  65970697666560  0.5712341        1.5123        ...
```

## Sorting

When running a list or monitor operation, include the `--sort` option to sort a column of the output in ascending or descending order.

The sort option adheres to the following syntax:

**--sort SORT**

SORT represents a comma-separated list of columns to sort. Use the unformatted title name (`--raw`) of the column to represent each column listed. To sort a column in descending order, append the minus (-) sign to the column name.

For example, run the following commands to get the unformatted column titles in the `purehost list` output, and then sort the same output by host group in descending order:

```
$ purehost list --raw
name          ...  hgroup
ESXi-GRP-Cluster02-H0001  ...  ESXi-GRP-Cluster02-HG003
ESXi-GRP-Cluster02-H0002  ...  ESXi-GRP-Cluster02-HG003
ESXi-IT-Cluster01-H0001  ...  ESXi-IT-Cluster01-HG001
ESXi-IT-Cluster02-H0001  ...  ESXi-IT-Cluster02-HG002
ESXi-STG-Cluster03-H0001  ...  ESXi-STG-Cluster03-HG005

$ purehost list --sort hgroup-
Name          ...  Host Group
ESXi-STG-Cluster03-H0001  ...  ESXi-STG-Cluster03-HG005
ESXi-IT-Cluster02-H0001  ...  ESXi-IT-Cluster02-HG002
ESXi-IT-Cluster01-H0001  ...  ESXi-IT-Cluster01-HG001
ESXi-GRP-Cluster02-H0002  ...  ESXi-GRP-Cluster02-HG003
ESXi-GRP-Cluster02-H0001  ...  ESXi-GRP-Cluster02-HG003
```

If multiple columns are specified, the output is sorted by the order of the columns listed.

If a sorted column contains non-unique values and a secondary sort argument is not specified, Purity//FA automatically performs a secondary sort using the default sort criteria (typically by `Name`).

## Examples

Example 1: Display a list of volumes sorted in descending order by physical space occupied.

```
$ purevol list --space --sort "total-"
```

Example 2: Display a list of protection groups sorted in ascending order by source array name.

```
$ purepgroup list --sort "source"
```

Example 3: Display a list of volumes that are sorted in descending order by volume size. If any of volumes are identical in size, sort those volumes in descending order by physical space occupied.

```
$ purevol list --space --sort size-,total-
```

## Filtering

When running a list or monitor operation, include the `--filter` option to narrow the results of the output to only the rows that meet the filter criteria. The following commands support the `--filter` option: purehgroup, purehost, purepgroup, pureport, and purevol. Filtering can be performed on any column of the list output.

### Filtering with Operators

When filtering with operators, use the following syntax:

```
--filter "RAW_TITLE OPERATOR VALUE"
```

`RAW_TITLE` represents the unformatted title name of the column by which to filter. Run the list or monitor operation with the `--raw` option to get the unformatted title name.

`OPERATOR` represents the type of filter match (`=`, `!=`, `<`, `>`, `<=`, or `>=`) used to compare `RAW_TITLE` to `VALUE`.

`VALUE` represents the value (number, date, or string) that determines the results to be included in the list output. Literal strings must be wrapped in quotes.

Filtering supports the following operators:

Operator	Description
<code>=</code>	Equals
<code>!=</code>	Does not equal
<code>&lt;</code>	Less than
<code>&gt;</code>	Greater than
<code>&lt;=</code>	Less than or equal to
<code>&gt;=</code>	Greater than or equal to

Filtering supports the asterisk wildcard character. For example, the value "`*esx*`" in the `purepgroup list --filter "hosts=*esx*"` command displays a list of all protection groups with hosts members that contain "esx" in the host name.

The AND and OR operators can be used to further refine your filter. The AND operator displays only the results that meet all of the filter criteria in the command. The OR operator displays the results that meet at least one of the filter criteria in the command.

## Examples

Example 1: Display a list of protection groups configured on source array `pure-001`.

```
$ purepgroup list --filter "source = 'pure-001'"
```

Example 2: Display a list of volumes that were created on or before `2016-05-23 13:09:39`.

```
$ purevol list --filter "created <= '2016-05-23 13:09:39 PDT'"
```

Example 3: Display a list of volume snapshots that are greater than 2 gigabytes in size.

```
$ purevol list --space --snap --filter "snapshots > '2G'"
```

Example 4: Display a list of volumes that are currently inactive.

```
$ purevol monitor --filter "output_per_sec=0 and input_per_sec=0"
```

Example 5: Display a list of volumes that begin with "vol10" or are greater than 100 gigabytes in size.

```
$ purevol list --filter "name = 'vol10*' or size > '100G'"
```

## Filtering with Functions

The filter option supports the CONTAINS and NOT functions.

`--filter FUNCTION(PARAMETERS)`

Function	Description
<code>contains(raw_title,string)</code>	Contains the enclosed string.  Takes exactly two parameters, where <code>raw_title</code> represents the unformatted title name of the column by which to filter and <code>string</code> represents the string to search within the column. Run the list or monitor operation with the <code>--raw</code> option to get the unformatted title name.
<code>not(expression)</code>	Inverse of the enclosed expression.

## Examples

Example 1: Get a list of all volumes that include "cluster03" in the volume name.

```
$ purevol list --filter "contains(name, 'cluster03')"
```

Example 2: Get a list of all hosts that are associated with host groups with "PROD" in the host group name.

```
$ purehost list --filter "not(contains(hgroup, 'PROD'))"
```

Example 3: Get a list of all hosts that are not associated with host groups with "PROD" in the host group name.

```
$ purehost list --filter "not(contains(hgroup, 'GRP'))"
```

## Existence Checks

The Purity//FA CLI supports existence checks. For example, the `purevol list --filter "name"` command checks to see if the "name" column exists in the volume list output.

## Examples

Example 1: Get a list of all hosts associated with a host group.

```
$ purehost list --filter "hgroup"
```

Example 2: Get a list of all volumes that were not created from another source.

```
$ purevol list --filter "not(source)"
```

Example 3: Get a list of all hosts that are not associated with Fibre Channel WWNs.

```
$ purehost list --filter "not(wwn)"
```

## Setting Limits

When running a list or monitor operation, include the `--limit` option to restrict the result size to the limit specified. The following commands support the `--limit` option: purehgroup, purehost, purepgroup, pureport, and purevol.

The limit option adheres to the following syntax:

`--limit ROWS`

`ROWS` represents the maximum number of rows to return.

For example, run the following command to list only the first 5 rows of the `purevol list` output:

```
$ purevol list --limit 5
Name          Size  Source  Created           Serial
ESXi-Cluster01-vol001  100G  -      2016-05-23 13:09:40 PDT  B143778B8ACE487B00011021
ESXi-Cluster01-vol002  100G  -      2016-05-23 13:09:40 PDT  B143778B8ACE487B0001101D
ESXi-Cluster01-vol003  100G  -      2016-05-23 13:09:40 PDT  B143778B8ACE487B00011024
ESXi-Cluster01-vol004  100G  -      2016-05-23 13:09:40 PDT  B143778B8ACE487B00011018
```

ESXi-Cluster01-vol005 100G - 2016-05-23 13:09:40 PDT B143778B8ACE487B00011025

## Combining Sorting, Filtering, and Limit Options

The `--sort`, `--filter`, and `--limit` options can all be combined together.

### Examples

Example 1: Display the top 10 volumes that occupy the largest amount of physical space and include "esx" in the volume name.

```
$ purevol list --space --sort "total-" --limit 10 --filter "name = '*esx*'"
```

Example 2: Display the top 10 volumes that have the largest physical space occupied by data unique to one or more snapshots.

```
purevol list --space --sort "snapshots-" --limit 10
```

Example 3: Display the top 10 volumes with the highest data reduction ratios.

```
$ purevol list --space --sort "data_reduction-" --limit 10
```

Example 4: Display the top 10 volumes that use the most read bandwidth.

```
$ purevol monitor --sort "output_per_sec-" --limit 10
```

Example 5: Display the top 10 volumes with the largest provisioned sizes and have shared connections to host groups with "PROD" in the host group name.

```
$ purevol list --connect --filter "hgroup = '*PROD*'" --sort size- --limit 10
```

## CLI Login

Log in to the Purity//FA CLI to query and administer the FlashArray. Logging in to the Purity//FA CLI requires a virtual IP address or fully-qualified domain name (FQDN) and a login username and password; this information is determined during the FlashArray installation. The CLI has been tested with the following remote access packages:

- SSH (all common UNIX and Linux distributions)
- PuTTY

## Logging in to the Purity//FA CLI

To log in to the Purity//FA CLI, select one of the following two options:

- **UNIX.** Start a secure shell (SSH) session, and then connect to the array using the login username and password obtained during the FlashArray installation.
- **Windows.** Start a remote terminal emulator (such as PuTTY), and then connect to the array using the login username and password obtained during the FlashArray installation.

Type **exit** or **logout** to log out of Purity//FA and exit the shell or terminal emulator.

## CLI Help

The Purity//FA CLI includes documentation for how to invoke and use each Purity//FA CLI command. Two types of interactive help are available through the Purity//FA CLI:

- The built-in Purity//FA CLI command help facility provides brief descriptions and usage information for each Purity//FA CLI command, subcommand, and option.
- The Purity//FA CLI man pages (or manual pages) is a more formal version of documentation that provides detailed information for each Purity//FA CLI command.

Run the **pureman** command to see a list of Purity//FA CLI commands.

### Purity//FA CLI Command Help

CLI command help is available at both the command and subcommand levels. To use it, type the Purity//FA CLI command or subcommand followed by **-h** or **--help**. Type the Purity//FA CLI command with the **-h** or **--help** switch to display usage information, supported syntax, and a list of subcommands for the specified command.

#### COMMAND -h

For example (truncated for brevity),

```
$ purevol -h
usage: purevol [-h]
    {connect,copy,create,destroy,disconnect,eradicate,
     list,listobj,monitor,recover,rename,setattr,snap,truncate}
...
positional arguments:
{connect,copy,create,destroy,disconnect,eradicate,
 list,listobj,monitor,recover,rename,setattr,snap,truncate}
connect          connect one or more volumes to a host
copy            copy a volume or snapshot to one or more volumes
create          create one or more volumes
destroy         destroy one or more volumes or snapshots
disconnect      disconnect one or more volumes from a host
eradicate       eradicate one or more volumes or snapshots
...
optional arguments:
-h, --help        show this help message and exit
```

Type the Purity//FA CLI subcommand with **-h** or **--help** switch to display usage information, supported syntax, and a list of options for the specified subcommand.

#### COMMAND SUBCOMMAND -h

For example,

```
$ purevol create -h
usage: purevol create [-h] --size SIZE VOL ...

positional arguments:
VOL          volume name

optional arguments:
-h, --help    show this help message and exit
--size SIZE  virtual size as perceived by hosts (e.g., 100M, 10G, 1T)
```

## Purity//FA CLI Man Pages

CLI man pages display extensive help for each CLI command. To display the man page for a Purity//FA CLI command, run the **pureman** command with the Purity//FA CLI command or subcommand name.

```
pureman COMMAND
```

or

```
pureman COMMAND-SUBCOMMAND
```

For example (truncated for brevity),

```
$ pureman purevol
PUREVOL(1)                               Purity//FA CLI Man Pages                  PUREVOL(1)

NAME
purevol, purevol-copy, purevol-create, purevol-destroy,
purevol-eradicate
...
- manage the creation, naming, and destruction of Purity//FA
virtual storage volumes and snapshots of their contents,
as well as the reclamation of physical storage occupied
by the data in them
purevol-monitor - monitor volume I/O performance

SYNOPSIS
purevol copy [--overwrite] SOURCE TARGETVOL
purevol create --size SIZE VOL...
purevol destroy VOL...
purevol eradicate VOL...
...

ARGUMENTS
SOURCE
Volume or snapshot from where data is copied.
The data is copied to the TARGETVOL volume.
```

**TARGETVOL**

Volume to where data is copied.

The data is copied from the SOURCE volume or snapshot.

**VOL**

Volume to be created, destroyed, eradicated, recovered,  
or snapped.

...

**OPTIONS**

--interval SECONDS (purevol monitor only)

Sets the number of seconds between displays of data.

At each interval, the system displays a point-in-time snapshot  
of the performance data.

If omitted, the interval defaults to every 5 seconds.

**--overwrite**

Allows purevol copy to overwrite an existing volume.

Without this option, if the target volume already exists,  
the command fails.

...

**DESCRIPTIONS**

This page describes management of volumes. The purevol create command...

**EXAMPLES****Example 1**

purevol create --size 100G vol1 vol2 vol3

Creates three volumes called vol1, vol2, and vol3, each  
with a host-visible capacity of 100 gigabytes ( $10^2 * 2^{30}$  bytes).

...

**SEE ALSO**

purevol-list(1), purevol-setattr(1), purehgroup(1),  
purehgroup-connect(1), purehost(1), purehost-connect(1)

**AUTHOR**

Pure Storage Inc.

## Purity//FA CLI Commands

This chapter describes each of the Purity//FA CLI commands.

The commands listed in this chapter administer and/or manage the FlashArray, with the exception of the following information-only commands:

**purelicense**

Outlines the End User Agreement (EUA) between Pure Storage Inc. and end users of the company's products.

**pureman**

Describes the pureman command and displays a list of available Purity//FA commands.



## pureadmin

pureadmin, pureadmin-create, pureadmin-delete, pureadmin-global, pureadmin-list, pureadmin-refresh, pureadmin-setattr — management of administrative accounts

### Synopsis

```
pureadmin create { --api-token | --role ROLE } [--timeout PERIOD] USER...
pureadmin delete [--api-token] USER...
pureadmin global list [ --cli | --csv | --nvp ] [--notitle] [--page] [--raw]
[USER...]
pureadmin global disable {--single-sign-on}
pureadmin global enable {--single-sign-on}
pureadmin global setattr --min-password-length LENGTH
pureadmin list [--api-token] [--expose] [--publickey] [ --cli | --csv | --nvp ] [--notitle] [--page] [--raw] [USER...]
pureadmin refresh [--clear] [USER...]
pureadmin setattr { --password | --publickey | --role } USER...
```

### Arguments

#### **USER**

User logon name. Sometimes referred to as sAMAccountName.

### Options

#### -h | --help

Can be used with any command or subcommand to display a brief syntax description.

#### --api-token

Displays users with API tokens, or generates an API token, granting the user access to the REST API. When listing users with API tokens, optionally include the **--expose** option to unmask the API token.

#### --clear

Clears the user permission cache.

#### --expose

Unmasks the current user's API token.

#### --min-password-length **LENGTH**

Displays or sets the global minimum character limit for local account passwords. New passwords must be at least **LENGTH** characters long to be accepted. The minimum password length must be greater than 0 characters. Empty passwords are not allowed. The default value is 1 character. Minimum password length changes do not apply to existing passwords.

**--password**

Changes the password for the specified local user. The password is entered through interactive prompt. Local users can change their own passwords through the **pureadmin setattr --password** command. Only array administrators can change the password on behalf of other users. By default, the new password must be at least 1 character and cannot exceed 100 characters in length. If a global minimum password length has been specified, the new password must abide by that setting.

**--publickey**

Displays users with public keys, or sets the public key for the specified user for SSH access. Only array administrators can change public keys on behalf of other users.

If changing the public key, the public key is entered through interactive prompt. If no users are specified, the public key change will be for the current user.

**--role *ROLE***

Sets the local user's access level to the array. Valid values are: **readonly**, **ops\_admin**, **storage\_admin**, and **array\_admin**.

Users with the Read-Only (**readonly**) role can perform operations that convey the state of the array. Read Only users cannot alter the state of the array.

Users with the Ops Admin (**ops\_admin**) role can perform the same operations as Read Only users plus enable and disable remote assistance sessions. Ops Admin users cannot alter the state of the array.

Users with the Storage Admin (**storage\_admin**) role can perform the same operations as Read Only users plus storage related operations, such as administering volumes, hosts, and host groups. Storage Admin users cannot perform operations that deal with global and system configurations.

Users with the Array Admin (**array\_admin**) role can perform the same operations as Storage Admin users plus array-wide changes dealing with global and system configurations. In other words, Array Admin users can perform all operations.

**--single-sign-on**

Enables or disables single sign-on (SSO) on the current array.

Enabling single sign-on gives LDAP users the ability to navigate seamlessly from Pure1 Manage to the current array through a single login. Enabling and disabling single sign-on takes effect immediately. By default, single sign-on is not enabled.

**--timeout *PERIOD***

Sets the validity period of the API token. The validity period is specified as an integer, followed by one of the suffix letters **s** (seconds), **m** (minutes), **h** (hours), **d** (days), or **w** (weeks).

Options that control display format:

**--cli**

Displays output in the form of CLI commands that can be issued to reproduce the current configuration. The **--cli** output is not meaningful when combined with immutable attributes.

--csv

Lists information in comma-separated value (CSV) format. The --csv output can be used for scripting purposes and imported into spreadsheet programs.

--notitle

Lists information without column titles.

--nvp

Lists information in name-value pair (NVP) format, in the form **ITEMNAME=VALUE**. Argument names and information items are displayed flush left. The --nvp output is designed both for convenient viewing of what might otherwise be wide listings, and for parsing individual items for scripting purposes.

--page

Turns on interactive paging.

--raw

Displays the unformatted version of column titles and data. For example, in the **purearray monitor** output, the unformatted version of column title **us/op (read)** is **usec\_per\_read\_op**. The --raw output is used to sort and filter list results.

## Description

The FlashArray array is delivered with a single administrative account named **pureuser**. The account is password protected and may alternatively be accessed using a public-private key pair.

The **pureuser** account is set to the array administrator role, which has array-wide permissions. The **pureuser** account cannot be deleted.

Users can be added to the array either locally by creating and configuring a local user directly on the array, or through Lightweight Directory Access Protocol (LDAP) by integrating the array with a directory service, such as Active Directory or OpenLDAP. Account information, such as group membership or password policy, for users that are added to the array through LDAP integration are managed through the directory. For more information about configuring directory services and LDAP integration, refer to **pureds(1)** [265].

Local users can only be created by array administrators. The name of the local user must be unique to all other users, including both local users and LDAP users.

If an LDAP user appears with the same name as a local user, the local user always has priority.

The **pureadmin create** command creates a local user. The name of the new user must be between 1 and 32 characters (alphanumeric and '-') in length and begin and end with a letter or number. The name must include at least one letter or '-'. All letters must be in lowercase. Include the required **--role** option to specify the user's access level to the array. Each local user on the array must be assigned to one of the following roles: **readonly**, **storage\_admin**, **array\_admin**. To log in to the Purity//FA system, each user requires a login password. For local users, this password is manually assigned by the array administrator through interactive command prompt during user creation. Local users can change their own passwords through the **pureadmin setattr --password** command.

The **pureadmin create --api-token** command generates a REST API token, granting the specified user access to the REST API. The regenerated API token replaces any token that has already

been created for the user. Users have permission to manage their own API tokens. By default, API tokens are created without expiry dates. To set an expiry date for an API token, include the **--timeout** option when creating the token. The validity period is specified as an integer, followed by one of the suffix letters **s** (seconds), **m** (minutes), **h** (hours), **d** (days), or **w** (weeks). When an API token expires and is therefore no longer valid, the user cannot access the REST API until the token is recreated.

The **pureadmin delete** command deletes a local user. Once deleted, the user is no longer able to access the array. Deleted users cannot be restored. To delete the API token of a user, include the **--api-token** command. Once the API token has been deleted, the user can no longer access the REST API. The API token can be recreated at any time.

The **pureadmin global** command displays and changes global administrative account configuration. The **pureadmin global setattr --min-password-length** command sets the minimum password length of all local account passwords. Existing passwords are not affected, but all future password assignments and changes must meet the new minimum password length requirement. The **pureadmin global enable --single-sign-on** command enables single sign-on on the current array. Enabling single sign-on gives LDAP users the ability to navigate seamlessly from Pure1 Manage to the current array through a single login. The **pureadmin global disable --single-sign-on** command disables single sign-on. Enabling and disabling single sign-on takes effect immediately. By default, single sign-on is not enabled. Enabling single sign-on is a two-step process: first, configure single sign-on and LDAP integration through Pure1 Manage, and second, enable single sign-on on the array through Purity//FA. For more information about SSO and LDAP integration with Pure1 Manage, refer to the Pure1 Manage - SSO Integration article in the Pure1 Knowledge site at <https://support.purestorage.com>. The **pureadmin global list** command displays the global configuration.

The **pureadmin list** command displays a list of Purity user accounts and their attributes. The list displays the following types of users:

- **pureuser** administrative account.
- Local users that have been created on the array.
- LDAP users with a public key and/or API token. LDAP users that do not have a public key or API token do not appear in the list.

The Type column displays the way in which the user was added to the array, either as Local or LDAP. The Role column displays the user's access level to the array. Include the **--api-token** option to display the API token details for each user. The Created column displays the date the API token was created while the Expires column displays the date the API token expires. When the API token expires and is therefore no longer valid, the user cannot access the REST API until the API token has been recreated. A dash (-) in the Expires column indicates an API token with no expiry date, meaning the API token is valid until it is recreated or deleted. Combine the **--api-token** option with the **--expose** option to unmask the current user's API token. Include the **--publickey** option to determine which users have public keys configured.

Directory service enabled accounts are also subject to role-based access control, but the permission level of those users correlate to the configured directory group(s) to which they are members. To prevent binding and querying the directory server too frequently, permissions are cached on the array. Run the **pureadmin refresh** command to refresh the cache entries for the specified user. Cache

entries are also automatically updated for a user when starting a new session. Include the **--clear** option to empty the entire permissions cache for all users. After running the **pureadmin refresh --clear** command, the first action by each user causes a query to the directory service, both to confirm that the user has permission for that action and to refresh that user's permission cache entry. These queries to the directory service eventually refresh the permission cache entries for all active users.

The **pureadmin setattr** command changes the attributes of existing users.

Include the **--password** option to change the password for the specified local user. The password is entered through interactive prompt. Local users can change their own passwords. Array administrators can change the password on behalf of other users. By default, the new password must be between 1 and 100 characters in length. If a minimum password length has been specified, the new password must abide by that setting. Password management for directory service enabled accounts is performed in the directory.

Include the **--publickey** option to set the public key for the specified user for SSH access. The public key is entered through interactive prompt. If no users are specified, the public key change will be for the current user. A new public key is typically entered by copying a value from a key generation application running in a local window on the administrative workstation and pasting it into the administrative session window. Each public key must correspond to a private key in the account from which a session is being conducted. Public key access can be configured for any user on the array, local or LDAP.

Include the **--role** option to specify the local user's access level to the array. All users in the array, whether created locally or added to the array through LDAP integration, are assigned one of the following roles in the array:

- **Read Only.** Users with the Read-Only (**readonly**) role can perform operations that convey the state of the array. Read Only users cannot alter the state of the array.
- **Ops Admin.** Users with the Ops Admin (**ops\_admin**) role can perform the same operations as Read Only users plus enable and disable remote assistance sessions. Ops Admin users cannot alter the state of the array.
- **Storage Admin.** Users with the Storage Admin (**storage\_admin**) role can perform the same operations as Read Only users plus storage related operations, such as administering volumes, hosts, and host groups. Storage Admin users cannot perform operations that deal with global and system configurations.
- **Array Admin.** Users with the Array Admin (**array\_admin**) role can perform the same operations as Storage Admin users plus array-wide changes dealing with global and system configurations. In other words, Array Admin users can perform all operations.

## Exceptions

None.

## Examples

### Example 1

```
pureadmin list --publickey
```

Lists the current administrative accounts for which SSH key access has been configured.

#### Example 2

```
pureadmin create --role storage_admin localuser1  
Enter password:  
Retype password:
```

Creates a local user named **localuser1** with the manually-created login password that was entered twice through interactive prompt. The password is not shown while typing. If the passwords do not match, the user is not created.

#### Example 3

```
pureadmin create --api-token pureuser
```

Creates an API token for the **pureuser** administrative account to grant REST API access.

#### Example 4

```
pureadmin create --api-token --timeout 1h pureuser
```

Creates an API token for the **pureuser** administrative account to grant REST API access, and sets the API token to expire one hour after being created.

#### Example 5

```
pureadmin delete localuser1
```

Deletes the **localuser1** user from the system. The **localuser1** user is no longer able to access the array.

#### Example 6

```
pureadmin setattr --password pureuser  
Enter old password:  
Enter new password:  
Retype new password:
```

Sets the password for local user **pureuser** to the new password. The old and new passwords are entered through interactive prompts. The password is not shown while typing.

#### Example 7

```
pureadmin setattr --publickey  
Enter key:
```

Indicates a request to change the public key for the workstation account from which the session is being conducted. The public key is entered through interactive prompt. The public key is typically copied from a key generation tool running in a local window on the administrative workstation.

## Example 8

```
pureadmin setattr --role array_admin localuser1
```

Assigns the array administrator role to local user `localuser1`, giving the ability to perform array-wide changes.

## Example 9

```
pureadmin refresh --clear
```

Clears the contents of the entire user permission cache.

## Example 10

```
pureadmin global enable --single-sign-on
```

Enables single sign-on (SSO) on the current array. SSO and LDAP integration must already be configured through Pure1 Manage before single sign-on can be enabled on the array.

## See Also

`pureds(1)` [265]

## Author

Pure Storage Inc. <[documentfeedback@purestorage.com](mailto:documentfeedback@purestorage.com)>

## purealert

purealert, purealert-create, purealert-delete, purealert-disable, purealert-enable, purealert-list, purealert-listobj, purealert-test — manages alert history and the list of designated email addresses to which Purity//FA sends alert messages when significant events occur in an array

### Synopsis

```
purealert create ADDRESS...
purealert delete ADDRESS...
purealert disable ADDRESS...
purealert enable ADDRESS...
purealert list [ --cli | --csv | --nvp ] [--notitle] [--page] [--raw] [ADDRESS...]
purealert listobj [--csv] [--type address] [ADDRESS...]
purealert test [ADDRESS...]
```

### Arguments

#### **ADDRESS**

Any valid electronic mail address.

### Options

**-h | --help**

Can be used with any command or subcommand to display a brief syntax description.

**--type {address} (purealert listobj)**

Outputs a whitespace-separated list containing the specified email addresses. If no addresses are specified, contains all addresses designated to receive alert messages, whether enabled or not.

**--type (purealert list)**

Type of information to be displayed. If not specified, defaults to designated email recipients.

Options that control display format:

**--cli**

Displays output in the form of CLI commands that can be issued to reproduce the current configuration. The **--cli** output is not meaningful when combined with immutable attributes.

**--csv**

Lists information in comma-separated value (CSV) format. The **--csv** output can be used for scripting purposes and imported into spreadsheet programs.

**--notitle**

Lists information without column titles.

--nvp

Lists information in name-value pair (NVP) format, in the form **ITEMNAME=VALUE**. Argument names and information items are displayed flush left. The --nvp output is designed both for convenient viewing of what might otherwise be wide listings, and for parsing individual items for scripting purposes.

--page

Turns on interactive paging.

--raw

Displays the unformatted version of column titles and data. For example, in the **purearray monitor** output, the unformatted version of column title **us/op (read)** is **usec\_per\_read\_op**. The --raw output is used to sort and filter list results.

## Description

Purity//FA generates log records called alerts when significant events occur within an array. Administrators can designate electronic mail addresses to which Purity//FA will send electronic mail messages when alert-generating events occur.

The **purealert create** subcommand can designate any valid electronic mail address to receive Purity//FA alert messages. Up to 20 addresses can be designated in an array (19 in addition to the built-in `flasharray-alerts@purestorage.com`). The **purealert delete** subcommand removes email addresses from the designated list.

FlashArray systems are delivered with the address `flasharray-alerts@purestorage.com` pre-designated to receive alerts. This address can be disabled (see below), temporarily suspending the transmission of alert messages to Pure Storage Technical Support, but cannot be deleted.

The **purealert test** subcommand tests an array's ability to send email to any email address (designated or not). If no email address arguments are specified, test messages are sent to all designated addresses. Verification of successful test message transmission is done at the destination. The only console response to the purealert test subcommand is the next Purity//FA prompt.

The **purealert list** subcommand lists any or all designated email addresses and their states (enabled or disabled).

The **purealert listobj** subcommand creates whitespace-separated (no formatting option specified) or comma-separated (--csv specified) lists of designated email addresses for scripting. If no addresses are specified, produces a list of all designated addresses. The same output list is produced whether or not the --type address option is specified.

The **purealert enable** and **purealert disable** subcommands respectively enable and disable the sending of alert messages to one or more designated email addresses. They do not send alert messages themselves. If no email address arguments are specified, the subcommands enable and disable the sending of alert messages to all designated addresses, including the built-in `flasharray-alerts@purestorage.com`.

When sending alerts to a designated email address is no longer appropriate, the **purealert delete** subcommand removes it from the list of designated addresses.

The sending account name for Purity//FA email alert messages is the array name. This name and the sender domain for alert messages can be viewed and altered by the **purearray list** and **purearray setattr** subcommands respectively.

The **purealert list --cli** subcommand displays the CLI command line that would produce the array's current list of designated email addresses in their current states. This list can, for example, be copied and pasted to create an identical email alert configuration in another array, or saved as a backup.

## Examples

### Example 1

```
purealert test admin1@mydomain.com
# verify at the destination that mail was received successfully
purealert create admin1@mydomain.com
purealert test
```

Sends a test message to admin1@mydomain.com. After receipt of the message at the destination has been verified by external means, designates admin1@mydomain.com to receive Purity//FA alert messages. The second purealert test subcommand sends test messages to all designated addresses, whether they are enabled or disabled.

### Example 2

```
purealert disable admin2@mydomain.com
purevol connect --host HOST1 VOL1 VOL2
purealert enable admin2@mydomain.com
```

Inhibits Purity//FA from sending alert messages to admin2@mydomain.com, so that the account does not receive the messages generated when the private connections between HOST1 and VOL1 and HOST1 and VOL2 are established. Re-enables sending to the admin2@mydomain.com account after the connections are established, so that the account receives subsequent alert messages.

## See Also

[purearray\(1\)](#) [230], [puremessage\(1\)](#) [354]

## Author

Pure Storage Inc. <[documentfeedback@purestorage.com](mailto:documentfeedback@purestorage.com)>

## pureapp

pureapp, pureapp-disable, pureapp-enable, pureapp-list, pureapp-node — displays and manages Purity apps

### Synopsis

```
pureapp disable [--vnc] APP
pureapp enable [--vnc] APP
pureapp list [--vnc] [ --cli | --csv | --nvp ] [--notitle] [--page] [--raw] [--filter FILTER] [--limit LIMIT] [--sort SORT] [APP...]
pureapp node list [--vnc] [ --cli | --csv | --nvp ] [--notitle] [--page] [--raw] [--filter FILTER] [--limit LIMIT] [--sort SORT] [APP...]
```

### Arguments

#### APP

App name.

### Options

#### -h | --help

Can be used with any command or subcommand to display a brief syntax description.

#### --vnc

Specifies VNC access for the installed apps.

Options that control display format:

#### --cli

Displays output in the form of CLI commands that can be issued to reproduce the current configuration. The **--cli** output is not meaningful when combined with immutable attributes.

#### --csv

Lists information in comma-separated value (CSV) format. The **--csv** output can be used for scripting purposes and imported into spreadsheet programs.

#### --notitle

Lists information without column titles.

#### --nvp

Lists information in name-value pair (NVP) format, in the form **ITEMNAME=VALUE**. Argument names and information items are displayed flush left. The **--nvp** output is designed both for convenient viewing of what might otherwise be wide listings, and for parsing individual items for scripting purposes.

#### --page

Turns on interactive paging.

--raw

Displays the unformatted version of column titles and data. For example, in the `purearray monitor` output, the unformatted version of column title `us/op (read)` is `usec_per_read_op`. The --raw output is used to sort and filter list results.

Options that manage display results:

--filter

Displays only the rows that meet the filter criteria specified.

--limit

Limits the size of the list output to the specified maximum number of rows.

--sort

Sorts the list output in ascending or descending order by the column specified.

## Description

The Purity Run platform extends array functionality by integrating add-on services into the Purity//FA operating system. Each service that runs on the platform is provided by an app.

In the following example, an app named `linux` has been installed on the array. The `linux` app represents an Ubuntu Linux distribution with Docker pre-installed, giving `pureuser` the ability to run Docker-based containers on the array.

```
$ pureapp list
Name    Version   Status    Description                               Details
linux  1.0.0     healthy   Ubuntu Linux 16.04 pre-installed with Docker -
```

Note that if an app migrates between controllers, it briefly stops and restarts.

Apps require CPU, memory, network, and storage resources. For this reason, apps by default are not installed. Apps are installed through the `puresw app` command. To determine if apps are installed on your array, run the `pureapp list` command. If apps are installed on your array, Purity//FA displays a list of the installed apps.

Once an app has been installed, perform the following tasks:

1. View the app attributes (`pureapp list`) to verify that the app is running.
2. View the volume attributes (`purevol list`) to verify that the boot and data volumes have been created.
3. View the host attributes (`purehost list`) to verify that the app host has been created, and view the host connection details (`purehost list --connect`) to verify that the boot and data volumes are connected to the app host.
4. View the app interfaces (`purenetwork list`) to verify that they have been created.
5. Configure the app interface(s) by assigning an IP address to the interface (`purenetwork setattr --address`) and then enabling the interface (`purenetwork enable`).
  - To give `pureuser` the ability to log into the app, configure the app management interface.

- To use a separate interface to transfer data in high volumes and at high speed, configure the app data interface.
6. Optionally connect FlashArray volumes to the app host (`purevol connect --host` or `purehost connect --vol`).

## Viewing App Attributes

The `pureapp list` command displays a list of apps that are installed on the array, along with the following attributes for each app:

- **Name:** App name. The app name is pre-assigned and cannot be changed.
- **Enabled:** App enable/disable status. An app must be enabled so the array can reach the app service. Apps are disabled by default.
- **Version:** App version that is currently installed on the array.
- **Status:** App status. A status of `healthy` means the app is running. A status of `unhealthy` means the app is not running.

There are various factors that contribute to an unhealthy app. In most cases, the unhealthy status is temporary, such as when the app is being restarted; upon successful restart, the app returns to healthy status. The app might also be unhealthy if, upon enabling the app, Purity//FA determines that there are insufficient resources to run it. An accompanying message appears in the Details column stating that there are insufficient resources to operate the app. Disable any apps that are currently not in use to free up some resources and try to enable the app again.

If the app is in an unhealthy status for a longer than expected period of time, contact Pure Storage Support.

- **Description:** Description of the app.
- **Details:** Additional details specific to the app.

## Viewing App Volumes

For each app that is installed, a boot volume is created. For some apps, a data volume is also created. Boot and data volumes are known as app volumes. Run the `purevol list` command to see a list of volumes, including app volumes.

Boot and data app volume names begin with a distinctive @ symbol. The naming convention for app volumes is `@APP_boot` for boot volumes and `@APP_data` for data volumes, where APP denotes the app name.

The following example displays the boot and data volumes for the `linux` app.

```
$ purevol list @linux*
Name      Size  Source  Created          Serial
...
@linux_boot  15G   -    2016-11-09 15:13:37 MST  AC97A330F2544A3C00011010
@linux_data  16T   -    2016-11-09 15:14:17 MST  AC97A330F2544A3C00011012
```

The boot volume represents a copy of the boot drive of the app. Do not modify or save data to the boot volume. When an app is upgraded, the boot volume is overwritten, completely destroying its contents including any other data that is saved to it. The data volume is used by the app to store data.

The following example shows that the drives were correctly mounted inside the `linux` app.

```
pureuser@linux:~$ df
Filesystem      1K-blocks    Used   Available  Use%  Mounted on
udev              8198768      0    8198768   0%   /dev
tmpfs             1643272    8756    1634516   1%   /run
/dev/sdal        15348720  1721392   12824616  12%   /
/dev/sdb         17177782208 33608  17177748600  1%   /data
```

Disk device `/dev/sdb`, which corresponds to the app data volume, is mounted on `/data`, meaning the data will be saved to the data volume (and not the boot volume), and disk device `/dev/sdal`, which corresponds to the app boot volume, is mounted on `/`.

## Viewing App Hosts

Each app has a dedicated host, known as an app host. The app host is connected to the associated boot and data volumes. The app host is also used to connect FlashArray volumes to the app.

App host names begin with a distinctive @ symbol. The naming convention for app hosts is `@APP`, where `APP` denotes the app name.

The following example displays the app host for the `linux` app.

```
$ purehost list @linux*
Name      WWN  IQN  Host Group
@linux    -    -    -
```

The following example displays one app host named `@linux` and its associated boot and data volumes.

```
$ purehost list --connect @linux*
Name      LUN  Vol          Host Group
@linux    1    @linux_boot  -
@linux    2    @linux_data  -
```

Unlike regular FlashArray hosts, app hosts cannot be deleted, renamed, or modified in any way. Furthermore, app hosts cannot be added to host groups or protection groups.

## Viewing App Interfaces

For each app that is installed, one app management interface is created per array management interface. An app data interface may also be created for high-speed data transfers.

The naming convention for app interfaces is `APP.datay` for the app data interface, and `APP.mgmty` for the app management interface, where `APP` denotes the app name, and `y` denotes the interface.

In the following example, for the `linux` app, two app management interfaces, one named `linux.mgmt0` and another named `linux.mgmt1`, have been created to correspond to each of the

array management interfaces. An app data interface (named `linux.data0`) with a data transfer rate of 10 gigabytes per second has also been created.

Name	Enabled	Address	Mask	...	Speed	Services
<code>ct0.eth0</code>	True	10.7.102.60	255.255.255.0	...	1.00 Gb/s	management
<code>ct0.eth1</code>	False	-	-	...	1.00 Gb/s	management
<code>ct0.eth2</code>	True	10.7.102.62	255.255.255.0	...	10.00 Gb/s	iscsi
<code>ct1.eth0</code>	True	10.7.102.70	255.255.255.0	...	1.00 Gb/s	management
<code>ct1.eth1</code>	False	-	-	...	1.00 Gb/s	management
<code>ct1.eth2</code>	True	10.7.102.72	255.255.255.0	...	10.00 Gb/s	iscsi
<code>linux.data0</code>	False	-	-	...	10.00 Gb/s	app
<code>linux.mgmt0</code>	False	-	-	...	1.00 Gb/s	app
<code>linux.mgmt1</code>	False	-	-	...	1.00 Gb/s	app
<code>vir0</code>	True	10.7.102.80	255.255.255.0	...	1.00 Gb/s	management
<code>vir1</code>	False	-	-	...	1.00 Gb/s	management

## Configuring App Interfaces

Configure an app interface to give `pureuser` the ability to log into the app or transfer data through a separate interface. Configuring an app interface involves assigning an IP address to the interface and then enabling the interface.

Optionally set the gateway. Note that only one of the app interfaces of a particular app can have a gateway set.

Before you configure an app interface, make sure the corresponding external interface is physically connected.

Configure one or more of the following app interfaces:

- **App Management Interface**

Configure the app management interface to give `pureuser` the ability to log into the app with the same Purity//FA login credentials. If a public key has been created for the user, it can be used to log into the app. Purity//FA password changes are automatically applied to the app.

To configure the app management interface, run the `purenetwork setattr --address` command to assign an IP address to one of the app management interfaces, and then run the `purenetwork enable` command to enable the interface.

- **App Data Interface**

Configure the app data interface to use a separate interface for high-speed data transfers.

To configure the app data interface, run the `purenetwork setattr --address` command to assign an IP address to the app data interface, and then run the `purenetwork enable` command to enable the interface.

In the following example, app management interface `linux.mgmt0` has been enabled and IP address `10.8.102.96` has been assigned to the interface, giving `pureuser` the ability to log directly into the `linux` app with the same Purity//FA login credentials.

```
$ purenetwork list
```

Name	Enabled	Address	...	Speed	Services
...					
linux.data0	False	-	...	10.00 Gb/s	app
linux.mgmt0	True	10.8.102.96	...	1.00 Gb/s	app
linux.mgmt1	False	-	...	1.00 Gb/s	app
...					

## Establishing Connections between FlashArray Volumes and Apps

FlashArray volumes are connected to apps via the app host.

To connect a FlashArray volume to an app, connect the volume to the app host that is associated with the app. In the following example, five FlashArray volumes, along with the `linux` boot and data volumes, are connected to the `@linux` app via the `@linux` app host.

```
$ purehost list --connect @linux*
Name      LUN  Vol          Host Group
@linux    1    @linux_boot  -
@linux    2    @linux_data   -
@linux    3    app_vol001   -
@linux    4    app_vol002   -
@linux    5    app_vol003   -
@linux    6    app_vol004   -
@linux    7    app_vol005   -
```

FlashArray volumes are connected to app hosts in the same way that they are connected to regular FlashArray hosts. Run the `purevol connect --host` command to connect one or more volumes to an app host. The connection can also be established by running the `purehost connect --vol` command.

A FlashArray volume can only be connected to one app host at a time. Furthermore, the FlashArray volume cannot be connected to other hosts or host groups while it is connected to an app host.

After a FlashArray volume has been connected to an app host, rescan the SCSI bus to ensure the newly-connected volumes are visible from inside the app. The following example displays five FlashArray volumes (and their target LUNs) as SCSI devices from inside the `linux` app, ready to be mounted.

```
pureuser@linux:~$ cat /proc/scsi/scsi
Attached devices:
Host: scsi2 Channel: 00 Id: 01 Lun: 03
  Vendor: PURE      Model: FlashArray        Rev: 9999
  Type:  Direct-Access                         ANSI  SCSI revision: 06
Host: scsi2 Channel: 00 Id: 01 Lun: 04
  Vendor: PURE      Model: FlashArray        Rev: 9999
  Type:  Direct-Access                         ANSI  SCSI revision: 06
Host: scsi2 Channel: 00 Id: 01 Lun: 05
  Vendor: PURE      Model: FlashArray        Rev: 9999
  Type:  Direct-Access                         ANSI  SCSI revision: 06
```

```

Host: scsi2 Channel: 00 Id: 01 Lun: 06
  Vendor: PURE      Model: FlashArray          Rev: 9999
  Type: Direct-Access                         ANSI SCSI revision: 06
Host: scsi2 Channel: 00 Id: 01 Lun: 07
  Vendor: PURE      Model: FlashArray          Rev: 9999
  Type: Direct-Access                         ANSI SCSI revision: 06

```

To disconnect a FlashArray volume from an app host, run the `purevol disconnect --host` or `purehost disconnect --vol` command.

### Displaying the Nodes of an App

A node of an app is a dedicated instance running the app. Some apps are made up of multiple nodes. For easy identification, nodes are indexed starting at 0.

To list the node details of the installed apps, use the `pureapp node list` command. The following example shows that the `wfs` app is running on two nodes.

```
$ pureapp node list
Name      Index    Status
WFS       0         healthy
           1         unhealthy
```

### Configuring VNC Access for an App

VNC (Virtual Network Computing) enables you to remotely access an installed app on the array in graphical mode from anywhere over the network. If an app supports VNC access, you can access the app through VNC when VNC access is enabled. Enabling VNC access opens a TCP port; the array management IP address and the TCP port create an endpoint on which the VNC server listens for connections.

To enable VNC access for an installed app, specify the `--vnc` option with the `pureapp enable` command. To disable VNC access for an installed app, specify the `--vnc` option with the `pureapp disable` command.

To check the status of VNC for the installed apps, use the `pureapp list --vnc` command.

The following example shows that VNC Enabled is `True` to allow VNC access for the `wfs` app. The default value of VNC Enabled is `False`.

```
$ pureapp list --vnc
Name  Enabled  Version  Status   VNC Enabled  Description          Details
WFS   True     2.0.0    healthy  True        Windows Server 2016 Standard -
```

To list the node details including VNC endpoints, specify the `--vnc` option with the `pureapp node list` command.

The following example shows that the `wfs` app consists of two nodes (indexes 0 and 1) and each node has two VNC endpoints in the format `IP address:port` as shown in the VNC column, where `IP address` represents the array management IP address and `port` represents the VNC port. When VNC access is disabled, the VNC column displays a null value.

```
$ pureapp node list --vnc
Name      Index  Version  Status    VNC
WFS        0       2.0.0   healthy  10.14.72.220:5900
                           10.14.72.237:5900
                           1       2.0.0   healthy  10.14.72.221:5900
                           10.14.72.238:5900
```

## Enabling and Disabling Apps

The **pureapp enable** command enables an app. An app must be enabled so the array can reach the app service. Apps are disabled by default.

The **pureapp disable** command disables an app, effectively stopping the service.

## Installing Apps

The **puresw app install** command installs a new app or upgrades a currently installed app. For more information about the app installation process, refer to *puresw(1)* [452].

## Examples

### Example 1

```
pureapp list
```

Displays a list of apps that have been installed on the array and the attributes of each app, including name, enabled status, version number, description, and details.

### Example 2

```
purevol list @linux*
purehost list --connect @linux
purenetwork setattr --address 10.8.108.165 linux.mgmt0
purenetwork enable linux.mgmt0
```

Displays the boot and data volumes for the **linux** app, displays the names of all volumes that are connected to app host **linux**, assigns IP address **10.8.108.165** to app management interface **linux.mgmt0**, and enables interface **linux.mgmt0**, giving *pureuser* the ability to log into the **linux** app.

### Example 3

```
purevol connect --host @linux app_vo1001
```

Connects FlashArray volume **app\_vo1001** to the **linux** app via app host **@linux**. Note that the connection can also be established by running the **purehost connect --vol** command.

### Example 4

```
pureapp enable --vnc linux
```

```
pureapp node list --vnc
```

Enables VNC access for the `linux` app and lists the node details of the app including VNC endpoints.

#### Example 5

```
purehost disconnect --vol app_vol002 @linux
```

Breaks the connection between FlashArray volume `app_vol002` and the `linux` app. Note that the connection can also be established by running the `purevol disconnect --host` command.

#### Example 6

```
pureapp disable linux
```

Disables the `linux` app.

#### See Also

`purehost(1)` [292], `purenetwork(1)` [360], `puresw(1)` [452], `purevol(1)` [469], `purevol-list(1)` [488]

#### Author

Pure Storage Inc. <[documentfeedback@purestorage.com](mailto:documentfeedback@purestorage.com)>

## purearray

purearray, purearray-list, purearray-rename, purearray-replace, purearray-setattr — manages attributes that pertain to the array as a whole  
 purearray-monitor — monitors array I/O performance  
 purearray-disable, purearray-enable, purearray-phonehome, purearray-remoteassist — manages support-related functions  
 purearray-diagnose, purearray-test — performs array diagnostic functions  
 purearray-connect, purearray-disconnect — manages connections between arrays  
 purearray-eula — manages the Pure Storage End User Agreement (EULA)

### Synopsis

```

purearray connect --management-address MANAGEMENT-ADDRESS [--replication-address
REPLICATION-ADDRESS] --type replication --connection-key

purearray diagnose { --email | --log | --phonehome }

purearray disable { console-lock | phonehome }

purearray disconnect ARRAY

purearray enable { console-lock | phonehome | security-token }

purearray eula { accept | list [ --acceptance ]}

purearray list [ --banner | --connect | --connection-key | --console-lock | --
controller | --idle-timeout | --ntpserver | --phonehome | --proxy | --relayhost | --
scsi-timeout | --security-token | --senderdomain | --space | --syslogserver | --
throttle ] [--path] [ --historical { 1h | 3h | 24h | 7d | 30d | 90d | 1y } ] [ --cli
| --csv | --nvp ] [--notitle] [--page] [--raw]

purearray monitor [--csv] [ --historical { 1h | 3h | 24h | 7d | 30d | 90d | 1y } ] [
[--interval SECONDS] [--latency] [--mirrored] [--repeat REPEAT-COUNT] [--size] ]
[ --notitle | --raw ]

purearray phonehome { --cancel | --send-all | --send-today | --send-yesterday | --
status }

purearray remoteassist { --connect | --disconnect | --status }

purearray rename NEW-NAME

purearray replace {security-token}

purearray setattr { --banner | --idle-timeout IDLE-TIMEOUT | --ntpserver NTP-
SERVER | --proxy HTTPS-PROXY | --relayhost RELAY-HOST | --scsi-timeout | --
senderdomain SENDER-DOMAIN | --syslogserver SYSLOG-SERVER }

purearray throttle --connect { --default-limit DEFAULT-LIMIT | --window WINDOW |
--window-limit WINDOW-LIMIT } ARRAY

purearray test { ntp | pinghome | phonehome | security-token | syslogserver }

```

## Options

**-h | --help**

Can be used with any command or subcommand to display a brief syntax description.

**--acceptance**

Displays the acceptance details for the Pure Storage End User Agreement. Acceptance details include the name and title of the individual at the company who accepted the terms of the agreement, company name, and agreement acceptance date.

**--banner**

Displays or sets text that Purity users see in the GUI at login, and in the CLI just before the password prompt.

**--cancel**

Cancels any in-progress transmission of event logs to Pure Storage Support.

**--connect**

When used with **purearray list**, displays array connection details, including array name, management and replication addresses, connection status (true or false), throttle status (true or false), and connection type (for example, replication).

When used with **purearray remoteassist**, enables Pure Storage Support to initiate a remote assistance session.

For **purearray throttle**, must be used with any of the other **purearray throttle** options to set network bandwidth throttle limits.

**--connection-key**

When used with **purearray list**, displays a key that can be used to connect to this array from another array.

When used with **purearray connect**, prompts for the connection key of the remote array.

**--controller**

Displays the name, mode, FlashArray model, Purity//FA version, and status information for the array's controllers.

**--default-limit *DEFAULT-LIMIT***

Sets the default maximum network bandwidth threshold for outbound traffic when transferring data to the specified target array. Once the threshold is exceeded, bandwidth throttling occurs.

**--disconnect**

Terminates an in-progress remoteassist session with Pure Storage Support.

**--email**

Sends subcommand output to Pure Storage Support.

**--historical**

When used with **purearray list --space**, displays the amount of usable physical storage on the array and the amount of storage occupied by data and metadata over the specified range of time. Valid time ranges include: 1 hour, 3 hours, 24 hours, 7 days, 30 days, 90 days, and one year.

When used with **purearray monitor**, displays historical performance data over the specified range of time. Valid time ranges include: 1 hour, 3 hours, 24 hours, 7 days, 30 days, 90 days, and one year.

--idle-timeout **IDLE-TIMEOUT**

Displays or sets an idle time limit in minutes. CLI and GUI sessions that are idle for more than **IDLE-TIMEOUT** minutes are automatically logged out. Values between 5 and 180 minutes are valid. Specifying a value of zero disables the automatic logout feature. The default timeout value is 30 minutes. Idle timeout changes apply to existing sessions.

--interval **SECONDS**

Sets the number of seconds between displays of real-time performance data. At each interval, the system displays a point-in-time snapshot of the performance data. If omitted, the interval defaults to every 5 seconds.

--latency

Displays real-time and historical I/O latency information across the entire array.

--log

Sends subcommand output to the array's log.

--management-address **MANAGEMENT-ADDRESS**

Specifies the management address. Enter the virtual (**vir0**) IP address or FQDN of the target array.

--mirrored

In a synchronous replication configuration, includes performance data for mirrored writes (i.e., writes to volumes in stretched pods). If one array of a stretched pod is offline, the writes processed on the other array are reported as local writes rather than mirrored ones. Once the pod is back online, in sync, and processing each write on both sides, the writes to its volumes are reported as mirrored writes again.

--ntpserver [**NTP-SERVER**]

Displays or sets the hostnames or IP addresses of the NTP servers currently being used by the array to maintain reference time.

When used with **purearray setattr**, specify up to four NTP servers. If specifying an IP address, for IPv4, specify the IP address in the form **ddd.ddd.ddd.ddd**, where **ddd** is a number ranging from 0 to 255 representing a group of 8 bits. For IPv6, specify the IP address in the form **xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx**, where **xxxx** is a hexadecimal number representing a group of 16 bits. When specifying an IPv6 address, consecutive fields of zeros can be shortened by replacing the zeros with a double colon (::).

--path

Displays the connection paths between this array and each of the connected arrays. This option is only valid when run on the array from where the array connection was made. Must be used with the **--connect** option.

**--phonehome**

For **purearray diagnose**, sends subcommand output to Pure Storage Support via the phonehome channel.

For **purearray list**, displays the current state of the Purity//FA phonehome automatic hourly log transmission facility (enabled or disabled).

**--proxy [HTTPS-PROXY]**

Displays or sets the proxy host for the phonehome facility when https is the phonehome protocol (the phonehome facility itself determines which protocol to use). The format for the option value in the **purearray setattr** subcommand is: **HTTP(S)://HOSTNAME:PORT**, where **HOSTNAME** is the name of the proxy host, and **PORT** is the TCP/IP port number used by the proxy host.

**--relayhost [RELAY-HOST]**

Displays or sets the hostname or IP address of the email relay server currently being used as a forwarding point for email alerts generated by the array. If specifying an IP address, enter the IPv4 or IPv6 address.

For IPv4, specify the IP address in the form **ddd.ddd.ddd.ddd**, where **ddd** is a number ranging from 0 to 255 representing a group of 8 bits. If a port number is also specified, append it to the end of the address in the format **ddd.ddd.ddd.ddd:PORT**, where **PORT** represents the port number.

For IPv6, specify the IP address in the form

**xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx**, where **xxxx** is a hexadecimal number representing a group of 16 bits. Consecutive fields of zeros can be shortened by replacing the zeros with a double colon (**::**). If a port number is also specified, enclose the entire address in square brackets (**[ ]**) and append the port number to the end of the address. For example, **[xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx]:PORT**, where **PORT** represents the port number.

**--repeat REPEAT-COUNT**

Sets the number of times to log I/O statistics or display real-time performance data. If omitted, the repeat count is 1.

**--replication-address REPLICATION-ADDRESS**

Specifies the replication address. Enter the IP address or FQDN of the replication bond (**replbond**) interface on the target array.

**--scsi-timeout**

Displays or sets the amount of time (in seconds) that can lapse during an I/O interruption before the target ports log out of the fabric. The default timeout value is 60 seconds.

The **--scsi-timeout** option is an advanced option. Changing the default timeout value may cause an initiator to mistakenly interpret the status of the FlashArray as failed or generate a host timeout.

Contact the Pure Storage support team before you change the **--scsi-timeout** value.

**--security-token**

Displays the status of the Rapid Data Locking feature (enabled or disabled) and the signature of the attached security token. For more information about the Rapid Data Locking (RDL) feature, refer to the FlashArray Enhanced Data Security Guide in the Pure1 Knowledge site at <https://support.purestorage.com>.

**--send-all**

Sends all log information stored in the array to Pure Storage Support via the phonehome channel.

**--senderdomain [ *SENDER-DOMAIN* ]**

Displays or sets the domain name. The domain name determines how Purity//FA logs are parsed and treated by Pure Storage Support and Escalations. The domain name is also appended to alert email messages. The default domain name is `please-configure.me`. If this is not a Pure Storage test array, set the sender domain to the actual customer domain name.

**--send-today**

Sends log information from the current day (in the array's time zone) to Pure Storage Support via the phonehome channel.

**--send-yesterday**

Sends log information from the previous day (in the array's time zone) to Pure Storage Support via the phonehome channel.

**--size**

Displays the average I/O sizes per read and write operation.

**--space**

Displays the amount of usable physical storage on the array and the amount of storage occupied by data and metadata.

#### Capacity

Total physical usable space on the array. Replacing a drive may result in a dip in usable capacity. This is intended behavior. RAID striping splits data across an array for redundancy purposes, spreading a write across multiple drives. A newly added drive cannot use its full capacity immediately but must stay in line with the available space on the other drives as writes are spread across them. As a result, usable capacity on the new drive may initially be reported as less than the amount expected because the array will not be able to write to the unallocatable space. Over time, usable capacity fluctuations will occur, but as data is written to the drive and spreads across the array, usable capacity will eventually return to expected levels.

#### Parity

Percentage of data that is fully protected. The percentage value will drop below 100% if the data isn't fully protected, such as when a module is pulled and the array is rebuilding the data to bring it back to full parity.

#### Thin Provisioning

Percentage of volume sectors that do not contain host-written data because the hosts have not written data to them or the sectors have been explicitly trimmed.

#### Data Reduction

Ratio of mapped sectors within a volume versus the amount of physical space the data occupies after data compression and deduplication. The data reduction ratio does not include thin provisioning savings.

For example, a data reduction ratio of 5:1 means that for every 5 MB the host writes to the array, 1 MB is stored on the array's flash modules.

#### Total Reduction

Ratio of provisioned sectors within a volume versus the amount of physical space the data occupies after reduction via data compression and deduplication *and* with thin provisioning savings. Total reduction is data reduction with thin provisioning savings.

For example, a total reduction ratio of 10:1 means that for every 10 MB of provisioned space, 1 MB is stored on the array's flash modules.

#### Volumes

Physical space occupied by volume data that is not shared between volumes, excluding array metadata and snapshots.

#### Snapshots

Physical space occupied by data unique to one or more snapshots.

#### Shared Space

Physical space occupied by deduplicated data, meaning that the space is shared with other volumes and snapshots as a result of data deduplication.

#### System

Physical space occupied by internal array metadata.

#### Total

Total physical space occupied by system, shared space, volume, and snapshot data.

#### --status

For **purearray phonehome --status**, displays the status of in-progress log transmission jobs. A status of "running" indicates a log transmission of the displayed action type (send-all, send-today, or send-yesterday) is in progress.

For **purearray remoteassist --status**, displays the status of a remote assist session as enabled or disabled.

#### --stop

Stops logging I/O statistics.

#### --syslogserver [*SYSLOG-SERVER*]

Displays or sets the URI of the remote syslog server. For example, `tcp://MyHost.com`.

Syslog servers deleted using this command will not receive an audit record on the remote syslog server. Instead, use the **purelog** command.

Note: The **--syslogserver** option is going to be replaced by the **purelog** command in a future release. Therefore, **--syslogserver** is considered deprecated and should not be used. To set remote syslog servers, run the **purelog** command.

--throttle  
 Displays the network bandwidth throttle settings for all connected arrays. Must be used with the --connect option.

--type {replication}  
 The type(s) of connection desired. The only option supported in this version is 'replication'.

--window **WINDOW**  
 Range of time the --window-limit threshold is in effect.

--window-limit **WINDOW-LIMIT**  
 Sets the maximum network bandwidth threshold during the specified --window time range.  
 Once the threshold is exceeded, bandwidth throttling occurs.

Options that control display format:

--cli  
 Displays output in the form of CLI commands that can be issued to reproduce the current configuration. The --cli output is not meaningful when combined with immutable attributes.

--csv  
 Lists information in comma-separated value (CSV) format. The --csv output can be used for scripting purposes and imported into spreadsheet programs.

--notitle  
 Lists information without column titles.

--nvp  
 Lists information in name-value pair (NVP) format, in the form **ITEMNAME=VALUE**. Argument names and information items are displayed flush left. The --nvp output is designed both for convenient viewing of what might otherwise be wide listings, and for parsing individual items for scripting purposes.

--page  
 Turns on interactive paging.

--raw  
 Displays the unformatted version of column titles and data. For example, in the **purearray monitor** output, the unformatted version of column title **us/op (read)** is **usec\_per\_read\_op**. The --raw output is used to sort and filter list results.

## Arguments

### **ARRAY**

Name of the remote array.

### **NEW-NAME**

Name by which the array is to be known after the command executes.

## Object Names

Valid characters are letters (A-Z and a-z), digits (0-9), and the hyphen (-) character. The first and last characters of the name must be alphanumeric, and the name must contain at least one letter or '-'.

Most objects in Purity//FA that can be named, including host groups, hosts, volumes, protection groups, volume and protection group suffixes, SNMP managers, and subnets, can be 1-63 characters in length.

Array names can be 1-56 characters in length. The array name length is limited to 56 characters so that the names of the individual controllers, which are assigned by Purity//FA based on the array name, do not exceed the maximum allowed by DNS.

Names are case-insensitive on input. For example, **vol1**, **Vol1**, and **VOL1** all represent the same volume. Purity//FA displays names in the case in which they were specified when created or renamed.

Pods and volume groups provide a namespace with unique naming conventions.

All objects in a pod have a fully qualified name that include the pod name and object name. The fully qualified name of a volume in a pod is **POD::VOLUME**, with double colons (::) separating the pod name and volume name. The fully qualified name of a protection group in a pod is **POD::PGROUP**, with double colons (::) separating the pod name and protection group name. For example, the fully qualified name of a volume named **vol01** in a pod named **pod01** is **pod01::vol01**, and the fully qualified name of a protection group named **pgroup01** in a pod named **pod01** is **pod01::pgroup01**.

If a protection group in a pod is configured to asynchronously replicate data to a target array, the fully qualified name of the protection group on the target array is **POD:PGROUP**, with single colons (:) separating the pod name and protection group name. For example, if protection group **pod01::pgroup01** on source array **array01** asynchronously replicates data to target array **array02**, the fully qualified name of the protection group on target array **array02** is **pod01:pgroup01**.

All objects in a volume group have a fully qualified name that includes the volume group name and the object name, separated by a forward slash (/). For example, the fully qualified name of a volume named **vol01** in a volume group named **vgroup01** is **vgroup01/vol01**.

## Description

The **purearray diagnose** subcommand executes a series of CLI subcommands that capture a snapshot of an array's I/O performance, configuration, and hardware status to be collected by Pure Storage Support. The output format must be declared with one of the following options:

**--email**

Mail the output to Pure Storage Support.

**--log**

Write the output to the array's log.

**--phonehome**

Transmit the output to Pure Storage Support via the secure channel.

The **purearray enable phonehome** and **purearray disable phonehome** commands enable and disable, respectively, automatic hourly transmission of array logs to Pure Storage Support.

The **purearray enable console-lock** and **purearray disable console-lock** commands enable and disable, respectively, root user login at the physical (serial or VGA) console.

The **purearray enable security-token** enables Rapid Data Locking if the required hardware is present. The command must be run on the primary controller.

The **purearray list** command with no option specified displays the array name, the array ID, and the Purity//FA version and revision numbers. Options can be specified to display controller information, connected arrays, the NTP (Network Time Protocol) server name, the state of and proxy host for the phonehome facility, the state of the console lock, the relay host via which electronic mail is sent, and the sender domain for electronic mail alert messages.

The **purearray list --connect** command displays a list of arrays that are connected to this array. In the list output, the management and replication addresses only appear for the arrays from where an array connection was made. If the array connection was made from its peer array, the Management Address and Replication Address columns display dashes (-). Include the **--path** option to display the connection paths for each of the connected arrays. The **--path** option is only valid when run on the array from where the array connection was made. A status of **connected** means that the path between the source and destination is healthy. A status of **connecting** means that the connection is unhealthy, possibly due to network issues. When a path goes into **connecting** status, the array generates an alert citing a degraded connection, which may result in replication issues. Check the network connectivity between the replication interfaces.

The **purearray list --controller** command displays the name, mode, FlashArray model, Purity//FA version, and status information for the array's controllers. The following are the controller modes:

**primary**

Running as the primary controller.

**secondary**

Running as the secondary controller.

**not present**

Not installed.

With FA-400 systems, also represents a controller that is installed but not running Purity//FA at the moment (including a controller that is powered off).

**offline**

Installed but not running Purity//FA at the moment. For example, represents a controller that is powered off or is undergoing a Purity//FA or firmware upgrade.

This status does not apply to FA-400 systems, where physical presence could not reliably be determined.

The Status column contains one of the following:

**ready**

Functioning normally.

**not ready**

In the process of starting Purity//FA.

updating

Undergoing a firmware upgrade. This status does not apply to FA-400 systems.

unknown

Status could not be determined.

For arrays to which a second controller has never been connected, only a single controller is displayed. Once an additional controller has been connected, its information is always shown.

The **purearray list --space** command displays current physical storage capacity, both total configured and occupied by data. Include the **--historical** option to display historical space data at the specified resolution.

The **purearray monitor** command displays real-time and historical I/O performance information for the whole array. The output includes the following data about bandwidth, IOPS, latency, and queue depth:

- **Time**: Current time.
- **B/s**: Bandwidth. Number of bytes read/written.
- **op/s**: IOPS. Number of read, write, or mirrored write requests processed per second.
- **us/op**: Service time. Average time, measured in microseconds, it takes the array to serve a read, write, or mirrored write I/O request. Service time does not include SAN time, queue time, or QoS rate limit time.
- **B/op**: IOPS. Average I/O size per read, write, mirrored write, and both read and write (all) operations. Must include the **--size** option to see the B/op columns.
- **Queue us/op**: Queue time. Average time, measured in microseconds, that a read, write, or mirrored write I/O request spends in the array waiting to be served.
- **Depth**: Queue depth. Average number of queued I/O requests for all volumes.

By default, the **purearray monitor** command displays real-time performance data. Include the **--repeat** option to specify the number of times to repeat the real-time update. If not specified, the repeat value defaults to 1. Include the **--interval** option to specify the number of seconds between each real-time update. At each interval, the system displays a point-in-time snapshot of the performance data. If not specified, the interval value defaults to every 5 seconds. The **--interval** and **--repeat** options can be combined.

In a synchronous replication configuration, include the **--mirrored** option to display performance data for mirrored writes (i.e., writes to volumes in stretched pods). If one array of a stretched pod is offline, the writes processed of the other array are reported as local writes rather than mirrored ones. Once the pod is back online, in sync, and processing each write on both sides, the writes to its volumes are reported as mirrored writes again.

The **purearray monitor --historical** command displays historical performance data over any of the following ranges of time: 1 hour, 3 hours, 24 hours, 7 days, 30 days, 90 days, or 1 year.

The **purearray monitor --latency** command displays real-time and historical I/O latency information across the entire array. The **purearray monitor --latency** output includes the following data:

- **Time**: Current time.
- **SAN us/op**: SAN time. Average time, measured in microseconds, required to transfer data between the initiator and the array. Slow data transfers will result in higher SAN times.
- **Queue us/op**: Queue time. Average time, measured in microseconds, that an I/O request spends in the array waiting to be served. The time is averaged across all I/Os of the selected types.
- **QoS Rate Limit us/op**: QoS rate limit time. Average time, measured in microseconds, that all reads, writes, or mirrored writes spend in queue as a result of bandwidth limits reached on one or more volumes.
- **us/op**: Service time. Average time, measured in microseconds, it takes the array to serve read, write, or mirrored write I/O requests across all volumes. Service time does not include SAN time, queue time, or QoS rate limit time.

Include the **--mirrored** option to display latency data for mirrored writes (i.e., writes to volumes in stretched pods).

The **purearray phonehome** subcommand initiates immediate transmission of event logs stored in an array to Pure Storage support. Options determine whether the current day, previous day, or all log information in the array are transmitted. Only one transmission may be in progress at any instant. A transmission in progress can be canceled by specifying the **--cancel** option in the subcommand. The phonehome facility automatically selects either the ssh or the https protocol for communicating with Pure Storage support. If a proxy host is required by https, use the **purearray setattr --proxy** command to specify its hostname and port. Specify the full URL string as described in the OPTIONS section above.

Purity//FA can transmit an array's log contents to Pure Storage Support (a) automatically once per hour, or (b) on demand. Pure Storage Support stores log contents for immediate or later analysis.

On-demand log transmission may be complete (**--send-all** option) or selective (**--send-today** or **--send-yesterday** options). Run the **--status** option to check the status of an in-progress log transmission. Run the **--cancel** option to cancel an in-progress log transmission.

Purity//FA can collect samples of hardware status, monitored performance, and array configuration on demand, and either (a) mail them to enabled alert recipients, (b) write them into the array's log, or (c) transmit them immediately to Pure Storage Support for analysis.

The **purearray remoteassist** subcommand enables or terminates a detached remote assistance session with Pure Storage Support. When the **purearray remoteassist --connect** command enables a session, a port ID is displayed. The administrator relays this ID to a Pure Storage Support representative, usually by phone, to enable the technician to connect to the array and perform diagnostic functions. The administrative session from which the **purearray remoteassist --connect** command is issued is unaffected, and can terminate the session at any time via the **purearray remoteassist --disconnect** subcommand.

One remote assistance session can be active at a time. Executing the `purearray remoteassist --connect` command while a session is active results in an error message. The `purearray remoteassist --status` command displays remote assistance status (capability disabled, capability enabled, or session active).

Use the `purearray rename` command to change an array's name. See the Object Names section below for information on Purity//FA object naming rules.

Use the `purearray setattr` command to change other attributes. To set a null value, use an empty string ("").

If Rapid Data Locking was enabled during `puresetup`, the `purearray replace` command substitutes new security tokens for those currently in use. Additional security tokens should be connected to both controllers prior to issuing the command. The command updates flash module keys using the passphrase on the new security tokens.

Purity//FA can test certain array functionality on demand. The `purearray test ntp` command verifies that the configured NTP servers can be used to synchronize time on both controllers with Coordinated Universal Time (UTC). The `purearray test pinghome` command tests an array's ability to connect with Pure Storage Support by establishing a secure shell or secure http connection. The `purearray test phonehome` command extends the `purearray test pinghome` command by additionally verifying that test messages can be exchanged. The `purearray test syslogserver` command sends a test message to configured syslog servers.

The `purearray test security-token` command tests the functionality of attached security tokens by regenerating and displaying an enhanced secret used to encrypt flash modules. The command verifies that attached security tokens are accessible and contain the passphrase used to generate the flash module keys currently in use. The command returns an error if:

- Rapid Data Locking is not enabled (configured during `puresetup`).
- The security token is not present or not functional.
- The attached security token does not contain the correct passphrase.

For more information about security tokens and the Rapid Data Locking (RDL) feature, refer to the FlashArray Enhanced Data Security Guide in the Pure1 Knowledge site at <https://support.purestorage.com>.

## Sender Domain and Purity//FA Logs

The `purearray setattr --senderdomain` command determines how Purity//FA logs are parsed and treated by Pure Storage Support and Escalations. It is crucial that you set the sender domain to the correct domain name.

If the array is not a Pure Storage test array, set the sender domain to the actual customer domain name. For example, `mycompany.com`.

If the sender domain is set to the default `please-configure.me` domain name, the array will be treated as an internal Pure Storage test array, potentially blocking alerts and other important warnings.

Purity//FA also includes the sender domain name in the email addresses of outgoing alert messages. For example, `purearray-ct0@mycompany.com`.

Run `purearray list --senderdomain` to display the sender domain name.

## Connecting Two Arrays

To transfer data from one array to another, the two arrays must be connected. For example, arrays must be connected to replicate data from one array to another.

Arrays are connected using a connection key, which is supplied from one array and entered into the other array.

When two arrays are connected, the array where data is being transferred from is called the source (local) array, and the array where data is being transferred to is called the target (remote) array. A source array can connect to multiple target arrays.

Connecting two arrays for data replication involves the following steps:

1. From the target array, obtain the connection key (`purearray list --connection-key`).
2. From the source array, connect the source array to the target array (`purearray connect --management-address --type --connection-key`).
3. From the source array, create the protection group (`purepgroup create`), configure the protection group replication schedule (`purepgroup schedule`), and enable the protection group replication schedule (`purepgroup enable`).

The `purearray connect --management-address --type --connection-key` command connects the local array to a remote array. The `--management-address` option represents the virtual (`vir0`) IP address or FQDN of the target array. The `--connection-key` option represents the connection key of the remote array. To obtain the connection key, run `purearray list --connection-key` from the remote array. The `-type` option represents the type of connection. Purity//FA supports connection type `replication`. Include the `--replication-address` option, which represents the IP address or FQDN of the replication bond (`replbond`) interface on the target array.

Once two arrays are connected, optionally configure network bandwidth throttling to set maximum threshold values for outbound traffic.

The `purearray disconnect` command disconnects the local array from the specified remote array. The command can only be run on the array that established the connection. Disconnecting two arrays suspends any in-progress data transfer processes. The process resumes when the arrays are reconnected.

If two arrays are connected for replication purposes, disconnecting the arrays does not delete data from the replication target array. However, after the two arrays have been disconnected, removing a target array from a protection group immediately deletes all of the data on the target array that was sent from its source array. As a result, replicating data again from the source array to the target array would cause another baseline transfer.

## Network Bandwidth Throttling

Once two arrays are connected, optionally configure **purearray throttle --connect** to regulate when and how much data should be transferred between the arrays. The **purearray throttle --connect** command must be used with at least one of the following options: **--default-limit**, **--window-limit**, or **--window**.

Network bandwidth throttling on the source array sets a network bandwidth cap for the amount of outbound data, measured by number of bytes per second, that can be transferred from the local array to a target array before network bandwidth throttling occurs.

Two different network bandwidth limits can be set:

- The **--default-limit** option sets a default maximum network bandwidth threshold for outbound traffic.

and/or

- The **--window-limit** option sets a maximum network bandwidth threshold for outbound traffic during the specified **--window** time range.

If both thresholds are set, the “window” limit overrides the “default” limit.

The maximum threshold must be between one megabyte per second and four gigabytes per second. The maximum threshold value represents an average data rate, so actual data transfer rates can fluctuate slightly above the configured limit.

The **--window-limit** setting overrides the **--default-limit** setting.

The **--default-limit** and **--window-limit** values are specified as integers, followed by one of the suffix letters **M** or **G**, denoting 512-byte sectors MiB and GiB, respectively.

The **--window** start and end times must be set on the hour, and the time can be specified in 12-hour or 24-hour format.

For example, run the following command to set a default network bandwidth cap of 4 gigabytes per second of data to be transferred from the local array to array **ARRAY1**, except between the hours of 11:00am and 3:00pm, when the network bandwidth limit drops to 2 gigabytes per second.

```
$ purearray throttle --connect --default-limit 4gb  
                  --window 11am-3pm --window-limit 2gb ARRAY1
```

To clear the **--default-limit**, **--window-limit**, or **--window** settings, use an empty string ("").

To completely stop the data transfer process, set the **default-limit** or **window-limit** option to "0". During this time, all in-progress and scheduled data transfer processes are aborted.

For example, run the following command to stop all data transfers from the local array to **ARRAY1** between 9:00am and 5:00pm.

```
$ purearray throttle --connect --window 9am-5pm --window-limit 0 ARRAY1
```

Bandwidth limit changes take effect immediately.

## End User Agreement (EULA)

The Pure Storage End User Agreement (EULA) represents a contract between Pure Storage and users of Pure Storage products. The most recent version of the agreement governs use of the FlashArray hardware and software and can be found at <http://www.purestorage.com/legal/productenduserinfo.html>.

Run the **purearray eula list** command to view the terms of the Pure Storage End User Agreement. Include the **--acceptance** option to view the acceptance details for the Pure Storage End User Agreement, including the name and title of the individual at the company who accepted the terms of the agreement, company name, and date and time when the agreement was accepted. If the agreement has not been accepted, all the columns, including the Accepted column, of the output display dashes (-).

Run the **purearray accept** command to accept the terms of the agreement. Only array administrators (i.e., users with the Array Admin role) have the necessary permissions to run the command.

During the prompt-based acceptance process, if any of the prompts are left blank, Purity//FA will revert to the previous signor's name, title, and company name.

Accepting the agreement requires the following information:

- **Name** - Full legal name of the individual at the company who has the authority to accept the terms of the agreement.
- **Title** - Individual's job title at the company.
- **Company** - Full legal name of the entity.

The name, title, and company name must each be between 1 and 64 characters in length.

If the agreement is not accepted, Purity//FA generates an alert notifying all Purity//FA alert watchers that the agreement is pending acceptance. A warning alert also appears in the Purity//FA GUI. Pure Storage is not notified of the alert. The alert remains open until the agreement is accepted. Furthermore, whenever a user logs in to Purity//FA GUI, the End User Agreement window pops up as a reminder that the agreement is pending acceptance.

Once the terms of the agreement have been accepted, Purity//FA closes the alert and stops generating the End User Agreement pop-up window.

## Note on Array Time

The installation technician sets the proper time zone for an array when it is installed. During operation, arrays maintain time synchronization by interacting with a Network Time Protocol (NTP) server (by default, time.purestorage.com). The **purearray setattr --ntpserver NTP-SERVER** command can be executed to specify an alternate NTP server by IP address or hostname.

## Examples

### Example 1

```
purearray enable phonehome
```

Enables automatic hourly transmission of log contents to the Pure Storage Support web site.

#### Example 2

```
purearray diagnose --phonehome
```

Takes a snapshot of performance, configuration, and hardware status and transmits it to Pure Storage Support via the phonehome channel.

#### Example 3

```
purearray list --phonehome
```

Displays the current state of the phonehome automatic hourly log transmission facility (enabled or disabled).

#### Example 4

```
purearray list --space --historical 3h --csv
```

Displays a CSV output of the amount of usable physical storage on the array and the amount of storage occupied by data and metadata over the past 3 hours.

#### Example 5

```
purearray list --throttle --connect
```

Displays network bandwidth limits, if set, for all connected arrays.

#### Example 6

```
purearray monitor --repeat 20 --interval 10
```

Displays real-time performance data for the whole array. Twenty (20) point-in-time updates are displayed, with each update taken every ten (10) seconds.

#### Example 7

```
purearray monitor --historical 24h --csv
```

Displays a CSV output of historical performance data for the local array over the past 24 hours.

#### Example 8

```
purearray phonehome --send-today
```

Initiates immediate transmission of the current day's event logs to Pure Storage Support via the phonehome network channel.

#### Example 9

```
purearray remoteassist --connect
```

Enables a Pure Storage Support representative to initiate a remoteassist session with the array.

#### Example 10

```
purearray rename NEWARRAYNAME
```

Changes the name of an array to **NEWARRAYNAME**.

#### Example 11

```
purearray setattr --ntpserver MyNTPServer1.com,MyNTPServer2.com
```

Assigns the NTP servers **MyNTPServer1.com** and **MyNTPServer2.com** as the array's sources for reference time. Supersedes any previous NTP server assignments.

#### Example 12

```
purearray setattr --relayhost ""
```

Sets the relayhost attribute value to null, causing Purity//FA to send alert email messages directly to recipient addresses rather than routing them via a relay (mail forwarding) server.

#### Example 13

```
purearray setattr --relayhost fake.relay.purestorage.com:26  
purealert test example@purestorage.com
```

Sets up a non-standard SMTP relay port number and sends a test alert.

#### Example 14

```
purearray setattr --senderdomain mycompany.com
```

Sets the sender domain name to **mycompany.com**.

#### Example 15

```
purearray list --connection-key
```

Displays the connection key of the array.

#### Example 16

```
purearray connect --management-address 10.78.0.188 --type replication  
--connection-key 6207d123-d123-0b5c-5fa1-95fab5c7123
```

Connects the local array to remote array **10.78.0.188** using connection key **6207d123-d123-0b5c-5fa1-95fab5c7123** (taken from array **10.78.0.188**).

#### Example 17

```
purearray disconnect 10.78.0.188
```

Disconnects array 10.78.0.188 from the local (current) array.

#### Example 18

```
purearray throttle --connect --default-limit 2mb  
--window 11pm-5am --window-limit 4mb ARRAY1
```

Sets the default maximum network bandwidth limit for outbound traffic to 2 MB/s for array **ARRAY1**, and sets the maximum network bandwidth limit for outbound traffic to 4 MB/s between 11:00pm and 5:00am.

#### Example 19

```
purearray eula list
```

Displays the terms of the Pure Storage End User Agreement.

#### Example 20

```
purearray eula list --acceptance
```

Displays the acceptance details for the Pure Storage End User Agreement, including the name and title of the individual at the company who accepted the terms of the agreement, company name, and agreement acceptance date.

#### Example 21

```
purearray eula accept
```

Displays a series of command prompts for the current array administrator to complete in order to accept the Pure Storage End User Agreement. The individual who accepts the terms of the agreement must have the authority to do so.

### See Also

[purealert\(1\)](#) [218], [puredns\(1\)](#) [257]

[purehgroup\(1\)](#) [275], [purehost\(1\)](#) [292], [purepgroup\(1\)](#) [381], [purepod\(1\)](#) [412],  
[purevol\(1\)](#) [469]

### Author

Pure Storage Inc. <[documentfeedback@purestorage.com](mailto:documentfeedback@purestorage.com)>

## pureaudit

pureaudit, pureaudit-list — displays and manages audit records

### Synopsis

```
pureaudit list [ --csv | --nvp ] [--notitle] [--page] [--raw] [--filter FILTER]  
[--limit LIMIT] [--sort SORT]
```

### Options

**-h** | **--help**

Can be used with any command or subcommand to display a brief syntax description.

Options that control display format:

**--csv**

Lists information in comma-separated value (CSV) format. The **--csv** output can be used for scripting purposes and imported into spreadsheet programs.

**--notitle**

Lists information without column titles.

**--nvp**

Lists information in name-value pair (NVP) format, in the form **ITEMNAME=VALUE**. Argument names and information items are displayed flush left. The **--nvp** output is designed both for convenient viewing of what might otherwise be wide listings, and for parsing individual items for scripting purposes.

**--page**

Turns on interactive paging.

**--raw**

Displays the unformatted version of column titles and data. For example, in the **purearray monitor** output, the unformatted version of column title **us/op (read)** is **usec\_per\_read\_op**. The **--raw** output is used to sort and filter list results.

Options that manage display results:

**--filter**

Displays only the rows that meet the filter criteria specified.

**--limit**

Limits the size of the list output to the specified maximum number of rows.

**--sort**

Sorts the list output in ascending or descending order by the column specified.

## Description

Purity//FA generates *audit trail* records, which represent administrative actions performed by a Purity//FA user to modify the configuration of the array. For example, creating, destroying, or eradicating a volume creates an audit record.

The **pureaudit list** command displays a list of audit records. For example,

pureaudit list							
ID	Time	User	Command	Subcommand	Name	Arguments	
250	2019-03-25 03:00:40 PDT	root	purevol	disconnect	vol3	--host host3	
251	2019-03-25 03:00:41 PDT	root	purevol	connect	vol3	--host host3	
252	2019-02-27 12:57:30 PST	pureuser	purehost	setattr	host1	--addwwnlist 21000024F	
253	2019-03-15 23:45:42 PDT	os76	purevol	disconnect	vol5	--host host2	
254	2019-02-24 18:56:21 PST	pureuser	purevol	create	vol4	--size 10T	
255	2018-12-09 17:14:14 PST	root	purevol	destroy	vol1		
256	2018-12-09 17:14:23 PST	root	purevol	eradicate	vol1		

The **pureaudit list** command displays the following audit record details:

### ID

Unique number assigned by the array to the audit record. ID numbers are assigned to audit records in chronological ascending order.

### Time

Date and time the action occurred to trigger the audit event.

### User

User name of the Purity//FA user who issued the command.

### Command

Purity//FA CLI command that was issued.

### Subcommand

Purity//FA CLI subcommand that was issued.

### Name

Name of the object on which the command was issued.

### Arguments

Options that were included with the Purity//FA CLI command that was issued.

## Examples

### Example 1

```
pureaudit list --filter "name = 'hgroup*'"
```

Displays a list of audit records of host groups with the prefix of "hgroup" in the host group name.

## Exceptions

None.

## See Also

[purealert\(1\)](#) [218], [purearray\(1\)](#) [230], [puremessage\(1\)](#) [354]

## Author

Pure Storage Inc. <[documentfeedback@purestorage.com](mailto:documentfeedback@purestorage.com)>

## purecert

purecert, purecert-construct, purecert-create, purecert-list, purecert-setattr — creates and manages FlashArray SSL certificate

### Synopsis

```
purecert construct [--common-name COMMON_NAME] [--country COUNTRY] {--certificate-signing-request} [--email EMAIL] [--locality LOCALITY] [--organization ORG] [--organizational-unit ORG_UNIT] [--state STATE] NAME
purecert create [--common-name COMMON_NAME] [--country COUNTRY] [ --certificate | --self-signed ] [--certificate-signing-request] [--days DAYS] [--email EMAIL] [--intermediate-certificate] [--key] [--key-size KEY_SIZE] [--locality LOCALITY] [--organization ORG] [--organizational-unit ORG_UNIT] [--passphrase] [--state STATE] NAME
purecert list [ --certificate | --intermediate-certificate ] [ --cli | --csv | --nvp ] [--notitle] [--page] [--raw] NAME
purecert setattr [--common-name COMMON_NAME] [--country COUNTRY] [ --certificate | --self-signed ] [--certificate-signing-request] [--days DAYS] [--email EMAIL] [--intermediate-certificate] [--key] [--key-size KEY_SIZE] [--locality LOCALITY] [--new-key] [--organization ORG] [--organizational-unit ORG_UNIT] [--passphrase] [--state STATE] NAME
```

### Arguments

#### **NAME**

Name of the certificate to be created, modified, or displayed.

### Options

#### **-h | --help**

Can be used with any command or subcommand to display a brief syntax description.

#### **--certificate**

Displays, exports, or imports the certificate. This option is mutually exclusive with the **--self-signed** option.

#### **--certificate-signing-request**

Constructs a certificate signing request.

#### **--common-name**

Fully qualified domain name (FQDN) of the current array. For example, the common name for https://purearray.example.com is purearray.example.com, or \*.example.com for a wildcard certificate. The common name can also be the management IP address of the array or the short name of the current array. Common names cannot have more than 64 characters.

--country

Country name. Two-letter ISO code for the country where the organization is located.

--days(**purecert setattr --self-signed** only)

Number of valid days for the self-signed certificate being generated. If not specified, the self-signed certificate expires after 3650 days.

--email

Email address used to contact the organization.

--intermediate-certificate

Displays, exports, or imports the intermediate certificate.

--key

Imports the private key.

--key-size(**purecert setattr --self-signed** only)

Specifies key size in bits. Valid values are 1024, 2048 (default), or 4096. A key size smaller than 2048 is considered insecure.

--locality

The city where the organization is located.

--new-key(**purecert setattr --self-signed** only)

Generates a new private key when creating the self-signed certificate. If a new private key is not generated, the certificate uses the existing private key.

--organization

Full and exact name in which the organization is legally registered. Organization name should not be abbreviated and should include suffixes such as Inc, Corp, or LLC.

--organizational-unit

Name of the department within the organization that is managing the certificate.

--passphrase

Passphrase used to decrypt the private key.

--self-signed

Generates a self-signed certificate. This option is mutually exclusive with the **--certificate** option.

--state

Full name of the state or province where the organization is located.

Options that control display format:

--cli

Displays output in the form of CLI commands that can be issued to reproduce the current configuration. The **--cli** output is not meaningful when combined with immutable attributes.

--csv

Lists information in comma-separated value (CSV) format. The --csv output can be used for scripting purposes and imported into spreadsheet programs.

--notitle

Lists information without column titles.

--nvp

Lists information in name-value pair (NVP) format, in the form **ITEMNAME=VALUE**. Argument names and information items are displayed flush left. The --nvp output is designed both for convenient viewing of what might otherwise be wide listings, and for parsing individual items for scripting purposes.

--page

Turns on interactive paging.

--raw

Displays the unformatted version of column titles and data. For example, in the **purearray monitor** output, the unformatted version of column title **us/op (read)** is **usec\_per\_read\_op**. The --raw output is used to sort and filter list results.

## Description

The **purecert** command allows you to perform tasks to create and manage your Pure Storage SSL certificates. Purity//FA creates a self-signed certificate and private key when you start the system for the first time. You can use the default certificate, change the certificate attributes, create a new self-signed certificate, or import an SSL certificate signed by a certificate authority.

### Self-signed Certificate

You can create a new self-signed certificate on the current array by using the **purecert create --self-signed** command. The command generates a certificate object that stores the private key and a certificate containing the public key. You can then submit this certificate to a certificate authority (CA) to obtain a CA certificate. To display the self-signed certificate, use the **purcert list --certificate** command.

When you create a self-signed certificate to replace the current certificate, you can also include the --key-size option to generate a private key with the specified key size. The default key size is 2048 bits.

By default, self-signed certificates are valid for 3650 days. Include the --days option to change the valid number of days.

### Certificate Signing Request

Signing in with a certificate from a certificate authority (CA) involves importing an SSL certificate issued by the CA.

To obtain a CA certificate, you must first construct a certificate signing request (CSR) on the array. The CSR represents a block of encrypted data specific to your organization. Run the **purecert construct --certificate-signing-request** command to construct a CSR.

You can change the certificate attributes when you construct the CSR; otherwise, Purity//FA will reuse the attributes of the current certificate (self-signed or imported) to generate the new one. Note that the certificate attribute changes will only be visible after you import the signed certificate from the CA.

Send the CSR to a certificate authority for signing. The certificate authority returns the SSL certificate for you to import. Verify that the signed certificate is PEM formatted (Base64 encoded), includes the "-----BEGIN CERTIFICATE-----" and "-----END CERTIFICATE-----" lines, and does not exceed 3000 characters in length. If the intermediate certificate is bundled with the SSL certificate, run the `purecert setattr --certificate` to import the certificates. If the certificates are sent separately, run the `purecert setattr --certificate --intermediate-certificate` to import the SSL certificate and its accompanying intermediate certificate. The certificates are entered interactively.

If the certificate is signed with the CSR that was constructed on the current array and you did not change the private key, you do not need to import the key. However, if the CSR was not constructed by the current array, include the `--key` option to import the private key. If the private key is encrypted, include the `--passphrase` option with `--key`.

## Changing the Certificate

To change the certificate attributes, run the `purecert setattr` command. When you change the attributes of a self-signed certificate, Purity//FA replaces the existing certificate with a new certificate, along with its specified attributes. Certificate attributes include organization-specific information, such as country, state, locality, organization, organizational unit, common name, and email address.

You can also generate a new private key for the current certificate by using the `--new-key` option. To change the length (in bits) of the private key, include the `--key-size` option with the `--new-key` option.

To list the attributes of the current certificate, run the `purecert list` command.

## Certificate Administration

To import the certificate signed by a certificate authority (CA), run the `purecert setattr --certificate`.

To export a certificate or an intermediate certificate, such as for backup purposes, run the `purecert list --certificate` or `purecert list --intermediate-certificate` command, respectively.

## Examples

### Example 1

```
purecert setattr cert_1 --self-signed --common-name db.example.com --country US --state CA  
--locality 'Mountain View' --organization 'Example, Inc.'
```

Modifies the existing self-signed certificate `cert_1` with the common name `db.example.com` (the FQDN of the current array) and various attributes. Because the private key is not specified, the new self-signed certificate will use the existing private key.

## Example 2

```
purecert construct cert_2 --certificate-signing-request --common-name app.example.com
```

Constructs a certificate signing request with the common name app.example.com (the FQDN of the current array) and all the existing attributes.

## Example 3

```
purecert setattr cert_3 --certificate --intermediate-certificate
```

Imports the certificate signed by a certificate authority (CA) along with its intermediate certificate.

## Example 4

```
purecert create DSM_cert --self-signed --common-name c14-v7-w11  
purecert list DSM_cert --certificate
```

Creates a new DSM\_cert self-signed certificate with the common name **c14-v7-w11**, and then displays the certificate.

## Example 5

```
purecert create my_cert --self-signed  
purecert construct my_cert --certificate-signing-request --common-name c14-v7-w11
```

Creates a new certificate **my\_cert** containing a private and public key pair, and then generates the certificate signing request with the common name **c14-v7-w11**.

## Author

Pure Storage Inc. <[documentfeedback@purestorage.com](mailto:documentfeedback@purestorage.com)>

## pureconfig

pureconfig — displays commands required to reproduce an array's current volume, host, host group, connection, network, alert, and array configuration

### Synopsis

```
pureconfig list
```

### Options

None

### Description

Displays an array's current configuration of volumes, hosts, shared and private connections, administrative network parameters, alert email addresses, and array parameters in the form of CLI commands that would be required to reproduce the configuration on a newly-installed or otherwise unconfigured array. (The output does not contain delete or destroy subcommands).

The output of this command can be captured on the administrative workstation and used as a script to configure a previously unconfigured array identically to the array on which the command is executed.

Executing the **pureconfig list** command is roughly equivalent to executing the following commands in sequence:

```
purevol list --cli  
purehost list --cli  
purehgroup list --cli  
purehost list --connect --cli  
purehgroup list --connect --cli  
purenetwork list --cli  
purealert list --cli  
purearray list --cli
```

### See Also

[purealert\(1\)](#) [218], [purearray\(1\)](#) [230], [purehgroup-list\(1\)](#) [287], [purehost-list\(1\)](#) [307], [purenetwork\(1\)](#) [360], [purevol-list\(1\)](#) [488]

### Author

Pure Storage Inc. <[documentfeedback@purestorage.com](mailto:documentfeedback@purestorage.com)>

## puredns

puredns — manages the DNS attributes of an array

### Synopsis

```
puredns list [ --cli | --csv | --nvp ] [--notitle] [--page] [--raw]
```

```
puredns setattr [--domain DOMAIN] [--nameservers NAMESERVERS]
```

### Options

**-h | --help**

Can be used with any command or subcommand to display a brief syntax description.

**--domain**

Domain suffix to be appended by the array when performing DNS lookups. To remove the domain suffix from Purity//FA DNS queries, set to an empty string ("").

**--nameservers**

Comma-separated list of up to three DNS server IP addresses.

For IPv4, specify the IP address in the form **ddd.ddd.ddd.ddd**, where **ddd** is a number ranging from 0 to 255 representing a group of 8 bits. For IPv6, specify the IP address in the form **xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx**, where **xxxx** is a hexadecimal number representing a group of 16 bits. When specifying an IPv6 address, consecutive fields of zeros can be shortened by replacing the zeros with a double colon (::).

To unassign the DNS server IP addresses, set to an empty string (""). Once the DNS server IP addresses have been unassigned, the array no longer makes DNS queries.

Options that control display format:

**--cli**

Displays output in the form of CLI commands that can be issued to reproduce the current configuration. The **--cli** output is not meaningful when combined with immutable attributes.

**--csv**

Lists information in comma-separated value (CSV) format. The **--csv** output can be used for scripting purposes and imported into spreadsheet programs.

**--notitle**

Lists information without column titles.

**--nvp**

Lists information in name-value pair (NVP) format, in the form **ITEMNAME=VALUE**. Argument names and information items are displayed flush left. The **--nvp** output is designed both for convenient viewing of what might otherwise be wide listings, and for parsing individual items for scripting purposes.

**--page**

Turns on interactive paging.

--raw

Displays the unformatted version of column titles and data. For example, in the `purearray monitor` output, the unformatted version of column title `us/op (read)` is `usec_per_read_op`. The --raw output is used to sort and filter list results.

## Description

The `puredns` command manages the DNS attributes for an array's administrative network.

The `puredns setattr` command sets the DNS parameters of the array. Include the `--domain` option to set the domain suffix to be appended to DNS queries. The list of DNS name server IP addresses specified in the `--nameservers` option replaces the list of name servers in effect prior to command execution.

The `puredns list` command displays the current DNS parameters, including domain suffix and name servers, of the array. Include the `--cli` option to display the CLI command line that would reproduce the array's current DNS configuration. This can, for example, be copied and pasted to create an identical DNS configuration in another array, or saved as a backup.

## Examples

### Example 1

```
puredns setattr --domain mydomain.com --nameservers 192.168.0.125,192.168.2.125
```

Specifies the IPv4 addresses of two DNS servers for Purity//FA to use to resolve hostnames to IP addresses, and the domain suffix mydomain.com for DNS searches.

### Example 2

```
puredns setattr --nameservers 192.168.0.125,2001:0db8:85a3::ae26:8a2e:0370:7334
```

Specifies the IP addresses of two DNS server for Purity//FA to use to resolve hostnames to IP addresses.

### Example 3

```
puredns setattr --domain ""
```

Removes the domain suffix from Purity//FA DNS queries.

### Example 4

```
puredns setattr --nameservers ""
```

Unassigns DNS server IP addresses (Purity//FA ceases to make DNS queries).

## See Also

`purearray(1)` [230], `purenw(1)` [360]

**Author**

Pure Storage Inc. <[documentfeedback@purestorage.com](mailto:documentfeedback@purestorage.com)>

## puredrive

puredrive, puredrive-admit, puredrive-list — display and manage an array's flash and NVRAM modules

### Synopsis

**puredrive** admit

**puredrive** list [ --csv | --nvp ] [--notitle] [--page] [--raw] [--spec] [--total] [**DRIVE**...]

### Arguments

#### **DRIVE**

Name of a flash module or NVRAM module about which information is to be displayed. Includes the shelf identifier (for example SH1.BAY0, which designates the module in bay #0 of storage shelf #1).

### Options

-h | --help

Can be used with any command or subcommand to display a brief syntax description.

--spec

Displays hardware specifications for the module.

--total

Follows output lines with a single line containing column totals in columns where they are meaningful.

Options that control display format:

--csv

Lists information in comma-separated value (CSV) format. The --csv output can be used for scripting purposes and imported into spreadsheet programs.

--notitle

Lists information without column titles.

--nvp

Lists information in name-value pair (NVP) format, in the form **ITEMNAME=VALUE**. Argument names and information items are displayed flush left. The --nvp output is designed both for convenient viewing of what might otherwise be wide listings, and for parsing individual items for scripting purposes.

--page

Turns on interactive paging.

--raw

Displays the unformatted version of column titles and data. For example, in the `purearray monitor` output, the unformatted version of column title `us/op (read)` is `usec_per_read_op`. The --raw output is used to sort and filter list results.

## Description

The FlashArray array consists of flash SSD (solid state drive) and NVRAM (non-volatile random access memory) modules. Flash modules are used for the persistent storage of user data, while NVRAM modules are used as non-volatile write caches. The `puredrive` command manages the flash and NVRAM modules of an array.

The `puredrive list` command lists an array's flash and NVRAM modules. The list output includes the following information:

### Name

Name by which Purity//FA identifies the module in administrative operations. Flash and NVRAM module names identify physical locations in terms of shelf and bay numbers.

A bay with the name (`unknown`) means that something unexpected has happened to the bay and the array is attempting to bring it back. If the module remains in `unknown` status, contact Pure Storage Support.

### Type

Drive type. Possible drive types include `ssd` (Solid State Drive) and `nvramp` (Non-Volatile Random-Access Memory).

A dash (-) in the Type column represents a module that this array does not recognize. The module is initiated as a FlashArray module and is functional, but it does not belong to this array. For example, the module may have been pulled from another array and mistakenly placed into this array.

### Status

Condition of the module. Possible module statuses include:

#### healthy

The module is functioning and responds to I/O commands.

#### unadmitted

The module has been added to the bay, and it is waiting to be admitted into the array. Run `puredrive admit` to admit all of the unadmitted modules, including this one. Once a module has been successfully admitted, it changes to `healthy` status. If the module returns to `unadmitted` status after the admission process, run the `puredrive admit` command again. If the subsequent "admit" attempt is not successful, contact Pure Storage Support.

#### unused

The bay does not have any modules. New modules can be added to this empty bay.

#### unrecognized

The module is initiated as a FlashArray module and is functional, but it does not belong to this array. For example, the module may have been pulled from another array and mistakenly placed into this array.

**unhealthy**

The module is not working as intended, but it may still be functional. Contact Pure Storage Support.

**missing**

The array is expecting a module to be in the bay, but the module is missing. Contact Pure Storage Support.

**failed**

The module has experienced a failure and is not responding to I/O. As a result, Purity//FA is evacuating or has evacuated data off the module. Contact Pure Storage Support.

Modules might also go into any of the following temporary statuses:

**identifying**

The module is in the process of coming online as Purity//FA tries to identify the drive. The module is not yet ready to respond to I/O commands. The **identifying** status is transitory and will eventually change to **unadmitted** or **healthy** status. If the module remains in **identifying** status for more than 10 minutes, contact Pure Storage Support.

**recovering**

The module is undergoing a drive profiling procedure or rehabilitation. The **recovering** status is transitory and will eventually change to **healthy** status. If the module remains in **recovering** status or transitions to **failed** status, contact Pure Storage Support.

**updating**

A firmware update is taking place. The updating status is transitory and will eventually return to **healthy** status. If the module does not return to **healthy** status, contact Pure Storage Support.

**Capacity**

Physical storage capacity of the module.

**Evac Remaining**

For a missing module, the amount of data that remains to be evacuated (reconstructed and stored on other flash modules).

**Last Failure**

Time at which a module became non-responsive.

**Last Evac Completed**

Time at which the evacuation of data from a non-responsive module completed.

**Protocol**

Storage protocol, such as NVMe and SAS. Must include the **--spec** option to display the Protocol column.

## Capacity Upgrade and Drive Admission

Increase storage capacity by adding data packs or individual drives to a shelf.

Data packs can be added to any shelf that has open space.

Individual drives can be added to any unused slot within a shelf. The drives must be DirectFlash modules, and the shelf must already contain similar-sized DirectFlash modules. The array will not admit individual drives of other types or sizes. Add each drive to the unused slots from left to right, starting with the lowest open slot.

Before you begin the upgrade, ensure your shelf has enough open spaces to hold the new packs or individual drives.

Performing a capacity upgrade is a two-step process: first, add the modules to the array; and second, admit the newly added modules.

When a module has been added to the array, its status changes from `unused` to `identifying` as Purity works to identify the module. The module transitions to its final `unadmitted` status when it has been successfully added to the array.

Run the `puredrive list` command to verify that the newly added shelves and drives are in `unadmitted` status, indicating that the modules have been successfully connected but not yet admitted to the array.

After a module has been added (connected) to the array, it must be admitted.

Run the `puredrive admit` command to admit all modules that have been added (connected) but not yet admitted to the array.

Once a drive has been successfully admitted to the array, its module status changes from `unadmitted` to `healthy`.

Run the `puredrive list` command to verify that all of the shelves and drives are in `healthy` status, indicating that the modules have been successfully admitted and are in use by the system. This completes the drive admission process.

If issues arise during the admission process, the module status changes to `unrecognized` or `failed`, or reverts to `unadmitted`. If the module changes to `unrecognized` or `failed` status, contact Pure Storage Support. If the module returns to the `unadmitted` status, run the `puredrive admit` command again. If the subsequent “admit” attempt is not successful, contact Pure Storage Support.

## Examples

### Example 1

```
puredrive list SH0.BAY5
```

Lists information for the flash module in BAY5 of shelf SH0.

### Example 2

```
puredrive list CH0.NVB1
```

Lists information for the NVRAM module in NVRAM bay 1 of shelf CH0 (does not apply to FA-400 systems).

### Example 3

```
puredrive admit
```

**puredrive list**

Admits all modules that have been added (connected) to the array and are waiting to be admitted. Confirms that all modules that were in **unadmitted** status are now in **healthy** status, indicating successful admission.

**See Also**

n/a

**Author**

Pure Storage Inc. <[documentfeedback@purestorage.com](mailto:documentfeedback@purestorage.com)>

## pureds

pureds, pureds-disable, pureds-enable, pureds-list, pureds-role-list, pureds-role-setattr, pureds-setattr, pureds-test — manages FlashArray array integration with a directory service

### Synopsis

```
pureds disable [--check-peer]
pureds enable [--check-peer]
pureds list [--certificate] [ --cli | --csv | --nvp ] [--notitle] [--page] [--raw]
pureds role list [ --cli | --csv | --nvp ] [--filter FILTER] [--limit LIMIT] [--notitle] [--page] [--raw] [--sort SORT] [ROLE...]
pureds role setattr [--group GROUP] [--group-base GROUP-BASE] ROLE...
pureds setattr [--auto-fetch] [--base-dn BASE-DN] [--bind-password] [--bind-user BIND-USER] [--certificate] [--trust] [--uri URI-LIST]
pureds test
```

### Options

**-h | --help**

Can be used with any command or subcommand to display a brief syntax description.

**--array-admin-group **ARRAY-ADMIN-GROUP****

Common name (CN) of the directory service group containing administrators with full privileges to manage the array. The name should be just the common name of the group without the "CN=" specifier. If the configured groups are not in the same organization unit (OU), also specify the OU. For example, "pureadmins,OU=PureStorage", where pureadmins is the common name of the directory service group.

Note: The **--array-admin-group** option will be deprecated in a future release and should not be used. Instead, run the **pureds role setattr** command to define the LDAP groups for the **array\_admin** role.

**--auto-fetch**

Attempts to get CA certificate data from the configured URI or the first URI in the list if more than one URI is configured.

**--base-dn **BASE-DN****

Sets the base of the distinguished name (base DN) of the directory service groups. The base DN should consist of only domain components (DCs).

Specify "DC=" for each domain component. List multiple DCs in comma-separated format.

To clear the base DN setting, set to an empty string ("").

**--bind-password**

Sets the password of the bind user account. The password is entered through interactive prompt.

**--bind-user *BIND-USER***

Sets the user name used to bind to and query the directory.

For Active Directory, enter the user name - often referred to as sAMAccountName or User Logon Name - for the account that is used to perform directory lookups. The user name cannot contain the characters " [ ] : ; | = + \* ? < > / \, and cannot exceed 20 characters in length.

For OpenLDAP, enter the full DN of the user. For example, "CN=John,OU=Users,DC=example,DC=com".

**--certificate**

Sets the CA certificate data. The data is entered through interactive prompt. The data must be PEM formatted (Base64 encoded) and include the "-----BEGIN CERTIFICATE-----" and "-----END CERTIFICATE-----" lines. The certificate cannot exceed 3000 characters in total length. Press **Enter** and then **Control-d** to exit the interactive session. To clear the certificate, enter blank lines at the prompt.

**--check-peer**

Enables or disables server authenticity enforcement with the configured CA certificate. This option can only be enabled if CA certificate data has been provided. If this option is enabled and certificate data is cleared, it will revert back to disabled.

**--group *GROUP***

Sets the common name (CN) of the configured group in the directory tree that maps to a role in the array. The group name should be just the common name of the group without the "CN=" specifier. Common names cannot exceed 64 characters in length.

All users in a configured group must map to the same role in the array. For example, all users in configured LDAP group "purereadonly" must map to the same role, such as the **readonly** role, in the array.

To clear the group setting, set to an empty string ("").

**--group-base *GROUP-BASE***

Specifies where the configured groups are located in the directory tree. This field consists of organizational units (OU) that, when combined with the base DN and the configured group CNs, complete the full DN of each group.

Specify "OU=" for each OU. List multiple OUs in comma-separated format.

The order of OUs should get smaller in scope from right to left. For example, for "OU=PureGroups,OU=SANManagers", the "SANManagers" OU contains the sub OU "PureGroups".

To clear the group base setting, set to an empty string ("").

**--readonly-group *READONLY-GROUP***

Common name (CN) of the configured group in the directory tree containing users with read-only privileges on the array. The name should be just the common name of the group without the "CN=" specifier. If the configured groups are not in the same OU, also specify the OU. For example, "purereadonly,OU=PureStorage", where "purereadonly" is the common name of the directory service group.

Note: The `--readonly-group` option will be deprecated in a future release and should not be used. Instead, run the `pureds role setattr` command to define the LDAP groups for the `readonly` role.

**--storage-admin-group *STORAGE-ADMIN-GROUP***

Common name (CN) of the configured group in the directory tree containing administrators with storage related privileges on the array. The name should be the command name of the group without the "CN=" specifier. If the configured groups are not in the same OU, also specify the OU. For example, "pureusers,OU=PureStorage", where pureusers is the common name of the directory service group.

Note: The `--storage-admin-group` option will be deprecated in a future release and should not be used. Instead, run the `pureds role setattr` command to define the LDAP groups for the `storage_admin` role.

**--trust**

Skips the certificate chain trust verification.

**--uri *URI-LIST***

Sets the URIs of the directory servers. Up to 30 URIs can be specified. List multiple URIs in comma-separated format.

Each URI must include the scheme `ldap://` or `ldaps://` (for LDAP over SSL), a hostname, and a domain name or IP address. For example, `ldap://ad.company.com` configures the directory service with the hostname "ad" in the domain "company.com" while specifying the unencrypted LDAP protocol.

If specifying a domain name, it should be resolvable by the configured DNS servers. See `puredns(1)` [257] for more information.

If specifying an IP address, for IPv4, specify the IP address in the form `ddd.ddd.ddd.ddd`, where `ddd` is a number ranging from 0 to 255 representing a group of 8 bits.

For IPv6, specify the IP address in the form

`[xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx]`, where `xxxx` is a hexadecimal number representing a group of 16 bits. Enclose the entire address in square brackets (`[]`). Consecutive fields of zeros can be shortened by replacing the zeros with a double colon (`::`).

If the scheme of the URIs is `ldaps://`, SSL is enabled. SSL is either enabled or disabled globally, so the scheme of all supplied URIs must be the same. They must also all have the same domain.

If base DN is not configured and a URI is provided, the base DN will automatically default to the domain components of the URIs.

Optionally specify a port. If a port number is specified, append it to the end of the address.

Default ports are 389 for Ldap, and 636 for Ldaps. Non-standard ports can be specified in the URI if they are in use.

To clear the URI setting, set to an empty string ("").

Options that control display format:

--cli

Displays output in the form of CLI commands that can be issued to reproduce the current configuration. The --cli output is not meaningful when combined with immutable attributes.

--csv

Lists information in comma-separated value (CSV) format. The --csv output can be used for scripting purposes and imported into spreadsheet programs.

--notitle

Lists information without column titles.

--nvp

Lists information in name-value pair (NVP) format, in the form **ITEMNAME=VALUE**. Argument names and information items are displayed flush left. The --nvp output is designed both for convenient viewing of what might otherwise be wide listings, and for parsing individual items for scripting purposes.

--page

Turns on interactive paging.

--raw

Displays the unformatted version of column titles and data. For example, in the **purearray monitor** output, the unformatted version of column title **us/op (read)** is **usec\_per\_read\_op**. The --raw output is used to sort and filter list results.

## Arguments

### **ROLE**

Role name. Valid role names are **readonly**, **ops\_admin**, **array\_admin**, and **storage\_admin**.

## Description

The **pureds** command manages the integration of FlashArray arrays with an existing directory service.

The Purity//FA release comes with a single local administrative account named "pureuser" with array-wide (**array\_admin**) permissions. The account is password protected and may alternatively be accessed using a public-private key pair.

Additional users can be added to the array by creating and configuring local users directly on the array. For more information about local users, refer to **pureadmin(1)** [211].

Users can also be added to the array through Lightweight Directory Access Protocol (LDAP) by integrating the array with an existing directory service. If a user is not found locally, the directory servers are queried. OpenLDAP and Microsoft's Active Directory (AD) are two implementations of LDAP that Purity//FA supports.

With LDAP integration, the array leverages the directory for authentication (validate user's password) and authorization (determine user's role in the array).

Configuring the Pure Storage directory service requires a URI, a base DN, a bind user, a bind password, and at least one group within the LDAP directory.

## Users

An LDAP user is an individual in the LDAP directory. To be able to access the array, an LDAP user must belong to a configured group in the LDAP directory.

LDAP users log in to the array via `ssh` by entering the following information, where `<user_name>` represents the sAMAccountName for Active Directory or uid for OpenLDAP, and `<array_name>` represents the name or the IPv4 or IPv6 address of the FlashArray array:

```
<user_name>@<array_name>
```

For IPv4, specify the IP address in the form `ddd.ddd.ddd.ddd`, where `ddd` is a number ranging from 0 to 255 representing a group of 8 bits.

For IPv6, specify the IP address in the form `[xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx]`, where `xxxx` is a hexadecimal number representing a group of 16 bits. Enclose the entire address in square brackets (`[]`). Consecutive fields of zeros can be shortened by replacing the zeros with a double colon (`::`).

For directory service enabled accounts, user passwords to the array are managed through the directory service, while public keys are configured through Purity//FA.

Accounts with user names that conflict with local accounts will not be authenticated against the directory. These account names include, but are not limited to: `pureuser`, `os76`, `root`, `daemon`, `sys`, `man`, `mail`, `news`, `proxy`, `backup`, `nobody`, `syslog`, `mysql`, `ntp`, `avahi`, `postfix`, `sshd`, `snmp`.

If an LDAP user has the same name as a locally created user, the locally created user always has priority.

Users with disabled accounts will not have access to the array.

## Groups

A group in the LDAP directory consists of users who share a common purpose.

Each configured group in the directory has a unique distinguished name (DN) representing the entire path of the object's location in the directory tree. The DN is comprised of the following attribute-value pairs:

- DC - Domain component base of the DN. For example, `DC=mycompany,DC=com`.
- OU - Organizational unit base of the group. For example, `OU=PureGroups,OU=SAN,OU=IT`.
- CN - Common name of the groups themselves. For example, `CN=purereadonly`.

For example, `CN=purereadonly,OU=PureGroups,OU=SAN,OU=IT,DC=mycompany,DC=com` is the DN for configured group `purereadonly` at group base `OU=PureGroups,OU=SAN,OU=IT` and with base DN `DC=mycompany,DC=com`.

The DN can contain multiple DC and OU attributes.

OUs are nested, getting more specific in purpose with each nested OU.

For OpenLDAP, for group configurations based on the `shadowAccount` class, groups must have the full DN of members in the `member` attribute. For group configurations based on the `posixAccount` class, groups must have the `uid` of members in the `memberUid` attribute.

When a user who is a member of a configured group logs in to the array, only the CLI actions that the user has permission to execute will be visible. Similarly, in the GUI, actions the user does not have permission to execute will be grayed out or disabled.

For Active Directory, two types of groups are supported: security groups and distribution groups. Distribution groups are used only with email applications to distribute messages to collections of users. Distribution groups are not security enabled. Security groups assign access to resources on your network. All groups configured on the array must be security groups.

## Role-Based Access Control

Role-based access control (RBAC) restricts the system access and capabilities of each user based on their assigned role in the array.

All users in the array, whether created locally or added to the array through LDAP integration, are assigned one of the following roles in the array:

- **Read Only.** Users with the Read-Only (`readonly`) role can perform operations that convey the state of the array. Read Only users cannot alter the state of the array.
- **Ops Admin.** Users with the Ops Admin (`ops_admin`) role can perform the same operations as Read Only users plus enable and disable remote assistance sessions. Ops Admin users cannot alter the state of the array.
- **Storage Admin.** Users with the Storage Admin (`storage_admin`) role can perform the same operations as Read Only users plus storage related operations, such as administering volumes, hosts, and host groups. Storage Admin users cannot perform operations that deal with global and system configurations.
- **Array Admin.** Users with the Array Admin (`array_admin`) role can perform the same operations as Storage Admin users plus array-wide changes dealing with global and system configurations. In other words, Array Admin users can perform all operations.

For LDAP users, role-based access control is achieved by configuring the groups in the LDAP directory to correspond to the different roles in the array. For example, a group named "purereadonly" in the directory might correspond to the `readonly` role in the array.

For security purposes, each user should be assigned to only one role in the array. If a user belongs to multiple configured groups that map to different roles in the array, modify the LDAP directory to ensure that the user belongs to only one group. If a user has multiple roles, one of which includes the `ops_admin` role, the user will be locked out of the system and an alert will be sent to all alert recipients. Modify the LDAP directory to ensure that the user has only one user role assigned. If a user has multiple roles, none of which include the `ops_admin` role, the user will have privileges corresponding to the least privileged group. For example, a user who has both the `readonly` and `array_admin` roles will have read-only privileges.

## Pure Storage Directory Service Configuration

Configuring the Pure Storage directory service requires a URI, a base DN, a bind user, a bind password, and at least one group within the LDAP directory.

Before you start the configuration process, note the DN of each group within the directory server. Each component of the DN will be used to configure Pure Storage directory service.

When you configure the array to integrate with a directory service, consider the following:

- If the directory service contains multiple groups, each group must have a common name (CN).
- All uniform resource identifiers (URIs) must be in the same, single domain.

Run the `pureds setattr` command to configure the URI, base DN, bind user name, and bind password. Include the `--uri` option to set the URIs of the directory servers with LDAP or LDAPS. Up to 30 URIs can be specified. List multiple URIs in comma-separated format. To clear the URI setting, set to an empty string (""). Include the `--base-dn` option to set the base distinguished name (base DN) of the LDAP group. The base DN should consist of only domain components. Specify "DC=" for each domain component. List multiple DCs in comma-separated format. To clear the base DN setting, set to an empty string (""). Include the `--bind-user` and `--bind-password` options to specify the user name and password used to bind to and query the directory, respectively. The bind password is entered through interactive prompt. The bind account must be configured to allow the array to read the directory. It is good practice for this account to not be tied to any actual person and to have different password restrictions, such as "password never expires". The bind account should also not be a privileged account, since only read access to the directory is required.

Run the `pureds role setattr` command to configure the roles in the array. Include the `--group` option to map the common name (CN) of the configured group in the directory tree to the specified role in the array. The group name should be just the common name of the group without the "CN=" specifier. For example, `purereadonly`. To clear the group setting, set to an empty string (""). Include the `--group-base` option to set the common organizational unit (OU) under which to search for the group. Specify "OU=" for each organizational unit. The order of OUs should get smaller in scope from right to left. List multiple OUs in comma-separated format. To clear the group base setting, set to an empty string ("").

For example, run the following command to set two URIs, set bind user credentials, and configure the "readonly" role to correspond to DN

```
DN=purereadonly,OU=PureGroups,OU=SAN,OU=IT,OU=US,DC=mycompany,DC=com:
```

```
pureds setattr --uri ldaps://dc01.domain.com:636,ldaps://dc02.domain.com:636
               --base-dn DC=mycompany,DC=com --bind-user ldapreader --bind-password
pureds role setattr --group purereadonly
                   --group-base OU=PureGroups,OU=SAN,OU=IT,OU=US readonly
```

After you have configured the directory service and created the roles, test the configuration to:

- Verify the Uniform Resource Identifiers (URIs) can be resolved and that the array can bind and query the tree using the bind user credentials.

- Verify the array can find all the configured groups to ensure the common names (CNs) and group base are correctly configured. For each configured group, the array binds and queries the directory service to find the configured group. If "check peer" is enabled, the initial bind and query test is repeated while enforcing server authenticity using the CA certificate.

The **pureds test** command runs a series of tests to verify that the URIs can be resolved and that the array can bind and query the tree using the bind user credentials. The test also verifies that the array can find all the configured groups to ensure the common names and group base are correctly configured. The directory service test can be run at any time. If "check peer" is enabled, the initial bind and query test is repeated while enforcing server authenticity using the CA certificate.

The **pureds enable** command enables the Pure Storage directory service, allowing users in the LDAP directory to log in to the array. Enable the directory service after you have configured the directory service, created the roles, and tested the directory service configuration. Include the **--check-peer** option to enable server authenticity enforcement using the configured CA certificate. Note that including the **--check-peer** option only enables "check peer"; it does not enable the actual directory service. The CA certificate must be configured before the **--check-peer** option can be enabled.

The **pureds disable** command disables the directory service, stopping all users in the directory server from logging in to the array. Include the **--check-peer** option to disable "check peer". Note that including the **--check-peer** option only disables "check peer"; it does not disable the actual directory service.

The **pureds list** command displays the current base configuration. Include the **--certificate** option to display currently-configured CA certificate data.

The **pureds role list** command displays the current role-to-group configurations for the directory service, including each role's group and group base. In order to log in to the array, a user must belong to a configured group in the LDAP directory, and that group must be mapped to an RBAC role in the array. A dash (-) represents an unconfigured attribute of the role.

The permission level of an individual user is cached locally to prevent frequently binding and querying the directory. The cache entries expire after a time limit at which point the directory is queried again and the cache entry is updated. Cache entries can be refreshed on demand using the **pureadmin refresh** command. Cache entries are also automatically updated when starting a new session.

## Certificates

If the configured directory servers have been issued certificates, the certificate of the issuing certificate authority can be stored on the array to validate the authenticity of the directory servers. When performing directory queries, the certificate presented by the server is validated using CA certificate.

Server authenticity is only enforced if "check peer" is enabled. "Check peer" can only be enabled if a CA certificate has been configured.

Only one certificate can be configured at a time, so the same certificate authority should be the issuer of all directory server certificates. The certificate must be PEM formatted (Base64 encoded) and include the **"-----BEGIN CERTIFICATE-----"** and **"-----END CERTIFICATE-----"** lines. The certificate cannot exceed 3000 characters in total length.

When certificate data with valid syntax is supplied, the certificate trust is checked to determine if the certificate is self-signed or signed by a trusted root certificate authority. If the trust cannot be determined, the certificate data can still be saved, but server authenticity enforcement using the certificate may fail.

As a convenience, the Pure Storage directory service can attempt to automatically fetch CA certificate data from the directory server. If certificate data is successfully retrieved from the server, it undergoes the same trust check as manually entered certificate data, however if trust cannot be determined the operation will not continue. If certificate data is successfully retrieved and the CA certificate is trusted, a final prompt to confirm the data is displayed.

## Examples

### Example 1

```
pureds setattr --uri ldaps://[2001:0db8:85a3::ae26:8a2e:0370:7334]
               --base-dn DC=mycompany,DC=com --bind-user ldapreader
```

Sets the URI to IPv6 address `2001:0db8:85a3::ae26:8a2e:0370:7334` and the scheme to `ldaps://` to enable SSL. Also sets the base DN to be the correct domain components and sets the bind user as the user name used to bind to and query the directory.

### Example 2

```
pureds setattr --uri ldaps://ad1.mycompany.com,ldaps://ad2.mycompany.com
               --base-dn DC=mycompany,DC=com
               --bind-user CN=John,OU=Users,DC=mycompany,DC=com
```

Sets the URI to both `ad1.mycompany.com` and `ad2.mycompany.com` and the scheme to `ldaps://` to enable SSL. Also sets the base DN to be the correct domain components and sets the bind user as the user name used to bind to and query the directory.

### Example 3

```
pureds setattr --bind-password
Enter bind password:
Retype bind password:
```

Shows the interactive prompt for entering a password for the bind user account. The password is not shown while typing, so a confirmation prompt is presented. If the passwords do not match, no change is made.

### Example 4

```
pureds role setattr --group purereadonly
                   --group-base OU=PureGroups,OU=SAN,OU=IT,OU=US readonly
pureds role setattr --group pureusers
                   --group-base OU=PureGroups,OU=SAN,OU=IT,OU=US storage_admin
pureds role setattr --group pureadmins
```

```
--group-base OU=PureGroups,OU=SAN,OU=IT,OU=US array_admin
```

Sets the group base to be the nested organizational units where the groups can be found in the tree. Also sets the groups to be the common names of the directory groups. Combined with Example 1, the full distinguished name of the read-only group would be `CN=purereadonly,OU=PureGroups,OU=SAN,OU=IT,OU=US,DC=mycompany,DC=com`.

### Example 5

```
pureds list
```

Displays the current base configuration of the Pure Storage directory service. Attributes include URI, base DN, bind user, "check peer" status, and enabled status.

### Example 6

```
pureds role list storage_admin
```

Displays the group and group base configuration for the `storage_admin` role in the array.

### Example 7

```
pureds test
Output
Feature Status: Enabled

Testing from ct0:
Resolving dc01.domain.com...          PASSED
Searching ldaps://dc01.domain.com:636... PASSED
Searching for group CN=purereadonly... PASSED
Searching for group CN=pureopsadmins... PASSED
Searching for group CN=pureusers...    PASSED
Searching for group CN=pureadmins...   PASSED

Resolving dc02.domain.com:636...          PASSED
Searching ldaps://dc02.domain.com...    PASSED
Searching for group CN=purereadonly... PASSED
Searching for group CN=pureopsadmins... PASSED
Searching for group CN=pureusers...    PASSED
Searching for group CN=pureadmins...   PASSED
```

Tests the current directory service configuration, showing successful test output.

## See Also

`pureadmin(1)` [211], `puredns(1)` [257]

## Author

Pure Storage Inc. <[documentfeedback@purestorage.com](mailto:documentfeedback@purestorage.com)>

## purehgroup

purehgroup, purehgroup-create, purehgroup-delete, purehgroup-rename, purehgroup-setattr — manage the creation, deletion, naming, and population of Purity//FA host groups.  
purehgroup-add, purehgroup-remove — manage adding and removing of Purity//FA host groups to and from protection groups, respectively  
purehgroup-monitor — monitor host group I/O performance.

### Synopsis

```
purehgroup create [--hostlist HOST-LIST] HGROUP...
purehgroup delete HGROUP...
purehgroup add --pgroup PGROUP HGROUP...
purehgroup remove --pgroup PGROUP HGROUP...
purehgroup monitor [--array] [--csv] [--filter FILTER] [--interval SECONDS] [--latency] [--limit LIMIT] [--mirrored] [--notitle] [--raw] [--repeat REPEAT-COUNT] [--size] [--sort SORT] [--total] [HGROUP...]
purehgroup rename OLD-NAME NEW-NAME
purehgroup setattr { --addhostlist HOST-LIST | --hostlist HOST-LIST | --remhostlist HOST-LIST } HGROUP
```

### Arguments

#### **HGROUP**

Host group name.

#### **NEW-NAME**

Name by which the host group is to be known after the command executes.

#### **OLD-NAME**

Current name of the host group to be renamed.

### Object Names

Valid characters are letters (A-Z and a-z), digits (0-9), and the hyphen (-) character. The first and last characters of the name must be alphanumeric, and the name must contain at least one letter or '-'.

Most objects in Purity//FA that can be named, including host groups, hosts, volumes, protection groups, volume and protection group suffixes, SNMP managers, and subnets, can be 1-63 characters in length.

Array names can be 1-56 characters in length. The array name length is limited to 56 characters so that the names of the individual controllers, which are assigned by Purity//FA based on the array name, do not exceed the maximum allowed by DNS.

Names are case-insensitive on input. For example, **vol1**, **Vol1**, and **VOL1** all represent the same volume. Purity//FA displays names in the case in which they were specified when created or renamed.

Pods and volume groups provide a namespace with unique naming conventions.

All objects in a pod have a fully qualified name that include the pod name and object name. The fully qualified name of a volume in a pod is **POD::VOLUME**, with double colons (::) separating the pod name and volume name. The fully qualified name of a protection group in a pod is **POD::PGROUP**, with double colons (::) separating the pod name and protection group name. For example, the fully qualified name of a volume named **vol01** in a pod named **pod01** is **pod01::vol01**, and the fully qualified name of a protection group named **pgroup01** in a pod named **pod01** is **pod01::pgroup01**.

If a protection group in a pod is configured to asynchronously replicate data to a target array, the fully qualified name of the protection group on the target array is **POD:PGROUP**, with single colons (:) separating the pod name and protection group name. For example, if protection group **pod01::pgroup01** on source array **array01** asynchronously replicates data to target array **array02**, the fully qualified name of the protection group on target array **array02** is **pod01:pgroup01**.

All objects in a volume group have a fully qualified name that includes the volume group name and the object name, separated by a forward slash (/). For example, the fully qualified name of a volume named **vol01** in a volume group named **vgroup01** is **vgroup01/vol01**.

## Options

**-h | --help**

Can be used with any command or subcommand to display a brief syntax description.

**--addhostlist**

Comma-separated list of one or more additional hosts to be associated with the host group. Has no effect on hosts already associated with the group.

**--array**

In a synchronous replication configuration, breaks down performance data by the array to which the I/O is directed.

**--hostlist**

Comma-separated list of one or more hosts to be associated with the host group. When specified in the **purehgroup setattr** command, replaces the entire membership of a host group. To disassociate all hosts from a host group, set to an empty string ("").

**--pgroup PGROUP**

Comma-separated list of protection groups to which the specified host groups are added or from which the specified host groups are removed. Has no effect on host groups already associated with the protection group.

**--interval SECONDS**

Sets the number of seconds between displays of real-time performance data. At each interval, the system displays a point-in-time snapshot of the performance data. If omitted, the interval defaults to every 5 seconds.

**--latency**

Displays real-time and historical I/O latency information.

--mirrored

In a synchronous replication configuration, includes performance data for mirrored writes (i.e., writes to volumes in stretched pods). If one array of a stretched pod is offline, the writes processed on the other array are reported as local writes rather than mirrored ones. Once the pod is back online, in sync, and processing each write on both sides, the writes to its volumes are reported as mirrored writes again.

--remhostlist

Comma-separated list of one or more hosts whose associations with the host group are to be broken.

--repeat *REPEAT-COUNT*

Sets the number of times to display real-time performance data. If omitted, the repeat count defaults to 1.

--size

Displays the average I/O sizes per read and write operation.

--total

Follows output lines with a single line containing column totals in columns where they are meaningful.

Options that control display format:

--csv

Lists information in comma-separated value (CSV) format. The --csv output can be used for scripting purposes and imported into spreadsheet programs.

--notitle

Lists information without column titles.

--raw

Displays the unformatted version of column titles and data. For example, in the `purearray monitor` output, the unformatted version of column title `us/op (read)` is `usec_per_read_op`. The --raw output is used to sort and filter list results.

Options that manage display results:

--filter

Displays only the rows that meet the filter criteria specified.

--limit

Limits the size of the list output to the specified maximum number of rows.

--sort

Sorts the list output in ascending or descending order by the column specified.

## Description

A Purity//FA *host group* is an abstraction that implements *consistent* connections between a set of hosts and one or more volumes. Connections are consistent in the sense that all hosts associated with

a host group address a volume connected to the group by the same LUN. Host groups are typically used to provide a common view of storage volumes to the hosts in a clustered application.

## Creating Host Groups

The **purehgroup create** command creates a host group and assigns it a name. Optionally, include the **--hostlist** option with the **purehgroup create** command to add hosts during host group creation.

## Adding Hosts to Host Groups

Adding a host to a host group automatically connects the host to all volumes associated with the group. A host can only belong to one host group.

Hosts can be added to or removed from a host group at any time.

To add hosts to an existing host group, include the **--addhostlist** or **--hostlist** option with the **purehgroup setattr** command. The **--addhostlist** option adds hosts to an existing host group. Hosts that are already associated with the host group are not affected. The **--hostlist** option adds hosts to an existing host group, completely replacing any existing hosts that are already associated with the host group.

To remove hosts from a host group, run the **purehgroup setattr --remhostlist** command. Alternatively, remove hosts from a host group by running the **purehgroup setattr --hostlist ""** command to replace the existing list of hosts with a null set.

## Creating Shared Host-Volume Connections

To connect the hosts within a host group to a volume, establish a shared connection by running the **purehgroup connect** or **purevol connect --hgroup** command. Shared connections can be created or broken at any time. Once a connection is established, the volume is assigned a logical unit number (LUN), which all hosts within the host group use to communicate with the volume.

For more information about shared connections and LUNs, refer to **purehgroup-connect(1)** [283].

## Deleting Host Groups

The **purehgroup delete** command removes host groups that are no longer required. A host group can only be deleted if it is empty, so remove all hosts from a host group before deleting the group.

## Renaming Host Groups

Run the **purehgroup rename** command to change the current name (**OLD-NAME**) of a host group to the new name (**NEW-NAME**). The name change is effective immediately and the old name is no longer recognized in CLI, GUI, or REST interactions. In the Purity//FA GUI, the new name appears upon page refresh.

## Adding Host Groups to Protection Groups

The **purehgroup add** command adds existing host groups to existing protection groups. Host groups can only be added to protection groups that replicate to target arrays through asynchronous replication. Hosts and host groups are not supported for replication to offload targets.

Multiple host groups can be added to multiple protection groups. Enter multiple protection groups in comma-separated format.

If a protection group already includes other host groups, those host groups are unaffected.

Only members of the same object type can belong to a protection group. For example, a host group cannot be added to a protection group that already contains hosts or volumes.

The `purehgroup add` command only accepts host groups that do not already belong to any of the specified protection groups. If any of the host groups already belong to any of the protection groups, the entire command fails.

Run the `purehgroup list --protect` command to see a list of all protected host groups and their associated protection groups.

The `purehgroup remove` command removes one or more host groups from one or more protection groups. All of the specified host groups must belong to all of the specified protection groups in the command; otherwise, the command fails.

Host groups can also be added to and removed from protection groups through the `purepgroup setattr` command.

## Host Group I/O Performance Monitoring

The `purehgroup monitor` command displays real-time I/O performance information for all or specified host groups. The `purehgroup monitor` output includes the following data about bandwidth, IOPS, and latency:

- **Name:** Object name.
- **Time:** Current time.
- **B/s:** Bandwidth. Number of bytes read/written.
- **op/s:** IOPS. Number of read, write, or mirrored write requests processed per second.
- **us/op:** Service time. Average time, measured in microseconds, it takes the array to serve a read, write, or mirrored write I/O request. Service time does not include SAN time, queue time, or QoS rate limit time.
- **B/op:** IOPS. Average I/O size per read, write, mirrored write, and both read and write (all) operations. Must include the `--size` option to see the B/op columns.

Include the `--repeat` option to specify the number of times to repeat the real-time update. If not specified, the repeat value defaults to 1. Include the `--interval` option to specify the number of seconds between each real-time update. At each interval, the system displays a point-in-time snapshot of the performance data. If not specified, the interval value defaults to every 5 seconds. The `--interval` and `--repeat` options can be combined.

In a synchronous replication configuration, include the `--array` option to break down performance data by the array to which the I/O is directed. Include the `--mirrored` option to display performance data for mirrored writes (i.e., writes to volumes in stretched pods). If one array of a stretched pod is offline, the writes processed on the other array are reported as local writes rather than mirrored ones.

Once the pod is back online, in sync, and processing each write on both sides, the writes to its volumes are reported as mirrored writes again.

## Host Group Latency Monitoring

The **purehgroup monitor --latency** command displays real-time and historical I/O latency information for volumes connected to host groups. The **purehgroup monitor --latency** output includes the following data:

- **SAN us/op**: SAN time. Average time, measured in microseconds, required to transfer data between the initiator and the array. Slow data transfers will result in higher SAN times.
- **Queue us/op**: Queue time. Average time, measured in microseconds, that an I/O request spends in the array waiting to be served. The time is averaged across all I/Os of the selected types.
- **QoS Rate Limit us/op**: QoS rate limit time. Average time, measured in microseconds, that reads, writes, or mirrored writes spend in queue as a result of bandwidth limits reached on one or more volumes.
- **us/op**: Service time. Average time, measured in microseconds, it takes the array to serve a read, write, or mirrored write I/O request. Service time does not include SAN time, queue time, or QoS rate limit time.

Include the **--mirrored** option to display latency data for mirrored writes (i.e., writes to volumes in stretched pods).

## Exceptions

Purity//FA will not create a host group if:

- A host group with the specified name already exists in the array.
- The creation of the host group would exceed the limit of concurrent host groups.

Purity//FA will not delete a host group if:

- Any hosts are associated with the host group or any volumes are connected to it.

A host cannot be added to a host group if:

- The host is associated with another host group. A host can only be associated with one host group at a time.
- The host has a private connection to a volume associated with the host group.

## Examples

### Example 1

```
purehost create --wwnlist 0123456789abcde6,0123456789abcde7 HOST6
purehost create --wwnlist 0123456789abcde8,0123456789abcde9 HOST7
purevol create --size 100g VOL1 VOL2 VOL3 VOL4
purehgroup create --hostlist HOST6,HOST7 HGROUP3
```

```
purehgroup connect --vol VOL1 HGROUP3  
purehgroup connect --vol VOL2 HGROUP3  
purehgroup connect --vol VOL3 HGROUP3  
purehgroup connect --vol VOL4 --lun 25 HGROUP3
```

Typical usage of the **purehgroup create** command. Creates hosts **HOST6** and **HOST7**. Creates 100 gigabyte volumes **VOL1**, **VOL2**, **VOL3**, and **VOL4**. Creates host group **HGROUP3** and associates **HOST6** and **HOST7** with it. Connects **VOL1**, **VOL2**, **VOL3**, and **VOL4** to **HGROUP3**. Purity//FA automatically assigns LUNs to volumes **VOL1**, **VOL2**, and **VOL3** by counting down from 254 and assigning the first available numbers to each of the respective volumes. If available, LUN 25 is assigned to **VOL4**. Both hosts communicate with the volumes via the same LUNs.

#### Example 2

```
purehgroup create --hostlist HOST1,HOST2,HOST3 HGROUP1
```

Creates host group **HGROUP1** and associates hosts **HOST1**, **HOST2**, and **HOST3** with it. No volumes are connected to the group at the time of creation.

#### Example 3

```
purehgroup create HGROUP2
```

*... time passes...*

```
purehgroup connect --vol VOL1 --lun 11 HGROUP2
```

*... more time passes...*

```
purehgroup setattr --addhostlist HOST4,HOST5 HGROUP2
```

Creates host group **HGROUP2**, but does not associate any hosts with it. At a later time, **VOL1** is connected to the group, with LUN 11 assigned to it. Still later, **HOST4** and **HOST5** are associated with **HGROUP2**, causing Purity//FA to establish shared connections between them and **VOL1**, using LUN 11 for communication.

#### Example 4

```
purehgroup monitor --repeat 60 HGROUP1
```

Displays real-time performance data for host group **HGROUP1**. Sixty (60) updates are displayed, with each point-in-time update taken at the default interval of every five (5) seconds.

#### Example 5

```
purehgroup setattr --host "" HGROUP2  
purehgroup delete HGROUP2
```

Removes all hosts from **HGROUP2** and deletes the host group.

## See Also

[purehgroup-connect\(1\)](#) [283], [purehgroup-list\(1\)](#) [287]

[purehost\(1\)](#) [292], [purepod\(1\)](#) [412], [purevol\(1\)](#) [469]

## Author

Pure Storage Inc. <[documentfeedback@purestorage.com](mailto:documentfeedback@purestorage.com)>

## purehgroup-connect

purehgroup-connect, purehgroup-disconnect, purevol-connect, purevol-disconnect — manage shared connections between host groups and volumes

### Synopsis

```
purehgroup connect --vol VOL [--lun LUN] [--use-protocol-endpoint PROTOCOL-ENDPOINT] HGROUP...
```

```
purehgroup disconnect --vol VOL HGROUP...
```

```
purevol connect { --host HOST | --hgroup HGROUP } [--lun LUN] [--use-protocol-endpoint PROTOCOL-ENDPOINT] VOL...
```

```
purevol disconnect { --host HOST | --hgroup HGROUP } VOL...
```

### Arguments

#### **HGROUP**

Host group with which a shared connection to the specified volume is to be created or broken.

#### **VOL**

Name of a volume to be connected to or disconnected from a host group.

### Options

#### **-h** | **--help**

Can be used with any command or subcommand to display a brief syntax description.

#### **--hgroup**

Host group with which shared connections to the specified volumes are to be made or broken by **purevol connect** and **purevol disconnect** commands. Exactly one host group must be specified by this option. The option is mutually exclusive with the **--host** option.

#### **--host**

Host with which private connections to the specified volumes are to be made or broken by **purevol connect** and **purevol disconnect** commands. Exactly one host must be specified by this option. The option is mutually exclusive with the **--hgroup** option.

#### **--lun**

Logical unit number (LUN) by which hosts associated with the host group are to address the volume. If not specified, Purity//FA automatically assigns a LUN to the connection. To automatically assign a LUN to a shared connection, Purity//FA starts at 254 and counts down to the minimum LUN 1, assigning the first available LUN to the connection. If all LUNs in the [1...254] range are taken, Purity//FA starts at LUN 255 and counts up to the maximum LUN 4095, assigning the first available LUN to the connection.

#### **--use-protocol-endpoint**

Connects the FlashArray virtual volume to VMware ESXi host groups.

For more information about virtual volumes, including configuration steps, refer to the Pure Storage vSphere Web Client Plugin for vSphere User Guide in the Pure1 Knowledge site at <https://support.purestorage.com>.

--vol

Name of the volume to be connected to or disconnected from the specified hosts host groups. Exactly one volume must be specified.

## Description

Makes and breaks shared connections between hosts associated with host groups and volumes.

### Private vs. Shared Host-Volume Connections

Purity//FA supports two types of host-volume connections:

Private

Connects one volume to one host. Private connections are independent of one another. For example, the sequence:

```
purevol connect --host HOST1 VOL1 VOL2  
purehost disconnect --vol VOL1 HOST1
```

connects **HOST1** to **VOL1** and **HOST1** to **VOL2**, and then disconnects **HOST1** and **VOL1**, leaving **HOST1** connected to **VOL2**.

Private connections are typically used for boot volumes and for stand-alone (non-clustered) host applications.

To establish a private connection, run the **purevol connect --host** or **purehost connect --vol** command. Both commands are functionally identical.

To break a private connection that is no longer required, run the **purevol disconnect --host** or **purehost disconnect --vol** command. Both commands are functionally identical. When a connection has been broken, its LUN is available for reuse.

For more information about private connections and the **purehost connect** command, refer to **purehost-connect(1)** [302].

Shared

Connects a designated set of hosts (via a host group) to a designated set of volumes, providing the hosts with a consistent "view" of the volumes. All associated hosts use the same LUN to address a given associated volume. All hosts and volumes associated with a host group are automatically connected to each other by virtue of their associations with the group.

To establish a shared connection (assuming the host belongs to a host group), run the **purevol connect --hgroup** or **purehgroup connect --vol** command. Both commands are functionally identical. For example, the command:

```
purevol connect --hgroup HGROUP1 VOL1 VOL2
```

is equivalent to the sequence:

```
purehgroup connect --vol VOL1 HGROUP1
purehgroup connect --vol VOL2 HGROUP1
```

Both commands establish shared connections between the hosts associated with **HGROUP1** and **VOL1** and **VOL2**.

Shared connections are useful for cluster applications in which several related hosts require consistent (same LUN) connectivity to a set of storage volumes.

To break a shared connection that is no longer required, run the

**purevol disconnect --hgroup** or **purehgroup disconnect --vol** command. Both commands are functionally identical. When a connection has been broken, its LUN is available for reuse.

A host can have only one connection to a given volume at any given time. If you attempt to make a second connection between a host and a volume, private or shared, the attempt will fail.

## LUN Management

When a volume is connected to a host group, it is assigned a logical unit number (LUN) ID, which all hosts within the host group use to communicate with the volume. A volume can be connected to multiple host groups as well as to individual hosts simultaneously.

To manually assign a LUN ID to a shared connection, include the **--lun** option with the **purehgroup connect** or **purevol connect --hgroup** command.

Purity//FA automatically assigns a LUN to the shared connection. To do this, Purity//FA starts at LUN 254 and counts down to the minimum LUN 1, assigning the first available LUN to the connection. If all LUNs in the [1...254] range are taken, Purity//FA starts at LUN 255 and counts up to the maximum LUN (4095), assigning the first available LUN to the connection.

To change the LUN associated with a shared connection, the connection must first be broken and then recreated by **purehgroup connect**.

## Exceptions

Purity//FA will not establish a (shared) connection between a volume and host group if:

- An unavailable LUN was specified.
- The volume is already connected to the host group.
- The volume is already connected to a host associated with the host group.

## Examples

### Example 1

```
purehgroup connect --vol VOL1 HGROUP1 HGROUP2
```

Establishes shared connections between **VOL1** and the hosts associated with **HGROUP1** and those associated with **HGROUP2**. Purity//FA assigns a LUN to the connections with **HGROUP1**'s hosts and another to those with **HGROUP2**'s hosts; the two LUNs may be the same or different.

## Example 2

```
purehgroup connect --vol VOL2 --lun 15 HGROUP3 HGROUP4
```

Establishes shared connections between **VOL2** and the hosts associated with **HGROUP3** and those associated with **HGROUP4**. If LUN 15 is already being used by shared connections to either of the host groups, no connections are made between that group's hosts and **VOL2**.

## Example 3

```
purehgroup disconnect --vol VOL6 $(purevol listobj --type hgroup VOL6)
```

Breaks all shared connections between **VOL6** and host groups. (The inner **purevol listobj** command produces a list of all host groups with shared connections to **VOL6**.)

Private connections to **VOL6** are unaffected.

## See Also

**purehgroup(1)** [275], **purehgroup-list(1)** [287]

**purehost(1)** [292], **purevol(1)** [469]

## Author

Pure Storage Inc. <[documentfeedback@purestorage.com](mailto:documentfeedback@purestorage.com)>

## purehgroup-list

purehgroup-list, purehgroup-listobj — display host groups' attributes and storage consumption

### Synopsis

```
purehgroup list [ --cli | --csv | --nvp ] [ --connect | --protect | --remote | --space ] [--filter FILTER] [--limit LIMIT] [--notitle] [--page] [--raw] [--sort SORT] [HGROUP...]
```

```
purehgroup listobj [--csv] [--remote] [ --type { hgroup | host | vol } ] [HGROUP...]
```

### Arguments

#### **HGROUP**

Host group for which the information specified by options is to be displayed.

### Options

Options that control information displayed:

**-h** | **--help**

Can be used with any command or subcommand to display a brief syntax description.

**default** (no content option specified in **purehgroup list** command)

Displays names and associated hosts for the specified host groups.

**--connect** (**purehgroup list** only)

Displays volumes associated with the specified host groups, and the LUNs used to address them.

**--protect** (**purehgroup list** only)

Displays all protected host groups and their associated protection groups.

**--remote**

Includes a list of remote host groups.

**--space** (**purehgroup list** only)

Displays size and space consumption information for the volumes connected to each host group.

**--type** (**purehgroup listobj** only)

Specifies the type of information about the specified host groups that is to be produced in whitespace-separated format suitable for scripting.

Options that control display format:

**--cli**

Displays output in the form of CLI commands that can be issued to reproduce the current configuration. The **--cli** output is not meaningful when combined with immutable attributes.

--csv

Lists information in comma-separated value (CSV) format. The --csv output can be used for scripting purposes and imported into spreadsheet programs.

--notitle

Lists information without column titles.

--nvp

Lists information in name-value pair (NVP) format, in the form **ITEMNAME=VALUE**. Argument names and information items are displayed flush left. The --nvp output is designed both for convenient viewing of what might otherwise be wide listings, and for parsing individual items for scripting purposes.

--page

Turns on interactive paging.

--raw

Displays the unformatted version of column titles and data. For example, in the **purearray monitor** output, the unformatted version of column title **us/op (read)** is **usec\_per\_read\_op**. The --raw output is used to sort and filter list results.

Options that manage display results:

--filter

Displays only the rows that meet the filter criteria specified.

--limit

Limits the size of the list output to the specified maximum number of rows.

--sort

Sorts the list output in ascending or descending order by the column specified.

## Description

The **puregroup list** command displays the information indicated by content options for the specified host groups. If no host groups are specified, the display includes the specified information for all host groups.

- If no options are specified, the display lists the hosts associated with each specified host group.
- If the --connect option is specified, the display lists volumes associated with the specified host groups, and the LUNs used to address them.
- If the --protect option is specified, the display lists all protected host groups and their associated protection groups.
- If the --remote option is specified, the display includes a list of remote host groups, meaning a list of host groups that have been created on remote arrays. Host groups on remote arrays have the naming convention **ARRAY:HGROUP**, where **ARRAY** represents the name of the remote array, and **HGROUP** represents the name of the host group on the remote array.

- If the **--space** option is specified, the display lists the following size and space consumption information for the volumes connected to each host group:

**Size**

Total provisioned size of all volumes connected to the host group. Represents storage capacity reported to hosts.

**Thin Provisioning**

Percentage of volume sectors that do not contain host-written data because the hosts have not written data to them or the sectors have been explicitly trimmed.

**Data Reduction**

Ratio of mapped sectors within a volume versus the amount of physical space the data occupies after data compression and deduplication. The data reduction ratio does not include thin provisioning savings.

For example, a data reduction ratio of 5:1 means that for every 5 MB the host writes to the array, 1 MB is stored on the array's flash modules.

**Total Reduction**

Ratio of provisioned sectors within a volume versus the amount of physical space the data occupies after reduction via data compression and deduplication *and* with thin provisioning savings. Total reduction is data reduction with thin provisioning savings.

For example, a total reduction ratio of 10:1 means that for every 10 MB of provisioned space, 1 MB is stored on the array's flash modules.

**Volumes**

Physical space occupied by volume data that is not shared between volumes, excluding array metadata and snapshots.

**Snapshots**

Physical space occupied by data unique to one or more snapshots.

**Total**

Total physical space occupied by system, shared space, volume, and snapshot data.

The **purehgroup listobj** command creates lists of certain attributes of specified host groups in either whitespace or comma-separated form, suitable for scripting.

If the **--remote** option is specified, the display includes a list of remote host groups, meaning a list of host groups that have been created on remote arrays. Host groups on remote arrays have the naming convention **ARRAY:HGROUP**, where **ARRAY** represents the name of the remote array, and **HGROUP** represents the name of the host group on the remote array.

Include the **--type** option to display one of the following types of lists:

**--type hgroup** (default if **--type** option not specified)

Produces a list of the specified host group names. If no host group names are specified, the list contains the names of all host groups.

--type host

Produces a list of hosts associated with the specified host groups. If no host groups are specified, the list contains names of all hosts associated with host groups.

--type vol

Produces a list of volumes associated with the specified host groups. If no host group argument is specified, the list contains all volumes that are associated with any host group.

Lists are whitespace-separated by default. Specify the **--csv** option to produce a comma-separated list.

## Exceptions

None.

## Examples

### Example 1

```
purehgroup list --connect
```

For all host groups, displays names of associated volumes, and the logical units used by associated hosts to address them.

### Example 2

```
purehgroup list
```

Displays the names of all host groups and their member hosts.

### Example 3

```
purehgroup list --space --csv HGROUP1 HGROUP2 HGROUP3
```

Displays the virtual and physical space consumption for volumes associated with each of **HGROUP1**, **HGROUP2**, and **HGROUP3**. The output is displayed in comma-separated value format.

### Example 4

```
purehgroup list --remote
```

Displays a list of all host groups on both the local and remote arrays, and their member hosts.

### Example 5

```
purehgroup list --connect $(purevol listobj --type hgroup VOL1)
```

The inner **purehgroup listobj** command produces a list of all host groups with which **VOL1** is associated. This list is input to the outer **purehgroup list** command to display a list of all volumes associated with host groups with which **VOL1** has an association.

## Example 6

```
purevol list --space --total $(purehgroup listobj --type vol HGROUP1)
```

The inner **purehgroup listobj** command produces a list of the volumes associated with **HGROUP1**. This list is input to the outer **purevol list** command to display the space occupied by each of these volumes, as well as the total space occupied by all volumes associated with **HGROUP1**.

## See Also

[purehgroup\(1\)](#) [275], [purehgroup-connect\(1\)](#) [283]

[purehost\(1\)](#) [292], [purevol\(1\)](#) [469]

## Author

Pure Storage Inc. <[documentfeedback@purestorage.com](mailto:documentfeedback@purestorage.com)>

## purehost

purehost, purehost-create, purehost-delete, purehost-rename, purehost-setattr — manage creation, deletion, naming, and attributes of the Purity//FA hosts used to identify computers ("hosts") that use FlashArray storage services  
purehost-add, purehost-remove — manage adding and removing of Purity//FA hosts to and from protection groups, respectively  
purehost-monitor — monitor host I/O performance

### Synopsis

```
purehost create [--iqnlist IQN-LIST] [--nqnlist NQN-LIST] [--wwnlist WWN-LIST] HOST...  
purehost delete HOST...  
purehost add --pgroup PGROUP HOST...  
purehost remove --pgroup PGROUP HOST...  
purehost monitor [--array] [--balance] [--csv] [--filter FILTER] [--interval SECONDS] [--latency] [--limit LIMIT] [--mirrored] [--notitle] [--raw] [--repeat REPEAT-COUNT] [--size] [--sort SORT] [--total] [HOST]...  
purehost rename OLD-NAME NEW-NAME  
purehost setattr { --addiqnlist IQN-LIST | --addnqnlist NQN-LIST | --addwwnlist WWN-LIST | --host-password | --host-user HOST-USER | --iqnlist IQN-LIST | --nqnlist NQN-LIST | --personality PERSONALITY | --remiqnlist IQN-LIST | --remnqnlist NQN-LIST | --remwwnlist WWN-LIST | --target-password | --target-user TARGET-USER | --wwnlist WWN-LIST } HOST
```

### Arguments

#### *HOST*

Host name.

#### *NEW-NAME*

Name by which the host is to be known after the command executes.

#### *OLD-NAME*

Current name of the host to be renamed.

### Object Names

Valid characters are letters (A-Z and a-z), digits (0-9), and the hyphen (-) character. The first and last characters of the name must be alphanumeric, and the name must contain at least one letter or '-'.

Most objects in Purity//FA that can be named, including host groups, hosts, volumes, protection groups, volume and protection group suffixes, SNMP managers, and subnets, can be 1-63 characters in length.

Array names can be 1-56 characters in length. The array name length is limited to 56 characters so that the names of the individual controllers, which are assigned by Purity//FA based on the array name, do not exceed the maximum allowed by DNS.

Names are case-insensitive on input. For example, **vol1**, **Vol1**, and **VOL1** all represent the same volume. Purity//FA displays names in the case in which they were specified when created or renamed.

Pods and volume groups provide a namespace with unique naming conventions.

All objects in a pod have a fully qualified name that include the pod name and object name. The fully qualified name of a volume in a pod is **POD::VOLUME**, with double colons (::) separating the pod name and volume name. The fully qualified name of a protection group in a pod is **POD::PGROUP**, with double colons (::) separating the pod name and protection group name. For example, the fully qualified name of a volume named **vol01** in a pod named **pod01** is **pod01::vol01**, and the fully qualified name of a protection group named **pgroup01** in a pod named **pod01** is **pod01::pgroup01**.

If a protection group in a pod is configured to asynchronously replicate data to a target array, the fully qualified name of the protection group on the target array is **POD:PGROUP**, with single colons (:) separating the pod name and protection group name. For example, if protection group **pod01::pgroup01** on source array **array01** asynchronously replicates data to target array **array02**, the fully qualified name of the protection group on target array **array02** is **pod01:pgroup01**.

All objects in a volume group have a fully qualified name that includes the volume group name and the object name, separated by a forward slash (/). For example, the fully qualified name of a volume named **vol01** in a volume group named **vgroup01** is **vgroup01/vol01**.

## Options

**-h | --help**

Can be used with any command or subcommand to display a brief syntax description.

**--addiqnlist**

Adds the iSCSI Qualified Names (IQNs) in the comma-separated list to those already associated with the specified host.

**--addnqnlist**

Adds the NVMe Qualified Names (NQNs) in the comma-separated list to those already associated with the specified host.

**--addwwnlist**

Adds the Fibre Channel World Wide Names (WWNs) in the comma-separated list to those already associated with the specified host.

**--array**

In a synchronous replication configuration, breaks down performance data by the array to which the I/O is directed.

**--balance**

Displays I/O balance details.

--host-password

Sets the host password for CHAP authentication. The CLI prompts for the password interactively.

--host-user

Sets the host username for CHAP authentication.

--interval *SECONDS*

Sets the number of seconds between displays of real-time performance data. At each interval, the system displays a point-in-time snapshot of the performance data.

If used with the --balance option, sets the interval at which I/O balance details are logged.

Each I/O balance update consists of the I/O statistics collected during the entire interval.

If omitted, the interval defaults to every 5 seconds.

--iqnlist

Comma-separated list of one or more iSCSI Qualified Names (IQNs) to be associated with the specified host. In the **purehost setattr** command, this option replaces all IQNs previously associated with the specified host with those in the list.

--latency

Displays real-time and historical I/O latency information.

--nqnlist

Comma-separated list of one or more NVMe Qualified Names (NQNs) to be associated with the specified host. In the **purehost setattr** command, this option replaces all NQNs previously associated with the specified host with those in the list.

--mirrored

In a synchronous replication configuration, includes performance data for mirrored writes (i.e., writes to volumes in stretched pods). If one array of a stretched pod is offline, the writes processed on the other array are reported as local writes rather than mirrored ones. Once the pod is back online, in sync, and processing each write on both sides, the writes to its volumes are reported as mirrored writes again.

--personality

Determines how the Purity//FA system tunes the protocol used between the array and the initiator. To ensure the array works optimally with the host, set the personality to the name of the host operating or virtual memory system. Valid values are **aix**, **esxi**, **hitachi-vsp**, **hpx**, **oracle-vm-server**, **solaris**, and **vms**. If your system is not listed as one of the valid host personalities, do not set the option. By default, the host personality is not set. To clear the host personality setting, set to an empty string ("").

--pgroup *PGROUP*

Comma-separated list of protection groups to which the specified hosts are added or from which the specified hosts are removed. Has no effect on hosts already associated with the protection group.

--remiqnlist

Disassociates the iSCSI Qualified Names (IQNs) in the comma-separated list from the specified host.

--remnqnlist  
 Disassociates the NVMe Qualified Names (NQNs) in the comma-separated list from the specified host.

--remwwnlist  
 Disassociates the Fibre Channel World Wide Names (WWNs) in the comma-separated list from the specified host.

--repeat *REPEAT-COUNT*  
 Sets the number of times to display real-time performance data. If omitted, the repeat count defaults to 1.

--size  
 Displays the average I/O sizes per read and write operation.

--target-password  
 Sets the target password for CHAP authentication. The CLI prompts for the password interactively.

--target-user  
 Sets the target username for CHAP authentication.

--total  
 Follows output lines with a single line containing column totals in columns where they are meaningful.

--wwnlist  
 Comma-separated list of one or more Fibre Channel World Wide Names (WWNs) to be associated with the specified host. In the **purehost setattr** command, this option replaces all WWNs previously associated with the specified host with those in the list.

Options that control display format:

--csv  
 Lists information in comma-separated value (CSV) format. The --csv output can be used for scripting purposes and imported into spreadsheet programs.

--notitle  
 Lists information without column titles.

--raw  
 Displays the unformatted version of column titles and data. For example, in the **purearray monitor** output, the unformatted version of column title **us/op (read)** is **usec\_per\_read\_op**. The --raw output is used to sort and filter list results.

Options that manage display results:

--filter  
 Displays only the rows that meet the filter criteria specified.

--limit  
 Limits the size of the list output to the specified maximum number of rows.

--sort

Sorts the list output in ascending or descending order by the column specified.

## Description

The host is the abstraction used by Purity//FA to organize the storage network addresses (iSCSI Qualified Names, NVMe qualified names, or Fibre Channel world wide names) of client computers and to control communications between clients and volumes. The host's only attributes are lists of one or more iSCSI Qualified Names (IQNs), NVMe Qualified Names (NQNs), or Fibre Channel World Wide Names (WWNs) that identify the host's initiators.

### Creating Hosts

The **purehost create** command creates a host and optionally associates one or more IQNs, NQNs, or WWNs with it.

To add IQNs, NQNs, or WWNs to a host during host creation, include the respective **--iqnlist**, **--nqnlist**, or **--wwnlist** option with the **purehost create** command.

### Associating IQNs, NQNs, or WWNs with Hosts

The host cannot communicate with the array until at least one IQN, NQN, or WWN has been associated with it.

IQNs, NQNs, and WWNs can be added to or removed from a host at any time.

To add IQNs, NQNs, or WWNs to an existing host, include the respective **--addiqnlist**, **--addnqnlist**, or **--addwwnlist** option with the **purehost setattr** command. IQNs, NQNs, and WWNs that are already associated with the host are not affected.

To add IQNs, NQNs, or WWNs to an existing host, completely replacing the ones that are currently associated with the host, include the respective **--iqnlist**, **--nqnlist**, or **--wwnlist** option with the **purehost setattr** command.

To remove IQNs, NQNs, or WWNs from an existing host, include the respective **--remiqnlist**, **--remnqnlist**, or **--remwwnlist** option with the **purehost setattr** command.

Once the IQNs, NQNs, or WWNs have been specified, enable communication between the host and volumes by establishing connections, either private or shared, between the two.

### Creating Host-Volume Connections

After a host has been created and the IQNs, NQNs, or WWNs have been specified, establish a connection, either private or shared, between the host and volumes. Host-volume connections can be established or broken at any time.

To establish a private connection, run the **purehost connect** or **purevol connect --host** command. To establish a shared connection, run the **purehgroup connect** or **purevol connect --hgroup** command.

For more information about private connections, refer to **purehost-connect(1)** [302]. For more information about shared connections, refer to **purehgroup-connect(1)** [283].

## Deleting Hosts

The **purehost delete** command removes hosts that are no longer required. A host cannot be deleted while it has connections to volumes, either private or shared.

## Renaming Hosts

Run the **purehost rename** command to change the current name (**OLD-NAME**) of a host to the new name (**NEW-NAME**). The name change is effective immediately and the old name is no longer recognized in CLI, GUI, or REST interactions. In the Purity//FA GUI, the new name appears upon page refresh.

## Adding Hosts to Protection Groups

The **purehost add** command adds existing hosts to existing protection groups. Hosts can only be added to protection groups that replicate to target arrays through asynchronous replication. Hosts and host groups are not supported for replication to offload targets.

Multiple hosts can be added to multiple protection groups. Enter multiple protection groups in comma-separated format.

If a protection group already includes other hosts, those hosts are unaffected.

Only members of the same object type can belong to a protection group. For example, a host cannot be added to a protection group that already contains host groups or volumes.

The **purehost add** command only accepts hosts that do not already belong to any of the specified protection groups. If any of the hosts already belong to any of the protection groups, the entire command fails.

Run the **purehost list --protect** command to see a list of all protected hosts and their associated protection groups.

The **purehost remove** command removes one or more hosts from one or more protection groups. All of the specified hosts must belong to all of the specified protection groups in the command; otherwise, the command fails.

Hosts can also be added to and removed from protection groups through the **purepgroup setattr** command.

## Preferred Arrays

For synchronous replication configurations, Purity//FA supports both non-uniform and uniform storage access. In a non-uniform storage access configuration, a connected host has access to the array that is local to the host. In a uniform storage access configuration a connected host has access to both arrays within a pod.

If you have uniform storage access, but accessing one array over another is more efficient for reasons such as shorter distance and lower latency, then you may want to configure a preferred array for each host (also known as Asymmetric Logical Unit Access, or ALUA, preferred paths). The **purehost setattr --preferred-array** command sets a host's preferred array to specify which array exposes active/optimized paths to that host. If a preferred array is set for a host, then the other arrays in the same pod will expose active/non-optimized paths to that host.

A host can have more than one preferred array.

### Host I/O Performance and I/O Balance Monitoring

The **purehost monitor** command displays real-time I/O performance and I/O balance information for all or specified hosts. The **purehost monitor** displays I/O performance information, including the following data about bandwidth, IOPS, and latency:

- **Name:** Object name.
- **Time:** Current time.
- **B/s:** Bandwidth. Number of bytes read/written.
- **op/s:** IOPS. Number of read, write, or mirrored write requests processed per second.
- **us/op:** Service time. Average time, measured in microseconds, it takes the array to serve a read, write, or mirrored write I/O request. Service time does not include SAN time, queue time, or QoS rate limit time.
- **B/op:** IOPS. Average I/O size per read, write, mirrored write, and both read and write (all) operations. Must include the **--size** option to see the B/op columns.

Include the **--interval** option to specify the number of seconds between each point-in-time update. Include the **--repeat** option to specify the number of times to repeat the update. The **--interval** and **--repeat** options can be combined.

Include the **--balance** option to displays the I/O counts on each path between the host and the controller. Ideally, I/O counts on all the host paths should be as close to each other as possible. The **purehost monitor --balance** output includes the following I/O balance data:

- **Name:** Host name.
- **Time:** Current time.
- **Initiator WWN:** Fibre Channel initiator port name.
- **Initiator IQN:** iSCSI initiator port name.
- **Initiator NQN:** NVMe initiator port name.
- **Target:** For WWN, target Fibre Channel component name. For IQN, target primary or secondary controller. For NQN, NVMe storage target device.
- **Target WWN:** Target WWN port name. Only applies to WWN.
- **Failover:** Port to which this array port has failed over. The port name only appears if the array port has failed over.
- **I/O Count:** I/O count for the host path.
- **I/O Relative to Max:** Percentage of I/O counts for this path relative to the path with highest number of I/O counts. The path with the highest number of I/O counts is displayed with an I/O Relative to Max percentage of 100%. The percentage values of all other paths in the host are then calculated relative to the path with the highest number of I/O counts.

The **--interval** and **--repeat** options can be used with the **--balance** option. Include the **--interval** option to set the interval at which I/O balance details are logged. Each I/O balance update consists of the I/O statistics collected during the entire interval. Include the **--repeat** option to specify the number of times to repeat the update.

In a synchronous replication configuration, include the **--array** option to break down performance data by the array to which the I/O is directed. Include the **--mirrored** option to display performance data for mirrored writes (i.e., writes to volumes in stretched pods). If one array of a stretched pod is offline, the writes processed on the other array are reported as local writes rather than mirrored ones. Once the pod is back online, in sync, and processing each write on both sides, the writes to its volumes are reported as mirrored writes again.

The **purehost monitor --latency** command displays real-time and historical I/O latency information for volumes connected to hosts. The **purehost monitor --latency** output includes the following data:

- **SAN us/op:** SAN time. Average time, measured in microseconds, required to transfer data between the initiator and the array. Slow data transfers will result in higher SAN times.
- **Queue us/op:** Queue time. Average time, measured in microseconds, that an I/O request spends in the array waiting to be served. The time is averaged across all I/Os of the selected types.
- **QoS Rate Limit us/op:** QoS rate limit time. Average time, measured in microseconds, that reads, writes, or mirrored writes spend in queue as a result of bandwidth limits reached on one or more volumes.
- **us/op:** Service time. Average time, measured in microseconds, it takes the array to serve a read, write, or mirrored write I/O request. Service time does not include SAN time, queue time, or QoS rate limit time.

Include the **--mirrored** option to display latency data for mirrored writes (i.e., writes to volumes in stretched pods).

## Exceptions

Purity//FA will not create a host if:

- The specified name is already associated with another host in the array.
- Any of the specified IQNs, NQNs, or WWNs is already associated with an existing host in the array.
- The creation of the host would exceed the limit of concurrent hosts, or the creation of the IQN, NQN or WWN would exceed the limit of concurrent initiators.

Purity//FA will not delete a host while it has connections to volumes, either private or shared.

Purity//FA will not associate an IQN, NQN or WWN with a host if:

- The creation of the IQN, NQN or WWN would exceed the maximum number of concurrent initiators.

- The specified IQN, NQN or WWN is already associated with another host on the array.

Hosts are configured through the Purity//FA GUI (**Storage** tab) and Purity//FA CLI (**purehost** command).

## Examples

### Example 1

```
purehost create HOST1
```

Creates a host called **HOST1**. **HOST1** cannot be connected to volumes or associated with a host group until at least one WWN has been associated with it.

### Example 2

```
purehost create --wwnlist 0123456789abcde1,0123456789abcde2 HOSTFC  
purehost create --iqnlist iqn.2001-04.com.example:diskarrays-sn-a8675309 HOSTISCSI  
purehost create --nqnlist nqn.2001-04.com.example:diskarrays-sn-a8675309 HOSTNVME
```

Creates a host called **HOSTFC** and associates WWNs 01:23:45:67:89:ab:cd:e1 and 01:23:45:67:89:ab:cd:e2 with it.

Creates a second host called **HOSTISCSI** and associates IQN iqn.2001-04.com.example:diskarrays-sn-a8675309 with it.

Creates a third host called **HOSTNVME** and associates NQN nqn.2001-04.com.example:diskarrays-sn-a8675309 with it.

### Example 3

```
purehost delete HOST5
```

Deletes host **HOST5**. **HOST5**'s private connections and host group association (if any) must previously have been broken.

### Example 4

```
purehost add --pgroup PGROUP1,PGROUP2 HOST1 HOST2
```

Adds hosts **HOST1** and **HOST2** to protection groups **PGROUP1** and **PGROUP2**, respectively.

### Example 5

```
purehost monitor --repeat 60 --interval 10 HOST1
```

Displays real-time performance data for host **HOST1**. Sixty (60) point-in-time updates are displayed, with each update taken every ten (10) seconds.

### Example 6

```
purehost monitor --balance
```

Displays I/O balance details for all hosts.

#### Example 7

```
purehost monitor --balance --repeat 2 --interval 2 HOST3
```

Displays I/O balance details for host **HOST3** containing 2 seconds' worth of data with 2 seconds between each update.

#### Example 8

```
purehost setattr --personality esxi HOST3
```

Sets the host personality of host **HOST3** to the **esxi** virtual memory system to ensure the array works optimally with the host.

#### Example 9

```
purehost setattr --wwnlist 0123456789abcdef,01:23:45:67:89:ab:cd:ee HOST3
```

Replaces all WWN previously associated with **HOST3** with the two specified. This example also illustrates the two formats for entering WWNs.

#### Example 10

```
purehost setattr --remwwnlist 01:23:45:67:89:ab:cd:ed HOST4  
purehost setattr --addwwnlist 0123456789abcdef,0123456789abcdeb HOST4
```

Disassociates WWN 01:23:45:67:89:ab:cd:ed from **HOST4** and replaces it with 01:23:45:67:89:ab:cd:ec and 01:23:45:67:89:ab:cd:eb. Other WWNs associated with **HOST4** are unaffected.

#### Example 11

```
purehost setattr --preferred-array ARRAY1 HOST1
```

Sets the preferred array for **HOST4** to **ARRAY1**. Once the preferred array is set, **HOST1** exposes active/optimized paths to **HOST1**.

#### See Also

[purehost-connect\(1\)](#) [302], [purehost-list\(1\)](#) [307]

[purearray\(1\)](#) [230], [purehgroup\(1\)](#) [275], [purepod\(1\)](#) [412], [purevol\(1\)](#) [469]

#### Author

Pure Storage Inc. <[documentfeedback@purestorage.com](mailto:documentfeedback@purestorage.com)>

## purehost-connect

purehost-connect, purehost-disconnect, purevol-connect, purevol-disconnect — manage private connections between hosts and volumes

### Synopsis

```
purehost connect --vol VOL [--lun LUN] [--use-protocol-endpoint PROTOCOL-ENDPOINT] HOST...  
purehost disconnect --vol VOL HOST...  
purevol connect { --host HOST | --hgroup HGROUP } [--lun LUN] [--use-protocol-endpoint PROTOCOL-ENDPOINT] VOL...  
purevol disconnect { --host HOST | --hgroup HGROUP } VOL...
```

### Arguments

#### **HOST**

Name of a host to be connected to or disconnected from a volume.

#### **VOL**

Name of a volume to be connected to or disconnected from a host.

### Options

#### **-h** | **--help**

Can be used with any command or subcommand to display a brief syntax description.

#### **--hgroup**

Host group with which shared connections to the specified volumes are to be made or broken by **purevol connect** and **purevol disconnect** commands. Exactly one host group must be specified by this option. The option is mutually exclusive with the **--host** option.

#### **--host**

Host with which private connections to the specified volumes are to be made or broken by **purevol connect** and **purevol disconnect** commands. Exactly one host must be specified by this option. The option is mutually exclusive with the **--hgroup** option.

#### **--lun**

Logical unit number (LUN) by which specified hosts are to address specified volume. If not specified, Purity//FA automatically assigns a LUN to the connection. To automatically assign a LUN to a private connection, Purity//FA starts at LUN 1 and counts up to the maximum LUN 4095, assigning the first available LUN to the connection.

#### **--use-protocol-endpoint**

Connects the FlashArray virtual volume to VMware ESXi hosts.

For more information about virtual volumes, including configuration steps, refer to the Pure Storage vSphere Web Client Plugin for vSphere User Guide in the Pure1 Knowledge site at <https://support.purestorage.com>.

**--vol**

Volume with which private connections to the specified hosts are to be made or broken by **purehost connect** and **purehost disconnect** commands. Exactly one volume must be specified by this option.

## Description

Makes and breaks private connections between hosts and volumes and shared connections between hosts associated with host groups and volumes.

For private connections:

- The **purehost connect** and **purehost disconnect** commands are used to manage private connections between a single volume and one or more hosts.
- The **purevol connect** and **purevol disconnect** commands are used with the **--host** option to manage private connections between a single host and one or more volumes.

For shared connections:

- The **purevol connect** and **purevol disconnect** commands are used with the **--hgroup** option to manage shared connections between a single host group and one or more volumes. See **purehgroup-connect(1)** [283] for additional commands used to manage shared connections.

## Private vs. Shared Host-Volume Connections

Purity//FA supports two types of host-volume connections:

### Private

Connects one volume to one host. Private connections are independent of one another. For example, the sequence:

```
purevol connect --host HOST1 VOL1 VOL2  
purehost disconnect --vol VOL1 HOST1
```

connects **HOST1** to **VOL1** and **HOST1** to **VOL2**, and then disconnects **HOST1** and **VOL1**, leaving **HOST1** connected to **VOL2**.

Private connections are typically used for boot volumes and for stand-alone (non-clustered) host applications.

To establish a private connection, run the **purevol connect --host** or **purehost connect --vol** command. Both commands are functionally identical.

To break a private connection that is no longer required, run the **purevol disconnect --host** or **purehost disconnect --vol** command. Both commands are functionally identical. When a connection has been broken, its LUN is available for reuse.

### Shared

Connects a designated set of hosts (via a host group) to a designated set of volumes, providing the hosts with a consistent "view" of the volumes. All associated hosts use the same LUN to

address a given associated volume. All hosts and volumes associated with a host group are automatically connected to each other by virtue of their associations with the group.

To establish a shared connection (assuming the host belongs to a host group), run the **purevol connect --hgroup** or **purehgroup connect --vol** command. Both commands are functionally identical. For example, the command:

```
purevol connect --hgroup HGROUP1 VOL1 VOL2
```

is equivalent to the sequence:

```
purehgroup connect --vol VOL1 HGROUP1  
purehgroup connect --vol VOL2 HGROUP1
```

Both commands establish shared connections between the hosts associated with **HGROUP1** and **VOL1** and **VOL2**.

Shared connections are useful for cluster applications in which several related hosts require consistent (same LUN) connectivity to a set of storage volumes.

To break a shared connection that is no longer required, run the

**purevol disconnect --hgroup** or **purehgroup disconnect --vol** command. Both commands are functionally identical. When a connection has been broken, its LUN is available for reuse.

For more information about shared connections and the **purehgroup connect** command, refer to **purehgroup-connect(1)** [283].

A host can have only one connection to a given volume at any given time. If you attempt to make a second connection between a host and a volume, private or shared, the attempt will fail.

## App Hosts

The Purity Run platform extends array functionality by integrating add-on services into the Purity//FA operating system. Each service that runs on the platform is provided by an app.

Each app has a dedicated host, known as an app host. The app host is used to connect FlashArray volumes to the app.

Run the **purehost connect --vol** or **purevol connect --host** command to connect FlashArray volumes to an app host, and thereby its associated app. Likewise, run the **purehost disconnect --vol** or **purevol disconnect --host** command to break the connection between the FlashArray volumes and the app.

For more information about Pure Apps and the **pureapp** command, refer to **pureapp(1)** [221].

## LUN Management

When a volume is connected to a host, it is assigned a logical unit number (LUN) ID, which the host uses to communicate with the volume. A volume can be connected to individual hosts as well as to multiple host groups simultaneously. Include the **--lun** option with the **purehost connect** or **purevol connect --host** command to manually assign a LUN ID anywhere in the **[1...4095]** range.

Purity//FA automatically assigns a LUN to the private connection. To do this, Purity//FA starts at LUN 1 and counts up to the maximum LUN 4095, assigning the first available LUN to the connection.

If multiple hosts are specified in a single `purehost connect` command, there is no guarantee that the same LUN will be associated with each connection established. When `--lun` and `--host` are both specified in a `purehost connect` or `purevol connect` command, exactly one host and one volume must be specified.

To change the LUN associated with a private connection, the connection must first be broken and then recreated by `purehost connect` or `purevol connect`.

## Exceptions

Purity//FA will not establish a (private) connection between a volume and a host if:

- An unavailable LUN was specified.
- The volume is already connected to the host, either through a private or shared connection.

## Examples

### Example 1

```
purevol connect --host HOST1 VOL1 VOL2
```

Establishes private connections between `HOST1` and `VOL1` and between `HOST1` and `VOL2`.

Purity//FA assigns a LUN to each connection. If `HOST1` is associated with a host group, Purity//FA assigns a LUN according to the LUN assignment guidelines for shared connections. If `HOST1` is not associated with a host group, Purity//FA assigns a LUN according to the LUN assignment guidelines for private connections.

### Example 2

```
purehost connect --vol VOL3 --lun 4 HOST2
purevol connect --host HOST2 --lun 5 VOL4
```

Establishes private connections between `HOST2` and `VOL3` and between `HOST2` and `VOL4`. If LUN 4 or LUN 5 is already in use by another connection to `HOST2`, the corresponding connection fails.

### Example 3

```
purevol connect --host @linux app_vo1001
```

Connects FlashArray volume `app_vo1001` to the `linux` app via app host `@linux`.

### Example 4

```
purehost disconnect --vol app_vo1002 @linux
```

Breaks the connection between FlashArray volume `app_vo1002` and the `linux` app.

## See Also

[purehost\(1\)](#) [292], [purehost-list\(1\)](#) [307]  
[pureapp\(1\)](#) [221], [purehgroup\(1\)](#) [275], [purevol\(1\)](#) [469]

## Author

Pure Storage Inc. <[documentfeedback@purestorage.com](mailto:documentfeedback@purestorage.com)>

## purehost-list

purehost-list, purehost-listobj — display information about Purity//FA host objects, host-volume connections, and storage provisioning and consumption.

### Synopsis

```
purehost list [ --all | --chap | --connect | --personality | --protect | --remote |  
--space ] [ --cli | --csv | --nvp ] [--filter FILTER] [--limit LIMIT] [--notitle]  
[--page] [ --private | --shared ] [--raw] [--sort SORT] [HOST...]  
purehost listobj [--csv] [--remote] [ --type { host | iqn | nqn | vol | wwn } ]  
[HOST...]
```

### Arguments

#### **HOST**

Host object for which the information specified by options is to be displayed.

### Options

Options that control information displayed:

**-h | --help**

Can be used with any command or subcommand to display a brief syntax description.

**default** (no content option specified with **purehost list**)

Displays associated world wide names and host groups for the specified hosts.

**--all** (**purehost list** only)

Displays all visible attributes of the specified hosts.

**--chap** (**purehost list** only)

Displays CHAP authentication settings.

**--connect** (**purehost list** only)

Displays volumes connected to the specified hosts and the LUNs used to address them.

**--personality** (**purehost list** only)

Displays host personality settings. The host personality determines how the Purity//FA system tunes the protocol used between the array and the initiator. To ensure the array works optimally with the host, the host personality should be the name of the host operating or virtual memory system. The host personality is set through the **purehost setattr --personality** command.

**--private** (**purehost list connect** only)

Restricts the list or display of volumes connected to specified hosts to those with private connections. Invalid when combined with other options.

**--protect** (**purehost list** only)

Displays all protected hosts and their associated protection groups.

--remote

Includes a list of remote hosts.

--shared (**purehost list connect** only)

Restricts the display of volumes connected to specified hosts to those with shared connections.  
Invalid when combined with other options.

--space (**purehost list** only)

Displays size and space consumption information for the volumes connected to each host.

--type (**purehost listobj** only)

Specifies the type of information about specified hosts that is to be produced in whitespace-separated format suitable for scripting.

Options that control display format:

--cli

Displays output in the form of CLI commands that can be issued to reproduce the current configuration. The --cli output is not meaningful when combined with immutable attributes.

--csv

Lists information in comma-separated value (CSV) format. The --csv output can be used for scripting purposes and imported into spreadsheet programs.

--notitle

Lists information without column titles.

--nvp

Lists information in name-value pair (NVP) format, in the form **ITEMNAME=VALUE**. Argument names and information items are displayed flush left. The --nvp output is designed both for convenient viewing of what might otherwise be wide listings, and for parsing individual items for scripting purposes.

--page

Turns on interactive paging.

--raw

Displays the unformatted version of column titles and data. For example, in the **purearray monitor** output, the unformatted version of column title **us/op (read)** is **usec\_per\_read\_op**. The --raw output is used to sort and filter list results.

Options that manage display results:

--filter

Displays only the rows that meet the filter criteria specified.

--limit

Limits the size of the list output to the specified maximum number of rows.

--sort

Sorts the list output in ascending or descending order by the column specified.

## Description

The **purehost list** command displays a list of hosts on the array. The list includes host attributes such as host group association, interface, connected volumes (both shared and private), provisioned size, and storage consumption.

The information to be displayed is specified by including one of the following options:

- If no options are specified, displays names, associated world wide names, and host groups for specified hosts.
- If the **--all** option is specified, displays all visible attributes of the specified hosts. Display includes associated IQNs, NQNs or WWNs, host groups, and the connected volumes and the LUNs used to address them.
- If the **--chap** option is specified, displays host and target usernames. Also indicates whether host and target passwords have been set.
- If the **--connect** option is specified, displays volumes associated with the specified hosts, and the LUNs used to address them.
- If the **--personality** option is specified, displays the personality setting associated with the specified hosts.
- If the **--protect** option is specified, displays all protected hosts and their associated protection groups.
- If the **--remote** option is specified, the display includes a list of remote hosts, meaning a list of hosts that have been created on remote arrays. Hosts on remote arrays have the naming convention **ARRAY:HOST**, where **ARRAY** represents the name of the remote array, and **HOST** represents the name of the host on the remote array.
- If the **--space** option is specified, the display lists the following size and space consumption information for the volumes connected to each host:

### Size

Total provisioned size of all volumes connected to the host. Represents storage capacity reported to hosts.

### Thin Provisioning

Percentage of volume sectors that do not contain host-written data because the hosts have not written data to them or the sectors have been explicitly trimmed.

### Data Reduction

Ratio of mapped sectors within a volume versus the amount of physical space the data occupies after data compression and deduplication. The data reduction ratio does not include thin provisioning savings.

For example, a data reduction ratio of 5:1 means that for every 5 MB the host writes to the array, 1 MB is stored on the array's flash modules.

#### Total Reduction

Ratio of provisioned sectors within a volume versus the amount of physical space the data occupies after reduction via data compression and deduplication *and* with thin provisioning savings. Total reduction is data reduction with thin provisioning savings.

For example, a total reduction ratio of 10:1 means that for every 10 MB of provisioned space, 1 MB is stored on the array's flash modules.

#### Volumes

Physical space occupied by volume data that is not shared between volumes, excluding array metadata and snapshots.

#### Snapshots

Physical space occupied by data unique to one or more snapshots.

#### Total

Total physical space occupied by system, shared space, volume, and snapshot data.

By default, space consumption for all connected volumes, both private and shared, is displayed. The display can be restricted to volumes with private or shared connections by specifying the **--private** or the **--shared** option.

The **purehost listobj** command produces lists of certain attributes of specified hosts in either whitespace or comma-separated form, suitable for scripting.

If the **--remote** option is specified, the display includes a list of remote hosts, meaning a list of hosts that have been created on remote arrays. Hosts on remote arrays have the naming convention **ARRAY:HOST**, where **ARRAY** represents the name of the remote array, and **HOST** represents the name of the host on the remote array.

Include the **--type** option to display one of the following types of lists:

**--type host** (default if **--type** option not specified)

List contains the specified host names. If no host names are specified, the list contains the names of all host objects.

**--type iqn**

List contains the IQNs associated with each specified host. If no hosts are specified, the list contains all IQNs (administratively assigned and discovered) known to the array.

**--type nqn**

List contains the NQNs associated with each specified host. If no hosts are specified, the list contains all NQNs (administratively assigned and discovered) known to the array.

**--type vol**

List contains the volumes connected to the specified hosts. If no hosts are specified, list contains names of all volumes connected to any host. List can be restricted to show only private connections by specifying the **--private** option.

```
--type wwn
```

List contains the world wide names associated with each specified host. If no hosts are specified, the list contains all world wide names (administratively assigned and discovered) known to the array.

Lists are whitespace-separated by default. Specify the **--csv** option to produce a comma-separated list.

## App Hosts

App hosts are used to connect FlashArray volumes to Pure apps. If an app is installed on the array, its app host will appear in the **purehost list** output.

App host names begin with a distinctive @ symbol. The naming convention for app hosts is @**APP**, where **APP** denotes the app name.

The following example displays the app host for the **linux** app.

```
$ purehost list @linux*
Name      WWN    IQN    NQN  Host Group
@linux    -      -      -     -
```

Include the **--connect** option to display all of the volumes associated with the app host, and thereby connected to the app. In the following example, five FlashArray volumes, along with the **linux** boot and data volumes, are associated with the **@linux** app host, and thereby connected to the **@linux** app.

```
$ purehost list --connect @linux*
Name      LUN    Vol          Host Group
@linux    1      @linux_boot  -
@linux    2      @linux_data  -
@linux    3      app_vol001   -
@linux    4      app_vol002   -
@linux    5      app_vol003   -
@linux    6      app_vol004   -
@linux    7      app_vol005   -
```

For more information about Pure apps and the **pureapp** command, refer to **pureapp(1)** [221].

## Exceptions

None.

## Examples

### Example 1

```
purehost list
```

Displays names, associated world wide names, and associated host groups (if any) for all hosts.

## Example 2

```
purehost list --connect
```

Displays names, connected volumes and logical units for all hosts. Both private and shared volume connections are displayed.

## Example 3

```
purehost list --space --private --csv --notitle HOST1 HOST2 HOST3
```

Displays the above mentioned virtual and physical space consumption for volumes associated with each of **HOST1**, **HOST2**, and **HOST3**.

## Example 4

```
purehost list --remote
```

Displays a list of all hosts on both the local and remote arrays, their associated IQNs, NQNs, or WWNs and their associated host groups (if any).

## Example 5

```
purevol list --space $(purehost listobj --type vol HOST1 HOST2)
```

The inner **purehost listobj** command produces a whitespace-separated list of the volumes connected to **HOST1** and **HOST2**. The outer **purevol list** command displays space consumption for the volumes specified in the inner command.

## Example 6

```
purehost listobj --type vol --private HOST1 HOST2
```

Lists all volumes with private connections to **HOST1** and **HOST2**.

## Example 7

```
purevol list --connect --private $(purehost listobj --type vol HOST1)
```

The inner **purehost listobj** command produces a list of the volumes to which **HOST1** is connected. The list is input to the **purevol list** command to display all hosts with private connections to those volumes.

## Example 8

```
purehost list --connect @linux
```

Displays the names of all volumes associated with app host **linux**, and thereby connected to the **linux** app.

## See Also

purehost(1) [292], purehost-connect(1) [302]  
pureapp(1) [221], purehgroup(1) [275], purevol(1) [469]

## Author

Pure Storage Inc. <[documentfeedback@purestorage.com](mailto:documentfeedback@purestorage.com)>

## purehw

purehw, purehw-list, purehw-setattr — displays information about and controls visual identification of FlashArray hardware components

### Synopsis

```
purehw list [ --csv | --nvp ] [--notitle] [--page] [--raw] [--spec] [--type  
COMPONENT-TYPE] [COMPONENT...]  
  
purehw setattr [ --identify {off | on} | --index INDEX ] COMPONENT...
```

### Arguments

#### COMPONENT

Hardware component whose information is to be displayed or whose attribute is to be set to the specified value.

### Options

#### -h | --help

Can be used with any command or subcommand to display a brief syntax description.

#### --identify

Turns a visual identifier for the component on or off. Valid for controllers, NVRAM bays, storage bays, and storage shelves. Used with the **purehw setattr** command to set front panel LED identifiers on or off.

#### --index

In multi-shelf arrays, sets the top-level shelf component to a unique number. Once set, the number becomes part of the name of each component in the shelf.

#### --spec

Displays hardware component model names, part numbers, and serial numbers.

#### --type

Type of component for which information is to be displayed. When this option is specified, information is displayed for all components of the specified type. Valid values for **--type** are: **ch** (chassis), **ct** (controller), **bay** (flash module bay), **eth** (Ethernet port), **fan** (fan), **fc** (Fibre Channel port), **ib** (InfiniBand port), **iom** (I/O module), **nvb** (NVRAM module bay), **pwr** (power supply), **sas** (SAS port), **tmp** (temperature sensor), and **sh** (storage shelf).

Supported **--type** values vary by FlashArray model.

Options that control display format:

#### --csv

Lists information in comma-separated value (CSV) format. The **--csv** output can be used for scripting purposes and imported into spreadsheet programs.

#### --notitle

Lists information without column titles.

--nvp

Lists information in name-value pair (NVP) format, in the form **ITEMNAME=VALUE**. Argument names and information items are displayed flush left. The --nvp output is designed both for convenient viewing of what might otherwise be wide listings, and for parsing individual items for scripting purposes.

--page

Turns on interactive paging.

--raw

Displays the unformatted version of column titles and data. For example, in the **purearray monitor** output, the unformatted version of column title **us/op (read)** is **usec\_per\_read\_op**. The --raw output is used to sort and filter list results.

## Description

Most FlashArray hardware components report their operational status. The **purehw list** command displays information about the FlashArray hardware components. The **purehw setattr** command controls the visual identification of certain hardware components, and sets numerical values of top-level shelf components.

## FlashArray Hardware Components

Hardware components and their naming vary by FlashArray series. To see the hardware technical specifications for each FlashArray model, refer to the Products page at <https://www.purestorage.com>.

### Hardware Components in FlashArray//X and FlashArray//M

The //X and //M chassis, controller and storage shelf names have the form **xxm**. The names of other components have the form **xxm.yyn** or **xxm.yyyyn**, where:

**xx**

Denotes the type of component:

- **CH** - //X or //M chassis.
- **CT** - Controller.
- **SH** - Storage shelf.

**m**

Identifies the specific controller or storage shelf:

- For an //X or //M chassis, **m** has a value of 0. For example, **CH0**.
- For controllers, **m** has a value of 0 or 1. For example, **CT0**, **CT1**.
- For storage shelves, **m** represents the shelf number, starting at 0. For example, **SH0**, **SH1**. The assigned number can be changed on the shelf front panel or by running the **purehw setattr --id** command.

**YY or YYY**

Denotes the type of component. For example, **FAN** for cooling device, **FC** for Fibre Channel port.

**n**

Identifies the specific component by its index (its relative position within the //X or //M chassis, controller, or storage shelf), starting at 0.

The following table lists //X and //M hardware components that report status, grouped by their location on the array. The Identify Light column shows which components have an LED light on the physical component that can be turned on and off.

### **Chassis (CH0)**

<b>Component Name</b>	<b>Identify Light</b>	<b>Component Type</b>
CH0	Yes	Chassis
CH0.BAYn	Yes	Storage bay
CH0.NVBN	Yes	NVRAM bay
CH0.PWRN	--	Power module

### **Controller (CTm)**

<b>Component Name</b>	<b>Identify Light</b>	<b>Component Type</b>
CTm	Yes	Controller
CTm.ETHn	--	Ethernet port
CTm.FANn	--	Cooling fan
CTm.FCn	--	Fibre Channel port
CTm.IBn	--	InfiniBand port (included only with certain upgrade kits)
CTm.SASN	--	SAS port
CTm.TMPn	--	Temperature sensor

### **Storage Shelf (SHm)**

<b>Component Name</b>	<b>Identify Light</b>	<b>Component Type</b>
SHm	Yes	Storage shelf
SHm.BAYn	Yes	Storage bay
SHm.FANn	--	Cooling fan
SHm.IOMn	--	I/O module
SHm.PWRN	--	Power module
SHm.SASN	--	SAS port
SHm.TMPn	--	Temperature sensor

### FlashArray//X and FlashArray//M Chassis Components (CH0)

In //X and //M, the chassis (**CH0**) contains the controllers and the bays that host the Flash Modules and NVRAM modules. The controller component names use **CT0**. The component names for the bays use

**CH0.** The chassis status is generated from a combination of software conditions and the status of its components.

The components in an //X or //M chassis include:

- **Storage Bay (CH0.BAYn):**

Each storage shelf contains a row of hot-swappable storage bays, which contain flash modules used for the array's data. Storage bays are indexed from left to right starting at 0.

- **NVRAM Bay (CH0.NVBr):**

The chassis contains multiple NVRAM bays, which are hot-swappable and house NVRAM modules (temporary holding areas for host-written data). Bays are indexed from left to right, facing the front of the shelf.

- **Power Supply (CH0.PWRn):**

The internal shelf contains redundant power supplies that report status. Power supply replacement is a service technician operation.

#### FlashArray//X or FlashArray//M Controller Components (CTm)

Purity//FA reports an overall controller (CTm) status. The controller status is generated from a combination of software conditions and the status of its components. For example, the status `ok` is reported for properly functioning controllers. Controllers contain 2- or 4-port Fibre Channel interface cards, 2-port Ethernet I/O interface cards, or a mix of both types.

The components in an //X or //M controller include:

- **Ethernet Port (CTm.ETHn):**

Ethernet ports report status and link speed. Each controller contains at least 1x1GbE port for management (including one used for remote array administration) and 2x10GbE ports for replication.

- **Fan (CTm.FANn):**

Each controller contains multiple cooling fans that report status. Fan replacement is a service technician operation.

- **Fibre Channel Port (CTm.FCn):**

Fibre Channel ports report status and link speed. Most Fibre Channel-based controllers contain redundant 2- or 4-port Fibre Channel host bus adapters. Some controllers support a single Fibre Channel adapter. To see the Fibre Channel port details for each FlashArray model, refer to the Products page at <https://support.purestorage.com>.

- **InfiniBand Port (CTm.IBn):**

InfiniBand ports are shipped only in upgrade kits for FA-4xx-to-FlashArray//X or //M upgrades. FlashArray//X and //M use InfiniBand ports only for controller upgrades from FA-4xx systems. InfiniBand ports report status and communication speed.

- **SAS Ports (CTm.SAS):**

Each controller contains multiple dual-port SAS interface cards that communicate with flash modules. SAS ports report status and communication speed.

- **Temperature Sensor (CTm.TMPn):**

Each controller chassis contains multiple temperature sensors that report their status and the ambient temperature at various points within the chassis. In steady-state operation, Purity//FA generates alerts when a temperature sensor reports a value outside of normal operating range.

### Expansion Shelf Components (SHm)

Purity//FA reports an overall storage shelf (**SHm**) status generated from a combination of software conditions and the status of its components.

The optional expansion shelf is available with most //M models.

The components in an expansion shelf include:

- **Storage Bay (SHm.BAYn):**

Each storage shelf contains a row of hot-swappable storage bays, which contain flash modules used for the array's data. Storage bays are indexed from left to right starting at 0.

- **Fan (SHm.FANn):**

Each of a shelf's power and cooling modules (PCMs) reports cooling fan status.

- **IOM (SHm.IOMn):**

Redundant I/O Modules (IOMs) accessed from the rear of the shelf chassis contain the shelf's SAS ports.

- **Power Supply (SHm.PWRn):**

The power supply sub-component within each of a shelf's power and cooling modules (PCMs) reports its operating status. Power supply replacement is a service technician operation.

- **SAS Ports (SHm.SASN):**

Multiple SAS ports handle communication between the array's controllers and the shelf's flash modules.

- **Temperature Sensor (SHm.TMPn):**

Each controller chassis contains multiple temperature sensors that report their status and the ambient temperature at various points within the chassis. In steady-state operation, Purity//FA generates alerts when a temperature sensor reports a value outside of normal operating range.

### Hardware Components in the FA-400 Series

In the FA-400 series, the controller and storage shelf names have the form **xxm**. The names of other components have the form **xxm.yyn** or **xxm.yyyyn**, where:

**xx**

Denotes the type of chassis. Controllers use **CT**. Storage shelves use **SH**.

**m**

Identifies the specific controller or storage shelf.

- For controllers, **m** has a value of 0 or 1. For example, **CT0**, **CT1**.
- For storage shelves, **m** represents the shelf number, starting at 0. For example, **SH0**, **SH1**.

The assigned number can be changed on the shelf front panel or by running the `purehw setattr --id` command.

#### **YY or YYY**

Denotes the type of component. For example, `FAN` for cooling device, `FC` for Fibre Channel port.

#### **n**

Identifies the specific component by its index (its relative position within the array), starting at 0.

The following table lists FA-400 hardware components that report status, grouped by their location on the array. The Identify Light column shows which components have an LED light on the physical component that can be turned on and off.

### **Controller (CTm)**

<b>Component Name</b>	<b>Identify Light</b>	<b>Component Type</b>
CTm	Yes	Controller
CTm.ETHn	--	Ethernet port
CTm.FANn	--	Cooling fan
CTm.FCn	--	Fibre Channel port
CTm.IBn	--	InfiniBand port
CTm.PWRn	--	Power module
CTm.SASN	--	SAS port
CTm.TMPn	--	Temperature sensor

### **Storage Shelf (SHm)**

<b>Component Name</b>	<b>Identify Light</b>	<b>Component Type</b>
SHn	Yes	Storage shelf
SHn.BAYn	Yes	Storage bay
SHn.FANn	--	Cooling fan
SHn.IOMn	--	I/O module
SHn.PWRn	--	Power module
SHn.SASN	--	SAS port
SHn.TMPn	--	Temperature sensor

### FA-400 Controller Components (CTm)

Purity//FA reports an overall controller (`CTm`) status. The controller status is generated from a combination of software conditions and the status of its components. For example, the status `ok` is reported for properly functioning controllers. Controllers contain 2-port Fibre Channel interface cards, 2-port Ethernet I/O interface cards, or a mix of both types.

The components in a FA-400 controller include:

- **Ethernet Port (CTm.ETHn):**

Ethernet ports report status and link speed. Each controller contains at least 1x1GbE port for management (including one used for remote array administration) and 2x10GbE ports for replication.

- **FAN (CTm.FANn):**

Each controller contains multiple cooling fans that report status. Fan replacement is a service technician operation.

- **Fibre Channel Port (CTm.FCn):**

Fibre Channel ports report status and link speed. Most Fibre Channel-based controllers contain redundant 2-port Fibre Channel host bus adapters. Certain controllers, such as the FA-405 controller and controllers with mixed I/O configurations, support a single Fibre Channel adapter. To see the Fibre Channel port details for each FlashArray model, refer to the Products page at <https://support.purestorage.com>.

- **InfiniBand Port (CTm.IBn):**

Controllers within an FA-400 series array communicate via redundant InfiniBand interfaces located on a single PCI Express host bus adapter. InfiniBand ports report status and communication speed.

- **Power Supply (CTm.PWRn):**

Each controller contains redundant power supplies that report status. Power supply replacement is a service technician operation.

- **SAS Ports (CTm.SASN):**

Multiple SAS ports handle communication between the array's controllers and the shelf's flash modules and NVRAM modules. **Note:** The FA-405 model supports only a single SAS card.

- **Temperature Sensor (CTm.TMPn):**

Each controller chassis contains multiple temperature sensors that report their status and the ambient temperature at various points within the chassis. In steady-state operation, Purity//FA generates alerts when a temperature sensor reports a value outside of normal operating range.

## FA-400 Storage Shelf Components (SHm)

Purity//FA reports an overall storage shelf (SHm) status generated from a combination of software conditions and the status of its components.

The components in a FA-400 storage shelf include:

- **Shelf (SHm):**

The first two storage shelves contain both storage bays and NVRAM bays. Additional shelves, if present, contain only storage bays.

- **Storage Bay (SHm.BAYn):**

Each storage shelf contains a row of hot-swappable storage bays, which contain flash modules used for the array's data. Storage bays are indexed from left to right starting at 0.

- **Fan (SHm.FANn):**

Each of a shelf's power and cooling modules (PCMs) reports cooling fan status.

- **IOM (SHm.IOMn):**  
Redundant I/O Modules (IOMs) accessed from the rear of the shelf chassis contain the shelf's SAS ports.
- **Power Supply (SHm.PWRn):**  
The power supply sub-component within each of a shelf's power and cooling modules (PCMs) reports its operating status. Power supply replacement is a service technician operation.
- **SAS Port (SHm.SASN):**  
Multiple SAS ports handle communication between the array's controllers and the shelf's flash modules and NVRAM modules. **Note:** The FA-405 model supports only a single SAS card.
- **Temperature Sensor (SHm.TMPn):**  
Each controller chassis contains multiple temperature sensors that report their status and the ambient temperature at various points within the chassis. In steady-state operation, Purity//FA generates alerts when a temperature sensor reports a value outside of normal operating range.

## Viewing Hardware Component Details

The `purehw list` command displays information about array hardware components that are capable of reporting their status. The display is primarily useful for diagnosing hardware-related problems.

If one or more component names are specified in the command line, information is displayed for those components only. If component names are not specified in the command line, information is displayed for all components.

Include the `--spec` option to display the hardware specifications for each component wherever applicable. Hardware specifications, all of which are unique to Pure Storage, include FlashArray hardware model names, part numbers, and serial numbers. Component part numbers and serial numbers and are also printed on the physical hardware.

Include the `--type` option to display information for all components of the specified type.

The `purehw list` output includes the following columns:

### Status

Each component reports its status as either:

`ok`

Functioning properly at full capacity.

`critical`

Not functioning or requiring immediate attention.

`degraded`

Functioning, but not at full capability due to a non-fatal failure.

`device_off`

Installed, but powered off.

**identifying**

Functioning, but is not yet initialized.

**not\_installed**

Does not appear to be installed.

**unknown**

Insufficient information to determine a state for this device.

## Identify

State (on or off) of an LED used to visually identify the component. (Relevant only for controllers, NVRAM bays, storage bays, and storage shelves.)

## Slot

Slot number occupied by the PCI Express card that hosts the component. (Relevant only for ports hosted by PCI Express cards in controller chassis.)

## Index

Number that identifies the relative position of a hardware component within the array.

## Speed

Speed at which the component is operating. (Relevant only for interface ports [bits/second].)

## Temperature

Temperature reported by the component. (Relevant only for temperature sensors.)

## Voltage

Input voltage (VIN) of the power supplies in the chassis. Only applies to //M models.

## Changing Hardware Component Attributes

FlashArray controllers, storage shelves, NVRAM bays, and storage bays are equipped with LEDs that are illuminated to identify a physical component. The **purehw setattr --identify** command turns a component's identifying light on or off. The Identify column in the **purehw list** output displays the current state of the identifying light for each component that has one.

Storage shelf control panels contain numeric displays in which numbers can be set to uniquely identify shelves in multi-shelf arrays. Run the **purehw setattr --index** command to set the top-level storage shelf number. Once the top-level storage shelf number has been set, the number becomes part of the name of each component in the shelf. For example, the **purehw setattr --index 2 SH0** command sets shelf **SH0** to **SH2**, and all of the components in the shelf automatically assume names in the form **SH2.COMPONENT-NAME** (e.g., **SH2.BAY0**, **SH2.FAN1**, etc.).

## Examples

### Example 1

```
purehw list SH0.BAY11 SH0.BAY12
```

Displays information for flash modules in slots 11 and 12 of storage shelf 0.

#### Example 2

```
purehw list --type fan
```

Displays information for all cooling fans in the array.

#### Example 3

```
purehw setattr --identify on CH0.BAY1
```

Illuminates the identifying LED for storage bay 1 in shelf CH0.

#### Example 4

```
purehw setattr --index 2 SH0
```

Sets shelf SH0 and all of its components to SH2. For example, sets SH0 to SH2, SH0.BAY0 to SH2.BAY0, SH0.FAN1 to SH2.FAN1, and so on).

### See Also

[puredrive\(1\)](#) [260]

### Author

Pure Storage Inc. <[documentfeedback@purestorage.com](mailto:documentfeedback@purestorage.com)>

## purekmip

purekmip, purekmip-create, purekmip-delete — creates and deletes KMIP server objects, and imports CA certificates

purekmip-list — displays the attributes of KMIP server objects and CA certificates

purekmip-setattr — changes the attributes of a KMIP server object or imports CA certificates

purekmip-test — tests the KMIP server connectivity

### Synopsis

```
purekmip create [--ca-certificate] [--certificate CERTIFICATE] [--uri URI-LIST]
```

**KMIP**...

```
purekmip delete KMIP...
```

```
purekmip list [--ca-certificate KMIP] [KMIP...]
```

```
purekmip setattr [--ca-certificate] [--certificate CERTIFICATE] [--uri URI-LIST]
```

**KMIP**...

```
purekmip test KMIP...
```

### Arguments

#### **KMIP**

KMIP server object to be created, deleted, modified, tested, or displayed.

### Options

#### -h | --help

Can be used with any command or subcommand to display a brief syntax description.

#### --ca-certificate

Imports the CA certificate generated by a certificate authority (CA) as the KMIP server communication certificate.

#### --ca-certificate **KMIP**

Displays the CA certificate of a certificate authority (CA) as the KMIP server communication certificate for the specified KMIP server object.

#### --certificate **CERTIFICATE**

CSR or self-signed certificate used to establish a mutually authenticated connection between the FlashArray array and the KMIP server.

#### --uri **URI-LIST**

Comma-separated list of URIs assigned to the KMIP server object. Specify the URI in the format **PROTOCOL://HOSTNAME:PORT**. For example,  
**tls://192.0.2.100:80,tls://192.0.2.101:80**.

## Description

The Key Management Interoperability Protocol (KMIP) server is used in combination with the Pure Storage Rapid Data Locking (RDL) feature and EncryptReduce feature to further secure the FlashArray array's encrypted data.

For more information on how to configure the KMIP server with the Rapid Data Locking and EncryptReduce features, refer to the EncryptReduce Guide in the Pure1 Knowledge site at <https://support.purestorage.com>.

## Exceptions

None.

## Examples

### Example 1

```
purekmip create DSM_kmip --uri vormetric-dsm3.dev.purestorage.com:5696  
                      --certificate DSM_cert --ca-certificate
```

Creates the DSM\_kmip KMIP object of the remote data security manager (DSM) KMIP server with the **DSM\_cert** certificate for communication with the KMIP server. Before running this command, you should obtain the CA certificate from your certificate authority.

### Example 2

```
purekmip test DSM_kmip
```

Tests the communication between the FlashArray and the KMIP server using the DSM\_kmip object created in Example 2.

### Example 3

```
purekmip list --ca-certificate DSM_kmip
```

Displays the CA certificate of the DSM\_kmip KMIP object.

### Example 4

```
purekmip create DSM_kmip --uri vormetric-dsm3.dev.purestorage.com:5696,  
                      vormetric-dsm2.dev.purestorage.com:5696 --certificate DSM_cert --ca-certificate
```

Creates the DSM\_kmip KMIP object of two remote DSM KMIP servers with the **DSM\_cert** certificate.

## See Also

[purearray\(1\)](#) [230], [purecert\(1\)](#) [251]

## Author

Pure Storage Inc. <[documentfeedback@purestorage.com](mailto:documentfeedback@purestorage.com)>

## purelicense

purelicense — End User Agreement (EUA), Subscription Services Addendum, and Professional Services Addendum between Pure Storage Inc. and end users of the company's products

### END USER AGREEMENT (v27Mar19)

The most recent version of the End User Agreement governs use of this software. The most recent version of the agreement can be obtained at <http://www.purestorage.com/legal/productenduserinfo.html>.

**IMPORTANT: PLEASE READ THIS END USER AGREEMENT (“AGREEMENT”) BEFORE INSTALLING OR USING THE HARDWARE, SOFTWARE, OR SUBSCRIPTION SERVICES, INCLUDING ANY HARDWARE OR SOFTWARE COMPONENTS THEREOF (COLLECTIVELY, “PRODUCTS”) AND PURE-BRANDED PROFESSIONAL SERVICES (“SERVICES”) THAT YOU OR THE ENTITY THAT YOU REPRESENT (“END USER”) OBTAIN FROM PURE STORAGE, INC. (“PURE”) OR FROM ANY THIRD PARTY AUTHORIZED BY PURE TO RESELL THE PRODUCTS AND SERVICES. BY INSTALLING OR USING THE PRODUCTS OR SERVICES, YOU REPRESENT AND WARRANT THAT YOU HAVE THE AUTHORITY TO BIND END USER AND AGREE THAT END USER IS BOUND BY THIS AGREEMENT WITH PURE, UNLESS A SEPARATE WRITTEN AGREEMENT IS IN EFFECT THAT SPECIFICALLY GOVERNS THE SUBJECT MATTER HEREOF.**

#### 1. SOFTWARE LICENSE.

**1.1. Software License.** Subject to End User’s compliance with the terms and conditions of this Agreement, Pure grants to End User, and any third party that End User authorizes to perform services involving the Product solely for End User’s benefit, a nontransferable, nonexclusive, perpetual license to use and execute the Pure software provided with, or incorporated in, the Pure hardware (the “Software”), in executable object code format only, and solely for use in accordance with the applicable Product documentation and Product SKU description. Pure may make Software updates and new releases available for installation by the End User and such updates will be subject to the terms of this Agreement.

**1.2. Subscription Services.** Software that is offered by Pure subject to permissions and/or limitations identified in the applicable SKU, schedule or quote for a specified period of time is provided to End User under the terms of Pure’s Subscription Services Addendum, incorporated herein by reference.

#### 2. PRODUCT RESTRICTIONS AND TITLE.

**2.1. Restrictions.** End User will not directly or indirectly (i) reproduce, modify, distribute, assign, disclose or make available any portion of the Products (or any related documentation) to any third party (except as otherwise authorized herein); (ii) rent, lease or sublicense the Products, unless otherwise authorized by Pure in writing; (iii) reverse engineer, decompile, or disassemble any portion of the Products, or otherwise attempt to decrypt, extract or derive source code for, or any algorithms or data structures embodied within, any portion of the Products (except to the extent the foregoing restriction is expressly prohibited by applicable law); (iv) use the Products to develop a similar product or service; (v) transfer or copy the Software to, or use the Software on, any other product or device, including any second-hand or grey market hardware that End User has not purchased from Pure or a Pure authorized reseller; or (vi) publish or disclose to any third party any technical features, performance or

benchmark tests, or comparative or competitive analyses relating to the Products, except for internal use by the End User or as may be authorized by Pure in writing. End User will remain fully and primarily responsible to Pure for compliance with this Agreement if End User permits any third party to access the Products. Any future release, update, or other addition to functionality of the Products made available by Pure to End User shall be subject to the terms and conditions of this Agreement, unless Pure expressly states otherwise. End User shall preserve and shall not remove, obscure or alter any copyright labels required by law or other proprietary notices in the Products or related documentation.

**2.2. Title.** As between Pure and End User, title to Pure hardware purchased by End User will transfer to End User. Except as provided in the foregoing sentence, Pure and its suppliers shall exclusively retain all right, title and interest, in all intellectual property rights, including patent, trademark, trade name and copyright, whether registered or not registered, in and to the Products and related documentation. Pure and its suppliers reserve all rights not expressly granted herein, and no other license or other implied rights of any kind are granted or conveyed. In the event that items of software code provided with the Products are subject to “open source” or “free software” licenses, nothing herein limits End User’s rights under, or grants rights that supersede, the applicable license therefor.

### **3. PRE-RELEASE PRODUCTS AND FEEDBACK.**

**3.1. Pre-Release Products.** If mutually agreed by the parties, Pure may make available to End User beta or pre-release versions of the Products (“**Pre-Release Products**”). End User acknowledges that the Pre-Release Products (i) are not at the level of performance or compatibility of final, generally available products; (ii) may not operate correctly; (iii) may be modified prior to being made generally available; (iv) may not be made available for general release; and (v) may not be used in a production environment. End User agrees to notify Pure of any bugs or problems in the Pre-Release Products.

**3.2. Feedback.** End User may provide feedback to Pure regarding the use, operation, performance, and functionality of the Products and Pre-Release Products, including identifying potential errors and improvements (collectively, “**Feedback**”). End User grants to Pure a perpetual, irrevocable, worldwide, sublicenseable, fully paid-up and royalty-free right to modify and use the Feedback in any manner, provided that Feedback is anonymized and does not identify End User.

### **4. EVERGREEN SUBSCRIPTION; INSTALLATION.**

At its option, End User may purchase an innovation and support subscription for a purchased Product (“**Evergreen Subscription**”), which provides End User with additional software and hardware benefits. Under an applicable Evergreen Subscription, Pure will provide the generally available Product maintenance and technical support in accordance with the Pure Storage Customer Support Guide during the term for which End User has purchased such Evergreen Subscription. Depending on the Product or Evergreen Subscription purchased, certain benefits of the Evergreen Storage Program Description may also apply. Pure may designate support partners and authorized resellers to deliver the Evergreen Subscription in accordance with the terms of this Agreement. If End User purchases Pure-branded professional installation or other Services, such Services are provided to End User under the terms of Pure’s Professional Services Addendum, incorporated herein by reference.

### **5. WARRANTY AND DISCLAIMER.**

**5.1. Hardware Warranty.** Subject to this Section 5, Pure warrants that the Pure hardware will perform in substantial accordance with the corresponding Product documentation for three years from the date

of shipment by Pure. The Evergreen Subscription benefits described in Section 4 extend beyond the limited warranty for the Pure hardware for the term purchased by End User.

**5.2. Software Warranty.** Subject to this Section 5, Pure warrants that the Software will perform in substantial accordance with the corresponding Product documentation for 90 days from the date of shipment by Pure. The Evergreen Subscription benefits described in Section 4 extend beyond the limited warranty for the term purchased by End User.

**5.3. Limited Warranty Process.** End User may contact Pure via email at <[support@purestorage.com](mailto:support@purestorage.com)> or phone at +1 (866) 244-7121 for warranty service. If a return is required, End User must obtain a return material authorization number from Pure and return the Product in secure packaging, freight prepaid, as instructed by Pure. Under the hardware warranty, Pure, at its option, either (i) will repair or replace any defective Product with Product or components of equal or greater functionality as the returned Product, or (ii) will refund the purchase price paid to Pure for such Product, reduced on a straight-line basis over a three-year life. Replacement Products or components will continue to be warranted for the remainder of the applicable warranty term. Repair, replacement or refund is the sole and exclusive remedy for breach of this warranty. Under the Software warranty, Pure will provide End User access to bug fixes and emergency patches. This warranty is provided to the original End User only and is not transferable.

**5.4. Exclusions.** The warranties herein do not cover defects or damages resulting from: (a) use of Products other than in a normal and customary manner in accordance with Pure's documentation; (b) physical or electronic abuse or misuse, accident, or neglect; or (c) alterations or repairs made to Products that are not authorized by Pure in writing. Pre-Release Products are provided without warranty or liability of any kind, for use at End User's own risk. Pure will use reasonable efforts to destroy (but have no liability for any loss or inadvertent disclosure of) data stored or remaining on a Product returned to Pure. Under this Agreement, all returned Products and components become the property of Pure.

**5.5. Disclaimer.** EXCEPT FOR THE EXPRESS WARRANTIES IN THIS AGREEMENT, PURE DISCLAIMS ALL OTHER WARRANTIES, EXPRESS, IMPLIED OR STATUTORY, TO THE EXTENT WARRANTIES MAY BE DISCLAIMED UNDER APPLICABLE LAW, INCLUDING ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. PURE DOES NOT WARRANT AGAINST LOSS OR INACCURACY OF DATA OR THAT THE OPERATION OF THE PRODUCT WILL BE UNINTERRUPTED OR ERROR FREE. EXCEPT AS EXPRESSLY STATED IN SECTION 5.1, PURE PROVIDES THE PRODUCTS (INCLUDING ANY SOFTWARE) ON AN "AS IS" BASIS. THE PRODUCT IS NOT DESIGNED OR INTENDED FOR USE WHERE FAILURE OF THE PRODUCT COULD REASONABLY BE EXPECTED TO RESULT IN PERSONAL INJURY, LOSS OF LIFE OR PROPERTY DAMAGE. END USER IS RESPONSIBLE FOR ENSURING THAT IT HAS APPROPRIATE DATA BACK-UP, DATA RECOVERY, AND DISASTER RECOVERY MEASURES IN PLACE.

## 6. INDEMNIFICATION.

Pure will indemnify and defend End User, at Pure's expense, against any action brought by a third party against End User to the extent that the action is based upon a claim that the Products directly infringe any copyrights or U.S. patents or misappropriate any trade secrets, and Pure will pay those costs and damages finally awarded by a court of competent jurisdiction against End User in any such action that are specifically attributable to such claim or those costs and damages agreed to by Pure in a monetary

settlement of such action. If End User's use of the Product is, or in Pure's opinion is likely to become, enjoined as a result of an infringement claim, Pure will, at its option and expense, either (i) procure the right to continue using the Product; (ii) replace or modify the Product so that it becomes non-infringing and remains functionally equivalent; or (iii) if, despite its commercially reasonable efforts, Pure is unable to do either (i) or (ii), Pure will accept return of the Product, terminate the rights herein, and pay to End User a prorated refund of the money paid to Pure for the purchase of such Product reduced on a straight-line basis over a three-year life. Notwithstanding the foregoing, Pure will have no obligation with respect to any infringement claim based upon (a) any use of the Product that is not in accordance with this Agreement or the corresponding Product documentation; (b) any use of the Product in combination with other products, equipment, software, or data not supplied by Pure if such infringement would not have arisen but for such combination; (c) the use of any release of the Software other than the current and immediately preceding version; or (d) any modification of the Product by any person other than Pure if such infringement would not have arisen but for such modification. This Section 6 states Pure's entire liability, and End User's sole and exclusive remedy, for infringement claims and actions. The foregoing obligations are subject to End User notifying Pure promptly in writing of such action, giving Pure sole control of the defense thereof and any related settlement negotiations, and cooperating and assisting in such defense at Pure's reasonable request and expense (including reasonable attorneys' fees).

## 7. LIMITATION OF LIABILITY.

IN NO EVENT WILL PURE, ITS PARENTS, SUBSIDIARIES, AFFILIATES, AND THEIR RESPECTIVE DIRECTORS, OFFICERS, SHAREHOLDERS AND EMPLOYEES NOR ITS SUPPLIERS (COLLECTIVELY, THE "PURE PARTIES") BE LIABLE FOR ANY SPECIAL, CONSEQUENTIAL, EXEMPLARY, INCIDENTAL, OR INDIRECT DAMAGES, OR FOR LOST PROFITS, LOST OR CORRUPTED DATA, OR INTERRUPTION OF BUSINESS ARISING IN CONNECTION WITH THE USE OF THE PRODUCT OR SERVICES OR IN CONNECTION WITH ANY OTHER CLAIM ARISING FROM THIS AGREEMENT, EVEN IF THE PURE PARTIES HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. OTHER THAN PURE'S INDEMNIFICATION OBLIGATIONS UNDER SECTION 6, TO THE MAXIMUM EXTENT PERMITTED BY LAW, THE PURE PARTIES' AGGREGATE LIABILITY UNDER OR RELATING TO THIS AGREEMENT IS LIMITED TO DIRECT DAMAGES IN AN AMOUNT NOT TO EXCEED THE AMOUNT PAID BY END USER FOR PRODUCTS OR SERVICES THAT GAVE RISE TO SUCH CLAIM IN THE TWELVE (12) MONTHS PRECEDING THE CLAIM.

## 8. PRODUCT DIAGNOSTIC REPORTING.

End User acknowledges that the Products store certain diagnostic information about the routine operations of the Product, including performance, capacity usage, data reduction ratios, configuration data, and hardware faults ("**Pure1 Reports**") and, when enabled by End User, periodically transmit these Pure1 Reports to Pure and authorized End User partners. End User understands and agrees that End User data stored on the Products is not accessed, transmitted or provided to Pure or any third party as part of the Pure1 Reports. Pure retains all right, title, and interest in the Pure1 Reports. End User agrees that the collection and transmission of such Pure1 Reports is necessary to facilitate any Subscription Services and certain support services under an Evergreen Subscription.

## 9. CONFIDENTIAL INFORMATION.

**9.1. Data Privacy.** End User is solely responsible for data (including personal data) managed or stored using the Products and for compliance with all applicable data privacy laws related thereto. In the event End User provides Pure with personal data in connection with the performance of this Agreement, the parties will ensure that such personal data is disclosed and handled in accordance with applicable data protection laws.

**9.2. Confidentiality.** “Confidential Information” means any nonpublic information of a disclosing party (“Discloser”), whether disclosed orally or in written or digital media, received by the receiving party (“Recipient”), that is identified as “confidential” or with a similar legend at the time of such disclosure or that Recipient knows or should have known is the confidential or proprietary information of Discloser. Pure’s Confidential Information includes all non-public information relating to, or derived from, the Products and Services, including technical features, benchmark results, or performance results. Information does not constitute a party’s Confidential Information if it (a) is already known by Recipient without obligation of confidentiality; (b) is independently developed by Recipient without use of Discloser’s Confidential Information; (c) is publicly known without breach of this Agreement; or (d) is lawfully received from a third party without obligation of confidentiality. Recipient shall: (i) not use or disclose any Confidential Information except as expressly authorized by this Agreement or Discloser; (ii) protect Discloser’s Confidential Information using the same degree of care that it uses with respect to its own confidential information of a like nature, but in no event with safeguards less than a reasonably prudent business would exercise under similar circumstances; and (iii) limit access to Discloser’s Confidential Information to its employees, affiliates, agents, or authorized representatives having a need to know and who are bound by confidentiality obligations no less protective to those contained herein. Recipient shall take prompt and appropriate action to prevent unauthorized use or disclosure of Discloser’s Confidential Information. Recipient’s obligations under this Section 9.2 survive termination and continue for five (5) years from the date of termination of this Agreement. All tangible materials containing Confidential Information shall remain the property of Discloser. Upon termination, Recipient shall cease any use of Confidential Information. Upon Discloser’s written request, the receiving party shall promptly return (or at Discloser’s option, destroy) all documents and tangible materials containing any portion of, or summarizing, Discloser’s Confidential Information. At Discloser’s request, an authorized representative of Recipient shall provide a certificate attesting to compliance with this section. If any Confidential Information must be disclosed to any third party by reason of legal, accounting, or regulatory requirements, Recipient shall promptly notify Discloser of the order or request and permit Discloser (at its own expense) to seek an appropriate protective order.

## 10. GENERAL PROVISIONS.

**10.1. Governing Law and Venue.** This Agreement will be governed and interpreted by and under the laws of the State of California, without giving effect to any conflicts of laws principles. The parties expressly consent to the personal jurisdiction and venue in the state and federal courts in Santa Clara County, California for any lawsuit filed there arising from or related to this Agreement. The U.N. Convention on Contracts for the International Sale of Goods shall not apply to this Agreement.

**10.2. Termination.** The license in Section 1.1, and End User’s rights to use the Software, will terminate immediately in the event that: (i) End User returns the Product to Pure as provided herein; or (ii) End User materially breaches any provision of this Agreement and, if capable of cure, fails to cure such breach within 30 days from the date of Pure’s written notice to End User. Upon any such termination,

End User shall promptly discontinue all use of the Software. Sections 2, 3.2, 5.4, 6, 7, 9, and 10 will survive any termination of this Agreement.

**10.3. Notices.** Except as specifically stated, all notices or other communications required or permitted under this Agreement shall be in writing and shall be delivered by personal delivery, certified overnight delivery such as Federal Express, or registered mail (return receipt requested) and shall be deemed given upon personal delivery or upon confirmation of receipt.

**10.4. Compliance with Laws.** The parties agree to comply with all laws applicable to the distribution and use of the Products and performance of its obligations under this Agreement.

**10.5. Severability; Waiver.** If any provision of this Agreement is, for any reason, held to be invalid or unenforceable, the other provisions of this Agreement will remain enforceable and the invalid or unenforceable provision will be deemed modified so that it is valid and enforceable to the maximum extent permitted by law. Any waiver or failure to enforce any provision of this Agreement on one occasion will not be deemed a waiver of any other provision or of such provision on any other occasion.

**10.6. Export.** The Products and related technology are subject to U.S. export control laws and may be subject to export or import regulations in other countries. End User agrees not to export, re-export, or transfer, directly or indirectly, any technical data acquired from Pure, or any products incorporating such data, in violation of applicable export laws or regulations. For purposes of Pure's compliance with applicable export laws, End User agrees to provide Pure with applicable end use information upon Pure's request.

**10.7. No Assignment.** This Agreement, and End User's rights and obligations herein, may not be assigned by End User without Pure's prior written consent, which consent will not be unreasonably withheld, and any attempted assignment in violation of the foregoing will be null and void.

**10.8. U.S. Government End Users.** The Products and related documentation are "commercial off the shelf items" as defined in FAR 2.101 and their use is subject to the policies set forth in FAR 12.211, FAR 12.212 and FAR 227.7202, as applicable.

**10.9. Force Majeure.** Neither party shall be liable hereunder by reason of any failure or delay in the performance of its obligations under this Agreement on account of strikes, shortages, riots, insurrection, fires, flood, storm, explosions, acts of God, war, terrorism, governmental action, labor conditions, earthquakes, volcanic eruption, material shortages or any other cause that is beyond the reasonable control of the party.

**10.10. Privacy.** This Agreement is subject to Pure's Privacy Policy, which constitutes an integral part of this Agreement. End User is solely responsible for personal data managed or stored using the Products and for compliance with all applicable data privacy laws related thereto.

**10.11. Entire Agreement; Modification.** This Agreement, including any terms referenced herein, is the entire agreement between the End User and Pure with respect to the subject matter hereof. Any varying or additional terms relating to the subject matter hereof in any purchase order, discussion, or other written document will be of no effect. This Agreement, including any rights hereunder, may be extended or amended by the parties in writing.

## SUBSCRIPTION SERVICES ADDENDUM

The most recent version of the Subscription Services Addendum can be obtained at <http://www.purestorage.com/legal/productenduserinfo.html>.

This Subscription Services Addendum (“**Addendum**”) is applicable to the Pure-branded subscription-based hardware and/or software offerings and related services (collectively, “**Subscription Services**”), as purchased by the end user (“**End User**”) from Pure Storage, Inc. or its wholly owned affiliates (“**Pure**”) or a Pure authorized reseller (a “**Partner**”).

## 1. SUBSCRIPTION LIMITS.

Subscription Services are provided and licensed based on Subscription Limits. “**Subscription Limits**” means the permissions and/or limitations identified in the applicable SKU, schedule or quote issued by a Partner or Pure and accepted by End User (“**Schedule**”). Such Subscription Limits may include, but are not limited to: (i) the specified period of time during which End User is entitled to access and use the Subscription Services (“**Subscription Term**”); (ii) identified sizing and capacity limitations (and identified units of measurement: GiB, TiB, etc.); (iii) identified number of users; (iv) identified host, VM, instance, site, cluster, or other sub-org designation limits; (v) physical location; and (vi) inter-connectivity rights with other Pure Products or third party offerings, all as applicable to the specific Subscription Services purchased by End User.

## 2. ACCESS AND USE.

Subject to the terms of this Addendum and the applicable Schedule, Pure grants to End User, and any third party that End User authorizes to perform services involving the Subscription Services solely for End User’s benefit, a nontransferable, nonexclusive, revocable right to: (i) access and use the Subscription Services, and (ii) download, install, and use, in executable object code format only, any Pure plug-in or software necessary to use such Subscription Services (“**Subscription Software**”), solely in accordance with the applicable Subscription Limits, and solely during the Subscription Term. End User will not, nor will End User allow any third party to, use the Subscription Services or Subscription Software in excess of the Subscription Limits. For any Subscription Services that require access to third party cloud services, End User is responsible for all actions associated with such third party cloud service accounts, including, but not limited to, obtaining all necessary use and access rights, all costs, and any setup, configuration, or decommissioning activities.

## 3. PURE1 DATA REQUIRED.

Pure’s “phone home” feature, known as Pure1, is an integral component of Pure’s delivery of the Subscription Services. End User acknowledges and agrees that the rates charged for Subscription Services, and End User’s compliance with the Subscription Limits, are contingent upon Pure’s receipt of Pure1 Data. Therefore, End User shall ensure that Pure1 is enabled at all times. Notwithstanding any other rights or remedies available to Pure, Pure reserves the right to suspend or terminate the Subscription Services immediately if Pure1 is not enabled at any time during the Subscription Term, provided that Pure shall notify End User prior to any such suspension or termination.

## 4. RESERVE CAPACITY.

Depending on the Subscription Services purchased by End User, Subscription Limits may include a Reserve Capacity. Reserve Capacity may be increased, if requested by End User, with no change to the Subscription Term. However, End User acknowledges and agrees that Reserve Capacity may not be reduced at any time during the Subscription Term, as Pure’s ability to provide the flexibility associated with its Subscription Services is based on Pure’s reliance on End User’s commitment to a minimum Reserve Capacity during the Subscription Term.

## 5. SUBSCRIPTION HARDWARE.

The following terms apply to any Subscription Services requiring or otherwise utilizing Pure hardware (“**Subscription Hardware**”):

**5.1 Sizing and Installation.** Pure sizes the Subscription Hardware, subject to Pure’s reasonable discretion, based on End User’s identified workload requirements, as reported by End User to a Partner or Pure (the “**Workload**”). Except as otherwise agreed by the parties, the Workload excludes any encrypted or pre-compressed data where the data reduction ratio is less than 2:1, as determined by Pure1 (“**Encrypted Workloads**”). Pure will ship the Subscription Hardware following confirmation of the datacenter’s readiness, as determined by completion of Pure’s pre-site survey. The Subscription Hardware must not be moved without Pure’s written consent, which shall not be unreasonably withheld. Notwithstanding anything to the contrary in this Addendum, End User is responsible for ensuring Pure’s access to the datacenter for all Subscription Hardware installation and maintenance services.

**5.2 Deployed Capacity.** Pure may expand, modify, substitute, replace, or remove any component of Subscription Hardware deployed to provide the Subscription Services to meet the Workload (“**Deployed Capacity**”), based on a data reduction ratio Pure sees with similar end users across its install base. Pure will use reasonable efforts to ship additional Deployed Capacity within 14 business days of Pure determining that additional Deployed Capacity is required to meet the Workload. Notwithstanding the foregoing, (a) shipment of additional Deployed Capacity is subject to the payment of all current and outstanding invoices for Subscription Services; and (b) Pure may require End User to increase the Reserve Capacity if the Deployed Capacity exceeds 2.5x the then-current Reserve Capacity. In addition to Pure’s rights under Section 3 of this Addendum, for any and all periods that Pure1 is not enabled in excess of four days during any 30-day period, Partner or Pure may invoice End User for every TiB of Deployed Capacity for each day in excess of that four-day period that Pure1 is not enabled.

**5.3 Return of Subscription Hardware.** Upon termination or expiration of the Subscription Term, End User shall (a) promptly contact Pure regarding the return of Subscription Hardware to obtain an RMA number, packaging instructions, and shipping address; (b) promptly return the Subscription Hardware to Pure in accordance with Pure’s reasonable shipping instructions; (c) reimburse Pure for reasonable repair or reasonable replacement costs associated with any damage to the Subscription Hardware (other than normal wear and tear) while in End User’s possession; and (d) ensure that all information stored on the Subscription Hardware is removed in its entirety. Pure is not responsible for or liable to End User, or any third party, for any information remaining on the Subscription Hardware returned to Pure and Pure has the right to delete and destroy any such information or data.

## 6. SUBSCRIPTION TERM.

The start date of any Subscription Term shall be the date that Pure makes the Subscription Services available to End User. Notwithstanding the foregoing, for any Subscription Services requiring Subscription Hardware, the start date of any Subscription Term shall be: (i) 4 weeks from the Schedule acceptance date for any net-new Subscription Services, or (ii) the Schedule acceptance date, for any renewal or modification of existing Subscription Services. Upon expiration of the initial Subscription Term, the Subscription Services will automatically renew, for subsequent 12-month terms, unless End User provides Partner written notice of its intent not to renew at least 60 days prior to the expiration of the then-current Subscription Term.

## 7. AUDIT.

Pure and its independent accountants shall have the right, upon reasonable notice to End User, to examine End User's use of the Subscription Services to verify compliance with the Subscription Limits and this Addendum. If the audit identifies usage in excess of the Subscription Limits, then End User will promptly pay to Pure or Partner, as determined by Pure, any additional fees that Pure (or a Partner) is owed hereunder, and the reasonable costs of conducting the audit.

## 8. TERMINATION.

This Addendum, and End User's rights to use the Subscription Services, will terminate immediately in the event that: (i) any applicable Subscription Term expires; (ii) End User fails to make timely payments for the Subscription Services to Pure or its authorized resale partner; or (iii) End User materially breaches any provision of this Agreement and, if capable of cure, fails to cure such breach within 30 days from the date of Pure's written notice to End User. Upon any such termination, End User shall promptly: (a) discontinue all use of the Subscription Services, (b) return any applicable Subscription Hardware to Pure, and (c) pay all amounts due for Subscription Services. Pure reserves the right to enter End User's premises, or the premises where the Subscription Hardware is located, to access and retrieve Subscription Hardware. Subscription Services may not be cancelled or terminated by End User during the Subscription Term.

## 9. GENERAL TERMS.

This Addendum supplements the Pure End User Agreement (or other written agreement covering the same subject matter executed by Pure) for the applicable Subscription Services purchased by End User. Capitalized terms not specifically defined in this Addendum will have the same meaning as in the End User Agreement. Pure reserves the right to update the Addendum from time to time, as noted by the "Last Updated" date below.

## 10. SERVICES DESCRIPTIONS AND DEFINITIONS.

The following descriptions and definitions apply to this Addendum and any Schedules for Subscription Services:

**10.1 Evergreen Storage Service ("ES2").** ES2 is a subscription-based, data storage-as-a-service involving the use of Subscription Hardware, measured and billed on a consumption basis through connectivity to Pure1. Unless otherwise agreed to in the applicable Schedule: (a) Reserve Capacity is billed annually in advance for a minimum committed usage at the Reserve Rate; (b) On-Demand Capacity is billed calendar-quarterly in arrears at the On-Demand Rate; and (c) Reserve Capacity and On-Demand Capacity exclude Encrypted Workloads, which are billed calendar-quarterly in arrears at 2.5x the Reserve Rate.

### 10.2 Definitions for ES2:

- **Effective Used Capacity:** The logical capacity of data written by a host to a storage volume, along with any snapshots and copies thereof.
- **On-Demand Capacity (or Burst Capacity):** The total amount of Effective Used Capacity used by End User above the Reserve Capacity.
- **On-Demand Rate (or Burst Rate):** The rate applicable to usage of On-Demand Capacity.

- **Reserve Capacity:** A commitment of Effective Used Capacity for an agreed-upon Subscription Term.
- **Reserve Rate:** The rate applicable to usage of Reserve Capacity.

**10.3 ObjectEngine CloudDirect.** ObjectEngine CloudDirect is a subscription-based service that allows End User to store data, after deduplication and compression, up to the capacity limits identified in the Schedule. In addition to Pure's other rights and remedies hereunder, Pure may bill End User or Partner for any usage of ObjectEngine CloudDirect above the capacity limits identified in the Schedule. If ObjectEngine CloudDirect Subscription Services are purchased at the same time as ObjectEngine hardware, then the ObjectEngine CloudDirect Subscription Services are made available to End User on the hardware shipment date.

## PROFESSIONAL SERVICES ADDENDUM

The most recent version of the Professional Services Addendum can be obtained at <http://www.purestorage.com/legal/productenduserinfo.html>.

This Professional Services Addendum (“**Addendum**”) is applicable to the Pure-branded installation, configuration, data migration, or other professional services (“**Services**”) performed by or on behalf of Pure and the hardware/software modifications, work product and other deliverables developed by or on behalf of Pure in the course of performing the Services (“**Improvements**”), as purchased by the end user (“**End User**”) from Pure Storage, Inc. or its wholly owned affiliates (“**Pure**”) or a Pure authorized reseller. A description of Pure’s Services can be found in the SKU description and Pure’s Professional Services Guide (“**Services Guide**”). If necessary, Pure and End User may enter into Statements of Work (“**SOW(s)**”) that reference this Addendum to further describe the Services and Improvements, and any such SOW shall be incorporated herein by reference.

### 1. OWNERSHIP & LICENSE.

Each party owns all of its intellectual property rights and other proprietary rights existing, owned, or otherwise licensed by such party prior to the performance of Services (“Pre-Existing IP”). Pure owns all intellectual property rights and other proprietary rights in and to the Improvements. In no event will Improvements be deemed to include End User Pre-Existing IP or End User Confidential Information. Subject to End User’s payment of the fees for such Services and compliance with all terms and conditions of this Addendum, the Services Guide, and any applicable SOW, Pure hereby grants to End User, and any third party that End User authorizes to perform services involving Pure Products or Services solely for End User’s benefit, a fully paid-up, royalty-free, non-transferable, non-sublicensable license to install, operate, and use the Improvements, solely for End User’s internal business operations and in accordance with the Services Guide or any applicable SOW. End User shall not (i) re-distribute the Improvements to any third party except as expressly permitted by Pure; (ii) copy the Improvements, except as reasonably necessary for End User’s use of the Improvements as permitted herein; (iii) modify or create derivative works of the Improvements; (iv) reverse engineer, disassemble, decompile, decode or otherwise attempt to derive or gain access to the source code of the Improvements or any part thereof; (v) remove, delete, alter, or obscure any trademarks or any copyright, trademark, patent, or other intellectual property or proprietary rights notices from any Improvements, including any copy thereof; (vi) use any Improvements in a manner or for any purpose that infringes, misappropriates, or otherwise violates any law or intellectual property right; or (vii) otherwise use the Improvements beyond the scope of the license granted under this Addendum, the Services Guide, or any applicable SOW.

## **2. REPRESENTATIONS & WARRANTIES.**

Pure represents and warrants that (i) it shall perform the Services using personnel of required skill, experience and qualifications and in a professional and workmanlike manner in accordance with commercially reasonable industry standards for similar services; (ii) it is in compliance with, and shall perform the Services in compliance with, all applicable laws; (iii) the Services and Improvements will be in conformity in all material respects with the requirements or specifications stated in this Addendum, the Services Guide, and any applicable SOW for a period of 30 days after delivery to End User. If any element of the Services or Improvements does not conform to the foregoing warranty, End User shall notify Pure in writing within 30 days after delivery of such Service or Improvement, and Pure shall use commercially reasonable efforts to cure such nonconformance. This clause states Pure's entire liability, and End User's sole and exclusive remedy, for warranty claims for Services. EXCEPT FOR THE EXPRESS WARRANTIES DESCRIBED HEREIN, PURE HEREBY DISCLAIMS ALL WARRANTIES, EITHER EXPRESS, IMPLIED, STATUTORY, OR OTHERWISE, INCLUDING WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE AND NON-INFRINGEMENT, UNDER THIS ADDENDUM. END USER IS RESPONSIBLE FOR ENSURING THAT IT HAS APPROPRIATE DATA BACK-UP, DATA RECOVERY, AND DISASTER RECOVERY MEASURES IN PLACE.

## **3. ASSUMPTIONS.**

End User shall (i) provide such access to End User's premises and operating environment as may reasonably be requested by Pure for the purposes of performing the Services or developing the Improvements; (ii) respond promptly to any Pure request to provide direction, information, approvals, authorizations, or decisions that are reasonably necessary for Pure to perform the Services or develop the Improvements, and ensure that each of the foregoing is complete and accurate in all material respects; (iii) ensure that all End User equipment (other than Pure Products) is in good working order and suitable for the purposes for which it is used in relation to the Services and conforms to all relevant legal or industry standards or requirements; (iv) secure all necessary rights, licenses, consents, approvals and authorizations necessary for Pure to use any third party materials or intellectual property that are in End User's environment and necessary for Pure to perform the Services and develop the Improvements; and (v) perform any other end user obligations identified in the Services Guide or applicable SOW. If Pure's performance of its obligations under this Addendum is prevented or delayed by End User's breach of this section or any act or omission of End User, Pure shall not be liable for any costs, charges or losses sustained or incurred by End User, in each case, to the extent arising directly or indirectly from such prevention or delay.

## **4. SUBCONTRACTORS.**

Pure may subcontract the performance or development of any of the Services or Improvements. Pure shall be responsible and liable for: (i) the acts and omissions of such subcontractor to the same extent as if such acts or omissions were by Pure or its employees, and (ii) all fees and expenses payable to any subcontractor.

## **5. FEES AND EXPENSES.**

The cost for all Services and Improvements shall be determined between End User and Pure or the Pure-authorized reseller, as applicable, and shall be based on the scope and assumptions included in the SKU description, the Services Guide, or the applicable SOW. If specified in the SKU description,

Services Guide, or SOW, End User will pay actual out-of-pocket expenses, including travel, as reasonably incurred during the performance of the Services. Except as identified in an applicable SOW, all Services are non-cancellable.

## **6. ON-SITE SERVICES.**

While performing on-site Services at End User's facilities, Pure personnel will follow End User policies and procedures made available to such personnel in writing.

## **7. GENERAL TERMS.**

This Addendum supplements the Pure End User Agreement (or other written agreement covering the same subject matter executed by Pure) for the applicable Products and Services purchased by End User. Capitalized terms not specifically defined in this Addendum will have the same meaning as in the End User Agreement. Pure reserves the right to update the Addendum from time to time, as noted by the "Last Updated" date below

## purelog

purelog, purelog-create, purelog-delete, purelog-global, purelog-list, purelog-setattr, purelog-test —  
manages connections to syslog servers

### Synopsis

```
purelog create --uri URI NAME
purelog delete NAME...
purelog global disable --tls-audit
purelog global enable --tls-audit
purelog global list [--ca-certificate] [ --cli | --csv | --nvp ] [--notitle] [--page] [--raw]
purelog global setattr --ca-certificate
purelog list [ --cli | --csv | --nvp ] [ --notitle | --raw ] [NAME...]
purelog setattr --uri URI NAME
purelog test
```

### Arguments

#### **NAME**

Name used by Purity//FA to identify a syslog server.

### Options

#### **-h | --help**

Can be used with any command or subcommand to display a brief syntax description.

#### **--ca-certificate**

Import or display the Certificate Authority's (CA) certificate.

Purity//FA's syslog agent uses the CA's certificate in communicating with syslog servers using the TLS protocol.

#### **--tls-audit**

Forward messages needed for TLS auditing.

#### **--uri **URI****

Creates or sets the URI of the remote syslog server. For example, **tcp://MyHost.com**.

Specify the URI in the format **PROTOCOL://HOSTNAME:PORT**.

**PROTOCOL** is "tcp", "tls", or "udp".

**HOSTNAME** is the syslog server hostname or IP address. If specifying an IP address, for IPv4, specify the IP address in the form **ddd.ddd.ddd.ddd**, where **ddd** is a number ranging from 0 to 255 representing a group of 8 bits.

For IPv6, specify the IP address in the form

[xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx], where xxxx is a hexadecimal number representing a group of 16 bits. Enclose the entire address in square brackets ([]). Consecutive fields of zeros can be shortened by replacing the zeros with a double colon (::).

**PORT** is the port at which the server is listening. If a port number is specified, append it to the end of the address. If the port is not specified, it defaults to 514.

Options that control display format:

--cli

Displays output in the form of CLI commands that can be issued to reproduce the current configuration. The --cli output is not meaningful when combined with immutable attributes.

--csv

Lists information in comma-separated value (CSV) format. The --csv output can be used for scripting purposes and imported into spreadsheet programs.

--notitle

Lists information without column titles.

--nvp

Lists information in name-value pair (NVP) format, in the form **ITEMNAME=VALUE**. Argument names and information items are displayed flush left. The --nvp output is designed both for convenient viewing of what might otherwise be wide listings, and for parsing individual items for scripting purposes.

--page

Turns on interactive paging.

--raw

Displays the unformatted version of column titles and data. For example, in the **purearray monitor** output, the unformatted version of column title **us/op (read)** is **usec\_per\_read\_op**. The --raw output is used to sort and filter list results.

## Description

Purity//FA's syslog agent can forward syslog messages to syslog servers.

The **purelog** command can be used for configuring Purity//FA's syslog agent to communicate with syslog servers.

## Using Secure Transport

Syslog messages can be forwarded using the TLS protocol.

SSL certificate installed using the **purecert** command is used by Purity//FA's syslog agent to authenticate itself to syslog servers.

**purelog global setattr --ca-certificate** can be used to import the CA's certificate that Purity//FA's syslog agent will use for authenticating the syslog servers.

## purelog Subcommands

Subcommands of the **purelog** command configures, displays and sets attributes of syslog servers.

The **purelog create** subcommand configures a syslog server. Transmission of syslog messages is enabled immediately upon configuring the syslog server.

The **purelog delete** subcommand deletes the configured syslog server and stops forwarding syslog messages to the syslog server.

The **purelog global enable** subcommand enables global options.

The **purelog global disable** subcommand disables global options.

The **purelog global list** subcommand displays global configuration.

The **purelog global setattr** subcommand sets global attributes.

The **purelog list** subcommand displays the configured syslog servers.

The **purelog setattr** subcommand sets syslog server attributes.

The **purelog test** subcommand sends a test message to syslog servers. Verification of successful test message transmission is done at the destination.

## Examples

### Example 1

```
purelog create --uri tcp://MyHost.com LOGSERVER1
```

Configures a syslog server named **LOGSERVER1** running on host **MyHost.com**, using **TCP** and at the default port **514**.

This immediately enables transmission of all future syslog messages to **LOGSERVER1**.

### Example 2

```
purelog create --uri tcp://[2001:0db8:85a3::ae26:8a2e:0370:7334]:614 LOGSERVER2
```

Configures a syslog server named **LOGSERVER2** running on host **2001:0db8:85a3::ae26:8a2e:0370:7334**, using **TCP** and at port **614**.

This immediately enables transmissions of all future syslog messages to **LOGSERVER2**.

### Example 3

```
purelog delete LOGSERVER2
```

Stops transmission of future syslog messages to **LOGSERVER2** and deletes the configured syslog server.

### Example 4

```
purelog list LOGSERVER3
```

Displays the URI for LOGSERVER3.

#### Example 5

```
purelog global disable --tls-audit
```

Disables TLS audit for all syslog servers.

#### Example 6

```
purelog global enable --tls-audit
```

Enables TLS audit for all syslog servers.

Syslog messages needed for TLS auditing are forwarded to all configured syslog servers.

#### Example 7

```
purelog global list
```

Displays global options.

#### Example 8

```
purelog global list --ca-certificate
```

Displays the imported CA certificate.

#### Example 9

```
purelog global setattr --ca-certificate
```

Import CA certificate.

#### Example 10

```
purelog setattr --uri tcp://MyHost.com LOGSERVER1
```

Set URI for LOGSERVER1 to `tcp://MyHost.com`.

### See Also

`purearray(1)` [230]

### Author

Pure Storage Inc. <[documentfeedback@purestorage.com](mailto:documentfeedback@purestorage.com)>

## pureman

pureman — displays man pages for Purity//FA commands

### Synopsis

```
pureman { pureadmin | purealert | purearray | pureaudit | purecert | pureconfig  
| puredns | puredrive | pureds | purehgroup | purehost | purehw | purelicense |  
purelog | pureman | puremessage | purenetwork | purepgroup | pureplugin | pureport |  
puresmis | puresnmp | purevol }
```

### Description

The **pureman** command displays extensive help for each Purity//FA CLI command. To display the man page for a Purity//FA CLI command, run the **pureman** command with the Purity//FA CLI command name. For example, run the following command to display the man page for the **purevol** command:

```
pureman purevol
```

Help on Purity//FA subcommands is also supported by running the subcommand preceded with a hyphen. For example, run the following command to display the man page for the **purehost connect** subcommand:

```
pureman purevol-list
```

### CLI Conventions

The following list describes the conventions used in CLI help documentation:

- Text in fixed-width (Courier) font must be entered exactly as shown. For example, **purehost list**.
- Text not enclosed in brackets represents mandatory text.
- Text inside square brackets ("[ ]") represents optional items. Do not type the brackets.
- Text inside curly braces ("{ }") represents text, where one (and only one) item must be specified. Do not type the braces.
- The vertical bar (|) separates mutually exclusive items.
- Uppercase italic text represents entered text whose value is based on the nature of the subcommand or option. For example, **--size SIZE**, where SIZE represents the value to be entered, such as **100g**.

### CLI Command Syntax

Purity//FA CLI commands have the general form:

```
command subcommand --options OBJECT-LIST
```

The parts of a command are:

**COMMAND**

Type of FlashArray object to be acted upon, prefixed by "pure". For example, the **purevol** command acts on Purity//FA virtual storage volumes.

Run the **pureman** command to see a list of Purity//FA CLI commands.

**SUBCOMMAND**

Action to be performed on the specified object. Most CLI subcommands are common to some or all object types.

For example, **purehost list** lists all hosts on the array, while **purevol list** lists all volumes on the array.

The following subcommands are common to most or all object types:

**create**

Creates and names one or more FlashArray objects.

**delete**

Deletes one or more specified objects.

**list**

Lists information about one or more objects. To list information for all objects, do not specify the object in the command.

**listobj**

Creates whitespace-separated lists of objects or attributes related to one or more objects.

For example, **purevol listobj --type host** creates a list of the hosts to which volumes have connections. The **listobj** subcommand is primarily used to create lists of object and attribute names for scripting purposes.

**rename**

Changes the name of the specified object. Purity//FA identifies the object name in administrative operations and displays. The new name is effective immediately and the old name is no longer recognized in Purity//FA GUI and CLI interactions. In the Purity//FA GUI, the new name appears upon page refresh. Hardware object names cannot be changed.

**setattr**

Changes the attribute values of the specified objects.

**OPTIONS**

Options that specify attribute values or modify the action performed by the subcommand.

For example, in the following command, the **--addvollist** option adds volumes VOL1, VOL2, and VOL3 to protection group PGROUP1:

```
purepgroup setattr --addvollist VOL1,VOL2,VOL3 PGROUP1
```

Some options, such as **--hostlist**, inherently apply only to a single object. Other options, such as **--size**, can be set for multiple objects in a single command.

Some options can be multi-valued. For example, in the following command, the **--hostlist** option associates multiple hosts (HOST1, HOST2, and HOST3) with the host group HGROUP1.

```
purehgroup setattr --hostlist HOST1,HOST2,HOST3 HGROUP1
```

#### OBJECT-LIST

Object or list of objects upon which the command is to be operated.

If a subcommand changes the object state, then at least one object must be specified. Examples of subcommands that change the object state include **create**, **delete**, and **setattr**. For example, **purehost create HOST1** creates host **HOST1**. In the command synopses, OBJECT specifications that are not enclosed in square brackets (for example, "**HOST**") represent ones that are required.

Passive subcommands, such as **list**, which do not change object state, do not require object specification. Leaving out the object is equivalent to specifying all objects of the type. For example, **purevol list** with no volumes specified displays information about all volumes in an array. In the command synopses, OBJECT specifications enclosed in square brackets (for example, "[**HOST**]") represent ones that are optional.

Most subcommands act on a single object. For example, in the following command, the **setattr** subcommand can only be run on a single host group to change the attributes of that host group.

```
purehgroup setattr --addhostlist HOST1 HGROUP1
```

Certain subcommands can operate on multiple objects. For example, the following command connects volume **VOL1** to host groups **HGROUP1** and **HGROUP2**.

```
purehgroup connect --vol VOL1 HGROUP1 HGROUP2
```

In the command synopses, OBJECT specifications that are followed by ellipses (for example, **HGROUP...**) indicate that multiple objects can be entered.

## CLI Output

Various Purity//FA CLI subcommands, including **list** and **monitor**, generate list outputs. With the ability to create and manage hundreds of volumes and snapshots, list outputs can become very long.

The Purity//FA CLI includes options to control the way a list output is displayed and formatted. The CLI also includes sort, filter, and limit options to control the results you see and the order in which you see them.

### Interactive Paging

Pagination divides a large output into discrete pages. Pagination is disabled by default and is only in effect if the **--page** option is specified and the number of lines in the list output exceeds the size of the window.

To interactively move through a paginated list output:

- Press the **[Right Arrow]** key or space bar to move to the next page.
- Press the **[Left Arrow]** key to move to the previous page.

To quit interactive paging and exit the list view, press **q**.

With pagination, each page of the CLI list output begins with the column titles. To suppress the column titles, run the command with the `--notitle` option.

## Using Wildcards

The asterisk (\*) symbol denotes a wildcard character used in list or monitor operations to represent zero or more characters in an object name. The object name can include multiple asterisks.

When performing a list or monitor operation, include the asterisk in the name argument to expand the list results. For example, the `purehost list *cat*` command displays a list of all hosts that contain "cat", such as `cat`, `catnap`, `happycats`, and `lolcat`. If the asterisks were not included, only the host named `cat` would be returned. As another example, the `purevol list *vol*` command displays a list of all volumes that contain "vol", including ones that begin or end with "vol".

```
$ purevol list *vol*
Name      Size  Source  Created
Myvol    100G   -        2016-04-11 10:18:07 PDT
MyVol-01 100G   -        2016-04-11 10:18:07 PDT
vol       1G     -        2016-04-11 11:19:19 PDT
vol01    100G   -        2016-04-11 10:17:23 PDT
Volume   100G   -        2016-04-11 10:17:23 PDT
```

If Purity//FA cannot find matches for the wildcard argument specified, an error message appears.

Asterisks are also allowed in Purity//FA CLI options that take a list of object names. For example, the `purevol list --snap --pgrouplist *pg*` command displays a list of all volume snapshots that are created as part of a protection group snapshot with "pg" in the protection group name. As another example, the following command displays a list of volume snapshots for all volumes that end with "01" and are created as part of a protection group snapshot with "pg" in the protection group name.

```
$ purevol list --snap --pgrouplist *pg* *01
Name              Size  Source  Created
pgroup01.001.vol01 100G  vol01  2016-04-13 11:44:46 PDT
pgroup01.123.vol01 100G  vol01  2016-04-13 11:44:46 PDT
pgroup01.abc.vol01 100G  vol01  2016-04-13 11:44:46 PDT
pgroup01.backup.vol01 100G  vol01  2016-04-13 11:44:45 PDT
pgroup01.snap001.vol01 100G  vol01  2016-04-13 11:44:45 PDT
pgroup01.snap002.vol01 100G  vol01  2016-04-13 11:44:42 PDT
pgroup01.suffix.vol01 100G  vol01  2016-04-13 11:44:46 PDT
```

## Formatting

Format options control the way list outputs are displayed. The following format options are common to most `list` and `monitor` subcommands:

### `--cli`

Displays output in the form of CLI commands that can be issued to reproduce the current configuration. The `--cli` output is not meaningful when combined with immutable attributes.

**--csv**

Lists information in comma-separated value (CSV) format. The **--csv** output can be used for scripting purposes and imported into spreadsheet programs.

**--notitle**

Lists information without column titles.

**--nvp**

Lists information in name-value pair (NVP) format, in the form **ITEMNAME=VALUE**. Argument names and information items are displayed flush left. The **--nvp** output is designed both for convenient viewing of what might otherwise be wide listings, and for parsing individual items for scripting purposes.

**--page**

Turns on interactive paging.

**--raw**

Displays the unformatted version of column titles and data. For example, in the **purearray monitor** output, the unformatted version of column title **us/op (read)** is **usec\_per\_read\_op**. The **--raw** output is used to sort and filter list results.

Here is a comparison between the **purehost list** output with formatted column titles and data, and the same output with unformatted column titles and data:

```
$ purehost list --space
Name  Size  Thin Provisioning  Data Reduction  ...
H001  60T   57%                1.5 to 1       ...

$ purehost list --space --raw
name  size          thin_provisioning  data_reduction  ...
H001  65970697666560  0.5712341      1.5123        ...
```

## Sorting

When running a list or monitor operation, include the **--sort** option to sort a column of the output in ascending or descending order. The following commands support the **--sort** option: purehgroup, purehost, purepgroup, pureport, and purevol.

The sort option adheres to the following syntax:

**--sort SORT**

SORT represents a comma-separated list of columns to sort. Use the unformatted title name (**--raw**) of the column to represent each column listed. To sort a column in descending order, append the minus (-) sign to the column name.

For example, run the following commands to get the unformatted column titles in the **purehost list** output, and then sort the same output by host group in descending order:

```
$ purehost list --raw
name          ...  hgroup
ESXi-GRP-Cluster02-H0001  ...  ESXi-GRP-Cluster02-HG003
```

```
ESXi-GRP-Cluster02-H0002    ...    ESXi-GRP-Cluster02-HG003
ESXi-IT-Cluster01-H0001     ...    ESXi-IT-Cluster01-HG001
ESXi-IT-Cluster02-H0001     ...    ESXi-IT-Cluster02-HG002
ESXi-STG-Cluster03-H0001    ...    ESXi-STG-Cluster03-HG005
```

```
$ purehost list --sort hgroup-
Name                      ...  Host Group
ESXi-STG-Cluster03-H0001   ...  ESXi-STG-Cluster03-HG005
ESXi-IT-Cluster02-H0001    ...  ESXi-IT-Cluster02-HG002
ESXi-IT-Cluster01-H0001    ...  ESXi-IT-Cluster01-HG001
ESXi-GRP-Cluster02-H0002   ...  ESXi-GRP-Cluster02-HG003
ESXi-GRP-Cluster02-H0001   ...  ESXi-GRP-Cluster02-HG003
```

If multiple columns are specified, the output is sorted by the order of the columns listed.

If a sorted column contains non-unique values and a secondary sort argument is not specified, Purity//FA automatically performs a secondary sort using the default sort criteria (typically by **Name**).

## Examples

Example 1: Display a list of volumes sorted in descending order by physical space occupied.

```
$ purevol list --space --sort "total-"
```

Example 2: Display a list of protection groups sorted in ascending order by source array name.

```
$ purepgroup list --sort "source"
```

Example 3: Display a list of volumes that are sorted in descending order by volume size. If any of volumes are identical in size, sort those volumes in descending order by physical space occupied.

```
$ purevol list --space --sort size-,total-
```

## Filtering

When running a list or monitor operation, include the **--filter** option to narrow the results of the output to only the rows that meet the filter criteria. The following commands support the **--filter** option: purehgroup, purehost, purepgroup, pureport, and purevol. Filtering can be performed on any column of the list output.

### Filtering with Operators

When filtering with operators, use the following syntax:

```
--filter "RAW_TITLE OPERATOR VALUE"
```

**RAW\_TITLE** represents the unformatted title name of the column by which to filter. Run the list or monitor operation with the **--raw** option to get the unformatted title name.

**OPERATOR** represents the type of filter match (=, !=, <, >, <=, or >=) used to compare **RAW\_TITLE** to **VALUE**.

**VALUE** represents the value (number, date, or string) that determines the results to be included in the list output. Literal strings must be wrapped in quotes.

Filtering supports the following operators:

Operator	Description
=	Equals
!=	Does not equal
<	Less than
>	Greater than
<=	Less than or equal to
>=	Greater than or equal to

Filtering supports the asterisk wildcard character. For example, the value `"*esx*`" in the `purepgroup list --filter "hosts=*esx*"` command displays a list of all protection groups with hosts members that contain "esx" in the host name.

The AND and OR operators can be used to further refine your filter. The AND operator displays only the results that meet all of the filter criteria in the command. The OR operator displays the results that meet at least one of the filter criteria in the command.

## Examples

Example 1: Display a list of protection groups configured on source array `pure-001`.

```
$ purepgroup list --filter "source = 'pure-001'"
```

Example 2: Display a list of volumes that were created on or before `2016-05-23 13:09:39`.

```
$ purevol list --filter "created <= '2016-05-23 13:09:39 PDT'"
```

Example 3: Display a list of volume snapshots that are greater than 2 gigabytes in size.

```
$ purevol list --space --snap --filter "snapshots > '2G'"
```

Example 4: Display a list of volumes that are currently inactive.

```
$ purevol monitor --filter "output_per_sec=0 and input_per_sec=0"
```

Example 5: Display a list of volumes that begin with "vol10" or are greater than 100 gigabytes in size.

```
$ purevol list --filter "name = 'vol10*' or size > '100G'"
```

## Filtering with Functions

The filter option supports the CONTAINS and NOT functions.

**--filter FUNCTION(PARAMETERS)**

Function	Description
<code>contains(raw_title, string)</code>	Contains the enclosed string. Takes exactly two parameters, where <code>raw_title</code> represents the unformatted title name of the column by which to filter and <code>string</code> represents the string to search within the column. Run the list or monitor operation with the <code>--raw</code> option to get the unformatted title name.
<code>not(expression)</code>	Inverse of the enclosed expression.

## Examples

Example 1: Get a list of all volumes that include "cluster03" in the volume name.

```
$ purevol list --filter "contains(name, 'cluster03')"
```

Example 2: Get a list of all hosts that are associated with host groups with "PROD" in the host group name.

```
$ purehost list --filter "not(contains(hgroup, 'PROD'))"
```

Example 3: Get a list of all hosts that are not associated with host groups with "PROD" in the host group name.

```
$ purehost list --filter "not(contains(hgroup, 'GRP'))"
```

## Existence Checks

The Purity//FA CLI supports existence checks. For example, the `purevol list --filter "name"` command checks to see if the "name" column exists in the volume list output.

## Examples

Example 1: Get a list of all hosts associated with a host group.

```
$ purehost list --filter "hgroup"
```

Example 2: Get a list of all volumes that were not created from another source.

```
$ purevol list --filter "not(source)"
```

Example 3: Get a list of all hosts that are not associated with Fibre Channel WWNs.

```
$ purehost list --filter "not(wwn)"
```

## Setting Limits

When running a list or monitor operation, include the `--limit` option to restrict the result size to the limit specified. The following commands support the `--limit` option: purehgroup, purehost, purepgroup, purepod, pureport, and purevol.

The limit option adheres to the following syntax:

```
--limit ROWS
```

ROWS represents the maximum number of rows to return.

For example, run the following command to list only the first 5 rows of the `purevol list` output:

```
$ purevol list --limit 5
```

Name	Size	Source	Created	Serial
ESXi-Cluster01-vol001	100G	-	2016-05-23 13:09:40 PDT	B143778B8ACE487B00011021
ESXi-Cluster01-vol002	100G	-	2016-05-23 13:09:40 PDT	B143778B8ACE487B0001101D
ESXi-Cluster01-vol003	100G	-	2016-05-23 13:09:40 PDT	B143778B8ACE487B00011024
ESXi-Cluster01-vol004	100G	-	2016-05-23 13:09:40 PDT	B143778B8ACE487B00011018
ESXi-Cluster01-vol005	100G	-	2016-05-23 13:09:40 PDT	B143778B8ACE487B00011025

## Combining Sorting, Filtering, and Limit Options

The `--sort`, `--filter`, and `--limit` options can all be combined together.

## Examples

Example 1: Display the top 10 volumes that occupy the largest amount of physical space and include "esx" in the volume name.

```
$ purevol list --space --sort "total-" --limit 10 --filter "name = '*esx*'"
```

Example 2: Display the top 10 volumes that have the largest physical space occupied by data unique to one or more snapshots.

```
purevol list --space --sort "snapshots-" --limit 10
```

Example 3: Display the top 10 volumes with the highest data reduction ratios.

```
$ purevol list --space --sort "data_reduction-" --limit 10
```

Example 4: Display the top 10 volumes that use the most read bandwidth.

```
$ purevol monitor --sort "output_per_sec-" --limit 10
```

Example 5: Display the top 10 volumes with the largest provisioned sizes and have shared connections to host groups with "PROD" in the host group name.

```
$ purevol list --connect --filter "hgroup = '*PROD*'" --sort size- --limit 10
```

## Examples

### Example 1

```
pureman purehost
```

Displays the Purity//FA man page for the `purehost` command.

## Example 2

```
pureman purehost-connect
```

Displays the Purity//FA man page for the **purehost connect** subcommand.

## Related Command

The **pureman** command displays a list of the available Purity//FA commands.

## See Also

[pureadmin\(1\)](#) [211]

[purealert\(1\)](#) [218]

[purearray\(1\)](#) [230]

[pureaudit\(1\)](#) [248]

[purecert\(1\)](#) [251]

[pureconfig\(1\)](#) [256]

[puredns\(1\)](#) [257]

[puredrive\(1\)](#) [260]

[pureeds\(1\)](#) [265]

[purehgroup\(1\)](#) [275]

[purehgroup-connect\(1\)](#) [283]

[purehgroup-list\(1\)](#) [287]

[purehost\(1\)](#) [292]

[purehost-connect\(1\)](#) [302]

[purehost-list\(1\)](#) [307]

[purehw\(1\)](#) [314]

[purelicense\(7\)](#) [327]

[purelog\(1\)](#) [339]

[puremessage\(1\)](#) [354]

[purenetwork\(1\)](#) [360]

[purepgroup\(1\)](#) [381]

[purepgroup-list\(1\)](#) [389]

[pureplugin\(1\)](#) [409]

[purereport\(1\)](#) [433]

[puresmis\(1\)](#) [435]

puresnmp(1) [440]  
purevol(1) [469]  
purevol-list(1) [488]  
purevol-setattr(1) [497]

## Author

Pure Storage Inc. <[documentfeedback@purestorage.com](mailto:documentfeedback@purestorage.com)>

## puremessage

puremessage, puremessage-clear, puremessage-flag, puremessage-list — manage alerts, audit records, and user session logs

### Synopsis

**puremessage** clear *ID*...

**puremessage** flag *ID*...

**puremessage** list [ --audit | --login | --open ] [ --csv | --nvp ] [ --flagged ] [ --notitle ] [ --page ] [ --raw ] [ --recent ] [ --timeline ] [ --user **USER** ] [ *ID*... ]

### Arguments

*ID*

A unique, numeric ID assigned by the array to each message.

### Options

**-h** | **--help**

Can be used with any command or subcommand to display a brief syntax description.

**--audit**

Lists audit records instead of alert messages.

Note: The **--audit** option is going to be replaced by the **pureaudit** command in a future release. Therefore, **--audit** is considered deprecated and should not be used. To list audit records, run the **pureaudit** command.

**--flagged**

Lists only alert messages that have been flagged.

**--login**

Lists user session logs instead of alert messages. If used with the **--open** option, lists only open user sessions (i.e., users who are still logged in).

**--open**

Lists only open alert messages. Purity//FA closes the alert message after the issue has been resolved.

If used with the **--login** option, lists only open user sessions (i.e., users who are still logged in).

**--recent**

Lists only recent messages.

- An alert is considered recent if the situation that triggered it is unresolved, or has only been resolved within the past 24 hours. For example, if free space falls below 20%, an alert is generated. As long as free space remains below 20%, that alert is considered recent. Once the situation has been resolved, the alert is still considered recent for another 24 hours.

- A user session log event is considered recent if the login, logout, or authentication event occurred within the past 24 hours.

--timeline

Lists the opened, updated and closed times for alert messages.

--user (with **--login**)

Lists only login records for the specified user.

Options that control display format:

--csv

Lists information in comma-separated value (CSV) format. The **--csv** output can be used for scripting purposes and imported into spreadsheet programs.

--notitle

Lists information without column titles.

--nvp

Lists information in name-value pair (NVP) format, in the form **ITEMNAME=VALUE**. Argument names and information items are displayed flush left. The **--nvp** output is designed both for convenient viewing of what might otherwise be wide listings, and for parsing individual items for scripting purposes.

--page

Turns on interactive paging.

--raw

Displays the unformatted version of column titles and data. For example, in the **purearray monitor** output, the unformatted version of column title **us/op (read)** is **usec\_per\_read\_op**. The **--raw** output is used to sort and filter list results.

## Description

Purity//FA generates three types of messages:

- *Alert* messages represent exceptional events that have occurred in the Pure Storage array, hardware, or software.
- *Audit trail* messages represent administrative actions performed by a Purity//FA user that change the state of the array. For more information about audit trails, refer to **pureaudit(1)** [248].  
Note: The **--audit** option is going to be replaced by the **pureaudit** command in a future release. Therefore, **--audit** is considered deprecated and should not be used. To list audit records, run the **pureaudit** command.
- *User session logs* represent user login and authentication events performed in the Purity//FA GUI, Purity//FA CLI, and REST API.

The alert and user session messages that appear are retrieved from a list of log entries that are stored on the array. To conserve space, Purity//FA stores a reasonable number of log entries on the array. Older entries are deleted from the log as new entries are added.

To access the complete list of messages, configure the Syslog Server feature to forward all messages to your remote server. Run the `purelog setattr --uri` command to configure the Syslog Server feature.

Purity//FA assigns a unique numeric ID to each alert, audit trail, and user session event as it is created.

## Alerts

An *alert* represents an exceptional event that has occurred in the Pure Storage array, hardware, or software. The severity of an alert ranges from critical to informational, where critical alerts require immediate attention.

The array automatically flags the following alerts:

- New Critical and Warning alerts.
- Existing alerts that have been updated, for example, due to a state change.
- Alerts that have been closed for less than 24 hours and are being reopened.

Run the `puremessage list --flagged` command to view alerts that have been automatically flagged by Purity//FA or manually flagged by the user.

Alerts can be manually flagged and unflagged. Run the `puremessage flag` command to manually flag an alert, and run the `puremessage clear` command to clear an alert message that has been manually or automatically flagged.

Run the `puremessage list` command to display a list of alerts. For example,

puremessage list								
ID	Time	Severity	Category	Code	Component	Name		Event
150	2014-01-31 18:35:19 PST	warning	hardware	15	drive	sh1.bay17		failure
152	2014-02-02 15:01:42 PST	critical	array	4	controller	ct1		failure
153	2014-02-19 10:49:23 PST	warning	hardware	15	cooling	ct1.fan2		failure
445	2014-03-26 19:22:50 PDT	info	array	25	storage	array.capacity		high ut

The `puremessage list` command displays the following alert information:

### ID

Unique number assigned by the array to the alert, audit, or user session event. ID numbers are assigned to messages in chronological ascending order.

### Time

Date and time the event occurred to trigger the alert.

### Severity

Alert severity, categorized as Critical, Warning, or Info. Critical alerts are typically triggered by service interruptions, major performance issues, or risk of data loss, and require immediate attention. For example, Purity//FA triggers a critical alert if it detects a controller failure.

Warning alerts are of low to medium severity and require attention, though not as urgently as critical alerts. For example, Purity//FA triggers a warning alert if it detects that a flash module or NVRAM module is in an unhealthy state.

Informational (Info) alerts inform users of a general behavior change and require no action. For example, Purity//FA triggers an informational alert when an array exceeds 80% capacity.

#### Category

Categorizes the alert event as due to an array, hardware, or software issue.

#### Code

Pure Storage alert code number that identifies the component experiencing the alert event.

#### Component

General array, hardware or software component experiencing the alert event.

#### Name

Specific array, hardware or software component experiencing the alert event.

#### Event

Event that triggered the alert.

#### Expected

Status that Purity//FA expects.

#### Actual

Actual status. The event triggers an alert when the actual status does not meet the expected status.

#### Details

Additional alert details, if any.

Alerts can be open or closed. Open alerts are ones that need to be resolved. Once resolved, Purity//FA closes the alert. Run the `puremessage list --open` command to view open alerts.

## User Session Logs

*User session logs* represent user login and authentication events performed in the Purity//FA GUI, Purity//FA CLI, and REST API. For example, logging in to and out of the Purity//FA GUI, attempting to log in to the Purity//FA CLI with an invalid password, or opening a Pure Storage REST API session generates a user session log entry.

The `puremessage list` command with the `--login` option displays a list of user login and authentication events. For example,

puremessage list --login						
ID	Start Time	End Time	Interface	User	Location	
54	2014-08-14 11:45:00 PDT	2014-08-14 12:00:00 PDT	REST	pureuser	10.124.190.231	A
57	2014-08-14 11:45:00 PDT	2014-08-14 12:00:00 PDT	REST	-	10.124.190.231	f
59	2014-08-14 11:45:47 PDT	2014-08-14 11:45:47 PDT	REST	pureuser	10.124.190.231	u
63	2014-08-14 11:45:47 PDT	2014-08-14 11:45:47 PDT	REST	root	10.124.190.231	u
66	2014-08-14 11:45:00 PDT	2014-08-14 12:00:00 PDT	REST	pureuser	10.124.190.231	r
69	2014-08-14 14:55:11 PDT	2014-08-14 15:59:30 PDT	GUI	pureuser	10.123.14.220	u
71	2014-08-25 17:25:15 PDT	2014-08-25 19:22:30 PDT	CLI	pureadmin	10.123.13.62	u
75	2014-08-25 19:40:37 PDT	2014-08-25 19:40:37 PDT	CLI	pureuser	127.0.0.1	

The **puremessage list --login** command displays the following user session event information:

**ID**

Unique number assigned by the array to the alert, audit, or user session event. ID numbers are assigned to messages in chronological ascending order.

**Start Time**

For user login events, date and time the user logged in to the Purity//FA interface. For authentication events, start time of the time period in which the authentication action occurred.

**End Time**

For user login events, date and time the user logged out of the Purity//FA interface. For authentication events, end time of the time period in which the authentication action occurred.

**Interface**

Purity//FA GUI, Purity//FA CLI, or REST API through which the user session event was performed.

**User**

Username of the Purity//FA user who triggered the user session event.

**Location**

IP address of the user client connecting to the array.

**Event**

Event that was triggered. Login events include 'login' and 'user session'. Authentication events include 'failed authentication', 'API token obtained', and 'request without session'.

**Repeats**

Number of attempts in addition to an initial attempt that a user has performed an authentication action within the start and end time period.

**Method**

Method by which the user attempted to log in, log out, or authenticate. Methods include 'API token', 'password', and 'public key'.

Run the **puremessage list --login --recent** command to view a list of user session events that have issued within the past 24 hours.

Run the **puremessage list --login --user** command to view a list of user session events for the specified user.

## Examples

### Example 1

```
puremessage list
```

Displays a list of all alerts that have been generated by a hardware, software, or array event.

### Example 2

```
puremessage list --open
```

Displays a list of all unresolved alerts.

### Example 3

```
puremessage flag 1574
```

Flags message ID 1574.

### Example 4

```
puremessage list --login --user pureuser
```

Displays a list of user login/logout and authentication events performed by *pureuser*.

## Exceptions

None.

## See Also

[purealert\(1\)](#) [218], [purearray\(1\)](#) [230], [pureaudit\(1\)](#) [248], [puredrive\(1\)](#) [260]

## Author

Pure Storage Inc. <[documentfeedback@purestorage.com](mailto:documentfeedback@purestorage.com)>

## purenetwork

purenetwork, purenetwork-create, purenetwork-delete, purenetwork-disable, purenetwork-enable, purenetwork-list, purenetwork-monitor, purenetwork-setattr — manages and monitors the network interfaces used to connect a FlashArray system to a network

purenetwork-ping, purenetwork-trace — pings remote destinations and traces routes on the network

### Synopsis

```
purenetwork create vif [--address ADDRESS] --subnet SUBNET INTERFACE
purenetwork delete INTERFACE...
purenetwork disable INTERFACE
purenetwork enable INTERFACE
purenetwork list [ --cli | --csv | --nvp ] [--notitle] [--page] [--raw] [--service SERVICE] [--subnet SUBNET] --vlan VLAN
purenetwork monitor [--error] [--interval SECONDS] [--repeat REPEAT-COUNT] [--total] [--historical { 1h | 3h | 24h | 7d | 30d | 90d | 1y } ] [--csv] [--notitle] [--raw] [--filter FILTER] [--limit LIMIT] [--page] [--sort SORT] [INTERFACE...]
purenetwork ping [--count COUNT] [--interface INTERFACE] [--no-hostname] [--packet-size PACKET-SIZE] [--user-to-user-latency] DEST
purenetwork setattr [--address ADDRESS] [--gateway GATEWAY] [--mtu MTU-SIZE] [--netmask NETMASK] [ --subinterfacelist INTERFACELIST | --addsubinterfacelist INTERFACELIST | --remsubinterfacelist INTERFACELIST ] [--subnet SUBNET] INTERFACE
purenetwork trace [--dont-fragment] [--interface] [--method METHOD] [--mtu] [--no-hostname] [--port PORT] [--source SOURCE] DEST
```

### Arguments

#### **DEST**

Internet Protocol (IP) address or full hostname of a ping target or of a remote computer to which the network route is to be determined.

#### **INTERFACE**

Network interface name.

Ethernet interface names are in the form CTx.ETHy, where x denotes the controller (0 or 1) and y denotes the interface (0 or 1).

App interface names are in the form APP.datay and APP.mgmty, where APP denotes the app name and y denotes the interface (0 or 1).

Bond interface names are comprised of alphanumeric characters.

VLAN interface names include a VLAN ID number, which is appended to the name of the physical interface.

In CLI commands, interface names are case-insensitive. For example, *CT0.ETH0*, *Ct0.Eth0*, *CT0.eth0*, and *ct0.eth0* refer to the same Ethernet interface name. Likewise, *bond1*, *Bond1*, and *BOND1* refer to the same bond interface name.

## Options

**-h | --help**

Can be used with any command or subcommand to display a brief syntax description.

**--address *ADDRESS***

IP address to be associated with the specified Ethernet interface.

For IPv4, enter the address in CIDR notation *ddd.ddd.ddd.ddd/dd*. For example, **10.20.20.210/24**. Alternatively, specify the address *ddd.ddd.ddd.ddd* with a netmask (**--netmask**).

For IPv6, enter the address and prefix length in the form

**xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx/xxx**. For example, **2001:0db8:85a3::ae26:8a2e:0370:7334/64**. Consecutive fields of zeros can be shortened by replacing the zeros with a double colon (::). Alternatively, specify the address **xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx** with a prefix length (**--netmask**).

**--addsubinterfacelist (Bond interfaces only)**

Comma-separated list of one or more additional slave devices to be added to the specified bond interface. The slave device name should only include the interface name *ethX*. Do not include the controller name *ct0*.

Purity//FA disables the interface before it is added as a slave device to a bond interface. If the slave device that you are adding to the bond interface is an administrative interface, it will lose SSH connection and no longer be able to connect to the controller.

**--count *COUNT***

Number of Internet Control Message Protocol (ICMP) ping messages to send in sequence.

**--dont-fragment**

Do not fragment the probe packets. For IPv4 and IPv6, it also sets the Don't Fragment (DF) bit. By default, IP allows the packets to be fragmented when it goes through a segment with a smaller maximum transmission unit (MTU).

**--error**

Displays error statistics for all or the specified network interfaces, such as CRC, frame, carrier, and dropped errors.

**--gateway *GATEWAY***

IP address of the gateway through which the specified interface is to communicate with the network. For IPv4, specify the gateway IP address in the form *ddd.ddd.ddd.ddd*. For IPv6, specify the gateway IP address in the form **xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx**. When specifying an IPv6 address, consecutive fields of zeros can be shortened by replacing the zeros with a double colon (::).

To remove the gateway specification, set to a null value ("").

--historical **TIME**

Displays historical data over the specified range of time. Valid time ranges include: 1 hour, 3 hours, 24 hours, 7 days, 30 days, 90 days, and one year. This option is mutually exclusive with the --total option.

--interface **INTERFACE**

For **purenetwork ping**, IP address or interface name of the current controller. If the interface is an address, the source address is set to the specified interface address.

For **purenetwork trace**, interface name of the current controller.

--interval **SECONDS**

Sets the number of seconds between displays of real-time updates. If omitted, the interval defaults to every 30 seconds.

--method **METHOD**

Method for trace operations. By default, the trace operation uses the UDP method. Trace operations can also be specified as ICMP (icmp) or TCP (tcp).

--mtu (**purenetwork trace** only)

Maximum transmission unit (MTU) along the path being traced between two Ethernet hosts.

--mtu **MTU-SIZE** (**purenetwork setattr** only)

Specifies the maximum message transfer unit (packet) size for the interface in bytes. Valid values are integers between 568 and 9000 (inclusive). The default value is 1500.

--netmask **NETMASK**

Defines the range of IP addresses that make up a group of IP addresses on the same network.

For IPv4, if the address is not entered in CIDR notation, enter the subnet mask in the form **ddd.ddd.ddd.ddd**. For example, **255.255.255.0**. For IPv6, if the address entered did not include a prefix length, specify the prefix length. For example, **64**.

## --no-hostname

Do not map IP addresses to hostnames when displaying them.

--packet-size **PACKET-SIZE**

The number of data bytes to be sent. The default packet size is 56. For ICMP tracing, the default packet size is combined with 8 bytes of ICMP header data to equal 64 ICMP data bytes.

--port **PORT**

Initial destination port. The default destination port is 33434.

For UDP tracing, specifies the destination port base used by **purenetwork trace**. The destination port number will be incremented by each probe. For ICMP tracing, specifies the initial ICMP sequence value incremented by each probe. For TCP tracing, specifies the destination port to connect.

## --remsubinterfacelist (Bond interfaces only)

Comma-separated list of one or more slave devices to be removed from the specified bond interface.

--repeat *REPEAT-COUNT*

Sets the number of times to display real-time statistics. If omitted, the repeat count defaults to 1.

--service

Lists only the interfaces configured with the specified service. Supported services include (case-sensitive) app, iscsi, management, nvme-roce, and replication.

--subinterfacelist (Bond interfaces only)

Comma-separated list of one or more slave devices to be added to the specified bond interface.

In the **purenetwork setattr** command, this option replaces the entire list of slave devices that are currently associated with the specified bonding interface.

The slave device name should only include the interface name ethX. Do not include the controller name ct0.

Purity//FA disables the interface before it is added as a slave device to a bond interface. If the slave device that you are adding to the bond interface is an administrative interface, it will lose SSH connection and no longer be able to connect to the controller.

--source *SOURCE*

Alternative source address. The source address must be the address of one of the interfaces. By default, **purenetwork trace** uses the address of the outgoing interface.

--subnet *SUBNET*

For **purenetwork setattr**, name of the subnet which the physical, virtual, bond, or VLAN interface is to be attached. To detach a physical, virtual, or bond interface from a subnet, set to an empty string ("").

For **purenetwork list**, lists only the interfaces that belong to the specified subnet.

--total

Follows output lines with a single line containing column totals in columns where they are meaningful. This option is mutually exclusive with the --historical option.

--user-to-user-latency

Prints full user-to-user latency. By default, **purenetwork ping** prints network round-trip times.

--vlan

Lists only the interfaces configured with the specified VLAN ID.

Options that control display format:

--cli

Displays output in the form of CLI commands that can be issued to reproduce the current configuration. The --cli output is not meaningful when combined with immutable attributes.

--csv

Lists information in comma-separated value (CSV) format. The --csv output can be used for scripting purposes and imported into spreadsheet programs.

--notitle

Lists information without column titles.

--nvp

Lists information in name-value pair (NVP) format, in the form **ITEMNAME=VALUE**. Argument names and information items are displayed flush left. The --nvp output is designed both for convenient viewing of what might otherwise be wide listings, and for parsing individual items for scripting purposes.

--raw

Displays the unformatted version of column titles and data. For example, in the **purearray monitor** output, the unformatted version of column title **us/op (read)** is **usec\_per\_read\_op**. The --raw output is used to sort and filter list results.

Options that manage display results:

--filter

Displays only the rows that meet the filter criteria specified.

--limit

Limits the size of the list output to the specified maximum number of rows.

--page

Turns on interactive paging.

--sort

Sorts the list output in ascending or descending order by the column specified.

## Description

Manages the interfaces and the network connection attributes of the array.

Each FlashArray controller is equipped with two 1-gigabit Ethernet (1GbE) interfaces that connect to a data center network for array administration. The interfaces are called CTx.ETH0 and CTx.ETH1, where x denotes the array controller number. Physical interface ports are located on controller rear bulkheads, and are labeled ETH1 (left) and ETH0 (right).

The **purenetwork enable** and **purenetwork disable** subcommands respectively enable and disable a network interface.

Take caution when disabling an interface. If you disable an interface through which an administrative session is being conducted, it will lose SSH connection and no longer be able to connect to the controller.

The **purenetwork list** subcommand lists all the network interfaces for all controllers on the array.

App interfaces only appear for the apps that have been installed. For bond interfaces, the list also includes the bond interface slave devices, if any. For interfaces that belong to subnets, the list also includes the name of the subnet to which they belong. Include the **--service** or **--vlan** option to display only the interfaces that are configured with the specified service type or VLAN ID, respectively.

The **purenetwork setattr** subcommand changes the IP address, netmask, gateway, and MTU attributes of the specified physical, virtual, or bond interface. If the interface belongs to a subnet, you can only change the IP address.

The prefix length can either be specified with the **--address** option, or as a netmask (for IPv4) or prefix length (for IPv6) with the **--netmask** option.

In the following example, an IPv4 address is being associated with interface `ct0.eth1` (both commands are equivalent):

```
purenetwork setattr --address 192.168.0.25/24 ct0.eth1
purenetwork setattr --address 192.168.0.25 --netmask 255.255.255.0 ct0.eth1
```

In the following example, an IPv6 address is being associated with interface `ct0.eth1` (both commands are equivalent):

```
purenetwork setattr --address 2001:0db8:85a3::ae26:8a2e:0370:7334/64 ct0.eth1
purenetwork setattr --address 2001:0db8:85a3::ae26:8a2e:0370:7334 --netmask 64 ct0.eth1
```

Network interface IP addresses and netmasks are set explicitly (DHCP is not supported), along with the corresponding netmasks.

Gateways are specified by IP address. To remove a port's gateway specification, set the **--gateway** option to a null value (""). In the following example, a null value is specified to remove the port's gateway specification that was associated with `ct0.eth0`:

```
purenetwork setattr --gateway "" ct0.eth1
```

Change the maximum transmission unit (MTU) of an interface by specifying **--mtu**. If you are changing the MTU of a physical interface that is associated with a VLAN, verify the MTU of the physical interface is greater than or equal to (>=) the MTU of the VLAN interface.

## Apps

The Purity Run platform extends array functionality by integrating add-on services into the Purity//FA operating system. Each service that runs on the platform is provided by an app.

For each app that is installed, one app management interface is created per array management interface. An app data interface may also be created for high-speed data transfers.

The naming convention for app interfaces is `APP.datay` for the app data interface, and `APP.mgmty` for the app management interface, where `APP` denotes the app name, and `y` denotes the interface.

The following example lists three app interfaces for the `linux` app. The `linux.mgmt0` interface has been enabled and configured with an IP address to give `pureuser` the ability to log into the app.

```
$ purenetwork list
Name      Enabled  Address     ...  Speed       Services
...
linux.data0  False    -          ...  10.00 Gb/s  app
linux.mgmt0  True     10.8.102.96  ...  1.00 Gb/s  app
linux.mgmt1  False    -          ...  1.00 Gb/s  app
...
```

For more information about Apps, refer to `pureapp(1)` [221].

## Network Bonding

A bond interface combines two or more similar Ethernet interfaces, such as two or more 1GbE or two or more 10GbE interfaces, to form a single virtual "bonded" interface. A bond interface provides higher data transfer rates, load balancing, and link redundancy.

The **purenetwork setattr --addsubinterfacelist** command adds Ethernet interfaces to a bond interface. When you add an Ethernet interface to a bond interface, it becomes a slave to the bond interface. An Ethernet interface can only belong to one bond interface at a time. Administrative slaves cannot be added as slaves to bond interfaces. Furthermore, bond interfaces cannot be slaves to other bond interfaces.

The **purenetwork setattr --remslavelist** command removes slaves from a bond interface. When a slave is removed from its bond interface, it becomes an Ethernet interface.

The **purenetwork delete** command deletes a bond interface. Before a bond interface can be deleted, it must be empty of all slaves.

## Subnets

Interfaces with common attributes can be organized into subnetworks, or subnets, to enhance the efficiency of app, data (iSCSI or NVMe-RoCE), management, and replication traffic.

In Purity//FA, subnets can include physical, virtual, bond, and VLAN interfaces.

*Physical, virtual, and bond interfaces* can belong to the same subnet. To configure a subnet with physical, virtual, or bond interfaces:

1. Create the subnet. Run **puresubnet create** to create the subnet.
2. Add the physical, virtual, and bond interfaces to the subnet. Run **purenetwork setattr --subnet** to add a physical, virtual, or bond interface to a subnet.

To remove a physical, virtual, or bond interface from a subnet, set the **--subnet** option to a null value.

*VLAN interfaces* can belong to subnets, but they cannot be mixed with other interface types. All of the VLAN interfaces within a subnet must be in the same VLAN.

To configure a subnet with VLAN interfaces:

1. Create a subnet, assigning a VLAN ID to the subnet. Run **puresubnet create --vlan** to create the subnet.
2. Run **purenetwork create vif --subnet** to create a VLAN interface. The VLAN ID of the VLAN interface must match the VLAN ID of the subnet. Run the command for each corresponding physical network interface to be associated with the VLAN. To remove a VLAN interface from a subnet, run **purenetwork delete**.

For more information about subnets, refer to *puresubnet(1)* [447].

## Subnets and VLAN Interfaces

VLAN tagging allows customers to isolate traffic through multiple virtual local area networks (VLANs), ensuring data routes to and from the appropriate networks.

The **purenetwork create vif** command creates VLAN interfaces.

Create a VLAN interface for each of the corresponding physical network interfaces you want to associate with a VLAN.

In Purity//FA, VLAN interfaces have the naming structure `ctx.ETHy.z`, where `x` denotes the controller (0 or 1), `y` denotes the interface (0 or 1), and `z` denotes the VLAN ID number. For example, `ct0.eth1.500`.

The **purenetwork create vif** command requires the **--subnet** option, which adds the new VLAN interface to the appropriate subnet. Run **puresubnet list** to display a list of all subnets on the array. Run **puresubnet create** to create a new subnet.

All interfaces, including VLAN interfaces, inherit the MTU value of its subnet. Note that the MTU of a VLAN interface cannot exceed the MTU of the corresponding physical interface.

VLAN is only supported for the iSCSI service type, so before creating a VLAN interface, verify the iSCSI service is configured on the physical interface.

When creating a VLAN interface, include the **--address** option to create a reachable VLAN interface.

After a VLAN interface has been created and attached to a subnet, the interface inherits the mask, gateway, MTU, and VLAN ID from the subnet. Likewise, the subnet inherits the service type (for example, iSCSI) from its interfaces.

For more information about subnets and VLAN tagging, refer to **puresubnet(1)** [447].

## Network Ping and Trace

The **purenetwork ping** subcommand is used to determine whether a remote computer can be accessed by the array, provided that the remote computer is ICMP-enabled. The ping target can be specified by IP address. If a DNS service is available and has been configured for the array, the ping target can be specified by hostname.

The **purenetwork trace** subcommand traces and displays the route to a remote computer identified by IP address or, if a DNS service is available and configured for the array, by hostname.

## Monitoring Network Interfaces

You can display network statistics, historical bandwidth, and error reporting for all or the specified network interfaces by using the **purenetwork monitor** command. The following example displays the network statistics for interfaces `ct0.eth0` and `ct0.eth1`.

\$ purenetwork monitor							
Name	Time	B/s(rx)	B/s(tx)	Packets/s(rx)	Packets/s(tx)	Total	Errors
ct0.eth0	2019-06-25 13:34:11 PDT	8913	634	105	5		0
ct0.eth1	2019-06-25 13:34:11 PDT	7781	162	97	1		0

To show the aggregated data for each statistic of all network interfaces, specify the **--total** option, as shown in the following example.

\$ purenetwork monitor --total							
Name	Time	B/s(rx)	B/s(tx)	Packets/s(rx)	Packets/s(tx)	Total	Errors

<b>ct0.eth0</b>	2019-06-25	13:34:11	PDT	8913	634	105	5	0
<b>ct0.eth1</b>	2019-06-25	13:34:11	PDT	7781	162	97	1	0
<b>(total)</b>	2019-06-25	13:34:11	PDT	16694	796	202	6	0

To report error statistics of network interfaces, run the **purenetwork monitor --error** command.

The following types of errors are reported:

- **CRC errors**: Indicate that packets have incorrect checksums. A cyclic redundancy check (CRC) is an error-detecting code for data transmission.
- **Frame errors**: Indicate that the received packets have misaligned Ethernet frames.
- **Carrier errors**: Indicate duplex mismatch or faulty hardware issues.
- **Dropped errors**: Indicate network congestion. For example, the packets are dropped because the connected switch ports cannot process packets fast enough.
- **Other errors**: Indicate all other types of receive and transmit errors.

## Examples

### Example 1

```
purenetwork enable ct0.eth1
```

Enables controller **ct0**'s administrative network interface **eth1** to communicate with the administrative network.

### Example 2

```
purenetwork list
```

Lists the attributes of the all network interfaces on the array.

### Example 3

```
purenetwork setattr ct0.eth1 --address 192.168.0.24 --netmask 255.255.255.0
purenetwork setattr ct0.eth1 --address 192.168.0.24/24
```

Assigns IPv4 address **192.168.0.24** to administrative Ethernet interface **ct0.eth1**. Both commands are equivalent.

### Example 4

```
purenetwork setattr ct0.eth1 --address 2001:db8:85a3::8a2e:370:7334 --netmask 64
```

Assigns IPv6 address **2001:db8:85a3::8a2e:370:7334** to administrative Ethernet interface **ct0.eth1** with the prefix length of 64 bits.

### Example 5

```
purenetwork setattr --addsubinterfacelist eth2,eth3 bond009
```

Adds Ethernet interfaces **eth2** and **eth3** as slaves to bond interface **bond009**.

#### Example 6

```
purenetwork setattr --address 10.8.108.165 linux.mgmt0
purenetwork enable linux.mgmt0
```

Assigns IP address **10.8.108.165** to app management interface **linux.mgmt0** and enables the interface, giving **pureuser** the ability to log into the **linux** app.

#### Example 7

```
purenetwork create vif --address 192.168.0.24 --subnet ESXHost001 ct0.eth5.500
```

Creates VLAN interface **ct0.eth5.500** with IP address **192.168.0.24**, and adds it to existing subnet **ESXHost001**. The new VLAN interface inherits the mask, gateway, MTU, and VLAN ID from subnet **ESXHost001**.

#### Example 8

```
purenetwork delete ct0.eth5.500
```

Removes VLAN interface **ct0.eth5.500** from its subnet and then deletes it.

#### Example 9

```
purenetwork setattr --subnet "" ct0.eth5
```

Removes physical interface **ct0.eth5** from its subnet. The interface is not deleted.

#### Example 10

```
purenetwork disable replbond
purenetwork setattr --remslavelist eth2,eth3 --address "" --netmask "" --gateway "" replbond
purenetwork delete replbond
```

Disables the **replbond** bond interface, removes slaves **eth2** and **eth3** from the bond interface and clears the bond interface of all settings, and deletes the **replbond** bond interface.

#### Example 11

```
purenetwork ping --count 100 myhost.mydomain.com
```

Sends 100 ICMP ping messages to host **myhost.mydomain.com** and displays the response received for each one.

#### Example 12

```
purenetwork ping --interface eth0 8.8.8.8
```

Sends an ICMP ping message from Ethernet port **eth0** of the current controller to the host with IP address **8.8.8.8**, and displays the response received.

**Example 13**

```
purenetwork trace 192.168.0.1
```

Displays the route taken by a ping request to network address **192.168.0.1**.

**Example 14**

```
purenetwork monitor --historical 1h
```

Displays the network statistics of all the network interfaces over the past 1 hour.

**Example 15**

```
purenetwork monitor --error
```

Displays the error statistics of all the network interfaces.

**See Also**

[pureapp\(1\)](#) [221] [purearray\(1\)](#) [230] [puredns\(1\)](#) [257] [pureds\(1\)](#) [265] [puresubnet\(1\)](#) [447] [purevol-list\(1\)](#) [488]

**Author**

Pure Storage Inc. <[documentfeedback@purestorage.com](mailto:documentfeedback@purestorage.com)>

## pureoffload

pureoffload, pureoffload-list, pureoffload-azure, pureoffload-nfs, pureoffload-s3 — manages offload targets

### Synopsis

```
pureoffload list [ --cli | --csv | --nvp ] [--notitle] [--page] [--raw] [NAME...]  
pureoffload azure connect --account-name ACCOUNT-NAME [--container-name  
CONTAINER-NAME] [--initialize] NAME  
pureoffload azure disconnect NAME  
pureoffload azure list [ --cli | --csv | --nvp ] [--notitle] [--page] [--raw]  
[NAME...]  
pureoffload nfs connect --address ADDRESS --mount-point MOUNT-POINT [--mount-  
options MOUNT-OPTIONS] NAME  
pureoffload nfs disconnect NAME  
pureoffload nfs list [ --cli | --csv | --nvp ] [--notitle] [--page] [--raw]  
[NAME...]  
pureoffload s3 connect --access-key-id ACCESS-KEY-ID --bucket BUCKET [--  
initialize] [--placement-strategy { aws-standard-class | retention-based } ] NAME  
pureoffload s3 disconnect NAME  
pureoffload s3 list [ --cli | --csv | --nvp ] [--notitle] [--page] [--raw]  
[NAME...]
```

### Arguments

#### **NAME**

Name used by Purity//FA to identify an offload target.

### Options

#### **-h | --help**

Can be used with any command or subcommand to display a brief syntax description.

#### **--access-key-id *ACCESS-KEY-ID***

Access key ID of the AWS account. The access key is 20 characters in length.

A secret access key of the AWS account is also required to authenticate requests between the array and S3 bucket. The secret access key is 40 characters in length, and is entered through interactive prompt during connection.

#### **--address *ADDRESS***

Hostname or IP address of the NFS server.

For IPv4, enter the address in CIDR notation `ddd.ddd.ddd.ddd/dd`. For example,  
`10.20.20.210/24`.

For IPv6, enter the address and prefix length in the form  
`xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxx/xxx`. For example,  
`2001:0db8:85a3::ae26:8a2e:0370:7334/64`. Consecutive fields of zeros can be  
shortened by replacing the zeros with a double colon (`::`).

--bucket ***BUCKET***

Name of the Amazon S3 bucket.

--container-name ***CONTAINER-NAME***

Name of the Microsoft Azure Blob container.

--initialize

Prepares an Azure Blob container or S3 bucket for offloading. Only required if this is the first time  
a FlashArray array is connecting to the Azure Blob container or S3 bucket. The array will only  
initialize an Azure Blob container or S3 bucket if it is empty.

--mount-options ***MOUNT-OPTIONS***

List mount options in comma-separated value (CSV) format. Supported mount options include  
**`port`**, **`rsize`**, **`wsize`**, **`nfsvers`**, and **`tcp`** or **`udp`** and are common options available to all NFS  
file systems.

--mount-point ***MOUNT-POINT***

NFS export on the NFS server. For example, `/mnt`.

--placement-strategy

Specifies the placement strategy for offloaded data. The valid values are as follows:

- **`aws-standard-class`**: selects the Amazon S3 Standard storage class.
- **`retention-based`**: selects the Amazon S3 storage class based on the retention period of  
the protection group.

If you do not specify this option, the default is the **`retention-based`** placement strategy when  
the array is initially connected to an Amazon S3 offload target.

Options that control display format:

--cli

Displays output in the form of CLI commands that can be issued to reproduce the current  
configuration. The **--cli** output is not meaningful when combined with immutable attributes.

--csv

Lists information in comma-separated value (CSV) format. The **--csv** output can be used for  
scripting purposes and imported into spreadsheet programs.

--notitle

Lists information without column titles.

--nvp

Lists information in name-value pair (NVP) format, in the form **ITEMNAME=VALUE**. Argument names and information items are displayed flush left. The --nvp output is designed both for convenient viewing of what might otherwise be wide listings, and for parsing individual items for scripting purposes.

--page

Turns on interactive paging.

--raw

Displays the unformatted version of column titles and data. For example, in the **purearray monitor** output, the unformatted version of column title **us/op (read)** is **usec\_per\_read\_op**. The --raw output is used to sort and filter list results.

## Object Names

Valid characters are letters (A-Z and a-z), digits (0-9), and the hyphen (-) character. The first and last characters of the name must be alphanumeric, and the name must contain at least one letter or '-'.

Most objects in Purity//FA that can be named, including host groups, hosts, volumes, protection groups, volume and protection group suffixes, SNMP managers, and subnets, can be 1-63 characters in length.

Array names can be 1-56 characters in length. The array name length is limited to 56 characters so that the names of the individual controllers, which are assigned by Purity//FA based on the array name, do not exceed the maximum allowed by DNS.

Names are case-insensitive on input. For example, **vol1**, **Vol1**, and **VOl1** all represent the same volume. Purity//FA displays names in the case in which they were specified when created or renamed.

Pods and volume groups provide a namespace with unique naming conventions.

All objects in a pod have a fully qualified name that include the pod name and object name. The fully qualified name of a volume in a pod is **POD::VOLUME**, with double colons (::) separating the pod name and volume name. The fully qualified name of a protection group in a pod is **POD::PGROUP**, with double colons (::) separating the pod name and protection group name. For example, the fully qualified name of a volume named **vol01** in a pod named **pod01** is **pod01::vol01**, and the fully qualified name of a protection group named **pgroup01** in a pod named **pod01** is **pod01::pgroup01**.

If a protection group in a pod is configured to asynchronously replicate data to a target array, the fully qualified name of the protection group on the target array is **POD:PGROUP**, with single colons (:) separating the pod name and protection group name. For example, if protection group **pod01::pgroup01** on source array **array01** asynchronously replicates data to target array **array02**, the fully qualified name of the protection group on target array **array02** is **pod01:pgroup01**.

All objects in a volume group have a fully qualified name that includes the volume group name and the object name, separated by a forward slash (/). For example, the fully qualified name of a volume named **vol01** in a volume group named **vgroup01** is **vgroup01/vol01**.

## Description

The offload target feature enables system administrators to replicate point-in-time volume snapshots from the array to an external storage system for long-term retention. Each snapshot is an immutable image of the volume data at that instance in time. The data is transmitted securely and stored unencrypted on the storage system.

Each offload target represents an external storage system, such as an Azure Blob container, NFS device, or S3 bucket, to where Purity//FA volume snapshots (generated via protection group snapshots) can be replicated.

Before you can connect to, manage, and replicate to an offload target, the respective Purity//FA app must be installed. For example, to connect to an NFS offload target, the Snap to NFS app must be installed. To connect to an Azure Blob container or S3 bucket, the Snap to Cloud app must be installed. To determine if apps are installed on your array, run the `pureapp list` command. To install the Snap to NFS or Snap to Cloud app, contact Pure Storage Support.

To replicate volume snapshots to an offload target, the array must be able to connect to and reach the external storage system. Before you configure an offload target on the array, perform the following steps to prepare the offload target and verify that the network is set up to support the offload process:

1. Verify that at least one interface with the replication service is configured on the array. Run the `purenw` command lists and manages network interfaces. Assign an IP address to the port; this will be the interface that will be used to connect to the target device, such as an Azure Blob container, a NAS/NFS device, an NFS storage system, an S3 bucket, or a generic Linux server. For optimum performance, an Ethernet interface of at least 10GbE is recommended.
2. Prepare the offload target.
  - For Azure Blob, create a Microsoft Azure Blob container and set the storage account to the hot access tier. Grant basic read and write ACL permissions, and verify that the container contains no blobs. By default, server-side encryption is enabled for the container and cannot be disabled.
  - For NFS, create the NFS export, granting read and write access to the array for all users.
  - For S3, create an Amazon S3 bucket. Grant basic read and write ACL permissions, and enable default (server-side) encryption for the bucket. Also verify that the bucket is empty of all objects and does not have any lifecycle policies.
3. Verify that the array can reach the offload target.

The `pureoffload` command displays and manages the connections between the array and offload targets.

The `pureoffload list` command displays a list of all offload targets that are connected to the array. Each offload target represents an external storage system, such as an Azure Blob container, NFS device, or S3 bucket, to where Purity//FA volume snapshots (generated via protection group snapshots) can be replicated. Offload targets that are disconnected from the array do not appear in the list of offload targets.

## Azure Blob Offload

The **pureoffload azure list** command displays the connection details for each Azure Blob offload target that is connected to the array. Details include connection status, protocol type, container name, account name, and secret access key (displayed in masked form).

To configure an offload target:

1. Connect the array to the offload target. For Azure, creating the connection to the Microsoft Azure Blob container requires the container account name and secret access key, both of which are created through the Microsoft Azure storage website.
2. Create a protection group, adding volume members and the offload target to the group.
3. Define the protection group snapshot and replication schedule.
4. Enable the protection group schedule.

The **pureoffload azure connect** command connects the array to an Azure Blob offload target. An array can be connected to one offload target at a time, while multiple arrays can be connected to the same offload target. Include the required **--account-name** option to specify the Microsoft Azure Blob account name when creating a connection between the array and an Azure Blob offload target. The secret access key of the Azure Blob account is required to authenticate requests between the array and Azure Blob container. The secret access key is entered twice through interactive prompt during connection. Specify the name of the Azure Blob container using the **--container-name** option. If not specified, the default is **offload**. If this is the first time any FlashArray array is connecting to the Azure Blob container, include the **--initialize** option to prepare the Azure Blob container for offloading.

The **purepgroup** command creates protection groups and generates the protection group snapshots through which Aure Blob offload snapshots are generated. The **purepgroup** command also generates the volume snapshots and replicates them, either on a scheduled basis or on demand, to the offload target via protection group snapshots. For more information about creating protection groups and generating protection group snapshots, refer to **purepgroup(1)** [381].

The **purevol** command displays and manages the individual volume snapshots on the offload target that are generated through the protection group snapshots. Through the **purevol** command, view a list of volume snapshots on the offload target and restore volume snapshots from an offload target to create a brand new volume. For more information about viewing and restoring volume snapshots, refer to **purevol(1)** [469].

The **pureoffload azure disconnect** command disconnects the array from an Azure Blob offload target. Whenever an array is disconnected from an offload target, any data transfer processes that are in progress are suspended. The processes resume when the connection is re-established.

In the following example, the first command connects the array to Azure Blob offload target **mytarget**, which will offload volume snapshots to Azure Blob container **mybackup**. Because this is the first time any FlashArray array is connecting to Azure Blob container **mybackup**, the **--initialize** option is included. The secret access key is entered interactively. The second command displays the connection details of each Azure Blob offload target that is connected to the array.

```
$ pureoffload azure connect --initialize  
--account-name mytest
```

```
--container-name mybackup mytarget
Enter secret access key:
Retype secret access key:
Name      Status      Protocol Container Name Account Name Secret Access Key
mytarget  connected  azure       mybackup      mytest        ****

$ pureoffload azure list
Name      Status      Protocol Container Name Account Name Secret Access Key
mytarget  connected  azure       mybackup      mytest        ****
```

## NFS Offload

The **pureoffload nfs list** command displays the connection details for each NFS offload target that is connected to the array. Details include connection status, hostname or IP address of the NFS server, mount point on the NFS server, protocol type, and mount options, if any.

To configure an offload target:

1. Connect the array to the offload target. For NFS, creating the connection requires the host name or IP address of the server (such as the NFS server) and the mount point on the server. If the protection group is in a stretched pod, for high availability, connect both arrays in the stretched pod to the offload target.
2. Create a protection group, adding volume members and the offload target to the group.
3. Define the protection group snapshot and replication schedule.
4. Enable the protection group schedule.

The **pureoffload nfs connect** command connects the array to an NFS offload target. Include the required **--address** and **--mount-point** options when creating a connection between the array and an NFS offload target. An array can be connected to one offload target at a time, while multiple arrays can be connected to the same offload target. If the protection group is in a stretched pod, for high availability, connect both arrays in the stretched pod to the offload target.

The **purepgroup** command creates protection groups and generates the protection group snapshots through which NFS offload snapshots are generated. The **purepgroup** command also generates the volume snapshots and replicates them, either on a scheduled basis or on demand, to the offload target via protection group snapshots. For more information about creating protection groups and generating protection group snapshots, refer to **purepgroup(1)** [381].

The **purevol** command displays and manages the individual volume snapshots on the offload target that are generated through the protection group snapshots. Through the **purevol** command, view a list of volume snapshots on the offload target and restore volume snapshots from an offload target to create a brand new volume. For more information about viewing and restoring volume snapshots, refer to **purevol(1)** [469].

The **pureoffload nfs disconnect** command disconnects the array from an NFS offload target. Whenever an array is disconnected from an offload target, any data transfer processes that are in progress are suspended. The processes resume when the connection is re-established.

## S3 Offload

The **pureoffload s3 list** command displays the connection details for each S3 offload target that is connected to the array. Details include connection status, protocol type, bucket name, access key ID, secret access key (displayed in masked form), and placement strategy.

To configure an offload target:

1. Connect the array to the offload target. For S3, creating the connection to the Amazon S3 bucket requires the bucket's access key ID and secret access key, both of which are created through Amazon Web Services.
2. Create a protection group, adding volume members and the offload target to the group.
3. Define the protection group snapshot and replication schedule.
4. Enable the protection group schedule.

The **pureoffload s3 connect** command connects the array to an S3 offload target. An array can be connected to one offload target at a time, while multiple arrays can be connected to the same offload target. Include the required **--access-key-id** and **--bucket** options when creating a connection between the array and an S3 offload target. The access key ID is the 20-character access key ID of the AWS account. The 40-character secret access key of the AWS account is also required to authenticate requests between the array and S3 bucket. The secret access key is entered twice through interactive prompt during connection. The bucket represents the name of the S3 bucket. If this is the first time any FlashArray array is connecting to the S3 bucket, include the **--initialize** option to prepare the S3 bucket for offloading.

The **purepgroup** command creates protection groups and generates the protection group snapshots through which S3 offload snapshots are generated. The **purepgroup** command also generates the volume snapshots and replicates them, either on a scheduled basis or on demand, to the offload target via protection group snapshots. For more information about creating protection groups and generating protection group snapshots, refer to **purepgroup(1)** [381].

The **purevol** command displays and manages the individual volume snapshots on the offload target that are generated through the protection group snapshots. Through the **purevol** command, view a list of volume snapshots on the offload target and restore volume snapshots from an offload target to create a brand new volume. For more information about viewing and restoring volume snapshots, refer to **purevol(1)** [469].

The **pureoffload s3 disconnect** command disconnects the array from an S3 offload target. Whenever an array is disconnected from an offload target, any data transfer processes that are in progress are suspended. The processes resume when the connection is re-established.

## Dynamic Placement

When you connect the array to an S3 offload target using the **pureoffload s3 connect** command, the command chooses a cost-effective storage class using dynamic placement. Dynamic placement stores objects automatically with an optimized storage class depending on the retention period of the protection groups. For data that is less frequently accessed, dynamic placement provides a more cost-effective use of object storage.

By default, dynamic placement chooses the retention-based placement strategy when the array is initially connected to an S3 offload target. To manually select a placement strategy for an offload target connection, specify the **--placement-strategy** option as follows:

- **--placement-strategy retention-based**

Specifies the retention-based placement strategy to configure the Amazon S3 storage class based on the retention period of the protection group.

This is the default placement strategy if you do not specify the **--placement-strategy** option.

- **--placement-strategy aws-standard-class**

Specifies this placement strategy to select the Amazon S3 Standard storage class to offload volume snapshots. This storage class is suitable for general-purpose storage of frequently accessed data.

When an S3 offload target is disconnected and then reconnected, the placement strategy is determined as follows:

- If no placement strategy is specified, the existing placement strategy is used.
- If a different placement strategy is specified, the system will move objects to the storage class selected by the placement strategy for the offload target connection regardless of storage costs.

## Examples

### Example 1

```
pureoffload list
```

Displays a list of all offload targets that are connected to the array, their connection statuses and their protocol types.

### Example 2

```
pureoffload nfs list
```

Displays the connection details for each NFS offload target that is connected to the array, including connection status, hostname or IP address of the NFS server, mount point on the NFS server, and mount options.

### Example 3

```
pureoffload nfs connect --address 192.168.1.10 --mount-point /mnt NFS-TARGET01
```

Connects the array to NFS offload target **NFS-TARGET01**, which will offload volume snapshots to mount point **/mnt** on NFS server **192.168.1.10**.

### Example 4

```
pureoffload s3 connect --initialize --bucket purebucket1  
--access-key-id ABCDEFGHIJKLMNOPQRST S3-TARGET01
```

```
Enter secret access key:  
Retype secret access key:
```

Connects the array to S3 offload target **s3-TARGET01**, which will offload volume snapshots to Amazon S3 bucket **purebucket1**. Since this is the first time any FlashArray array is connecting to S3 bucket **purebucket1**, the **--initialize** option is included. The secret access key is entered interactively.

#### Example 5

```
pureoffload s3 connect --bucket purebucket2  
    --access-key-id ZYXWVUTSRQPOMNLKJIHG S3-TARGET02  
Enter secret access key:  
Retype secret access key:
```

Connects the array to NFS offload target **s3-TARGET02**, which will offload volume snapshots to Amazon S3 bucket **purebucket1**. A FlashArray array has already initialized S3 bucket **purebucket1** from a previous S3 offload connection, so the bucket does not need to be initialized again. The secret access key is entered interactively.

#### Example 6

```
pureoffload s3 connect --bucket purebucket1  
    --access-key-id ABCDEFGHIJKLMNOPQRST S3-TARGET03  
    --placement-strategy aws-standard-class  
Enter secret access key:  
Retype secret access key:
```

Connects the array to S3 offload target **s3-TARGET03** and sets the placement strategy to the **aws-standard-class** storage class to offload volume snapshots to Amazon S3 bucket **purebucket1**. The secret access key is entered interactively.

#### Example 7

```
pureoffload s3 disconnect S3-TARGET
```

Disconnects the array from S3 offload target **S3-TARGET**.

#### Example 8

```
pureoffload azure disconnect AZCONN
```

Disconnects the array from Azure Blob offload target **AZCONN**.

#### Example 9

```
purepgroup create --vollist vol01,vol02 --targetlist NFS-TARGET01 pgroup01  
purepgroup schedule --replicate-frequency 4h pgroup01  
purepgroup enable --replicate pgroup01
```

Creates protection group pgroup01 with volumes **vol01** and **vol02** and offload target **NFS-TARGET01**. Configures the protection group schedule to replicate a snapshot of all volumes in pgroup01 to offload target **NFS-TARGET01** every four hours. Enables the replication schedule for protection pgroup01 to immediately begin replicating protection group snapshots to offload target **NFS-TARGET01**.

### Example 10

```
purevol list --snap --on NFS-TARGET01  
purevol get --on NFS-TARGET01 pure-001:pgroup01.25.vol03  
purevol copy pure-001:vol03.restore-of.pure-001-pgroup01-25-vol03 restore1
```

Displays a list of volume snapshots (generated from protection group snapshots) both local and remote to the array that have been replicated and retained on offload target **NFS-TARGET01**. Restores volume snapshot **pure-001:pgroup01.25.vol03** from offload target **NFS-TARGET01**, resulting in a volume snapshot with the name **pure-001:vol03.restore-of.pure-001-pgroup01-25-vol03**, which is then copied to create a new volume named **restore1**.

### See Also

[puregroup\(1\)](#) [381] [purevol\(1\)](#) [469]

### Author

Pure Storage Inc. <[documentfeedback@purestorage.com](mailto:documentfeedback@purestorage.com)>

## purepgroup

`purepgroup`, `purepgroup-create`, `purepgroup-copy`, `purepgroup-destroy`, `purepgroup-eradicate`, `purepgroup-recover`, `purepgroup-rename`, `purepgroup-snap` — manage the creation and deletion of the Purity//FA protection group (pgroup) objects.

### Synopsis

```
purepgroup copy [ --overwrite ] SOURCE DEST
purepgroup create [ --hgrouplist HGROUPS | --hostlist HOSTS | --vollist VOLS ]
[--targetlist TARGETS] PGROUP...
purepgroup destroy PGROUP...
purepgroup eradicate PGROUP...
purepgroup recover PGROUP...
purepgroup rename OLD-NAME NEW-NAME
purepgroup snap [ --apply-retention ] [ --replicate-now | --replicate ] [ --suffix
SUFFIX] PGROUP...
```

### Arguments

#### **PGROUP**

Protection group or protection group snapshot to be created, destroyed, eradicated, or recovered. For `purepgroup snap`, protection group to be snapped.

#### **SOURCE**

Protection group or protection group snapshot on the target array from where data is copied. The data is copied to the `DEST` protection group.

#### **DEST**

Protection group to where data is copied. The data is copied from the `SOURCE` protection group or protection group snapshot on the target array.

#### **OLD-NAME**

Name of the protection group to be renamed.

#### **NEW-NAME**

Name by which the protection group is to be known after the command executes.

### Options

#### **-h | --help**

Can be used with any command or subcommand to display a brief syntax description.

#### **--apply-retention**

Applies the retention schedule settings to the snapshot.

--hgrouplist

Comma-separated list of one or more host groups to be included in the specified protection groups. Host and host groups are not supported for replication to offload targets; instead, add volume members.

--hostlist

Comma-separated list of one or more hosts to be included in the specified protection groups. Hosts and host groups are not supported for replication to offload targets; instead, add volume members.

--overwrite

Allows **purepgroup copy** to overwrite existing volumes. Without this option, if the **DEST** protection group already contains volumes with matching names, then the command fails. With this option, if the **DEST** protection group exists, then the **SOURCE** and **DEST** protection groups must contain the same volumes (volume names).

--replicate, --replicate-now

Replicates the snapshot to all targets. For asynchronous replication, only replicates snapshots to target arrays that have allowed replication.

--suffix

Specifies a name suffix for the snapshots created. Specifying this option causes snapshots to have names of the form **PGROUP.SUFFIX** rather than the default **PGROUP.NNN** form. The names of all snapshots created by a single command that specifies this option have the same suffix.

--targetlist

Comma-separated list of one or more target arrays (for asynchronous replication) or offload targets (for replication to an offload target) to be included in the specified protection groups. The targets receive the replicated snapshots.

--vollist

Comma-separated list of one or more volumes to be included in the specified protection groups.

## Conventions

### Object Names

Valid characters are letters (A-Z and a-z), digits (0-9), and the hyphen (-) character. The first and last characters of the name must be alphanumeric, and the name must contain at least one letter or '-'.

Most objects in Purity//FA that can be named, including host groups, hosts, volumes, protection groups, volume and protection group suffixes, SNMP managers, and subnets, can be 1-63 characters in length.

Array names can be 1-56 characters in length. The array name length is limited to 56 characters so that the names of the individual controllers, which are assigned by Purity//FA based on the array name, do not exceed the maximum allowed by DNS.

Names are case-insensitive on input. For example, **vol1**, **Vol1**, and **VOL1** all represent the same volume. Purity//FA displays names in the case in which they were specified when created or renamed.

Pods and volume groups provide a namespace with unique naming conventions.

All objects in a pod have a fully qualified name that include the pod name and object name. The fully qualified name of a volume in a pod is **POD::VOLUME**, with double colons (::) separating the pod name and volume name. The fully qualified name of a protection group in a pod is **POD::PGROUP**, with double colons (::) separating the pod name and protection group name. For example, the fully qualified name of a volume named **vol01** in a pod named **pod01** is **pod01::vol01**, and the fully qualified name of a protection group named **pgroup01** in a pod named **pod01** is **pod01::pgroup01**.

If a protection group in a pod is configured to asynchronously replicate data to a target array, the fully qualified name of the protection group on the target array is **POD:PGROUP**, with single colons (:) separating the pod name and protection group name. For example, if protection group **pod01::pgroup01** on source array **array01** asynchronously replicates data to target array **array02**, the fully qualified name of the protection group on target array **array02** is **pod01:pgroup01**.

All objects in a volume group have a fully qualified name that includes the volume group name and the object name, separated by a forward slash (/). For example, the fully qualified name of a volume named **vol01** in a volume group named **vgroup01** is **vgroup01/vol01**.

## Protection Group Snapshot Names

The protection group snapshot naming convention is **PGROUP.NNN**, where:

- **PGROUP** is the name of the protection group.
- **NNN** is a unique monotonically increasing number or a manually-assigned protection group snapshot suffix name.

The protection group volume snapshot naming convention is **PGROUP.NNN.VOL**, where:

- **PGROUP** is the name of the protection group.
- **NNN** is a unique monotonically increasing number or a manually-assigned protection group snapshot suffix name.
- **VOL** is name of the volume member.

If you are viewing replicated snapshots on a target array or offload target, the snapshot name begins with the name of the source array from where the snapshot was taken.

## Description

A protection group defines a set of volumes, hosts, or host groups (called members) that are protected together through snapshots with point-in-time consistency across the member volumes. The members within the protection group have common data protection requirements and the same snapshot, replication, and retention schedules.

Protection group snapshots capture the content of all volumes on the source array for the specified protection group at a single point in time. The snapshot is an immutable image of the volume data at that instance in time. The volumes are either direct members of the protection group or connected to any of its hosts or host groups within a protection group.

Volumes are protected through protection group snapshots that are retained locally, replicated to other arrays or storage systems, or both. Purity//FA supports three types of replication:

- Asynchronous replication, where volume snapshots are replicated from the current (source) array to a target array. For asynchronous replication, a single protection group can consist of multiple hosts, host groups, and volumes. Likewise, hosts, host groups, and volumes can be associated with multiple protection groups.
- Synchronous replication, where volume snapshots are synchronously replicated between two arrays through stretched pods. For more information about synchronous replication, refer to `purepod(1)` [412].
- Replication to an offload target, where volume snapshots are replicated from the current array to an external storage system such as an NFS device or S3 bucket. For replication to offload targets, only volume members can be added to protection groups. For more information about offload targets, refer to `pureoffload(1)` [371].

Protection group snapshots cannot be connected to hosts or host groups for writing and reading. However, the volumes within a protection group snapshot, which are visible in both the Storage and Protection tabs, can be copied to new or existing live, host-accessible volumes.

## Creating Protection Groups

Run `purepgroup create` on the source array (source) to create a new protection group.

For asynchronous replication, include the `--hostlist`, `--hgroupelist`, or `--vollist` option to add hosts, host groups, or volumes, respectively, as members to the protection group. Only members of the same object type can belong to a protection group. For example, hosts or host groups cannot be added to a protection group that contains volumes. For replication to an offload target, include the `--vollist` option to add volumes as members to the protection group.

To replicate snapshots, the protection group must include at least one target array or offload (target) to where the replicated data is written. Include the `--targetlist` option to add the targets to the protection group. Once the targets have been added to the protection group, the source array must connect to each of the targets. To connect to a target array, run the command `purearray connect`. For more information about connecting arrays, refer to `puregroup-setattr(1)` [397]. To connect to an NFS offload target, run the command `pureoffload nfs connect`. To connect to an S3 offload target, run the command `pureoffload s3 connect`. For more information about connecting to an NFS or S3 offload target, refer to `pureoffload(1)` [371].

Once a protection group has been created with members and targets (for replication only), it is ready to generate and replicate snapshots.

## Renaming Protection Groups

Run `purepgroup rename` to change the current name (OLD-NAME) of the protection group to the new name (NEW-NAME). The name change is effective immediately and the old name is no longer recognized in CLI, GUI, or REST interactions. In the Purity//FA GUI, the new name appears upon page refresh.

Protection group snapshots cannot be renamed.

## Destroying Protection Groups

Protection groups can be destroyed, eradicated, and recovered by the respective `purepgroup` command.

The `purepgroup destroy` command destroys the specified protection group and, implicitly, all of its snapshots. Once a protection group has been destroyed, the replication process for the protection group stops. Destroyed protection groups and their snapshots undergo a 24-hour eradication pending period after which time the protection groups and their snapshots are completely eradicated and unrecoverable. Run `purepgroup list --pending` to display a list of protection groups and snapshots, including those that are pending eradication. Run `purepgroup list --pending-only` to only display a list of protection groups and their snapshots that are pending eradication.

During the eradication pending period, the destruction of the protection group and snapshots can be cancelled by running `purepgroup recover`, returning the protection group, including its replication schedule and snapshots, to its original state.

During the eradication pending period, the destroyed protection group and its snapshots can be eradicated by running `purepgroup eradicate`. This command terminates the eradication pending period and immediately begins reclamation of physical storage occupied by data "charged" to the snapshots of the destroyed protection group. Once eradication has begun, a protection group and its snapshots can no longer be recovered.

## Generating Protection Group Snapshots On Demand

There are two ways to generate and replicate protection group snapshots:

- **Scheduled.** Schedule the protection group to automatically generate local snapshots and/or replicate snapshot data to its targets on a regular, pre-defined basis.  
Configure protection group schedules through the `purepgroup schedule` command. For steps on how to configure a protection group snapshot and replication schedule, refer to `purepgroup-setattr(1)` [397].  
View protection group schedules through the `purepgroup list --schedule` command. For more information about viewing protection group attributes and schedules, refer to `purepgroup-list(1)` [389].
- **On Demand.** Manually generate local protection group snapshots or replicate protection group snapshot data to its targets.

An on-demand snapshot represents a snapshot that is manually generated by running `purepgroup snap`. If the `--replicate-now` option is used, the on-demand snapshot is replicated immediately to the targets. Alternatively, if the `--replicate` option is used, the on-demand snapshot is queued for replication. Purity//FA will begin replicating data to each target only when all earlier replication sessions for the same protection group have been completed to that target, excluding those started with `--replicate-now`.

By default, an on-demand snapshot is retained indefinitely or until it is manually destroyed. Include the `--apply-retention` option to apply the scheduled snapshot or replication retention policy to the on-demand snapshot. For example, when generating a local snapshot on demand, include the `--apply-retention` option to keep the snapshot on the source array for the length of time based

on the snapshot retention schedule. Likewise, when replicating a snapshot on demand, include the **--apply-retention** option to keep the snapshot on the target for the length of time based on the replication retention schedule.

To replace the **NNN** snapshot number in the snapshot name with a custom string, include the **--suffix** option in the **purepgroup snap** command.

## Destroying Protection Group Snapshots

Destroy a protection group snapshot if it is no longer required. Run **purepgroup destroy** to destroy the specified protection group snapshot.

Destroying a protection group snapshot destroys all of its protection group volume snapshots, thereby reclaiming the physical storage space occupied by its data.

Destroyed protection group snapshots follow the same eradication pending behavior as destroyed protection groups. When a protection group snapshot is destroyed, Purity//FA automatically takes an undo snapshot. The undo snapshot enters a 24-hour eradication pending period, after which time the snapshot is eradicated. During the 24-hour pending period, the undo snapshot can be viewed, recovered, or permanently eradicated.

Protection group volume snapshots cannot be destroyed individually. A protection group volume snapshot can only be destroyed by destroying the protection group snapshot to which it belongs.

## Restoring Volumes from a Target Array

The **purepgroup copy** command restores the state of the volumes within a protection group that resides on a target array to a previous protection group snapshot. The restored volumes are added as real volumes to a new or existing protection group. To restore volume snapshots that reside on an offload target, run the **purevol get --on** command. For more information about restoring volumes from an offload target, refer to **purevol(1)** [469].

The **purepgroup copy** command requires source (**SOURCE**) and destination (**DEST**) arguments.

The source argument represents the protection group snapshot to be restored. To restore a specific protection group snapshot, include the full protection group snapshot name in the source argument. For example, if source is specified as **pgroup1.5213**, Purity//FA restores the volumes from protection snapshot **pgroup1.5213**. To restore volumes from the most recent, fully generated protection group snapshot (either locally generated or replicated from another array), specify the protection group name only. For example, if source is specified as **pgroup1**, Purity//FA restores the volumes from the latest, fully generated protection group snapshot of **pgroup1**.

The source argument also determines whether the protection group snapshot to be restored was generated locally or replicated from another array. If the protection group snapshot was replicated from another array, include the name of the array in the source argument. For example, if source is specified as **array1:pgroup1.5213**, Purity//FA restores the volumes from protection group snapshot **pgroup1.5213**, which was replicated over from **array1**.

The destination argument represents the name of the protection group to where the volumes will be restored. If the destination protection group and all of its volumes already exist, include the **--overwrite** option to overwrite all of the existing volumes with the snapshot contents. When using

the **--overwrite** option, the names of the volumes that are being overwritten must match the names of the volumes that are being restored.

The following restrictions apply to the **purepgroup copy** command:

- You cannot restore a volume if it already exists on the array unless you include the **--overwrite** option.
- You cannot overwrite destination protection groups that contain hosts, host groups, or connected volumes.
- You cannot run the **purepgroup copy --overwrite** command to create multiple clones of the protection group or its volume snapshots.

Restoring volumes from a protection group snapshot does not automatically expose the restored volumes to hosts and host groups. Run the **purehgroup connect**, **purehost connect** or **purevol connect** command to establish connections.

If the protection group snapshot being restored is one that was replicated from another array, Purity//FA automatically includes the original source array as a target array in the new protection group.

After the volumes are restored, the created date of the volumes is set to the date of the **purepgroup copy** action.

## Examples

### Example 1

```
purepgroup create --vollist vol1,vol2,vol3 pgroup1
```

Creates protection group **pgroup1** and associates volumes **vol1,vol2,vol3** with it.

### Example 2

```
purepgroup create --vollist vol5,vol6 --targetlist nfs-target pgroup2
```

Creates protection group **pgroup2** with volumes **vol5** and **vol6** and offload target **nfs-target**.

### Example 3

```
purepgroup destroy pgroup12
```

*... less than 24 hours passes...*

```
purepgroup recover pgroup12
```

Destroys protection group **pgroup12** and its snapshots and then recovers the destroyed protection group and its snapshots.

### Example 4

```
purepgroup destroy pgroup5.10
```

... less than 24 hours passes...

```
purepgroup eradicate pgroup5.10
```

Destroys protection group snapshot **pgroup5.10** and then eradicates the destroyed snapshot. Purity//FA immediately begins reclaiming the physical storage space occupied by the snapshot. Protection group snapshot **pgroup5.10** can no longer be recovered.

#### Example 5

```
purepgroup snap --replicate-now pgroup10
```

Generates an on-demand snapshot of all volumes, hosts, or host groups in protection group **pgroup10** and replicates the snapshot to all of the **pgroup10** targets.

#### Example 6

```
purepgroup copy pgroup1 pgroup6
purevol connect --host host01 vol01 vol02
```

Restores the volumes from the most recent, fully and locally generated protection group snapshot of protection group **pgroup1** to a new protection group named **pgroup6** on the same array, and then connects newly restored volumes **vol01** and **vol02** to host **host01**.

#### Example 7

```
purepgroup copy array1:pgroup2.5213 pgroup25
```

Restores the volumes from protection group snapshot **pgroup2.5213**, which was replicated over from source array **array1**. The restored volumes are saved as real volumes to new protection group **pgroup25** on the current array. To replicate new protection group **pgroup25** and its restored volumes from the current array back to the original source array, enable replication and then allow replication from the source array.

#### Example 8

```
purepgroup copy --overwrite pgroup5.4765 pgroup5
```

Restores the volumes from locally generated protection group snapshot **pgroup1.123** to existing protection group **pgroup1**, overwriting all of the existing volumes with the snapshot contents.

### See Also

[purepgroup-list\(1\)](#) [389], [purepgroup-setattr\(1\)](#) [397]

[purehgroup\(1\)](#) [275], [purehost\(1\)](#) [292], [purevol\(1\)](#) [469], [purepod\(1\)](#) [412]

### Author

Pure Storage Inc. <[documentfeedback@purestorage.com](mailto:documentfeedback@purestorage.com)>

## purepgroup-list

purepgroup-list, purepgroup-listobj — display protection group attributes and storage consumption

### Synopsis

```
purepgroup list [--on OFFLOAD-TARGET] [ --pending | --pending-only ] [ --  
retention | --schedule | --snap | --space ] [--sort SORT] [ --source | --target ]  
[--total] [--transfer] [ --cli | --csv | --nvp ] [--filter FILTER] [--limit LIMIT]  
[--notitle] [--page] [--raw] [PGROUP...]  
  
purepgroup listobj [ --pending | --pending-only ] [ --source | --target ] [ --type  
{ --hgroup | --host | --pgroup | --snap | --source | --target | --vol } ] [--csv]  
[PGROUP...]
```

### Arguments

#### **PGROUP**

Protection group for which the information specified by options is to be displayed.

### Options

Options that control information displayed:

#### -h | --help

Can be used with any command or subcommand to display a brief syntax description.

#### --on *OFFLOAD-TARGET*

Displays a list of protection groups both local and remote to the array that are connected to the offload target.

#### --pending

Displays a list of protection groups or snapshots that have been destroyed and are in the eradication pending state.

#### --pending-only

Only display destroyed protection groups or snapshots that are in the eradication pending state.

#### --retention

Displays the retention schedule for each protection group.

#### --schedule

Displays the snapshot and asynchronous replication schedule for each protection group.

#### --snap

Displays a list of protection group snapshots taken in each source and target protection group.

When used with **--on**, displays a list of protection group snapshots, both local and remote to the array, that have been replicated and retained on the offload target.

--source  
 Displays a list of protection groups that were created on this array. When used with **--snap**, displays a list of protection group snapshots that were created and retained on this array.

--space  
 Displays the size and space consumption details for all snapshots in each protection group.

--target  
 Displays a list of protection groups or snapshots that were replicated to this array.

--total  
 Used with **--space** to display total physical space occupied by snapshot data for each protection group.

--transfer  
 Used with **--snap** to display asynchronous replication data transfer statistics, including data transfer start time, data transfer end time, data transfer progress, and amount of logical/physical data transferred.

--type  
 Displays a list of hosts, host groups, protection groups, snapshots, sources, targets or volumes associated with the specified protection groups.

Options that control display format:

--cli  
 Displays output in the form of CLI commands that can be issued to reproduce the current configuration. The **--cli** output is not meaningful when combined with immutable attributes.

--csv  
 Lists information in comma-separated value (CSV) format. The **--csv** output can be used for scripting purposes and imported into spreadsheet programs.

--notitle  
 Lists information without column titles.

--nvp  
 Lists information in name-value pair (NVP) format, in the form **ITEMNAME=VALUE**. Argument names and information items are displayed flush left. The **--nvp** output is designed both for convenient viewing of what might otherwise be wide listings, and for parsing individual items for scripting purposes.

--page  
 Turns on interactive paging.

--raw  
 Displays the unformatted version of column titles and data. For example, in the **purearray monitor** output, the unformatted version of column title **us/op (read)** is **usec\_per\_read\_op**. The **--raw** output is used to sort and filter list results.

Options that manage display results:

--filter

Displays only the rows that meet the filter criteria specified.

--limit

Limits the size of the list output to the specified maximum number of rows.

--sort

Sorts the list output in ascending or descending order by the column specified.

## Conventions

The protection group snapshot naming convention is **PGROUP.NNN**, where:

- **PGROUP** is the name of the protection group.
- **NNN** is a unique monotonically increasing number or a manually-assigned protection group snapshot suffix name.

The protection group volume snapshot naming convention is **PGROUP.NNN.VOL**, where:

- **PGROUP** is the name of the protection group.
- **NNN** is a unique monotonically increasing number or a manually-assigned protection group snapshot suffix name.
- **VOL** is name of the volume member.

If you are viewing replicated snapshots on a target array, the snapshot name begins with the name of the source array from where the snapshot was taken.

## Description

The **purepgroup list** command displays information about each protection group, including their associated source arrays, replication targets, hosts, host groups, and volumes. Optionally add the protection group argument to any of the commands to view details for the specified protection groups. The list includes protection groups that were created on the current array to replicate snapshot data to other arrays or offload targets, created on a remote array and replicated asynchronously to this array, or created inside a pod on a remote array and stretched to the current array.

The following **purepgroup list** sample output generated on array **array1** illustrates the different ways in which protection groups can be listed:

- Protection group **pgroup05** was created on array **array5**, and the current array was added to the protection group as a target array for asynchronous replication.
- Protection group **pgroup1** was created on the current array, and **array2** was added as a target array for asynchronous replication.
- Protection group **pgroup10** was created on the current array and **array3** was added as a target array for asynchronous replication. However, **array3** has disallowed asynchronous replication.
- Protection group **pgroup20** was created on the current array, and **nfs-target** was added as an offload target.
- Protection group **pod05::pgroup01** is on the current array. The protection group is in pod **pod05**, which was either created on the current array or created on another array and stretched to this array. The pod may or may not be stretched across another array.
- Protection group **pod05::pgroup02** is on the current array. The protection group is in pod **pod05**, which was either created on the current array or created on another array and stretched to this array. The pod may or may not be stretched across another array. Array **array4** was added to the protection group as a target array for asynchronous replication. If the pod is stretched across another array and both arrays in the stretched pod are connected to the target array, both arrays will share the load of replicating data to the target array.
- Protection group **pod25::pgroup100** is on a remote array, and the current array was added to the protection group as a target array for asynchronous replication. (On the target array, the fully qualified name of protection groups inside pods is **POD:PGROUP**, with single colons (:) separating the pod name and volume name.) The protection group is inside pod **pod25**, which may or may not be stretched across another array. If the pod is stretched across another array and both arrays in the stretched pod are connected to the target array, both arrays will share the load of replicating data to the current (target) array.

```
//From array1
$ purepgroup list
```

Name	Source	Targets	Host Groups	Hosts	Volumes
array5:pgroup05	array5	-	-	-	array:vol1
pgroup1	array1	array2	-	-	v1 v2 v3

<b>pgroup10</b>	<b>array1</b>	<b>array3 (disallowed)</b>	<b>hg1</b>	-	-
<b>pgroup20</b>	<b>array1</b>	<b>nfs-target</b>	-	-	<b>v17</b>
					<b>v18</b>
<b>pod05::pgroup01</b>	<b>pod05</b>	-	-	-	<b>v60</b>
<b>pod05::pgroup02</b>	<b>pod05</b>	<b>array4</b>	-	-	<b>v50</b>
			-	-	<b>v51</b>
			-	-	<b>v52</b>
<b>pod25:pgroup100</b>	<b>pod25</b>	<b>array1</b>	-	-	<b>v85</b>
			-	-	<b>v92</b>

The `purepgroup list` command includes the mutually exclusive `--schedule`, `--retention`, and `--space` options.

The `--schedule` option displays the snapshot and asynchronous replication schedule for each protection group.

In the following sample output, the snapshot and asynchronous replication schedules have been enabled for protection group `pgroup1`. The protection group has been configured to enable snapshot generation every hour and replication every 2 hours. Replication is suspended during the blackout period between 9am and 5pm. Replication from the source to targets will not occur during the blackout period, but hourly snapshots will continue to be taken on the source array.

The snapshot schedule for protection group `pgroup05` on source array `array5` has also been configured to enable snapshot generation every hour. However, replication has been disabled. Once enabled, snapshot replication will occur every 2 days at or soon after 6pm.

\$ purepgroup list --schedule						
Name	Schedule	Enabled	Frequency	At	Blackout	
<code>array5:pgroup05</code>	snap	True	6h	-	-	
	replicate	False	2d	6pm	-	
<code>pgroup1</code>	snap	True	1h	-	-	
	replicate	True	2h	6pm	9am-5pm	

The `--retention` option displays the retention schedule for each protection group.

In the following sample output, the retention schedule has been set to keep all snapshots on the source array for 1 day. After that, Purity//FA will keep 2 of the snapshots for an additional 7 days and eradicate the others. Seven days later, Purity//FA will eradicate the 2 snapshots.

The retention schedule has also been set to keep all replicated snapshots on the targets for 7 days. After that, Purity//FA will keep 4 of the replicated snapshots for an additional 7 days and eradicate the others. Seven days later, Purity//FA will eradicate the 4 snapshots.

\$ purepgroup list --retention					
Name	Array	All	For	Per Day	Days
<code>pgroup1</code>	source	1d	2	7	
	target	7d	4	7	

The **--space** option displays the size and space consumption details for all snapshots in each protection group. Include the **--total** option to display the total space consumption for all protection groups.

The **--snap** option displays a list of scheduled and on-demand snapshots generated or replicated of each protection group. The list also displays the snapshot creation date. Include the **--transfer** option to display asynchronous replication data transfer statistics, including data transfer start time, data transfer end time, data transfer progress, and amount of logical/physical data transferred.

Include the **--on** option to display a list of protection groups both local and remote to the array that are connected to the offload target. Add the **--snap** option to display a list of protection group snapshots, both local and remote to the array, that have been replicated and retained on the offload target.

The **--pending** option includes protection groups or snapshots that have been destroyed and are in the eradication pending state in the output list. The **--pending** option can also be used in conjunction with the **--schedule**, **--retention**, **--space**, or **--snap** options to include a list of pending protection groups or snapshots.

The **--pending-only** option only displays protection groups or snapshots that have been destroyed and are in the eradication pending state in the output list. The **--pending-only** option can also be used in conjunction with the **--schedule**, **--retention**, **--space**, or **--snap** options to include a list of pending protection groups or snapshots.

The **puregroup listobj** command displays a list of protection groups. Include the protection group argument to view the list for the specified protection group.

Include the **--type** option to view a list of hosts, host groups, snapshots, sources, targets or volumes associated with a protection group. The **--type** option produces the following types of lists:

**--type hgroup**

Displays a list of host groups that belong to a protection group.

**--type host**

Displays a list of hosts that belong to a protection group.

**--type pgroup (default if --type option not specified)**

Displays a list of protection groups.

**--type snap**

Displays a list of snapshots that were taken for a protection group.

**--type source**

Displays a list of replication source arrays.

**--type target**

Displays a list of replication target arrays and offload targets.

**--type vol**

Displays a list of volumes that belong to a protection group.

The **--source** option displays a list of protection groups or snapshots that were created on this array.

The **--target** option displays a list of protection groups or snapshots that were replicated to this array from another array.

## Exceptions

None.

## Examples

### Example 1

```
purepgroup list
```

Displays information about each protection group, including their source arrays, targets, hosts, host groups, and volumes.

### Example 2

```
purepgroup list --pending --snap
```

Displays a list of snapshots, both on-demand and scheduled, including those which have been destroyed and are in the eradication pending state.

### Example 3

```
purepgroup list --pending-only --snap
```

Displays a list of snapshots, both on-demand and scheduled, that have been destroyed and are in the eradication pending state.

### Example 4

```
purepgroup list --schedule pgroup4
```

Displays the snapshot and asynchronous replication schedule for protection group **pgroup4**.

### Example 5

```
purepgroup list --on offload-target
```

Displays a list of protection groups both local and remote to the array that are connected to offload target **offload-target**.

### Example 6

```
purepgroup list --on s3-target --snap
```

Displays a list of snapshots, both local and remote to the array, that have been replicated and retained on offload target **s3-target**.

**Example 7**

```
purepgroup listobj
```

Displays a list of protection groups.

**Example 8**

```
purepgroup listobj --type vol pgroup7 pgroup10
```

Displays volumes that belong to protection group `pgroup7`, `pgroup10`, or both.

**Example 9**

```
purepgroup listobj --type snap --target
```

Displays protection group snapshots replicated to this array.

**See Also**

`pureoffload(1)` [371] `purepgroup(1)` [381], `purepgroup-setattr(1)` [397] `purepod(1)` [412]

**Author**

Pure Storage Inc. <[documentfeedback@purestorage.com](mailto:documentfeedback@purestorage.com)>

## purepgroup-setattr

purepgroup-allow, purepgroup-disable, purepgroup-disallow, purepgroup-enable, purepgroup-schedule, purepgroup-setattr — manage the attributes of the Purity//FA protection group (pgroup).

### Synopsis

```
purepgroup allow PGROUP...  
purepgroup disable [ --replicate | --snap ] PGROUP...  
purepgroup disallow PGROUP...  
purepgroup enable [ --replicate | --snap ] PGROUP...  
purepgroup retain [--all-for PERIOD] [--per-day SNAPS-PER-DAY] [--days COUNT]  
[--target-all-for PERIOD] [--target-per-day SNAPS-PER-DAY] [--target-days COUNT]  
PGROUP...  
purepgroup schedule [--snap-frequency FREQUENCY] [--snap-at TIME] [--replicate-  
frequency FREQUENCY] [--replicate-at TIME] [--replicate-blackout WINDOW]  
PGROUP...  
purepgroup setattr [--addhgrouplist HGROUPS] [--hgrouplist HGROUPS] [--  
remhgrouplist HGROUPS] [--addhostlist HOSTS] [--hostlist HOSTS] [--remhostlist  
HOSTS] [--addtargetlist TARGETS] [--targetlist TARGETS] [--remtargetlist TARGETS]  
[--addvollist VOLS] [--vollist VOLS] [--remvollist VOLS] PGROUP...
```

### Arguments

#### **PGROUP**

Protection group name. For **purepgroup allow** and **purepgroup disallow**, array name and protection group name of the source array and protection group from which to allow or disallow asynchronous replication.

### Options

#### **-h | --help**

Can be used with any command or subcommand to display a brief syntax description.

#### **--addhgrouplist**

Comma-separated list of one or more additional host groups to be included in the specified protection groups. Has no effect on host groups already associated with the group. Hosts and host groups are not supported for replication to offload targets.

#### **--addhostlist**

Comma-separated list of one or more additional hosts to be included in the specified protection groups. Has no effect on hosts already associated with the group. Hosts and host groups are not supported for replication to offload targets.

--addtargetlist

Comma-separated list of one or more additional targets to be included in the specified protection groups. Has no effect on targets already associated with the group.

--addvollist

Comma-separated list of one or more additional volumes to be included in the specified protection groups. Has no effect on volumes already associated with the group.

--all-for

Length of time to keep the snapshots on the source array before they are eradicated.

--days

Number of days to keep the --per-day snapshots beyond the --all-for period before they are eradicated.

--hgrouplist

Comma-separated list of one or more host groups to be included in the specified protection groups, replacing any host groups that currently belong to the protection group. Hosts and host groups are not supported for replication to offload targets.

--hostlist

Comma-separated list of one or more hosts to be included in the specified protection groups, replacing any hosts that currently belong to the protection group. Hosts and host groups are not supported for replication to offload targets.

--per-day

Number of per-day snapshots to keep beyond the --all-for period.

--remhgrouplist

Comma-separated list of one or more host groups to be removed from the specified protection group.

--remhostlist

Comma-separated list of one or more hosts to be removed from the specified protection group.

--remtargetlist

Comma-separated list of one or more targets whose associations with the specified protection group are to be broken.

--remvollist

Comma-separated list of one or more volumes to be removed from the specified protection group.

--replicate

Enables or disables replication.

--replicate-at

Preferred time, on the hour, at which to replicate the snapshots. To clear the preferred time, set to an empty string ("").

--replicate-blackout  
Range of time at which to suspend replication. To clear the blackout period, set to an empty string ("").

--replicate-frequency  
Replication frequency.

--snap  
Enables or disables snapshot creation on the source array.

--snap-at  
Preferred time, on the hour, at which to generate the snapshot. To clear the preferred time, set to an empty string ("").

--snap-frequency  
Snapshot frequency.

--target-all-for  
Length of time to keep the replicated snapshots on the targets.

--target-days  
Number of days to keep the --target-per-day snapshots beyond the --target-all-for period before they are eradicated.

--target-per-day  
Number of per-day snapshots to keep beyond the --target-all-for period.

--targetlist  
Comma-separated list of one or more targets to be associated with the protection group. When specified in the purepgroup setattr command, replaces the existing targets of the protection group.

--vollist  
Comma-separated list of one or more volumes to be included in the specified protection groups, replacing any volumes that currently belong to the protection group.

## Conventions

The protection group snapshot naming convention is **PGROUP.NNN**, where:

- **PGROUP** is the name of the protection group.
- **NNN** is a unique monotonically increasing number or a manually-assigned protection group snapshot suffix name.

The protection group volume snapshot naming convention is **PGROUP.NNN.VOL**, where:

- **PGROUP** is the name of the protection group.
- **NNN** is a unique monotonically increasing number or a manually-assigned protection group snapshot suffix name.
- **VOL** is name of the volume member.

If you are viewing replicated snapshots on a target array, the snapshot name begins with the name of the source array from where the snapshot was taken.

## Description

After a protection group has been created, configure the schedules to generate and replicate snapshots to another FlashArray array or to an external storage system. For more information about creating protection groups, refer to [purepgroup\(1\)](#) [381].

To configure the protection group snapshot schedule:

1. Define the snapshot schedule.
2. Define the retention schedule.
3. Enable protection group snapshots.

To configure the protection group replication schedule:

1. Connect the source and targets.  
To connect the source array to a target array, run the `purearray connect` command. If the protection group is in a stretched pod, for high availability, connect both arrays in the stretched pod to the target array. For more information about array connections, refer to [purearray\(1\)](#) [230].  
To connect the source array to a storage system, such as an NFS device or S3 bucket, run the respective `pureoffload nfs connect` or `pureoffload s3 connect` command. If the protection group is in a stretched pod, for high availability, connect both arrays in the stretched pod to the offload target. For more information about offload targets, refer to [pureoffload\(1\)](#) [371].
2. For asynchronous replication, verify that each target array has allowed replication.
3. Define the replication schedule.
4. Define the retention schedule.
5. Enable protection group replication.

The snapshot and replication schedules can be enabled and disabled at any time.

Note that protection group snapshots can also be generated and replicated on demand. For more information about generating on-demand snapshots, refer to [purepgroup\(1\)](#) [381].

## Connecting Arrays for Asynchronous Replication

Replication between a source and target array can only occur if the arrays are connected and the target array has allowed replication.

To connect a source array to a target array, log in to the target array and run `purearray connect --connection-key` to obtain a connection key, and then log in to the source array and run `purearray connect` to connect the target array to the source array. A source array can connect to multiple target arrays. For more information about connecting arrays, refer to [purearray\(1\)](#) [230].

Once two arrays are connected, run `purepgroup list` on either array to verify that the target array has allowed data to be replicated to it. If an array has disallowed replication, (`disallowed`) appears next to its name. To allow replication, run `purepgroup allow` on the target array.

The `purepgroup disallow` command stops replication to the target array. Allowing and disallowing replication on a target array will not impact the replication process between the source array and other

target arrays. If you disallow replication while a replication session is in progress, Purity//FA will wait until the session is complete and then stop any new replication sessions from being created.

When specifying the `purepgroup allow` and `purepgroup disallow` commands, include the source array name and protection group name in the argument, separating the two names with a colon (:).

In the following example, the array is allowing data to be replicated from protection group `PGROUP1` on array `ARRAY1`.

```
purepgroup allow ARRAY1:PGROUP1
```

## Defining the Protection Group Snapshot and Replication Schedules

Once a protection group has been created and connections between the source and targets have been established (for replication only), define the protection group snapshot and replication schedules.

Each protection group includes two schedules:

- **Snapshot Schedule.** Defines when and how frequently snapshots are taken and saved on the local (source) array. Run `purepgroup schedule --snap-frequency` to configure the snapshot schedule.
- **Replication Schedule.** Defines when and how frequently snapshots are taken and immediately replicated to its targets. Run `purepgroup schedule --replicate-frequency` to configure the replication schedule.

The snapshot schedule is independent of the replication schedule, meaning you can enable one schedule without enabling the other. You can also enable or disable both schedules at any time. The schedules are by default disabled.

The `purepgroup schedule` command configures the snapshot and replication schedules for the protection group. By default, the protection group schedules are configured to generate snapshots every hour and replicate snapshots to the targets every 4 hours.

### Snapshot Schedule

The `purepgroup schedule --snap-frequency` command specifies how frequently Purity//FA generates snapshots on the source array.

The frequency is specified as an integer, followed by one of the suffix letters `s` (seconds), `m` (minutes), `h` (hours), `d` (days), or `w` (weeks).

For example, run the following command to generate snapshots every 2 hours for protection group `pgroup1`.

```
purepgroup schedule --snap-frequency 2h pgroup1
```

If the `--snap-frequency` option is set to one or more days, set the `--snap-at` option to specify the preferred time, on the hour, at which to generate the snapshot on the source array. If the preferred time is set and you want to cancel it, meaning that you no longer want to specify a preferred time, set the `--snap-at` option to an empty string ("").

The time can be specified in 12-hour or 24-hour format. To specify the time in 12-hour format, enter the time integer followed by the suffix **am** or **pm**.

For example, run the following command to set the preferred snapshot time for protection group **pgroup3** to 6:00pm. A snapshot will be generated every 2 days at or soon after 6pm.

```
purepgroup schedule --snap-frequency 2d --snap-at 18 pgroup3
```

#### Replication Schedule

The **purepgroup schedule --replicate-frequency** command specifies how frequently Purity//FA replicates the snapshots from the source array to the targets.

The frequency is specified as an integer, followed by one of the suffix letters **s** (seconds), **m** (minutes), **h** (hours), **d** (days), or **w** (weeks).

For example, run the following command to replicate snapshots to the targets every 6 hours for protection group **pgroup2**.

```
purepgroup schedule --replication-frequency 6h pgroup2
```

If the **--replicate-frequency** option is set to one or more days, set the **--replicate-at** option to specify the preferred time, on the hour, at which to replicate the snapshots to the targets. If the preferred time is set and you want to cancel it, meaning that you no longer want to specify a preferred time, set the **--replicate-at** option to an empty string ("").

The time can be specified in 12-hour or 24-hour format. To specify the time in 12-hour format, enter the time integer followed by the suffix **am** or **pm**.

For example, run the following command to set the preferred replication time for protection group **pgroup3** to 6:00pm. Snapshot replication will occur every 2 days at or soon after 6pm.

```
purepgroup schedule --replicate-frequency 2d --replicate-at 18 pgroup3
```

The **--replicate-blackout** option (blackout period) specifies the range of time in which to suspend replication. Replication from the source to targets will not occur during the blackout period.

The start and end times for the blackout window must be set on the hour. The time can be specified in 12-hour or 24-hour format. To specify the time in 12-hour format, enter the time integer followed by the suffix **am** or **pm**.

For example, run the following command to suspend replication between 9:00am and 5:00pm for protection group **pgroup4**.

```
purepgroup schedule --replicate-blackout 9am-5pm pgroup4
```

Blackout periods only apply to scheduled replications. On-demand replications (**purepgroup snap --replicate-now**) do not observe the blackout period.

If the blackout period is set and you want to cancel it, meaning that you no longer want to specify a blackout period, set the **--replicate-blackout** option to an empty string ("").

## Defining the Protection Group Retention Schedule

The **purepgroup retain** command configures the snapshot retention schedule for the specified protection group.

By default, protection groups are scheduled to retain all snapshots for 1 day. After that, Purity//FA keeps 4 snapshots for an additional 7 days and eradicates the others. Seven days later, Purity//FA eradicates the 4 snapshots.

The following sample retention schedule output reflects the protection group schedule with the default schedule settings:

```
$ purepgroup list --retention
Name      Array    All For  Per Day  Days
pgroup1   source   1d      4        7
          target   1d      4        7
```

The **--all-for** option specifies the length of time to keep the snapshots on the source array before they are eradicated. The length of time is specified as an integer, followed by one of the suffix letters **s** (seconds), **m** (minutes), **h** (hours), **d** (days), or **w** (weeks).

For example, run the following command to keep all snapshots on the source array for for 5 days.

```
purepgroup retain --all-for 5d pgroup5
```

The **--per-day** option specifies the number of per-day snapshots to keep beyond the **--all-for** period, while the **--days** option specifies the number of days to keep the **--per-day** snapshots beyond the **--all-for** period. After the **--days** period, the snapshots are eradicated.

For example, run the following command to keep 8 snapshots per day for an additional 5 days on the source array *after* the **--all-for** period.

```
purepgroup retain --per-day 8 --days 5 pgroup6
```

The **--target-all-for** option specifies the length of time to keep the replicated snapshots on the targets before they are eradicated.

The length of time is specified as an integer, followed by one of the suffix letters **s** (seconds), **m** (minutes), **h** (hours), **d** (days), or **w** (weeks).

For example, run the following command to keep all replicated snapshots on the targets for 7 days.

```
purepgroup retain --target-all-for 7d pgroup7
```

The **--target-per-day** option specifies the number of per-day snapshots to keep beyond the **--target-all-for** period, while the **--target-days** option specifies the number of days to keep the **--target-per-day** snapshots beyond the **--target-all-for** period. After the **--target-days** period, the replicated snapshots are eradicated.

For example, run the following command to keep 4 replicated snapshots per day for an additional 3 days on the target *after* the **--target-all-for** period.

```
purepgroup retain --target-per-day 4 --target-days 3 pgroup8
```

## Enabling Protection Group Snapshots and Replication

Once the protection group snapshot and/or replication schedules have been defined, run **purepgroup enable** to enable snapshots and replications on the source array for the specified protection groups. Specify the **--snap** option to enable snapshots only. Specify the **--replicate** option to enable replication only. If the command is run without either option, Purity//FA enables both snapshots and replication.

Once you enable the snapshot schedule, Purity//FA immediately starts the snapshot process, with the following exception:

- If you are enabling the snapshot schedule and the **--snap-at** time is specified, Purity//FA starts the snapshot process at the specified **--snap-at** time.

Once you enable the replication schedule, Purity//FA immediately starts the replication process, with the following exceptions:

- If you are enabling the replication schedule during the blackout period, Purity//FA waits for the blackout period to end before it begins the snapshot and/or replication process.
- and the **--replicate-at** time is specified, If you are enabling the replication schedule Purity//FA starts the replication process at the specified **--replicate-at** time.

The **purepgroup disable** command disables snapshots and replication for the specified protection group. Specify the **--snap** option to disable snapshots only. Specify the **--replicate** option to disable replication only. If the command is run without either option, Purity//FA disables both snapshots and replication.

## Space Consumption Considerations

Consider space consumption when you configure the snapshot, replication, and retention schedules. The amount of space consumed on the source array depends on how many snapshots you want to generate, how frequently you want to generate the snapshots, how many snapshots you want to retain, and how long you want to retain the snapshots.

Likewise, the amount of space consumed on the target depends how many snapshots you want to replicate, how frequently you want to replicate the snapshots, how many replicated snapshots you want to retain, and how long you want to retain them.

In the following sample snapshot, replication, and retention schedules, the settings in Scenario 1 will consume more space than the settings in Scenario 2:

### SCENARIO 1

```
$ purepgroup list --schedule
Name      Schedule  Enabled  Frequency  At      Blackout
array1:pg1  snap     False    15m        -      -
              replicate False   1h         6pm     -
```

```
$ purepgroup list --retention
```

```
Name      Array   All For Per Day Days
pgroup1  source   2w     24    30
          target   2w     24    30
```

#### SCENARIO 2

```
$ purepgroup list --schedule
Name      Schedule Enabled Frequency At Blackout
array1:pg1 snap     False    1h      -   -
              replicate False    4h      6pm  -
```

```
$ purepgroup list --retention
Name      Array   All For Per Day Days
pgroup1  source   1d     4      7
          target   1d     4      7
```

### Adding and Removing Protection Group Members

The **purepgroup setattr** command adds members to a protection group or removes members from a protection group.

For asynchronous replication, only members of the same object type can belong to a protection group. For example, hosts or host groups cannot be added to a protection group that contains volumes. The **--addhgrouplist**, **--addhostlist**, and **--addvollist** options add one or more members to a protection group. Existing members in the protection group are unaffected. The **--hgrouplist**, **--hostlist**, and **--vollist** options add one or more members to a protection group, replacing all members that are currently in the protection group. The **--remhgrouplist**, **--remhostlist**, and **--remvollist** options remove members from a protection group.

For replication to offload targets, only volume members can be added to protection groups. The **--addvollist** option adds one or more volume members to a protection group. Existing members in the protection group are unaffected. The **--vollist** option adds one or more members to a protection group, replacing all volume members that are currently in the protection group. The **--remvollist** option removes volume members from a protection group.

Note that an alternate (and preferred!) method to add and remove protection group members is through the **add** and **remove** subcommands. Run the **add** subcommand with the respective **purehgroup**, **purehost**, or **purevol** command to add one or more members to one or more protection groups. For example, run **purehgroup add --pgroup PGROUP HGROUP** to add host groups to protection groups. Run the **remove** subcommand with the respective **purehgroup**, **purehost**, or **purevol** command to remove one or more members from one or more protection groups. For example, run **purevol remove --pgroup PGROUP VOL** to remove volumes from protection groups.

### Adding and Removing Target Arrays

The **purepgroup setattr** command adds target arrays to protection groups, and removes target arrays from protection groups.

The **--addtargetlist** option adds one or more target arrays to a protection group. Existing target arrays in the protection group are unaffected. Only arrays that are connected to the current array can

be added to a protection group as target arrays. Array connections are created through the **purearray connect** command.

If you add a target array to a protection group that is in a stretched pod, for high availability, make sure the peer array of the stretched pod is also connected to the target array. If only one of the arrays is connected to the target array, Purity//FA generates an alert notifying users of this misconfiguration.

The **--targetlist** option adds one or more target arrays to a protection group, replacing all target arrays that are currently in the protection group.

The **--remtargetlist** option removes target arrays from a protection group.

## Examples

### Example 1

```
purepgroup enable pgroup4
```

(Run from the source array)...

Enables snapshots and replication for protection group **pgroup4**.

### Example 2

```
purepgroup allow array1:pgroup4
```

(Run from the target array)...

Allows replication from source array **array1** to the target array for protection group **pgroup4**.

### Example 3

```
purepgroup schedule --snap --hourly 3 pgroup2
```

Of the snapshots generated on the **pgroup2** source array based on the snapshot frequency, keeps the 3 most recent hourly snapshots.

### Example 4

```
purepgroup schedule --replicate --blackout 9:00am-5:00pm pgroup10
```

Stops replicating snapshots from the source array to all targets between 9:00am and 5:00pm every day for protection group **pgroup10**.

### Example 5

```
purepgroup setattr --addhostlist host4,host7,host10 pgroup42
```

Adds hosts **host4**,**host7**,and **host10** to protection group **pgroup42**. The existing hosts in **pgroup42** are not affected.

## See Also

[pureoffload\(1\)](#) [371] [purepgroup\(1\)](#) [381], [purepgroup-list\(1\)](#) [389] [purepod\(1\)](#) [412]

## Author

Pure Storage Inc. <[documentfeedback@purestorage.com](mailto:documentfeedback@purestorage.com)>

## pureplugin

pureplugin, pureplugin-check, pureplugin-install, pureplugin-list, pureplugin-uninstall — manages the Pure Storage FlashArray plugins

### Synopsis

```
pureplugin check [-h | --help] --host HOST --user USER PLUGIN
pureplugin install [-h | --help] --host HOST [--reinstall] --user USER PLUGIN
pureplugin list [-h | --help] [ --csv | --nvp ] [--notitle] [--page] [--raw]
[PLUGIN...]
pureplugin uninstall [-h | --help] --host HOST --user USER PLUGIN
```

### Arguments

#### **PLUGIN**

Pure Storage FlashArray plugin name.

### Options

#### **-h | --help**

Can be used with any command or subcommand to display a brief syntax description.

#### **--host**

The target host on which to install the plugin.

#### **--reinstall**

If the plugin is already installed, use the **--reinstall** option to overwrite (upgrade) it.

#### **--user**

User name with which to log in to the target host.

Options that control display format:

#### **--csv**

Lists information in comma-separated value (CSV) format. The **--csv** output can be used for scripting purposes and imported into spreadsheet programs.

#### **--notitle**

Lists information without column titles.

#### **--nvp**

Lists information in name-value pair (NVP) format, in the form **ITEMNAME=VALUE**. Argument names and information items are displayed flush left. The **--nvp** output is designed both for convenient viewing of what might otherwise be wide listings, and for parsing individual items for scripting purposes.

#### **--page**

Turns on interactive paging.

```
--raw
```

Displays the unformatted version of column titles and data. For example, in the `purearray monitor` output, the unformatted version of column title `us/op (read)` is `usec_per_read_op`. The `--raw` output is used to sort and filter list results.

## Description

The `pureplugin check` command checks the plugin version that is currently installed on the target host. Values must be specified for the `--host` and `--user` options. If the plugin is not installed, the version displays a dash.

The `pureplugin install` command installs the specified plugin. Values must be specified for the `--host` and `--user` options. If the plugin is already installed, use `--reinstall` to overwrite (upgrade) it.

The `pureplugin list` command displays a list of plugins on the array that can be installed on the target host. Use `--csv` to display the output as a comma-separated list of values. Use `--nvp` to display the output as name-value pairs. Use `--notitle` to display the output without column titles. To display a list of specific plugins, type the list of plugin names at the end of the command.

The `pureplugin uninstall` command uninstalls the specified plugin. Values must be specified for the `--host` and `--user` options.

## Examples

### Example 1

```
pureplugin check --host vCenterHost --user vCenterAdmin vsphere
```

Checks the vSphere plugin version that is currently installed on vCenterHost.

### Example 2

```
pureplugin install --host vCenterHost --user vCenterAdmin vsphere
```

Installs the vSphere plugin on vCenterHost.

### Example 3

```
pureplugin install --host vCenterHost --user vCenterAdmin --reinstall vsphere
```

Installs or upgrades the vSphere plugin on vCenterHost.

### Example 4

```
pureplugin list vsphere
```

Displays the available version of the vSphere plugin that can be installed on vCenter.

### Example 5

```
pureplugin uninstall --host vCenterHost --user vCenterAdmin vsphere
```

Uninstalls the vSphere plugin.

## See Also

`purearray(1)` [230]

## Author

Pure Storage Inc. <[documentfeedback@purestorage.com](mailto:documentfeedback@purestorage.com)>

## purepod

purepod, purepod-add, purepod-clone, purepod-create, purepod-destroy, purepod-eradicate, purepod-recover, purepod-remove, purepod-rename, purepod-setattr — manages the Purity//FA pod.  
 purepod-list, purepod-listobj — display pod attributes and storage consumption  
 purepod-monitor — monitor pod I/O performance

### Synopsis

```

purepod add --array ARRAYLIST POD
purepod clone SOURCE TARGET
purepod create [--failover-preference ARRAY-LIST] POD...
purepod destroy POD...
purepod eradicate POD...
purepod list [ [--failover-preference] | --footprint | --mediator | --on REMOTE | --space [ --historical { 1h | 3h | 24h | 7d | 30d | 90d | 1y } ] ] [ --pending | --pending-only ] [ --total ] [ --cli | --csv | --nvp ] [ --notitle ] [ --raw ] [ --filter FILTER] [ --page ] [ --limit LIMIT] [ --sort SORT] [ POD... ]
purepod listobj [ --pending | --pending-only ] [ --type { array | pggroup | pod | vol } ] [ --csv ] [ POD... ]
purepod monitor [ --array ] [ --historical { 1h | 3h | 24h | 7d | 30d | 90d | 1y } ] [ --interval SECONDS] [ --latency ] [ --mirrored ] [ --repeat REPEAT-COUNT] [ --notitle ] [ --raw ] [ --filter FILTER] [ --page ] [ --limit LIMIT] [ --sort SORT] [ POD... ]
purepod recover POD...
purepod remove --array ARRAYLIST [ --with-unknown ] POD
purepod rename OLD-NAME NEW-NAME
purepod setattr { --failover-preference ARRAYLIST | --mediator MEDIATOR } POD

```

### Arguments

#### **POD**

Pod to be created, stretched/unstretched, destroyed, eradicated, or recovered. Also the pod for which the mediator is to be configured.

#### **OLD-NAME**

Name of the pod to be renamed.

#### **NEW-NAME**

Name by which the pod is to be known after the command executes.

#### **SOURCE**

Pod from where data is cloned. The data is cloned to the **TARGET** pod.

**TARGET**

Pod to where data is cloned. The data is clone from the *SOURCE* pod.

## Object Names

Valid characters are letters (A-Z and a-z), digits (0-9), and the hyphen (-) character. The first and last characters of the name must be alphanumeric, and the name must contain at least one letter or '-'.

Most objects in Purity//FA that can be named, including host groups, hosts, volumes, protection groups, volume and protection group suffixes, SNMP managers, and subnets, can be 1-63 characters in length.

Array names can be 1-56 characters in length. The array name length is limited to 56 characters so that the names of the individual controllers, which are assigned by Purity//FA based on the array name, do not exceed the maximum allowed by DNS.

Names are case-insensitive on input. For example, **vol1**, **Vol1**, and **voL1** all represent the same volume. Purity//FA displays names in the case in which they were specified when created or renamed.

Pods and volume groups provide a namespace with unique naming conventions.

All objects in a pod have a fully qualified name that include the pod name and object name. The fully qualified name of a volume in a pod is **POD::VOLUME**, with double colons (::) separating the pod name and volume name. The fully qualified name of a protection group in a pod is **POD::PGROUP**, with double colons (::) separating the pod name and protection group name. For example, the fully qualified name of a volume named **vol01** in a pod named **pod01** is **pod01::vol01**, and the fully qualified name of a protection group named **pgroup01** in a pod named **pod01** is **pod01::pgroup01**.

If a protection group in a pod is configured to asynchronously replicate data to a target array, the fully qualified name of the protection group on the target array is **POD:PGROUP**, with single colons (:) separating the pod name and protection group name. For example, if protection group **pod01::pgroup01** on source array **array01** asynchronously replicates data to target array **array02**, the fully qualified name of the protection group on target array **array02** is **pod01:pgroup01**.

All objects in a volume group have a fully qualified name that includes the volume group name and the object name, separated by a forward slash (/). For example, the fully qualified name of a volume named **vol01** in a volume group named **vgroup01** is **vgroup01/vol01**.

## Options

**-h | --help**

Can be used with any command or subcommand to display a brief syntax description.

**--array**

When used with **purepod add**, stretches the pod between two arrays for synchronous replication.

When used with **purepod remove**, unstretches a stretched pod.

When used with **purepod monitor**, breaks down performance data by the array to which the I/O is directed.

--failover-preference **ARRAY-LIST**

Displays or determines which array within a stretched pod should be given priority to stay online, should the arrays ever lose contact with each other. The current array and any peer arrays that are connected to the current array for synchronous replication can be added to a pod for failover preference. A failover preference of (`auto`) means that no arrays have been configured for failover preference. To clear the list of failover preferences, set to an empty string ("").

--footprint

Displays the maximum amount of physical space the pod would take up on any array.

--historical **TIME**

When used with `purepod list --space`, displays historical size and space consumption over the specified range of time.

When used with `purepod monitor`, displays historical performance data over the specified range of time.

Valid time ranges include: 1 hour, 3 hours, 24 hours, 7 days, 30 days, 90 days, and one year.

--interval **SECONDS**

Sets the number of seconds between displays of real-time performance data. At each interval, the system displays a point-in-time snapshot of the performance data. If omitted, the interval defaults to every 5 seconds.

--latency

Displays real-time and historical I/O latency information.

--mediator

When used with `purepod list`, includes mediator details for the local array and all peer arrays. Mediator details include name, version and status.

When used with `purepod setattr`, replaces the current mediator with the specified one. By default, the Pure1 Cloud Mediator (`purestorage`) serves as the mediator.

--mirrored

Includes performance data for mirrored writes (i.e., writes to volumes in stretched pods). If one array of a stretched pod is offline, the writes processed on the other array are reported as local writes rather than mirrored ones. Once the pod is back online, in sync, and processing each write on both sides, the writes to its volumes are reported as mirrored writes again.

--on **REMOTE**

Displays a list of pods that are on the specified remote array but not stretched to this array. Pods that are stretched to this array will not appear in the list. REMOTE can be set to (\*) to represent all remote arrays.

--pending

Includes destroyed pods that are in the eradication pending state. If not specified, pods that are pending eradication are not shown.

--pending-only

Only displays destroyed pods that are in the eradication pending state.

--repeat *REPEAT-COUNT*

Sets the number of times to display real-time performance data. If omitted, the repeat count defaults to 1.

--space

Displays size and space consumption information for the volumes and snapshots inside the pod. The space metrics are local to the array.

--total

Follows output lines with a single line containing column totals in columns where they are meaningful.

--type

Specifies the type of information (arrays, pods, protection groups, or volumes) about specified pods to be produced in whitespace-separated list format suitable for scripting.

Options that control display format:

--cli

Displays output in the form of CLI commands that can be issued to reproduce the current configuration. The --cli output is not meaningful when combined with immutable attributes.

--csv

Lists information in comma-separated value (CSV) format. The --csv output can be used for scripting purposes and imported into spreadsheet programs.

--notitle

Lists information without column titles.

--page

Turns on interactive paging.

--raw

Displays the unformatted version of column titles and data. For example, in the `purearray monitor` output, the unformatted version of column title `us/op (read)` is `usec_per_read_op`. The --raw output is used to sort and filter list results.

Options that manage display results:

--filter

Displays only the rows that meet the filter criteria specified.

--limit

Limits the size of the list output to the specified maximum number of rows.

--sort

Sorts the list output in ascending or descending order by the column specified.

## Description

This page describes the management of pods for synchronous replication and transparent failover.

Organizations replicate critical data for various reasons, such as for off-host backup, software development and testing, data distribution and consolidation, and perhaps most importantly, disaster protection.

Purity//FA offers two types of replication: asynchronous replication and synchronous replication.

Asynchronous replication is managed through protection groups consisting of member hosts, host groups, or volumes. Replication policies are created for each protection group to define when Purity//FA should take snapshots of the protection group's data set and send it off to its target arrays, where they are saved as replicas. Asynchronous replication is configured through the **purepgroup** command.

Synchronous replication is managed through pods. A pod representing a collection of protection groups and volumes is created on one array and stretched to another array, resulting in fully synchronized writes between the two arrays. A pod can contain a mix of volumes, and protection groups with member volumes. Writes to the pod coming into either array are immediately synchronized and seen on both arrays. Synchronous replication is configured through the **purepod** command.

## Configuring Synchronous Replication

The process of creating and stretching a pod so that volume data is synchronously replicated between two arrays involves the following broad steps:

1. Verify that you have four physical interfaces configured with the replication service. For more information on how to configure the interfaces for synchronous replication, refer to **purenetwork(1)** [360].
2. Create a connection between the two arrays over which the pod will be stretched. For more information on how to connect arrays, refer to **purearray(1)** [230].
3. From the array with the volumes you want to synchronously replicate, create a pod (**purepod create**), and then add volumes and protection groups to the pod.  
Volumes are created in pods through the **purevol create** command. Volumes are moved into pods and protection groups through the **purevol move** command. For more information on how to create and move volumes, refer to **purevol(1)** [469].  
Protection groups are created through the **purepgroup create** command. For more information on how to create protection groups, refer to **purepgroup(1)** [381].
4. Create hosts, if necessary, and connect them to the volumes in the pod. If you have uniform storage access, meaning each host is connected to both arrays, but accessing one array is more efficient for reasons such as shorter distance and lower latency, then you may want to configure a preferred array for each host (also known as Asymmetric Logical Unit Access, or ALUA, preferred paths). Hosts and preferred arrays are created and configured through the **purehost** command. For more information on how to create and configure hosts, refer to **purehost(1)** [292].
5. Stretch the pod (**purepod add**) to the remote array.

Once stretched, the pod immediately starts synchronizing the data across the two arrays. Once fully synchronized, all of the data in the pod is mirrored between the two arrays. For example, when a volume is resized or renamed, the change will be seen on both arrays instantaneously.

## Creating Pods

The **purepod create** command creates a pod on the local array.

Each pod must be given a name that is unique across the arrays to which they are stretched, so a pod cannot be stretched to an array that already contains a pod with the same name.

After a pod has been created, add volumes and protection groups to the pod, and then stretch the pod to another (connected) array. Volumes are created inside pods through the **purevol create** command. Volumes can be moved into or out of pods and protection groups through the **purevol move** command. Protection groups are created inside pods through the **purepgroup create** command. Protection groups cannot be moved into or out of pods.

Include the **--failover-preference** option to specify which array within a stretched pod should be given priority to stay online, should the arrays ever lose contact with each other.

### Pod Naming Conventions

Each pod is a separate namespace for the volumes and protection groups it contains.

Each volume in a pod consists of the pod namespace identifier and the volume name, separated by a double colon (::). The naming convention for a volume inside a pod is **POD::VOL**, where:

- **POD** is the name of the container pod.
- **VOL** is the name of the volume inside the pod.

For example, the fully qualified name of a volume named **vol01** inside a pod named **pod01** is **pod01::vol01**.

In the following example, an array has three volumes named **vol01** that are completely independent of one another - one on the array named **vol01**, another volume in pod01 named **pod01::vol01**, and a third volume in pod02 named **pod02::vol01**.

\$ purepod list *vol01					
Name	Size	Source	Created	Serial	
pod01::vol01	1T	-	2017-06-12 16:05:37 PDT	C54E7DD1D6F6408600011814	
pod02::vol01	1T	-	2017-06-12 16:05:37 PDT	C54E7DD1D6F6405305839572	
vol01	1T	-	2017-06-08 14:48:02 PDT	C54E7DD1D6F6408600011010	

When administering a volume that is in a pod, always include the pod namespace in the volume name. For example, to create a volume named **vol02** in pod01 that is 1T in size, run the command **purepod create --size 1T pod01::vol02**. To increase the virtual size of volume **vol11** in pod pod02 to 2 GB, run the command **purepod setattr --size 2G pod02::vol11**.

Each protection group in a pod consists of the pod namespace identifier and the protection group name, separated by a double colon (::). The naming convention for a protection group inside a pod is **POD::PGROUP**, where:

- **POD** is the name of the container pod.
- **PGROUP** is the name of the protection group inside the pod.

For example, the fully qualified name for a protection group named pgroup02 inside a pod named pod09 is `pod09::pgroup02`.

When administering a protection group that is in a pod, always include the pod namespace in the protection group name. For example, to create a protection named pgroup02 in pod01, run the command `purepgroup create pod01::pgroup02`.

## Configuring Hosts and Preferred Arrays

For a host to read and write data on a volume within a pod, a connection between the two must be established. Verify that each volume in the pod is connected to a host.

If you have uniform storage access, meaning each host is connected to both arrays, but accessing one array is more efficient for reasons such as shorter distance and lower latency, then you may want to configure a preferred array for each host (also known as Asymmetric Logical Unit Access, or ALUA, preferred paths). If a preferred array is configured for a host, then other arrays will expose active/non-optimized paths to that host.

Hosts are created and preferred arrays are configured through the `purehost` command. For more information on how to create and configure hosts, refer to `purehost(1)` [292].

## Stretching Pods

The `purepod add --array` command stretches a pod to another array. When a pod is stretched, the volume data inside the pod is synchronously replicated between the two arrays. The arrays of a stretched pod are considered peer to one another - there is no concept of "source" or "target" array.

When the pod is first stretched, the pod data is asynchronously replicated to the other array before transitioning into synchronous mode. An array can host multiple stretched pods, each one replicated to a different peer.

Once synchronized, the peer arrays of a stretched pod present the pod volumes, including their contents, volume serial numbers, and snapshots, identically to all connected hosts at all times, and pod volume reads always return the same data regardless of which array receives and executes them.

Array administrators are also peers with respect to managing the pod and the volumes, protection groups, and snapshots it contains. For example, either administrator can change the size of a volume in the pod, create and destroy volumes in the pod, move volumes into and out of the pod, connect hosts to the volumes in the pod, add volumes to protection groups in the pod, schedule, take, and destroy snapshots of volumes and protection groups in the pod, and create clone volumes from snapshots of volumes in the pod. These operations take effect immediately and are visible on both arrays.

Before a pod can be stretched to another array, the two arrays must first be connected. Arrays are connected by running the command `purearray connect`.

After the arrays are connected, run the `purepod add --array` command to stretch the pod to the other array. For example, run the following command from array `array01` to stretch pod `pod01` on array `array01` to remote array `array02`,

```
$ purepod add --array array02 pod01
```

A pod can be stretched across no more than two arrays.

Stretched pods cannot be destroyed.

Volumes cannot be added to or removed from a pod that is stretched. To move volumes into or out of a stretched pod, unstretch the pod, move the volumes into or out of the unstretched pod, and then restretch the pod.

Protection groups cannot be created in a pod that is stretched. To create protection groups in a stretched pod, unstretch the pod, create the protection groups inside the unstretched pod, and then restretch the pod.

If an unstretched pod contains a protection group that is asynchronously replicating to another array, stretching the pod to any other array will not disrupt the asynchronous replication connection. If an unstretched pod contains a protection group that is asynchronously replicating to another array and the pod is stretched to that same array, any in-progress data transfer processes stop, the target array disallows asynchronous replication, and the synchronous replication process begins.

## Unstretching Pods

When the volumes within a stretched pod no longer need to replicate synchronously, unstretch the pod to collapse it to a single array.

Run the `purepod remove --array` command to unstretch a pod, removing it from the specified array.

After a pod has been unstretched, synchronous replication stops. A destroyed version of the pod with “restretch” appended to the pod name is created on the array that no longer has the pod. The restretch pod represents a point-in-time snapshot of the pod, just before it was unstretched. Run the `purepod list --pending-only` command to display the details of the restretch pod.

The restretch pod enters a 24-hour eradication pending period starting from the time that the pod was unstretched.

If within the 24-hour pending period the same pod is stretched to the array again, the pod is restored by first recovering the destroyed (\*.restretch) pod, and then replicating the new pod data until the stretched pod is in sync. Once fully synchronized, pod activity resumes on both arrays.

After the 24-hour pending period has lapsed, the restretch pod is permanently eradicated. If the pod is restretched after the 24-hour pending period, the arrays synchronize the pod data as if it were the first time the pod was stretched.

A restretch can pod can be cloned or destroyed, but it cannot be explicitly recovered.

In the event that a local array goes offline while a pod is still stretched across two arrays, the status of the remote array becomes unknown and there is no guarantee that the pod is online elsewhere. In such cases, you will only be able to unstretch the pod from the local array by including the `--with-unknown` option to force the unstretch.

In the following example, pod01 is stretched across arrays array01 and array02. Local array01 has gone offline. As a result, the status of its peer array (array02) is unknown. To unstretch pod01 while it is in offline/unknown status, you must include the `--with-unknown` option to force the unstretch.

```
$ purepod list
Name      Source  Array    Status   Frozen At
```

```
pod01      -      array01  offline  2017-11-14 17:25:40 PST
           -      array02  unknown   -
```

Note that by using the `--with-unknown` option, you risk losing data if there is a problem with the pod on the remote array.

## Cloning Pods

In the event of an extreme disaster where one array of a stretched pod is in an offline status while the other array is in an online status, and then the online array becomes permanently disabled, the pod or restretch pod in the offline array can be cloned to create a point-in-time consistent copy of the pod and its contents.

Run the `purepod clone` command to clone an existing pod (**SOURCE**) to create a new pod (**TARGET**).

When cloned, the new pod takes on the entire history and configuration of the source pod, including its protection groups, protection group snapshot and replication policies, volumes, volume snapshots, and historical performance metrics. The new pod does not take on the synchronous replication configuration of the source pod, and the serial numbers of its volumes differ from those of the source pod's volumes.

Once cloned, the new pod can be configured to resume synchronous replication and the source pod can be deleted from the arrays over which it was stretched.

After cloning a pod, run the `purepod list` command to display the new pod and its details, including and the name of the source pod from which the new pod was cloned. In the following example, pod `pod01`, which is currently in "offline" status, has been cloned to create a new pod named `clonedpod01`.

```
$ purepod clone pod01 clonedpod01
Name      Source  Array      Status  Frozen At
clonedpod01  pod01  pure-001  online   -
$ purepod list
Name      Source  Array      Status  Frozen At
pod01      -      pure-001  offline  2017-11-14 17:25:40 PST
clonedpod01  pod01  pure-001  online   -
```

Notice that the source pod for `clonedpod01` is `pod01`. If the source pod is eradicated, the name of the source pod will no longer appear.

## Destroying Pods

When a pod is no longer needed, it can be destroyed. Run the `purepod destroy` command to destroy a pod.

A pod can only be destroyed if it is empty, so before destroying a pod, ensure all volumes and protection groups inside the pod have been either moved out of the pod or destroyed.

You cannot destroy a stretched pod. To destroy a stretched pod, unstretch the pod before destroying it.

The destruction of pods and their contents is not immediate. Instead, the pod is placed in a 24-hour eradication pending period. The pod can be manually recovered or permanently eradicated within the eradication pending period. After 24 hours, the pod is completely eradicated and not recoverable.

## Recovering Pods

A destroyed pod can be recovered at any time during the 24-hour eradication pending period. Run the **purepod recover** command to recover a pod.

Once a pod has been recovered, its destroyed volumes and protection groups can also be recovered to bring the pod and its contents back to their previous state.

## Eradicating Pods

A destroyed pod can be permanently eradicated at any time during the 24-hour eradication pending period. Run the **purepod eradicate** command to eradicate a pod.

A pod can only be eradicated after its destroyed volumes and protection groups are eradicated. Therefore, before eradicating a pod, first eradicate all its destroyed volumes and protection groups.

Once a pod has been eradicated, it can no longer be recovered.

## Renaming Pods

Run the **purepod rename** subcommand to change the current name (**OLD-NAME**) of the pod to the new name (**NEW-NAME**). The name change is effective immediately and the old name is no longer recognized in CLI, GUI or REST interactions. All volumes, volume snapshots, and protection groups with the pod namespace are also renamed. In the Purity//FA GUI, the new name appears upon page refresh.

## Mediator

Pure Storage provides a passive mediation solution for synchronous replication that guarantees that a pod remains online on one array when the other array is offline.

During regular operation, each array periodically pings the mediator to confirm its status. In the event of an outage in the environment and two arrays lose contact with each other, the mediator takes control to determine which array remains online and continues data services.

When the arrays lose contact with each other, the first array to reach the mediator becomes the array that stays online and continues serving I/O while its peer array goes offline. If failover preference is configured for the pod, any of the arrays that are listed in the failover preference list are given priority to stay online. At this point, the stretched pod falls out of sync. To avoid inconsistent data between the two arrays (also known as "split brain"), the pod data in the offline array becomes frozen. The Frozen At time, also known as the recovery point, represents the time at which the data on the pod was frozen when the array went offline.

When the peer arrays re-establish contact with each other, the array that was online throughout the outage starts replicating pod data to its peer array until the pod is once again in sync, at which time pod activity resumes on both arrays.

The mediation process happens per pod, so if two arrays share multiple pods and the arrays lose contact with each other, one array could be online for some pods but offline for other pods, and vice versa for the other array.

View the mediator status for each pod by running the `purepod list --mediator` command. Possible mediator statuses include:

- **Online.** The array is successfully communicating with the mediator.
- **Unreachable.** The array cannot reach the mediator, either due to network issues or because the mediator is down. When a mediator is unreachable, synchronous replication continues to function provided all arrays are healthy and communicating, but a high availability event without mediator access can result in an outage.

If an array cannot successfully communicate with the mediator, an alert is generated. If after verifying your network connections the mediator status is still unreachable, contact Pure Storage Support.

By default, the mediator is set to `purestorage`, representing the Pure1© Cloud Mediator. The Pure1 Cloud Mediator requires no special configuration to work. View the mediator configuration for each pod by running the `purepod list --mediator` command.

The mediator service can be configured to use a different mediator, such as one that is hosted on premise, by running the `purepod setattr --mediator` command. Before you configure the mediator, contact Pure Storage Support to obtain the package and steps on how to install an alternate mediator.

To set the mediator service back to the default Pure1 Cloud Mediator, use an empty string ("").

## **Failover Preference**

Failover preference determines which array within a stretched pod should be given priority to stay online, should the arrays ever lose contact with each other. If both arrays are online and can reach the mediator, failover preference determines which of those two arrays should be given priority to stay online. If, however, only one of the two arrays can reach the mediator, that array stays online regardless of failover preference. Failover preference is set through the `--failover-preference` option.

The current array and any peer arrays that are connected to the current array for synchronous replication can be added to a pod for failover preference. A pod can be configured with multiple arrays for failover preference, even if the pod is not stretched to those arrays. For example, failover preference for a pod can be configured to include all of the arrays of a data center.

In the following example, pod `pod01` on current array `array01` has been configured with three arrays with failover preference: `arrayA`, `arrayB`, and `arrayC`. All three of the arrays are already connected to the current array for synchronous replication. The pod is then stretched to `arrayA`.

```
$ purepod list pod01 --failover-preference
Name  Source  Array      Status  Frozen At   Failover  Preference
pod01 -        array01  online   -          arrayA
                                         arrayB
                                         arrayC

$ purepod add --array arrayA pod01
Name  Arrays
```

```
pod01 array01
      arrayA
```

If the arrays lose contact with each other, **arrayA** will be given priority over **array01** to reach the mediator first. If **arrayA** cannot reach the mediator, **array01** will take over.

In the following example, pod **pod02** on current array **array02** has been configured with just one array - its own - with failover preference. The pod is then stretched to array **array04**.

```
$ purepod list pod02 --failover-preference
Name      Source   Array      Status  Frozen At  Failover Preference
pod02     -        array02    online   -        array02
$ purepod add --array array04 pod02
Name  Arrays
pod02 array02
      array04
```

If the arrays lose contact with each other, **array02** will be given priority over **array04** to reach the mediator first. If **array02** cannot reach the mediator, **array04** will take over.

The arrays listed for failover preference do not appear in any priority order, so if a pod is stretched over two arrays that are configured for failover preference, either array will have an opportunity to stay online for the given pod.

A failover preference of (**auto**) means that no arrays have been configured for failover preference. In such cases, if both arrays within a stretched pod can reach the mediator after losing contact with each other, either array will have an opportunity to stay online for the given pod.

To clear the list of failover preferences, set to an empty string ("").

## Pre-Election

In addition to passive mediation and failover preference, Purity provides the pre-election behavior to further ensure a stretched pod remains online. With pre-election, an array within a stretched pod is chosen by Purity to keep a pod online when other failures occur in the environment.

The pre-election behavior elects one array of the stretched pod to remain online in the rare event that:

- The mediator is inaccessible on both arrays within the stretched pod, preventing the arrays from racing to the mediator to determine which one keeps the pod online.

...and then later...

- The arrays within the stretched pod become disconnected from each other.

When the mediator becomes inaccessible on both arrays, Purity pre-elects an array per pod to keep the pod online. Then, if the arrays lose contact with each other, the pre-elected array for that pod takes over to keep the pod online while its peer array takes the pod offline.

If either array reconnects to the mediator before they lose contact with each other, the pre-election result is cancelled. The array with access to the mediator will race to the mediator and keep the pod online if its peer array fails or the arrays become disconnected from each other.

Run the `purepod list --mediator` command to display the pre-election status of an array. If the array is pre-elected to keep the pod online in the event of a communication failure between the arrays, the term (**pre-elected**) is appended to the mediator status in the Mediator Status column.

In the following example, pod `pod01` is stretched over arrays `array01` and `array02`. The arrays cannot reach the mediator, so Purity has pre-elected array `array01` to keep the pod online in the event of a communication failure between the arrays.

purepod list --mediator							
Name	Source	Mediator	Mediator Version	Array	Status	...	Mediator Status
pod01	-	purestorage	1.0	array01	online	...	unavailable (pre-elected)
				array02	online	...	unavailable

Then, if the arrays lose contact with each other, the pre-elected array for that pod takes over to keep the pod online while its peer array takes the pod offline.

One and only one array within each pod is pre-elected at a given point in time, so while a pre-elected array is keeping the pod online, the pod on its non-elected peer array remains offline during the communication failure.

Users cannot pre-elect arrays. Purity uses various factors, including the following ones (listed in order of precedence), to determine which array is pre-elected:

- If a pod has a failover preference set, then the array that is preferred will be pre-elected.
- If one of the arrays has no hosts connected to volumes in the pod, then the other array will be pre-elected.
- If neither of the above factors applies, one of the arrays is selected by Purity.

If the pre-elected array goes down while pre-election is in effect, the non-elected peer array will not bring the pod online.

If the non-elected array reconnects to the mediator while it is still disconnected from the pre-elected array, it is ignored and will still keep the pod offline. If the data in the non-elected pod must be accessed, clone it to create a point-in-time consistent copy of the pod and its contents, including its volumes and snapshot history. After the pod has been cloned, disconnect the hosts from the original volumes and reconnect the hosts to the volumes within the cloned pod.

If the arrays re-establish contact with each other but the mediator is still inaccessible, the array that was online throughout the outage starts replicating pod data to its peer array until the pod is once again in sync and both arrays serve I/O. One array will still be pre-elected, as indicated by the term (**pre-elected**), in case both arrays lose contact with each other again.

When the peer arrays re-establish contact with each other *and* can access the mediator, the array that was online throughout the outage starts replicating pod data to its peer array until the pod is once again in sync and both arrays serve I/O, at which time pod activity returns to normal.

## Listing Pod Details

The `purepod list` command displays a list of all pods on the local array. If a pod is stretched, the details of its peer array are also displayed.

The status of the array (Status) determines the ability of a pod to synchronously replicate data. The status details are unique to the local array and will differ from the status details when viewed from a peer array, so what one array knows about its own status and the status of its peer arrays will differ from what its peer arrays know.

Possible statuses on the local array include:

#### Online

The array is online and has the latest data for the pod. The array can handle I/O to the pod and take over during a high availability event.

#### Offline

The array is experiencing problems and may not have the latest data for the pod. The array cannot handle I/O to the pod and cannot take over during a high availability event.

#### Resyncing

The array is actively getting the latest data for the pod so that it becomes fully synchronized with its peer array. During the resyncing process, the array cannot handle I/O to the pod. Once the arrays are fully synchronized, the array changes to Online status.

#### Unknown

The status of the peer array is unknown because this array is offline and cannot determine the state of the pod on the peer array. Only the peer array can ever be in unknown status; this unknown status is unique to the local array and will differ when viewed from its peer array.

During normal operation, the status of each pod is `online`, meaning the arrays are online and have the latest data for the pod. If the pod is stretched over two arrays, the data is synchronously replicating as expected.

In the following example, local array `array01` has two pods. Pod `pod01` resides locally and is unstretched, while pod `pod02` is stretched between the local array and `array02`. Both arrays are online, which means that they are serving I/O and operating as expected.

```
$ purepod list
Name   Source  Array      Status   Frozen At
pod01  -       array01   online   -
                  array02   online   -
pod02  -       array01   online   -
```

When a pod is stretched to a peer array, the array goes into `resyncing` state as it actively gets the latest pod data. Once the peer array becomes synchronized with the local array, the status on the peer array changes to `online`. Writes to the pod coming into either array are immediately synchronized and seen on both arrays.

When the arrays lose contact with each other, they race to the mediator. The first array to reach the mediator becomes the array that stays `online` to continue serving I/O while its peer array goes `offline`. At this point, any stretched pods fall out of sync. The data in the offline array becomes frozen. Since the offline array has no contact with its peer array, it does not know its peer's status, so it gives its peer a status of unknown.

The mediation process happens per pod, so if two arrays share multiple pods and the arrays lose contact with each other, one array could be online for some pods but offline for other pods, and vice versa for the other array.

In the following example, arrays **array01** and **array02** lose contact with each other. Both arrays race to the mediator, where **array02** wins and stays online. **array01** goes offline and become frozen at 2017-11-14 17:25:40 PST. Since **array01** is offline, it cannot determine the status of peer **array02**, so its status is **unknown**. Here are the status details for both arrays, as seen from the offline array **array01**:

```
$ purepod list
Name   Source  Array      Status   Frozen At
pod01  -        array01  offline   2017-11-14 17:25:40 PST
      -        array02  unknown   -
```

When the peer arrays re-establish contact with each other, the array that was **online** throughout the outage starts replicating pod data to its peer array until the pod is once again in sync, at which time the **offline** array returns to **online** status and pod activity resumes across both arrays.

Here are the status details for both arrays after array01 comes back online:

```
$ purepod list
Name   Source  Array      Status   Frozen At
pod01  -        array01  online   -
      -        array02  online   -
```

Include the **--mediator** option to display the name and version number of the mediator that is used to determine which array will continue data services should an outage occur in the environment.

By default, the mediator is set to **purestorage**, representing the Pure1® Cloud Mediator.

The mediator status indicates if the mediator is available to mediate a high availability event. Possible mediator statuses include:

#### Online

The array is successfully communicating with the mediator, and the mediator is available to mediate a high availability event.

#### Unreachable

The array cannot reach the mediator, either due to network issues or because the mediator is down. When a mediator is unreachable, synchronous replication continues to function provided all arrays are healthy and communicating, but a high availability event without mediator access can result in an outage. If an array cannot successfully communicate with the mediator, an alert is generated. If after verifying your network connections the mediator status is still unreachable, contact Pure Storage Support.

If an array cannot successfully communicate with the mediator, an alert is generated.

In the following example, local array **array01** has two pods. Pod **pod01** is unstretched, while pod **pod02** is stretched to peer array **array02**. All of the arrays over which the pods are stretched are

online and serving I/O. Since pod **pod02** is stretched to peer array **array02**, any I/O that comes into one of the arrays is replicated to the other array so that the pod is always in sync.

\$ purepod list --mediator									
Name	Source	Mediator	Mediator Version	Mediator Status	Array	Status	Frozen	At	
pod01	-	purestorage	1.0	online	array01	online	-		
pod02	-	purestorage	1.0	online	array01	online	-		
	-	purestorage	1.0	online	array02	online	-		

Include the **--on** option to display a list of pods that are on the specified remote array but not stretched to this array. Pods that are stretched to this array will not appear in the list. REMOTE can be set to (\*) to represent all remote arrays.

In the following example, the current array, array **pure-abc**, is connected to three arrays (**pure-001**, **pure-021**, and **pure-050**) with some pods that are not stretched to the current array. Pod **pod01** is stretched over arrays **pure-001** and **pure-050**. Pod **pod02** is on array **pure-001** and not stretched to any other arrays. Pod **pod03** is stretched over arrays **pure-001** and **pure-021**.

```
//From array pure-abc
purepod list --on *
Name   Array
pod01  pure-001
      pure-050
pod02  pure-001
pod03  pure-001
      pure-021
```

Include the **--space** option to display the following size and space consumption details for all volumes and snapshots in each pod on the local array:

#### Size

Total provisioned size of all volumes inside the pod. Represents storage capacity reported to hosts.

#### Thin Provisioning

Percentage of volume sectors that do not contain host-written data because the hosts have not written data to them or the sectors have been explicitly trimmed.

#### Data Reduction

Ratio of mapped sectors within a volume versus the amount of physical space the data occupies after data compression and deduplication. The data reduction ratio does not include thin provisioning savings.

For example, a data reduction ratio of 5:1 means that for every 5 MB the host writes to the array, 1 MB is stored on the array's flash modules.

#### Total Reduction

Ratio of provisioned sectors within a volume versus the amount of physical space the data occupies after reduction via data compression and deduplication *and* with thin provisioning savings. Total reduction is data reduction with thin provisioning savings.

For example, a total reduction ratio of 10:1 means that for every 10 MB of provisioned space, 1 MB is stored on the array's flash modules.

#### Volumes

Physical space occupied by volume data that is not shared between volumes, excluding array metadata and snapshots.

#### Snapshots

Physical space occupied by data unique to one or more snapshots.

#### Total

Total physical space occupied by system, shared space, volume, and snapshot data.

In the following example, pod `pod01` is stretched across arrays `array01` and `array02`. Both arrays are online and the pod data between the two arrays is in sync, taking up space across both arrays.

Note that space consumption differs between the two arrays due to reasons such as compression, deduplication, hardware differences, resources, and so on.

<code>//From array01</code>							
<code>\$ purevol list --space</code>							
Name	Size	Thin Provisioning	Data Reduction	Total Reduction	Volumes	...	Total
<code>pod01::vol01</code>	1T	100%	7.6 to 1	>100 to 1	34.43M	...	34.43M
<code>pod01::vol02</code>	1T	100%	6.3 to 1	>100 to 1	28.64M	...	28.64M
<code>pod01::vol03</code>	1T	100%	6.3 to 1	>100 to 1	28.07M	...	28.07M

<code>//From array02</code>							
<code>\$ purevol list --space</code>							
Name	Size	Thin Provisioning	Data Reduction	Total Reduction	Volumes	...	Total
<code>pod01::vol01</code>	1T	100%	5.6 to 1	>100 to 1	46.65M	...	46.65M
<code>pod01::vol02</code>	1T	100%	5.6 to 1	>100 to 1	32.50M	...	32.50M
<code>pod01::vol03</code>	1T	100%	6.0 to 1	>100 to 1	29.67M	...	29.67M

Include the `--total` option with the `--space` option to display the total size and space consumption for all pods.

Include the `--footprint` option to display the maximum amount of physical space the pod would take up on any array. The footprint metric is mostly used for capacity planning.

In the following example, if pod `pod01` was stretched or migrated to an empty array, it would take up 2 gigabytes of (physical) space.

<code>purepod list --footprint</code>	
Name	Footprint
<code>pod01</code>	2.00G

Include the `--total` option with the `--footprint` option to display the total physical space the pod would take up on any array.

The **purepod list --pending** command displays a list of all pods, including ones that have been destroyed and are in the eradication pending state. The **purepod list --pending-only** command only displays a list of pods that have been destroyed and are in the eradication pending state.

The **purepod listobj** command displays a list of pod attributes, either in whitespace or comma-separated form, suitable for scripting. Include the **--type** option to list FlashArray objects that are associated with one or more pods. The **--type** option accepts the following arguments:

**--type array**

Display a list of arrays over which the specified pods reside, either stretched or unstretched. If no pods are specified, the list contains the names of all arrays that contain pods.

**--type pggroup**

Displays a list of protection groups that reside in the specified pods. If no pods are specified, the list contains the names of all protection groups that reside in pods.

**--type pod (default if --type option not specified)**

Displays a list of pods that have been created on the array or stretched to this array from another array. If no pods are specified, the list contains the names of all pods that reside on this array.

**--type vol**

Displays a list of volumes that reside in the specified pods. If no pods are specified, the list contains the names of all volumes that reside in pods.

## Monitoring Pod I/O Performance

The **purepod monitor** command displays real-time and historical I/O performance information for all of the specified pods. The output includes the following bandwidth, IOPS, and latency data:

- **Name:** Object name.
- **Time:** Current time.
- **B/s:** Bandwidth. Number of bytes read/written.
- **op/s:** IOPS. Number of read, write, or mirrored write requests processed per second.
- **us/op:** Service time. Average time, measured in microseconds, it takes the array to serve a read, write, or mirrored write I/O request. Service time does not include SAN time, queue time, or QoS rate limit time.
- **B/op:** IOPS. Average I/O size per read, write, mirrored write, and both read and write (all) operations. Must include the **--size** option to see the B/op columns.

Include the **--interval** option to specify the number of seconds between each real-time update.

Include the **--repeat** option to specify the number of times to repeat the real-time update. The **--interval** and **--repeat** options can be combined.

Include the **--array** option to break down performance data by the array to which the I/O is directed.

Include the **--latency** option to display real-time and historical I/O latency information for all of the volumes in each pod. The **purepod monitor --latency** output includes the following data:

- **SAN us/op:** SAN time. Average time, measured in microseconds, required to transfer data between the initiator and the array. Slow data transfers will result in higher SAN times.
- **Queue us/op:** Queue time. Average time, measured in microseconds, that an I/O request spends in the array waiting to be served. The time is averaged across all I/Os of the selected types.
- **QoS Rate Limit us/op:** QoS rate limit time. Average time, measured in microseconds, that reads, writes, or mirrored writes spend in queue as a result of bandwidth limits reached on one or more volumes.
- **us/op:** Service time. Average time, measured in microseconds, it takes the array to serve a read, write, or mirrored write I/O request. Service time does not include SAN time, queue time, or QoS rate limit time.

Include the **--mirrored** option to display performance or latency data for mirrored writes (i.e., writes to volumes in stretched pods). If one array of a stretched pod is offline, the writes processed on the other array are reported as local writes rather than mirrored ones. Once the pod is back online, in sync, and processing each write on both sides, the writes to its volumes are reported as mirrored writes again.

By default, the `purepod monitor` command displays real-time performance data. The `purepod monitor --historical` command displays historical performance data over any of the following ranges of time: 1 hour, 3 hours, 24 hours, 7 days, 30 days, 90 days, or 1 year.

## Examples

### Example 1

```
purepod create POD01
```

Creates a pod named `POD01`.

### Example 2

```
purearray connect --management-address ARRAY01 --type sync-replication
                  --connection-key
purepod add --array ARRAY01 POD01
```

Connects the current array to remote array `ARRAY01`, and then stretches `POD01` to array `ARRAY01`.

### Example 3

```
purearray connect --management-address ARRAYDC1 --type sync-replication
                  --connection-key
purepod create --failover-preference ARRAYDC1 POD02
purepod add --array ARRAYDC1 POD02
```

Connects the current array to remote array `ARRAYDC1`, creates a pod named `POD02` with failover preference given to array `ARRAYDC1`, and then stretches `POD02` to array `ARRAYDC1`.

## Example 4

```
purepod remove --array ARRAY01 POD01
```

Unstretches **POD01** from array **ARRAY01**, creating a destroyed pod with name **POD01.restretch** on array **ARRAY01**.

## Example 5

```
purepod clone POD10 POD10-CLONE
```

Clones pod **POD10** to create a new pod named **POD10-CLONE**.

## Example 6

```
purepod destroy POD10::VOL20
purepgroup destroy POD10::PGROUP05 POD10::PGROUP06
purepod destroy POD10
```

Destroys volume **VOL20** and protection groups **PGROUP05** and **PGROUP06** in **POD10**, clearing **POD10** of all volumes and protection groups. Destroys empty pod **POD10**.

## Example 7

purepod list --mediator									
Name	Source	Mediator	Mediator Version	Array	Status	Frozen At	Mediator Status		
pod01	-	purestorage	1.0	array01	online	-	unavailable		
				array02	online	-	unavailable		

Displays a list of all pods on the local array and the local array's mediator details. The local array **array01** has one pod named **pod01**. . The pod is stretched, so the details of peer array **array02** are also displayed.

## Example 8

```
purepod list --on ARRAY-001
Name      Array
pod01    ARRAY-001
          ARRAY-050
pod02    ARRAY-001
pod03    ARRAY-001
          ARRAY-021
```

Displays a list of pods that are on remote array **ARRAY-001**, but not stretched to this array.

## See Also

[purehgroup\(1\)](#) [275], [purehgroup-connect\(1\)](#) [283], [purehost\(1\)](#) [292], [purehost-connect\(1\)](#) [302]  
[purepgroup\(1\)](#) [381], [purevol\(1\)](#) [469]

**Author**

Pure Storage Inc. <[documentfeedback@purestorage.com](mailto:documentfeedback@purestorage.com)>

## pureport

pureport, pureport-list — manages an array's host connection ports

### Synopsis

```
pureport list [--initiator] [ --csv | --nvp ] [--filter FILTER] [--limit LIMIT]
[--notitle] [--page] [--raw] [--sort SORT]
```

### Options

**-h** | **--help**

Can be used with any command or subcommand to display a brief syntax description.

**--initiator**

Displays host iSCSI Qualified Names (IQNs), NVMe Qualified Names (NQNs), and Fibre Channel World Wide Names (WWNs) - both those discovered by Purity//FA and those manually assigned by system administrators - along with the array ports (targets) on which they are eligible to communicate.

Options that control display format:

**--csv**

Lists information in comma-separated value (CSV) format. The **--csv** output can be used for scripting purposes and imported into spreadsheet programs.

**--notitle**

Lists information without column titles.

**--nvp**

Lists information in name-value pair (NVP) format, in the form **ITEMNAME=VALUE**. Argument names and information items are displayed flush left. The **--nvp** output is designed both for convenient viewing of what might otherwise be wide listings, and for parsing individual items for scripting purposes.

**--page**

Turns on interactive paging.

**--raw**

Displays the unformatted version of column titles and data. For example, in the **purearray monitor** output, the unformatted version of column title **us/op (read)** is **usec\_per\_read\_op**. The **--raw** output is used to sort and filter list results.

Options that manage display results:

**--filter**

Displays only the rows that meet the filter criteria specified.

**--limit**

Limits the size of the list output to the specified maximum number of rows.

--sort

Sorts the list output in ascending or descending order by the column specified.

## Description

The **pureport list** command displays the iSCSI Qualified Names (IQNs), NVMe Qualified Names (NQNs), and Fibre Channel World Wide Names (WWNs) assigned to an array's target ports. If the array port has failed over, the Failover column displays the name of the port to which this port has failed over. Include the **--initiator** option to display all host IQNs, NQNs, and WWNs known to the array, either through discovery or manually assigned, and the array ports on which they are eligible to communicate. IQNs, NQNs, and WWNs are manually assigned through the **purehost** command.

The **pureport list** output does not include information about network interfaces. For information about network interfaces, refer to **purenetwork(1)** [360].

## Examples

### Example 1

```
pureport list --initiator --notitle
```

Displays initiator IQNs, NQNs, and WWNs known to the array, both discovered and manually assigned by system administrators. If an array port has failed over, also displays the name of the port to which the port has failed over. The output excludes column titles.

### See Also

[purearray\(1\)](#) [230], [purehost\(1\)](#) [292], [purenetwork\(1\)](#) [360]

### Author

Pure Storage Inc. <[documentfeedback@purestorage.com](mailto:documentfeedback@purestorage.com)>

## puresmis

puresmis, puresmis-enable, puresmis-disable, puresmis-list — manages the Pure Storage Storage Management Initiative Specification (SMI-S) provider

### Synopsis

**puresmis** disable [--slp] [--wbem-https]

**puresmis** enable [--slp] [--wbem-https]

**puresmis** list [ --cli | --csv | --nvp ] [--notitle] [--page] [--raw]

### Options

**-h | --help**

Can be used with any command or subcommand to display a brief syntax description.

**--slp**

Enables or disables the Service Location Protocol (SLP) and its ports, TCP 427 and UDP 427. SLP is optional.

**--wbem-https**

Enables or disables the SMI-S provider and its port, TCP 5989.

Options that control display format:

**--cli**

Displays output in the form of CLI commands that can be issued to reproduce the current configuration. The **--cli** output is not meaningful when combined with immutable attributes.

**--csv**

Lists information in comma-separated value (CSV) format. The **--csv** output can be used for scripting purposes and imported into spreadsheet programs.

**--notitle**

Lists information without column titles.

**--nvp**

Lists information in name-value pair (NVP) format, in the form **ITEMNAME=VALUE**. Argument names and information items are displayed flush left. The **--nvp** output is designed both for convenient viewing of what might otherwise be wide listings, and for parsing individual items for scripting purposes.

**--page**

Turns on interactive paging.

**--raw**

Displays the unformatted version of column titles and data. For example, in the **purearray monitor** output, the unformatted version of column title **us/op (read)** is **usec\_per\_read\_op**. The **--raw** output is used to sort and filter list results.

## Description

The **puresmis** command manages the Pure Storage Storage Management Initiative Specification (SMI-S) provider.

Enable the SMI-S provider to administer the array through an SMI-S client. The SMI-S provider is optional and must be enabled before its first use.

For more information about the SMI-S provider, including command usage and examples, refer to the Pure Storage SMI-S Provider Guide in the Pure1 Knowledge site at <https://support.purestorage.com>.

## Author

Pure Storage Inc. <[documentfeedback@purestorage.com](mailto:documentfeedback@purestorage.com)>

## puresmtp

puresmtp, puresmtp-list, puresmtp-setattr — manages connections to Simple Network Management Protocol (smtp) managers

### Synopsis

```
puresmtp list [ --cli | --csv | --nvp ] [--notitle]  
puresmtp setattr [--password] [--relay-host RELAY-HOST] [--sender-domain SENDER-DOMAIN] [--user-name USER-NAME]
```

### Options

**-h | --help**

Can be used with any command or subcommand to display a brief syntax description.

**--password**

Displays or sets the SMTP password used to authenticate into the relay host SMTP server. For SMTP servers that require authentication, specify the user name and password.

When used with **puresmtp list**, displays the password in masked form. A dash (-) in the Password column means that an SMTP password has not been set.

When used with **puresmtp setattr**, sets the SMTP password through interactive prompt. The SMTP user name and password must be set together. To clear the password, press **Enter** at the command prompts. Clearing the password implicitly clears the SMTP user name.

**--relay-host *RELAY-HOST***

Displays or sets the hostname or IP address of the email relay server currently being used as a forwarding point for email alerts generated by the array. If specifying an IP address, enter the IPv4 or IPv6 address. If a relay host is not configured, Purity//FA sends alert email messages directly to recipient addresses rather than route them via the relay (mail forwarding) server.

For IPv4, specify the IP address in the form **ddd.ddd.ddd.ddd**, where **ddd** is a number ranging from 0 to 255 representing a group of 8 bits. If a port number is also specified, append it to the end of the address in the format **ddd.ddd.ddd.ddd:PORT**, where **PORT** represents the port number.

For IPv6, specify the IP address in the form

**xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx**, where **xxxx** is a hexadecimal number representing a group of 16 bits. Consecutive fields of zeros can be shortened by replacing the zeros with a double colon (**::**). If a port number is also specified, enclose the entire address in square brackets ([ ]) and append the port number to the end of the address. For example, **[xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx]:PORT**, where **PORT** represents the port number.

**--sender-domain *SENDER-DOMAIN***

Displays or sets the sender domain name. The domain name determines how Purity//FA logs are parsed and treated by Pure Storage Support and Escalations. The domain name is also

appended to alert email messages. The default domain name is `please-configure.me`. If this is not a Pure Storage test array, set the sender domain to the actual customer domain name.

**--user-name *USER-NAME***

Displays or sets the SMTP account name used to authenticate into the relay host SMTP server. For SMTP servers that require authentication, specify the user name and password. The SMTP user name and password must be set together.

To clear the user name, set to an empty string (""). Clearing the user name implicitly clears the SMTP password. A dash (-) in the User Name column means that an SMTP user name has not been set.

Options that control display format:

**--cli**

Displays output in the form of CLI commands that can be issued to reproduce the current configuration. The `--cli` output is not meaningful when combined with immutable attributes.

**--csv**

Lists information in comma-separated value (CSV) format. The `--csv` output can be used for scripting purposes and imported into spreadsheet programs.

**--notitle**

Lists information without column titles.

**--nvp**

Lists information in name-value pair (NVP) format, in the form `ITEMNAME=VALUE`. Argument names and information items are displayed flush left. The `--nvp` output is designed both for convenient viewing of what might otherwise be wide listings, and for parsing individual items for scripting purposes.

## Description

The relay host represents the hostname or IP address of the email relay server currently being used as a forwarding point for alert email notifications generated by the array. The `puresmtp setattr --relay-host` command sets the relay host. For SMTP servers that require authentication, also include the `--user-name` and `--password` options to specify the username and password, respectively. If a relay host is not configured, Purity//FA sends alert email messages directly to recipient addresses rather than route them via the relay (mail forwarding) server.

The `puresmtp setattr --sender-domain` command sets the sender domain name to determine how Purity//FA logs are parsed and treated by Pure Storage Support and Escalations. It is crucial that you set the sender domain to the correct domain name. If the array is not a Pure Storage test array, set the sender domain to the actual customer domain name. For example, `mycompany.com`. If the sender domain is set to the default `please-configure.me` domain name, the array will be treated as an internal Pure Storage test array, potentially blocking alerts and other important warnings. Purity//FA also includes the sender domain name in the email addresses of outgoing alert messages. For example, `pure-ct0@mycompany.com`.

The `puresmtp list` command displays the SMTP relay host settings, including relay host name, SMTP user name and password, and sender domain.

## Examples

### Example 1

```
puresmtp list --csv
```

Displays a CSV output of the SMTP relay host settings, including relay host name, SMTP user name and password, and sender domain.

### Example 2

```
puresmtp setattr --relay-host relayhostname --user-name smtpusername --password  
Enter password for relayhost:  
Retype password for relayhost:
```

Sets the SMTP relay host to `relayhostname`. Also sets the SMTP user name and password, which are used to authenticate into the SMTP server. The password is entered interactively.

### Example 3

```
puresmtp setattr --user-name ""
```

Clears the SMTP user name, which also clears the SMTP password.

### Example 4

```
puresmtp setattr --password  
Enter password for relayhost: <Press Enter>  
Retype password for relayhost: <Press Enter>
```

Clears the SMTP password, which also clears the SMTP user name.

### Example 5

```
puresmtp setattr --sender-domain mycompany.com
```

Sets the sender domain name to `mycompany.com`.

## See Also

`purearray(1)` [230]

## Author

Pure Storage Inc. <[documentfeedback@purestorage.com](mailto:documentfeedback@purestorage.com)>

## puresnmp

puresnmp, puresnmp-create, puresnmp-delete, puresnmp-list, puresnmp-rename, puresnmp-setattr, puresnmp-test — manages connections to Simple Network Management Protocol (SNMP) managers

### Synopsis

```
puresnmp create [--auth-passphrase] [--auth-protocol AUTH-PROTOCOL {MD5 | SHA}]  
[--community] --host HOST [--notification {trap | inform}] [--privacy-passphrase]  
[--privacy-protocol PRIVACY-PROTOCOL {AES | DES}] [--user USER] [--version {v2c |  
v3}] MANAGER  
puresnmp delete MANAGER...  
puresnmp list [ --cli | --csv | --nvp ] [--engine-id] [--notitle] [--page] [--raw]  
[MANAGER...]  
puresnmp rename OLD-NAME NEW-NAME  
puresnmp setattr [--auth-passphrase] [--auth-protocol AUTH-PROTOCOL {MD5 | SHA}]  
[--community] --host HOST [--notification {trap | inform}] [--privacy-passphrase]  
[--privacy-protocol PRIVACY-PROTOCOL {AES | DES}] [--user USER] [--version {v2c |  
v3}] MANAGER...  
puresnmp test MANAGER
```

### Arguments

#### **MANAGER**

Name used by Purity to identify an SNMP Network Management System ("Manager").

### Options

#### -h | --help

Can be used with any command or subcommand to display a brief syntax description.

#### --auth-passphrase

SNMPv3 only. Passphrase used by Purity to authenticate the array with the specified managers. Purity prompts for the passphrase. The passphrase must be at least 8 characters in length. Allowed characters are [A-Z], [a-z], [0-9], \_ (underscore), and - (hyphen). To clear the passphrase, press **Enter** at the prompts. Note that the authentication protocol must be configured in order to set the authentication passphrase.

#### --auth-protocol **AUTH-PROTOCOL**

SNMPv3 only. Hash algorithm used to validate the authentication passphrase. Valid values are **MD5**, **SHA**, or null (set to ""). Values are case sensitive.

#### --community

SNMPv2c only. Manager community string under which Purity is to communicate with the specified managers. Purity prompts twice for the community string. Allowed characters are [A-Z],

[a-z], [0-9], \_ (underscore), - (hyphen). To remove the array from the community, press **Enter** at the prompts.

--engine-id

SNMPv3 only. Displays the SNMPv3 engine ID generated by Purity for the array. (Some managers require the engine ID in their configuration.)

--host *HOST*

DNS hostname or IP address of a computer that hosts an SNMP manager to which Purity is to send messages when it generates alerts. If specifying an IP address, enter the IPv4 or IPv6 address.

For IPv4, specify the IP address in the form **ddd.ddd.ddd.ddd**, where **ddd** is a number ranging from 0 to 255 representing a group of 8 bits. If a port number is also specified, append it to the end of the address in the format **ddd.ddd.ddd.ddd:PORT**, where **PORT** represents the port number.

For IPv6, specify the IP address in the form

**xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx**, where **xxxx** is a hexadecimal number representing a group of 16 bits. Consecutive fields of zeros can be shortened by replacing the zeros with a double colon (::). If a port number is also specified, enclose the entire address in square brackets ([ ]) and append the port number to the end of the address. For example, **[xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx]:PORT**, where **PORT** represents the port number.

**Note:** This host option refers to SNMP managers and is not related to host options in other CLI commands.

--notification

Notification type that determines whether the recipient (remote host) acknowledges receipt of SNMP messages. Valid options are **trap** (default) and **inform**. An SNMP trap is an unacknowledged (asynchronous) SNMP message, meaning the recipient does not acknowledge receipt of the message. An SNMP inform request is an acknowledged trap. Values are case sensitive.

--privacy-passphrase

SNMPv3 only. Passphrase used to encrypt SNMP messages. Purity prompts for the passphrase, which must be at least 8 non-space ASCII characters. To clear the passphrase, press **Enter** at the prompts. Note that the privacy protocol must be configured in order to set the privacy passphrase.

--privacy-protocol **PRIVACY-PROTOCOL**

SNMPv3 only. Encryption protocol for SNMP messages. Valid values are **AES**, **DES**, or null (set to ""). Values are case sensitive. Note that the authentication settings must be configured in order to set the privacy options.

--user

Required for SNMPv3. User ID that Purity uses in communications with SNMP managers. Allowed characters are [A-Z], [a-z], [0-9], \_ (underscore), and - (hyphen).

--version

Version of the SNMP protocol to be used by Purity in communications with the specified manager(s). Valid values are **v2c** (default) and **v3**. Values are case sensitive.

Options that control display format:

--cli

Displays output in the form of CLI commands that can be issued to reproduce the current configuration. The --cli output is not meaningful when combined with immutable attributes.

--csv

Lists information in comma-separated value (CSV) format. The --csv output can be used for scripting purposes and imported into spreadsheet programs.

--notitle

Lists information without column titles.

--nvp

Lists information in name-value pair (NVP) format, in the form **ITEMNAME=VALUE**. Argument names and information items are displayed flush left. The --nvp output is designed both for convenient viewing of what might otherwise be wide listings, and for parsing individual items for scripting purposes.

--page

Turns on interactive paging.

--raw

Displays the unformatted version of column titles and data. For example, in the **purearray monitor** output, the unformatted version of column title **us/op (read)** is **usec\_per\_read\_op**. The --raw output is used to sort and filter list results.

## Description

The Simple Network Management Protocol (SNMP) is used by SNMP agents and SNMP managers to send and retrieve information. FlashArray supports SNMP versions v2c and v3.

In the FlashArray, the built-in SNMP agent has local knowledge of the array. The agent collects and organizes this array information and translates it via SNMP to or from the SNMP managers running in hosts. The agent, named **localhost**, cannot be deleted or renamed. The managers are defined by creating SNMP manager objects on the array. The managers communicate with the agent via the standard TCP port 161, and they receive notifications on port 162.

In the FlashArray, the **localhost** SNMP agent has two functions, namely, responding to GET-type SNMP requests and transmitting alert messages.

The agent responds to GET-type SNMP requests made by the SNMP managers, returning values for an information block, such as **purePerformance**, or individual variables within the block, depending on the type of request issued. The variables supported are:

<b>pureArrayReadBandwidth</b>	Current array-to-host data transfer rate
<b>pureArrayWriteBandwidth</b>	Current host-to-array data transfer rate
<b>pureArrayReadIOPS</b>	Current read request execution rate

<code>pureArrayWriteIOPS</code>	Current write request execution rate
<code>pureArrayReadLatency</code>	Current average read request latency
<code>pureArrayWriteLatency</code>	Current average write request latency

The FlashArray Management Information Base (MIB) describes the purePerformance variables. Download the MIB through the **Settings > System > SNMP** panel of the GUI.

The SNMP agent generates and transmits messages to the SNMP manager as traps or inform requests (informs), depending on the notification type that is configured on the manager. An SNMP trap is an unacknowledged SNMP message, meaning the SNMP manager does not acknowledge receipt of the message. An SNMP inform is an acknowledged trap.

To configure the notification type, include the **--notification** option when creating or modifying the attributes of the SNMP manager. If the SNMP manager notification type is set to **trap**, the agent sends the SNMP message (trap) without expecting a response. If the SNMP manager is set to **inform**, the agent sends the SNMP message (inform) and waits for a reply from the manager confirming message retrieval. If the agent does not receive a response within a certain timeframe, it will retry until the inform has passed through successfully. If the notification type is not set, the manager defaults to **trap**.

The **puresnmp create** command creates a Purity SNMP manager object. Include the required **--host** and **--version** options when creating an SNMP manager object. Include the **--host** option to specify the DNS hostname or IP address of the computer that hosts the SNMP manager. Include the **--version** option to specify the version of the SNMP protocol. Valid versions are **v2c** and **v3**.

SNMPv2 uses a type of password called a community string to authenticate the messages that are passed between the agent and manager. The community string is sent in clear text, which is considered an unsecured form of communication. SNMPv3 supports secure communication between the agent and manager through the use of authentication and privacy encryption methods. As such, SNMPv2c and SNMPv3 have different security attributes.

To configure the SNMPv2c agent and managers, include the **--community** option to set the community string under which the agent is to communicate with the managers. The agent and manager must belong to the same community; otherwise, the agent will not accept requests from the manager. When setting the community, Purity prompts twice for the community string. To remove the agent or manager from the community, press **Enter** at the community prompts.

To configure the SNMPv3 agent and managers, include the **--user** option to specify the user ID that Purity uses to communicate with the SNMP manager. Also set the authentication and privacy encryption security levels for the agent and managers. SNMPv3 supports the following security levels:

- **noAuthNoPriv.** Authentication and privacy encryption is not set. Similar to SNMPv2c, communication between the SNMP agent and managers is not authenticated and not encrypted. noAuthNPriv security requires no configuration.
- **authNoPriv.** Authentication is set, but privacy encryption is not set. Communication between the SNMP agent and managers is authenticated but not encrypted. Password authentication is based on **MD5** or **SHA** hash authentication. To configure authNoPriv security, set the **--auth-protocol** and **--auth-passphrase** attributes.
- **authPriv.** Communication between the SNMP agent and managers is authenticated and encrypted. Password authentication is based on **MD5** or **SHA** hash authentication. Traffic

between the FlashArray and SNMP manager is encrypted using encryption protocol **AES** or **DES**. To configure authPriv security, set the **--auth-protocol**, **--auth-passphrase**, **--privacy-protocol**, and **--privacy-passphrase** attributes.

Note that privacy cannot be configured without authentication.

Once an SNMP manager is created, the FlashArray immediately starts transmitting SNMP messages and alerts to the manager.

The **puresnmp list** command displays the communication and security attributes of the SNMP agent and managers. Include the **--engine-id** option to display the array's engine ID, which may be required to configure some of the SNMP managers.

The **puresnmp rename** command changes the name of the specified SNMP manager object. SNMP manager names are used in Purity administrative commands, and have no external significance. The name change is effective immediately. The SNMP **localhost** agent cannot be renamed.

The **puresnmp setattr** command changes the hostname, IP address, notification, or SNMP version, corresponding protocol and security attributes, and notification type of the specified SNMP manager.

Changing the SNMP version clears all of the previous version settings.

The **puresnmp test** command sends a test SNMP message (trap or inform) to the specified manager. For SNMP traps, successful transmission is verified at the destination. For SNMP informs, the recipient sends a response, acknowledging receipt of the SNMP message.

The **puresnmp delete** command stops communication with the specified SNMP manager and deletes the SNMP manager object from Purity.

## Examples

### Example 1

```
puresnmp create --host mysnmp.example.com --community MySNMPManager  
Enter community:  
Retype community:
```

Creates an SNMP manager object named **MySNMPManager** for the SNMP manager running in host **mysnmp.example.com**. Purity prompts for the new community string. Purity uses SNMPv2c to communicate in the new SNMP community.

### Example 2

```
puresnmp delete MySNMPManager
```

Stops transmission of any future messages to **MySNMPManager** and deletes the SNMP manager. If **MySNMPManager** is recreated at a later time, its attributes must be re-defined.

### Example 3

```
puresnmp list MySNMPManager
```

Displays the protocol and security attributes for **MySNMPManager**.

## Example 4

```
puresnmp rename MySNMPManager MySNMPManager1
```

Changes the name of **MySNMPManager** to **MySNMPManager1**. None of the manager object's protocol or security attributes are changed.

## Example 5

```
puresnmp setattr --host 172.169.0.12 MySNMPManager
```

Changes the IPv4 address associated with **MySNMPManager** to **172.169.0.12**.

## Example 6

```
puresnmp setattr --host [2001:db8:85a3::8a2e:370:7334] MySNMPManager
```

Changes the IPv6 address associated with **MySNMPManager** to **2001:db8:85a3::8a2e:370:7334**.

## Example 7

```
puresnmp setattr --auth-passphrase MySNMPManager
```

Enter auth passphrase:

Retype auth passphrase:

Name	Host	Version	Community	User	Auth Protocol	Auth Pass	...
MyV3Mgr	mysnmp.example.com	v3	-	User1	SHA	****	...

For an existing SNMPv3 manager named **MySNMPManager**, sets the authentication passphrase to a new value. Purity prompts for the new authentication passphrase value.

## Example 8

```
puresnmp setattr --auth-protocol MD5 MySNMPManager
```

Name	Host	Version	Community	User	Auth Protocol	Auth Pass	...
MySNMPManager	mysnmp.example.com	v3	-	User1	MD5	****	...

For an existing SNMPv3 manager named **MySNMPManager**, sets the authentication protocol to **MD5**.

## Example 9

```
puresnmp setattr --auth-protocol SHA --auth-passphrase MySNMPManager
```

Enter auth passphrase:

Retype auth passphrase:

Name	Host	Version	Community	User	Auth Protocol	Auth Pass	...
MySNMPManager	mysnmp.example.com	v3	-	User1	SHA	****	...

For an existing SNMPv3 manager named **MySNMPManager**, sets the authentication protocol attribute to **SHA** and sets the authentication passphrase to a new value. Purity prompts for the new authentication passphrase value.

## Example 10

```
puresnmp setattr --community MySNMPManager
Enter community:
Retype community:
Name      Host          Version  Community  User    Auth Protocol  Auth Pass ...
MySNMPManager  mysnmp.example.com  v2c      ****      -      -      -      ...
...      ...      ...      ...      ...      ...      ...      ...      ...
```

For an existing SNMPv2c manager named **MySNMPManager**, sets the community string to a new value. Purity prompts for the new community string.

## Example 11

```
puresnmp setattr --version v3 --user User1 localhost
Name      Host          Version  Community  User    Auth Protocol  Auth Pass ...
localhost  localhost    v3      -          User1  -      -      -      ...
...      ...      ...      ...      ...      ...      ...      ...      ...
```

Sets the existing **localhost** agent to SNMPv3, with a user named **User1**. SNMP Managers can now communicate with the built-in SNMP agent using the SNMPv3 protocol.

## Example 12

```
puresnmp setattr --notification inform MySNMPManager
Name      Host          Version  Community  User    Auth Protocol  ...  Notification ...
MySNMPManager  mysnmp.example.com  v2c      ****      -      -      -      ...  inform
...      ...      ...      ...      ...      ...      ...      ...      ...
```

Sets the notification type for SNMP manager **MySNMPManager** to **inform**. SNMP messages sent will be acknowledged.

## See Also

[purearray\(1\)](#) [230], [puremessage\(1\)](#) [354]

## Author

Pure Storage Inc. <[documentfeedback@purestorage.com](mailto:documentfeedback@purestorage.com)>

## puresubnet

puresubnet, puresubnet-create, puresubnet-delete, puresubnet-disable, puresubnet-enable, puresubnet-list, puresubnet-rename, puresubnet-setattr — manages the subnets and VLANs used to organize the network interfaces

### Synopsis

```
puresubnet create [--gateway GATEWAY] [--mtu MTU] --prefix PREFIX [--vlan VLAN] SUBNET
puresubnet delete SUBNET
puresubnet disable SUBNET...
puresubnet enable SUBNET...
puresubnet list [ --cli | --csv | --nvp ] [--notitle] [--page] [--raw] [--vlan VLAN] [SUBNET...]
puresubnet rename OLD-NAME NEW-NAME
puresubnet setattr [--gateway GATEWAY] [--mtu MTU] --prefix PREFIX [--vlan VLAN] SUBNET
```

### Arguments

#### **SUBNET**

Subnet name.

#### **NEW-NAME**

Name by which the subnet is to be known after the command executes.

#### **OLD-NAME**

Current name of the subnet to be renamed.

### Options

#### **-h | --help**

Can be used with any command or subcommand to display a brief syntax description.

#### **--gateway *GATEWAY***

IP address of the gateway through which the specified interface is to communicate with the network. For IPv4, specify the gateway IP address in the form *ddd.ddd.ddd.ddd*. For IPv6, specify the gateway IP address in the form *xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx*. When specifying an IPv6 address, consecutive fields of zeros can be shortened by replacing the zeros with a double colon (`::`).

To remove the gateway specification, set to a null value (""). The interfaces in the subnet inherit the gateway address.

--mtu

Specifies the maximum message transfer unit (packet) size for the subnet in bytes. Valid values are integers between 568 and 9000 (inclusive). The default value is 1500. The interfaces in the subnet inherit the MTU value.

--prefix **PREFIX**

IP address of the subnet prefix and prefix length. For IPv4, specify the subnet prefix in the form **ddd.ddd.ddd.ddd/dd**. For IPv6, specify the subnet prefix in the form **xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx/xxx**. When specifying an IPv6 address, consecutive fields of zeros can be shortened by replacing the zeros with a double colon (**::**).

--vlan **VLAN**

VLAN ID number. Specify the VLAN ID if the subnet is being created for VLAN tagging purposes. The VLAN interfaces in the subnet inherit the VLAN ID. To remove the VLAN ID, set --vlan to 0.

Options that control display format:

--cli

Displays output in the form of CLI commands that can be issued to reproduce the current configuration. The --cli output is not meaningful when combined with immutable attributes.

--csv

Lists information in comma-separated value (CSV) format. The --csv output can be used for scripting purposes and imported into spreadsheet programs.

--notitle

Lists information without column titles.

--nvp

Lists information in name-value pair (NVP) format, in the form **ITEMNAME=VALUE**. Argument names and information items are displayed flush left. The --nvp output is designed both for convenient viewing of what might otherwise be wide listings, and for parsing individual items for scripting purposes.

--page

Turns on interactive paging.

--raw

Displays the unformatted version of column titles and data. For example, in the **purearray monitor** output, the unformatted version of column title **us/op (read)** is **usec\_per\_read\_op**. The --raw output is used to sort and filter list results.

## Description

In Purity//FA, interfaces with common attributes can be organized into subnetworks, or subnets, to enhance the efficiency of data (iSCSI or NVMe-RoCE), management, and replication traffic.

If the subnet is assigned a valid IP address, once it is created, all of its enabled interfaces are immediately available for connection. The subnet inherits the services from all of its interfaces.

Likewise, the interfaces contained in the subnet inherit the netmask, gateway, MTU, and VLAN ID (if applicable) attributes from the subnet.

Physical, virtual, and bond interfaces can belong to the same subnet. VLAN interfaces can only belong to subnets with other VLAN interfaces.

*Physical, virtual, and bond interfaces* in a subnet share common address, netmask, and MTU attributes. The subnet can contain a mix of physical, virtual, and bond interfaces, and the interface services can be of any type, such as iSCSI, management, NVMe-RoCE, or replication services. To add physical, virtual, or bond interfaces to a subnet, create the subnet and then run `purenw` `setattr --subnet` to add the interfaces to the subnet.

A *VLAN interface* is a dedicated virtual network interface that is designed to be used with an organization's virtual local area network (VLAN). Through VLAN interfaces, Purity//FA employs VLAN tags to ensure the data passing between the array and VLANs is securely isolated and routed properly.

## VLAN Tagging

VLAN tagging allows customers to isolate traffic through multiple virtual local area networks (VLANs), ensuring data routes to and from the appropriate networks. The array performs the work of tagging and untagging the data that passes between the VLAN and array.

VLAN is only supported for the iSCSI service type, so before creating a VLAN interface, verify the iSCSI service is configured on the physical interface.

Each port can support multiple VLANs. Note that each array supports up to 32 VLAN IDs, so if multiple hosts connect through multiple VLANs, it is possible to quickly reach the maximum number of paths that a host would support.

The `puresubnet create` command creates subnets.

To create and add VLAN interfaces to a subnet, first, create a subnet and assign it with a VLAN ID number, and second, create a VLAN interface for each of the corresponding physical network interfaces you want to associate with the VLAN. Run `puresubnet create --prefix --vlan` to create a subnet for each of the VLANs in the organization, assigning each subnet with the appropriate VLAN ID. For each of the physical network interfaces you want to associate with a VLAN, run `purenw` `create vif --subnet` to create a VLAN interface (`vif`) and add it to the subnet. Include the `--address` option to create a reachable VLAN interface that is ready to use. All of the VLAN interfaces within a subnet must be in the same VLAN.

In Purity//FA, VLAN interfaces have the naming structure `CTx.EThy.z`, where `x` denotes the controller (0 or 1), `y` denotes the interface (0 or 1), and `z` denotes the VLAN ID number. For example, `ct0.eth1.500`.

In the following example, subnet `192.168.100.0/24`, named `ESXHost001` and assigned to VLAN 50, is being created. The physical interfaces `ct0.eth4` and `ct0.eth5` are being added to subnet `192.168.100.0/24` (named `ESXHost001`) as VLAN interfaces, with VLAN ID number 50 appended to the interface name to match the VLAN ID number of the subnet.

```
puresubnet create --gateway 192.168.1.1 --mtu 9000 --prefix 192.168.100.0/24  
          --vlan 50 ESXHost001  
purenw create vif --address 192.168.1.10 --subnet ESXHost001 ct0.eth4.50
```

```
purenetwork create vif --address 192.168.1.11 --subnet ESXHost001 ct0.eth5.50
```

If a physical interface has multiple VLANs, create a subnet for each VLAN, and then create the VLAN interfaces mapping the physical interface to the given VLANs.

When creating the VLAN interface, the VLAN ID number in the VLAN interface name must match the VLAN ID of the subnet. In the following example, the second command will not work because the VLAN ID number of the `ESXHost002` subnet (600) does not match the VLAN ID number in the VLAN interface name (500).

```
puresubnet create --prefix 2001:db8:85a3::/64 --gateway 2001:0db8:85a3::1
                  --vlan 600 ESXHost002
purenetwork create vif --address 2001:0db8:85a3::ae26:8a2e:0370:7334
                  --subnet ESXHost002 ct0.eth1.500
```

If `--mtu` is not specified during subnet creation, the value defaults to 1500. Note that the MTU of a VLAN interface cannot exceed the MTU of the corresponding physical interface. Since an interface inherits the MTU value of its subnet, verify the MTU of the new subnet is valid.

Once a subnet is created, the interfaces within the subnet that are enabled are automatically available and ready to connect.

The `puresubnet delete` command deletes subnets that are no longer needed. A subnet can only be deleted if it is empty, so before you delete a subnet, remove all of its interfaces by running `purenetwork delete`.

The `puresubnet enable` and `puresubnet disable` commands respectively enable and disable a subnet.

When a subnet is enabled, the interfaces within the subnet that are enabled are automatically available and ready to connect. Interfaces within the subnet that are in disable status remain disabled and cannot be reached. Newly created subnets are automatically enabled.

Disabling a subnet disables all of its interfaces - including the ones that are enabled at the interface level.

Take caution when disabling a subnet. If you disable a subnet that contains interfaces through which administrative sessions are being conducted, the interfaces will lose SSH connection.

The `puresubnet list` command displays the attributes of the subnets, including its physical, virtual, bond, and VLAN interfaces. Include the `--vlan` option to list only the subnets that are configured with the specified VLAN ID.

The `puresubnet rename` command changes the current (OLD-NAME) name of a subnet to the new name (NEW-NAME). The name change is effective immediately and the old name is no longer recognized in CLI, GUI, or REST interactions. Renaming a subnet does not affect any of the interface connections attached to the subnet.

The `puresubnet setattr` subcommand modifies the subnet attributes, including subnet gateway, MTU size, prefix, and VLAN ID.

The **--gateway** and **--mtu** options can be set at any time. Note that the MTU of a VLAN interface cannot exceed the MTU of the corresponding physical interface. Since an interface inherits the MTU value of its subnet, verify the MTU is valid.

The **--prefix** and **--vlan** options can only be set if the subnet does not contain interfaces. Only specify the VLAN ID number if the subnet is being created for VLAN tagging purposes. To remove the VLAN ID from a subnet, set **--vlan** to 0.

## Examples

### Example 1

```
puresubnet create --prefix 192.168.1.0/24 --vlan 100 ESXHost001
```

Creates subnet **ESXHost001** with prefix **192.168.1.0/24** and VLAN ID **100**.

### Example 2

```
$ puresubnet create --prefix 2001:db8:85a3::/64 --vlan 200 ESXHost002
```

Creates subnet **ESXHost002** with prefix **2001:db8:85a3::/64** and VLAN ID **200**.

### Example 3

```
puresubnet delete ESXHost001
```

Deletes subnet **ESXHost001**. Subnets can only be deleted if they do not contain network interfaces.

### Example 4

```
puresubnet setattr --vlan 50 ESXHost001
```

Sets subnet **ESXHost001** to VLAN ID **50**.

### Example 5

```
puresubnet list --vlan 50
```

Displays only subnets set to VLAN ID **50**.

## See Also

**purearray(1)** [230]

## Author

Pure Storage Inc. <[documentfeedback@purestorage.com](mailto:documentfeedback@purestorage.com)>

## puresw

puresw, puresw-app, puresw-global, puresw-install, puresw-list — manages software updates and app installations and upgrades

### Synopsis

```
puresw app install APP
puresw app uninstall APP...
puresw app list [--catalog] [ --cli | --csv | --nvp ] [--notitle] [--page] [--raw]
[--filter FILTER] [--limit LIMIT] [--sort SORT] [APP...]
puresw global enable {--auto-download}
puresw global disable {--auto-download}
puresw global list [--auto-download] [ --cli | --csv | --nvp ] [--notitle] [--page]
[--raw]
puresw install {--version VERSION} SOFTWARE
puresw list [ --csv | --nvp ]
```

### Options

**-h | --help**

Can be used with any command or subcommand to display a brief syntax description.

**--auto-download**

When enabled, automatically downloads the latest software installation files to the array in preparation for a software update. Must be used with the **enable** or **disable** option.

**--catalog**

Displays a list of apps that are either installed or available to be installed on the array. Newer versions of currently installed apps are also listed in the app catalog.

**--version *VERSION***

Updates the software to the specified version.

Options that control display format:

**--cli**

Displays output in the form of CLI commands that can be issued to reproduce the current configuration. The **--cli** output is not meaningful when combined with immutable attributes.

**--csv**

Lists information in comma-separated value (CSV) format. The **--csv** output can be used for scripting purposes and imported into spreadsheet programs.

**--notitle**

Lists information without column titles.

--nvp

Lists information in name-value pair (NVP) format, in the form **ITEMNAME=VALUE**. Argument names and information items are displayed flush left. The --nvp output is designed both for convenient viewing of what might otherwise be wide listings, and for parsing individual items for scripting purposes.

--page

Turns on interactive paging.

--raw

Displays the unformatted version of column titles and data. For example, in the **purearray monitor** output, the unformatted version of column title **us/op (read)** is **usec\_per\_read\_op**. The --raw output is used to sort and filter list results.

Options that manage display results:

--filter

Displays only the rows that meet the filter criteria specified.

--limit

Limits the size of the list output to the specified maximum number of rows.

--sort

Sorts the list output in ascending or descending order by the column specified.

## Arguments

### **SOFTWARE**

Software name.

### **APP**

App name.

## Description

The **puresw** command manages Purity software updates and app installations.

The **puresw global** command manages the configuration setting across all software updates. The **puresw global enable --auto-download** command enables Auto Download. If Auto Download is enabled, any software installation files that Pure Storage Support sends to the array will be automatically downloaded and ready to install. Likewise, the **puresw global disable --auto-download** command disables Auto Download. If Auto Download is disabled, the software installation files that Pure Storage Support sends to the array will only be downloaded during the software update process. You cannot change the Auto Download status while a software update is in progress. Note that the Auto Download feature impacts software updates only. The Auto Download feature does not impact Purity apps. Auto Download is disabled by default.

The **puresw global list** command displays the current global configuration that affects all software updates. Include the **--auto-download** option to display the global configuration for the Auto Download feature.

In the following example, the `puresw global list --auto-download` command displays the global Auto Download status for software updates. Auto Download is set to `enabled`, which means that any software installation files that Pure Storage Support sends to the array will be automatically downloaded and ready to install. Purity apps are not affected by global settings, so as Purity apps become available, those installation files are not automatically downloaded to the array; instead, they are downloaded to the array during the app installation process.

```
$ puresw global list
Auto Download
enabled
```

## Software Updates

Software updates add or enhance Purity features and functionality. Perform periodic software updates to get the most out of your Purity system. To perform a software update, contact Pure Storage Support to obtain the software installation files required for the update.

Run the `puresw list` command to display a list of software versions available for update and the installation status of each software version. A software version that is available for update will have one of the following statuses:

- **Available:** A software update for this version is available, but the installation files have not been downloaded to the array. Instead, the files will be downloaded to the array during the installation process. When scheduling the software update, make sure to factor enough time for the download process.
- **Downloaded:** The installation files for this software version have been successfully downloaded to the array.

Run the `puresw install` command to run the software update. Include the required `--version` option to specify the software version to be updated. As the software update process progresses, the following statuses will appear:

- **Downloading:** Purity is downloading the installation files to the array for this software version. If Auto Download is enabled, any software installation files that Pure Storage Support sends to the array will be automatically downloaded and ready to install. If Auto Download is disabled, the software installation files that Pure Storage Support sends to the array will only be downloaded during the software update process.
- **Installing:** Purity is updating the software. Be prepared to be logged out of the software during the update process.

During the update process, you will be logged out of the software. After you have been logged out, log back in to continue monitoring the process. The software update process is complete when the software update no longer appears in the `puresw list` output.

If the software update fails to fully complete, the system reverts to the previous software version, ready to resume operations. If the update path spans multiple versions, the system stops the installation process at the most current successfully installed, highly-available software version. If you encounter any problems during the update process, contact Pure Storage Support.

## App Installations

The Purity Run platform extends array functionality by integrating add-on services into the Purity//FA operating system. Each service that runs on the platform is provided by an app.

The **puresw app** command manages the app installation process. Apps require CPU, memory, network, and storage resources. For this reason, apps are not installed by default.

Run the **puresw app list** command to display a list of apps that are currently installed on the array, along with the following attributes for each app:

- **Name:** App name. The app name is pre-assigned and cannot be changed.
- **Version:** App version that is currently installed on the array.
- **Status:** Status of the app installation. Possible app statuses include:
  - **Available:** App (new or upgraded version of an existing one) is available to be installed.
  - **Downloading:** App installation files are being downloaded in preparation for an installation.
  - **Downloaded:** App installation files have been successfully downloaded. Installation will begin shortly.
  - **Installing:** App is currently being installed.
  - **Installed:** App has been successfully installed.
  - **Uninstalling:** App is currently being uninstalled.
  - **Aborted:** App installation process has encountered issues. The installation is rolling back. If the app is no longer available, it will not reappear in the list; otherwise, it will eventually return to its previous "Available" status. If the app is available, try the installation again. If you continue to encounter issues, contact Pure Storage Support.
- **Progress:** Download progress during the installation process.
- **Description:** Description of the app.

The **puresw app list --catalog** command displays a list of apps, both available to and installed on the array. If a newer version of a currently installed app is available, it will also be listed in the app catalog.

The **puresw app install** command installs a new app or upgrades a currently installed app. An app can be installed at any time with no disruption to array activity, though the installation of larger, more intensive apps may impact performance.

The app installation process is three-fold: first, the latest installation files are downloaded to the array in preparation for the app installation, second, the app is installed, and third, the app is enabled. The app installation process is complete when the app is enabled.

Once the app installation process is complete, run the **pureapp list** command to verify its status. The app is ready to service requests when it is in a healthy status. If the app is in an unhealthy status, refer to **pureapp(1)** [221] to perform additional steps to bring the app to a healthy status.

After an app has been installed, run the `pureapp` command to manage the app. For more information about apps, refer to `pureapp(1)` [221].

When an app is no longer needed, run the `puresw app uninstall` command to uninstall the app. An app must be disabled before it can be uninstalled. Run the `pureapp disable` command to disable an app. After an app has been uninstalled, the volumes connected to the app host and the data volume that was created during app installation remain on the array, but they can no longer reach the app service. The boot volume that was created during app installation is deleted.

## Examples

### Example 1

```
puresw list
```

Displays a list of software versions that are either running on the array or available for update.

### Example 2

```
puresw global disable --auto-download
```

Disables Auto Download. Any software installation files that Pure Storage Support sends to the array will only be downloaded during the software update process.

### Example 3

```
puresw check --version 4.10.0 Purity
```

Performs a series of checks to ensure the array is healthy and prepared for the software update to Purity 4.10.0.

### Example 4

```
puresw install Purity//FA --version 5.0.6
```

Updates the Purity software to version 5.0.6.

### Example 5

```
puresw app list
```

Displays a list of installed apps and the installation status of each app.

### Example 6

```
puresw app list --catalog
```

Displays a list of apps, both available to and installed on the array.

### Example 7

```
puresw app install offload
```

Installs the **offload** app.

#### Example 8

```
puresw app list --catalog  
puresw app install linux  
puresw app list
```

Displays a list of apps that are available to be downloaded and installed. Downloads, installs, and enables the **linux** app. Displays the progress of the installation process.

#### Example 9

```
pureapp disable linux  
puresw app uninstall linux  
pureapp list
```

Disables the **linux** app in preparation for the uninstallation. Uninstalls the **linux** app. Displays a list of installed apps to verify that the **linux** app has been successfully uninstalled and no longer appears in the list.

#### See Also

[pureapp\(1\) \[221\]](#)

#### Author

Pure Storage Inc. <[documentfeedback@purestorage.com](mailto:documentfeedback@purestorage.com)>

## purevgroup

purevgroup, purevgroup-create, purevgroup-destroy, purevgroup-eradicate, purevgroup-recover, purevgroup-rename — manage the creation, naming, and destruction of Purity//FA volume groups  
 purevgroup-list, purevgroup-listobj — display volume group attributes and information about the virtual and physical storage space of all volumes within the volume group  
 purevgroup-monitor — monitor the I/O and QoS performance of all volumes in the volume group  
 purevgroup-setattr — manage the QoS limits of volume groups

### Synopsis

```

purevgroup create [--bw-limit BANDWIDTH-LIMIT] [--iops-limit IOPS-LIMIT]
VGROUP...

purevgroup destroy VGROUP...

purevgroup eradicate VGROUP...

purevgroup list [ --historical { 1h | 3h | 24h | 7d | 30d | 90d | 1y } ] [ --
pending | --pending-only ] [ --qos ] [ --space ] [ --cli | --csv | --nvp ] [ --notitle ]
[ --raw ] [ --filter FILTER] [ --limit LIMIT] [ --page ] [ --sort SORT] [ VGROUP... ]

purevgroup listobj [ --pending | --pending-only ] [ --type { vgroup | vol } ] [ --
csv ] [ VGROUP... ]

purevgroup monitor [ --interval SECONDS] [ --latency ] [ --qos ] [ --repeat REPEAT-COUNT ]
[ --size ] [ --total ] [ --historical { 1h | 3h | 24h | 7d | 30d | 90d | 1y } ] [ --
csv ] [ --notitle ] [ --raw ] [ --filter FILTER] [ --limit LIMIT] [ --page ] [ --sort
SORT] [ VGROUP... ]

purevgroup recover VGROUP...

purevgroup rename OLD-NAME NEW-NAME

purevgroup setattr [--bw-limit BANDWIDTH-LIMIT] [--iops-limit IOPS-LIMIT]
VGROUP...
  
```

### Arguments

#### **NEW-NAME**

Name by which the volume or volume snapshot is to be known after the command executes.

#### **OLD-NAME**

Current name of the volume or volume snapshot to be renamed.

#### **VGROUP**

Volume group to be created, renamed, destroyed, eradicated, recovered, viewed, or monitored.

### Object Names

Valid characters are letters (A-Z and a-z), digits (0-9), and the hyphen (-) character. The first and last characters of the name must be alphanumeric, and the name must contain at least one letter or '-'.

Most objects in Purity//FA that can be named, including host groups, hosts, volumes, protection groups, volume and protection group suffixes, SNMP managers, and subnets, can be 1-63 characters in length.

Array names can be 1-56 characters in length. The array name length is limited to 56 characters so that the names of the individual controllers, which are assigned by Purity//FA based on the array name, do not exceed the maximum allowed by DNS.

Names are case-insensitive on input. For example, **vol1**, **Vol1**, and **VOL1** all represent the same volume. Purity//FA displays names in the case in which they were specified when created or renamed.

Pods and volume groups provide a namespace with unique naming conventions.

All objects in a pod have a fully qualified name that include the pod name and object name. The fully qualified name of a volume in a pod is **POD::VOLUME**, with double colons (::) separating the pod name and volume name. The fully qualified name of a protection group in a pod is **POD::PGROUP**, with double colons (::) separating the pod name and protection group name. For example, the fully qualified name of a volume named **vol01** in a pod named **pod01** is **pod01::vol01**, and the fully qualified name of a protection group named **pgroup01** in a pod named **pod01** is **pod01::pgroup01**.

If a protection group in a pod is configured to asynchronously replicate data to a target array, the fully qualified name of the protection group on the target array is **POD:PGROUP**, with single colons (:) separating the pod name and protection group name. For example, if protection group **pod01::pgroup01** on source array **array01** asynchronously replicates data to target array **array02**, the fully qualified name of the protection group on target array **array02** is **pod01:pgroup01**.

All objects in a volume group have a fully qualified name that includes the volume group name and the object name, separated by a forward slash (/). For example, the fully qualified name of a volume named **vol01** in a volume group named **vgroup01** is **vgroup01/vol01**.

## Volume Sizes

Volume sizes are specified as an integer, followed by one of the suffix letters **K**, **M**, **G**, **T**, **P**, representing KiB, MiB, GiB, TiB, and PiB, respectively, where "Ki" denotes  $2^{10}$ , "Mi" denotes  $2^{20}$ , and so on.

Volumes must be between one megabyte and four petabytes in size. If a volume size of less than one megabyte is specified, Purity//FA adjusts the volume size to one megabyte. If a volume size of more than four petabytes is specified, the Purity//FA command fails.

Volume sizes cannot contain digit separators. For example, **1000g** is valid, but **1,000g** is not.

## Options

**-h | --help**

Can be used with any command or subcommand to display a brief syntax description.

**--historical TIME**

When used with **purevgroup list --space**, displays historical size and space consumption information over the specified range of time. Valid time ranges include: 1 hour, 3 hours, 24 hours, 7 days, 30 days, 90 days, and one year.

When used with **purevgroup monitor**, displays historical performance data over the specified range of time. Valid time ranges include: 1 hour, 3 hours, 24 hours, 7 days, 30 days, 90 days, and one year.

--bw-limit **BANDWIDTH-LIMIT**

Sets the maximum QoS bandwidth limit for the volume groups. Whenever throughput exceeds the bandwidth limit, throttling occurs. If set, the bandwidth limit must be between 1 MB/s and 512 GB/s. The bandwidth limit is specified as an integer, optionally followed by the suffix letter **M** (for megabytes) or **G** (gigabytes).

--interval **SECONDS**

Sets the number of seconds between displays of real-time performance data. At each interval, the system displays a point-in-time snapshot of the performance data. If omitted, the interval defaults to every 5 seconds.

--iops-limit **IOPS-LIMIT**

Sets the maximum IOPS limit for the volume groups. Whenever the number of I/O operations per second exceeds the IOPS limit, throttling occurs. If set, the IOPS limit must be between 100 and 100**M**. The IOPS limit is specified as an integer, optionally followed by the suffix **K** ( $10^3$ ) or **M** ( $10^6$ ).

--latency

Displays real-time and historical I/O latency information.

--pending

Includes destroyed volumes or snapshots that are in the eradication pending state. If not specified, volumes or snapshots that are pending eradication are not shown.

--pending-only

Only displays destroyed volumes or snapshots that are in the eradication pending state.

--qos

Displays the QoS information for each volume group, including the bandwidth limit and the IOPS limit. If the bandwidth limit is not set, the value appears as a dash (-), representing unlimited throughput. If the IOPS limit is not set, the value appears as a dash (-), representing unlimited IOPS.

--repeat **REPEAT-COUNT**

Sets the number of times to display real-time performance data. If omitted, the repeat count defaults to 1.

--size

For **purevol create**, sets the virtual capacity of the volume as perceived by hosts.

For **purevol monitor**, displays the average I/O sizes per read and write operation.

--space

Displays the following information about provisioned (virtual) size and physical storage consumption for each specified volume:

**Size**

Total provisioned size of the volume. Represents storage capacity reported to hosts.

**Thin Provisioning**

Percentage of volume sectors that do not contain host-written data because the hosts have not written data to them or the sectors have been explicitly trimmed.

**Data Reduction**

Ratio of mapped sectors within a volume versus the amount of physical space the data occupies after data compression and deduplication. The data reduction ratio does not include thin provisioning savings.

For example, a data reduction ratio of 5:1 means that for every 5 MB the host writes to the array, 1 MB is stored on the array's flash modules.

**Total Reduction**

Ratio of provisioned sectors within a volume versus the amount of physical space the data occupies after reduction via data compression and deduplication *and* with thin provisioning savings. Total reduction is data reduction with thin provisioning savings.

For example, a total reduction ratio of 10:1 means that for every 10 MB of provisioned space, 1 MB is stored on the array's flash modules.

**Volume**

Physical space occupied by volume data that is not shared across volumes, excluding array metadata and snapshots.

**Snapshots**

Physical space occupied by data unique to one or more snapshots.

**Shared Space**

Physical space occupied by deduplicated data, meaning that the space is shared with other volumes and snapshots as a result of data deduplication.

**System**

Physical space occupied by internal array metadata.

**Total**

Total physical space occupied by system, shared space, volume, and snapshot data.

**--total**

Follows output lines with a single line containing column totals in columns where they are meaningful.

**--type**

Specifies the type of information (connected hosts, snapshots, or echoed volume names) about specified volumes to be produced in whitespace-separated list format suitable for scripting.

Options that control display format:

**--cli**

Displays output in the form of CLI commands that can be issued to reproduce the current configuration. The **--cli** output is not meaningful when combined with immutable attributes.

--csv

Lists information in comma-separated value (CSV) format. The --csv output can be used for scripting purposes and imported into spreadsheet programs.

--nvp

Lists information in name-value pair (NVP) format, in the form **ITEMNAME=VALUE**. Argument names and information items are displayed flush left. The --nvp output is designed both for convenient viewing of what might otherwise be wide listings, and for parsing individual items for scripting purposes.

--notitle

Lists information without column titles.

--raw

Displays the unformatted version of column titles and data. For example, in the **purearray monitor** output, the unformatted version of column title **us/op (read)** is **usec\_per\_read\_op**. The --raw output is used to sort and filter list results.

Options that manage display results:

--filter

Displays only the rows that meet the filter criteria specified.

--page

Turns on interactive paging.

--limit

Limits the size of the list output to the specified maximum number of rows.

--sort

Sorts the list output in ascending or descending order by the column specified.

## Description

Volume groups organize FlashArray volumes into logical groupings.

If virtual volumes are configured, each volume group on the FlashArray array represents its associated virtual machine, and inside each of those volumes groups are the FlashArray volumes that are assigned to the virtual machine. Volume groups that are associated with virtual machines have names that begin with "vvol-" and end with the virtual machine name. For more information about virtual volumes, including configuration steps, refer to the Pure Storage vSphere Web Client Plugin for vSphere User Guide in the Pure Storage Knowledge Base at <https://support.purestorage.com>.

Volume groups can also be created through the **purevgroup create** command. Once a volume group has been created, create new volumes directly in the volume group or move existing ones into the volume group.

## Creating Volume Groups

The **purevggroup create** command creates a volume group on the array. The volume group itself does not contain any meaningful content; instead, it acts as a container that is used to organize volumes.

Once a volume group has been created, volumes can be created inside the volume group or moved into and out of the volume group.

## Creating Volumes in Volume Groups

The **purevol create --size** command creates one or more Purity//FA virtual storage volumes of the specified size. To create a volume in a volume group, include the volume group namespace in the volume name.

For example, run the following command to create a volume named **vol03** in volume group **vgroup01**:

```
$ purevol create vol03 vgroup01 --size 1T
Name      Size  Source  Created          Serial
vgroup01/vol03  1T    -      2019-02-02 16:55:15 PST  23364F86CC654C2900011013
```

A volume can only be created inside an existing volume group; you cannot create a volume group while creating its volume.

## Moving Volumes into and out of Volume Groups

The **purevol move** command moves one or more volumes between storage containers. Run the **purevol move** command to move volumes from the root of the array to a volume group, or from a volume group to the root of the array ("").

For example, run the following command to move two volumes named **vol03** and **vol04** from the root of the array to volume group **vgroup01**:

```
$ purevol move vol03 vol04 vgroup01
Name      Size  Source  Created          Serial
vgroup01/vol03  1T    -      2019-02-02 16:55:15 PST  23364F86CC654C2900011013
vgroup01/vol04  1T    -      2019-02-02 16:55:15 PST  23364F86CC654C2900011013
```

You can move volumes between volume groups, but you cannot move volumes between pods and volume groups; To move a volume between a pod and a volume group, first move the volume to the root of the array, and then move the volume from the root of the array to the destination storage container.

For more information about moving volumes, refer to **purevol(1)** [469].

## Renaming Volume Groups

The **purevggroup rename** command changes the current (OLD-NAME) name of a volume group to the new name (NEW-NAME). The name change is effective immediately and the old name is no longer recognized in CLI, GUI, or REST interactions. In the Purity//FA GUI, the new name appears upon page refresh.

## Destroying Volume Groups

When a volume group is no longer needed, it can be destroyed. Run the **purevgroup destroy** command to destroy a volume group.

A volume group can only be destroyed if it is empty, so before destroying a volume group, ensure all volumes inside the volume group have been either moved out of the volume group or destroyed.

The destruction of volume groups is not immediate. Instead, the volume group is placed in a 24-hour eradication pending period. The volume group can be manually recovered or permanently eradicated within the eradication pending period. After 24 hours, the volume group is completely eradicated and not recoverable.

## Recovering Volume Groups

A destroyed volume can be recovered at any time during the 24-hour eradication pending period. Run the **purevgroup recover** command to recover a volume group.

Once a volume group has been recovered, its destroyed volumes can also be recovered to bring the volume group and its contents back to their previous state.

## Eradicating Volume Groups

A destroyed volume group can be permanently eradicated at any time during the 24-hour eradication pending period. Run the **purevgroup eradicate** command to eradicate a volume group.

Once a volume group has been eradicated, it can no longer be recovered.

## Listing Volume Group Details

The **purevgroup list** command displays a list of all volume groups on the array and their volumes. Include the **--pending** option to display a list of all volume groups, including ones that have been destroyed and are in the eradication pending state. Include **--pending-only** command to only display a list of volume groups that have been destroyed and are in the eradication pending state.

The **purevgroup listobj** command displays a list of volume group attributes, either in whitespace or comma-separated form, suitable for scripting.

Include the **--pending** option to display a list of all volume groups, including ones that have been destroyed and are in the eradication pending state. Include the **--pending-only** option to only display a list of volume groups that have been destroyed and are in the eradication pending state.

Include the **--qos** option to display the bandwidth limit and the IOPS limit for each volume group. If the bandwidth limit for a volume group is not set, the value appears as a dash (-), representing unlimited throughput. If the IOPS limit for a volume group is not set, the value appears as a dash (-), representing unlimited IOPS.

Include the **--type** option to list FlashArray objects that are associated with one or more volume groups.

The **--type** option accepts the following arguments:

```
--type vgroup (default if --type option not specified)
  Display a list of volume groups that have been created on the array. If no volume groups are
  specified, the list contains the names of all volume groups.

--type vol
  Displays a list of volumes that reside in volume groups. If no volume groups are specified, the list
  contains the names of all volumes that reside in all volume groups.
```

## Monitoring Volume Group I/O Performance

The **purevgroup monitor** command displays real-time and historical I/O performance information for all of the specified volume groups. The output includes the following bandwidth, IOPS, and latency data:

- **Name:** Object name.
- **Time:** Current time.
- **B/s:** Bandwidth. Number of bytes read/written.
- **op/s:** IOPS. Number of read, write, or mirrored write requests processed per second.
- **us/op:** Service time. Average time, measured in microseconds, it takes the array to serve a read, write, or mirrored write I/O request. Service time does not include SAN time, queue time, or QoS rate limit time.
- **B/op:** IOPS. Average I/O size per read, write, mirrored write, and both read and write (all) operations. Must include the **--size** option to see the B/op columns.
- **Bandwidth Limit:** Maximum QoS bandwidth limit. Must include the **--qos** option to see the Bandwidth Limit column.
- **IOPS Limit:** Maximum QoS IOPS limit. Must include the **--qos** option to see the IOPS Limit column.

## Monitoring Volume Group Latency

The **purevgroup monitor --latency** command displays real-time and historical I/O latency information for volumes in volume groups. The **purevgroup monitor --latency** output includes the following data:

- **SAN us/op:** SAN time. Average time, measured in microseconds, required to transfer data between the initiator and the array. Slow data transfers will result in higher SAN times.
- **Queue us/op:** Queue time. Average time, measured in microseconds, that an I/O request spends in the array waiting to be served. The time is averaged across all I/Os of the selected types.
- **QoS Rate Limit us/op:** QoS rate limit time. Average time, measured in microseconds, that reads, writes, or mirrored writes spend in queue as a result of bandwidth limits reached on one or more volumes.

- **us/op:** Service time. Average time, measured in microseconds, it takes the array to serve a read, write, or mirrored write I/O request. Service time does not include SAN time, queue time, or QoS rate limit time.

## Quality of Service Limits

Quality of Service (QoS) limits define the maximum level of throughput and the maximum number of I/O operations per second for a volume group. You can set the following QoS limits:

- QoS bandwidth limit
- QoS IOPS limit

QoS limits are not enforced on volume groups that do not have the bandwidth limit or the IOPS limit set.

### QoS Bandwidth Limit

QoS bandwidth limit can be set on a volume group to enforce maximum allowable throughput. Whenever throughput exceeds the bandwidth limit, throttling occurs. This limit is the aggregate bandwidth for all the volumes in the volume group.

To set the absolute bandwidth limit of a volume group, include the `--bw-limit` option when creating or configuring the volume group. If set, the bandwidth limit must be between 1 MB/s and 512 GB/s. The bandwidth limit is specified as an integer, optionally followed by the suffix letter **M** (for megabytes) or **G** (gigabytes).

To clear the bandwidth limit, giving the volume group unlimited throughput, specify an empty string (""). To change the existing bandwidth limit of a volume group, use the `purevgroup setattr --bw-limit` command.

By default, the QoS bandwidth limit for a volume group is unlimited. When you move a volume to a volume group that has a specified bandwidth limit, the volume inherits the bandwidth limit of the volume group regardless of whether the volume has a previously set bandwidth limit. When you move the volume out of the volume group, the bandwidth limit of the volume defaults to unlimited bandwidth.

### QoS IOPS Limit

QoS IOPS limit can be set on a volume group to enforce maximum I/O operations processed per second. Whenever the number of I/O operations per second exceeds the IOPS limit, throttling occurs. This limit is the aggregate IOPS of all the volumes in the volume group.

To set the absolute IOPS limit of a volume group, include the `--iops-limit` option when creating or configuring the volume group. If set, the IOPS limit must be between 100 and 100M. The IOPS limit is specified as an integer, optionally followed by the suffix letter **K** ( $10^3$ ) or **M** ( $10^6$ ). To clear the IOPS limit, specify an empty string (""), giving the volume group unlimited IOPS. To change the existing IOPS limit of a volume group, use the `purevgroup setattr --iops-limit` command.

By default, the QoS IOPS limit of a volume group is unlimited. When you move a volume to a volume group that has a specified IOPS limit, the volume inherits the IOPS limit of the volume group regardless of whether the volume has a previously set IOPS limit. When you move the volume out of the volume group, the IOPS limit of the volume defaults to unlimited IOPS.

For example, run the following command to create a volume group named **VGROUP02** with the QoS bandwidth limit of 1 gigabyte and the IOPS limit of **5K**.

```
$ purevgroup create VGROUP02 --bw-limit 1G --iops-limit 5K
Name      Bandwidth Limit (B/s)  IOPS Limit
VGROUP02   1G                  5K
```

In the following example, the **purevgroup setattr** command changes the bandwidth limit and the IOPS limit of **VGROUP02** to 2 gigabytes and **10K**, respectively.

```
$purevgroup setattr VGROUP02 --bw-limit 2G --iops-limit 10K
Name      Bandwidth Limit (B/s)  IOPS Limit
VGROUP02   2G                  10K
```

## Examples

### Example 1

```
purevgroup create VGROUP10 VGROUP11
```

Creates two volume groups named **VGROUP10** and **VGROUP11**.

### Example 2

```
purevgroup create VGROUP01
purevol create --size 1T VGROUP01/VOL01 VGROUP01/VOL02
purevol move VOL10 VGROUP01
```

Creates volume group **VGROUP01**, creates two volumes in volume group **VGROUP01** named **VGROUP01/VOL01** and **VGROUP01/VOL02**, and moves volume **VOL10** from the root of the array to volume group **VGROUP01**.

### Example 3

```
purevol move VGROUP01/VOL01 VGROUP02
purevol move VGROUP01/VOL02 ""
purevol move VOL02 POD01
```

Moves volume **VGROUP01/VOL01** from volume group **VGROUP01** to volume group **VGROUP02**, and then moves volume **VGROUP01/VOL02** from volume group **VGROUP01** to pod **POD01** by first moving the volume to the root of the array and then moving the volume from the root of the array to the pod.

### Example 4

```
purevol move vgroup10/vol06 vgroup10/vol07 ""
purevgroup list
purevgroup destroy VGROUP10
```

Moves volumes **VGROUP10/VOL06** and **VGROUP10/VOL07** from volume group **VGROUP10** to the root of the array, lists the volume group details to verify that volume group **VGROUP10** is empty, and destroys volume group **VGROUP10**.

#### Example 5

```
purevggroup monitor --qos
```

Displays the QoS bandwidth limit and QoS IOPS limit for each volume group.

#### Example 6

```
purevggroup setattr VGROUP02 --bw-limit 2G --iops-limit 10K
```

Sets the bandwidth limit and the IOPS limit on volume group **VGROUP02** to 2 gigabytes and 10K, respectively.

#### Example 7

```
purevol setattr --iops-limit "" VGROUP02
```

Clears the IOPS limit set on volume group **VGROUP02**.

### See Also

[purepod\(1\)](#) [412], [purevol\(1\)](#) [469]

### Author

Pure Storage Inc. <[documentfeedback@purestorage.com](mailto:documentfeedback@purestorage.com)>

## purevol

purevol, purevol-copy, purevol-create, purevol-destroy, purevol-eradicate, purevol-recover, purevol-rename, purevol-snap — manage the creation, naming, QoS limits, and destruction of FlashArray virtual storage volumes and snapshots of their contents, as well as the reclamation of physical storage occupied by the data in them

purevol-add, purevol-remove — manage adding and removing of FlashArray virtual storage volumes to and from protection groups, respectively

purevol-monitor — monitor volume I/O performance

purevol-get — manage retrieval of volume snapshots from offload targets

purevol-move — manage adding and removing of FlashArray volumes to and from storage containers

### Synopsis

**purevol** add --pgroup **PGROUP VOL...**

**purevol** copy [--overwrite] **SOURCE TARGET**

**purevol** create --size **SIZE** [--bw-limit **BANDWIDTH-LIMIT**] [--iops-limit **IOPS-LIMIT**] [--protocol-endpoint] **VOL...**

**purevol** destroy **VOL...**

**purevol** eradicate **VOL...**

**purevol** get --on **OFFLOAD-TARGET** [--suffix **SUFFIX**] **VOL...**

**purevol** monitor [--array] [ --historical { 1h | 3h | 24h | 7d | 30d | 90d | 1y } ] [--interval **SECONDS**] [--latency] [--mirrored] [--qos] [--repeat **REPEAT-COUNT**] [--size] [--total] [--csv] [--notitle] [--page] [--raw] [--filter **FILTER**] [--limit **LIMIT**] [--sort **SORT**] [**VOL...**]

**purevol** move **VOL... CONTAINER**

**purevol** recover **VOL...**

**purevol** remove --pgroup **PGROUP VOL...**

**purevol** rename **OLD-NAME NEW-NAME**

**purevol** snap [--suffix **SUFFIX**] **VOL...**

### Arguments

#### **CONTAINER**

Storage container to where the volume is to be moved. A storage container can be the root of the array, a pod, or a volume group.

#### **NEW-NAME**

Name by which the volume or volume snapshot is to be known after the command executes.

#### **OLD-NAME**

Current name of the volume or volume snapshot to be renamed.

**SOURCE**

Volume or snapshot from where data is copied. The data is copied to the **TARGET** volume.

**TARGET**

Volume to where data is copied. The data is copied from the **SOURCE** volume or snapshot.

**VOL**

Volume or volume snapshot to be created, destroyed, eradicated, or recovered. For **purevol snap**, volume to be snapped. For **purevol move**, volume to be moved.

## Object Names

Valid characters are letters (A-Z and a-z), digits (0-9), and the hyphen (-) character. The first and last characters of the name must be alphanumeric, and the name must contain at least one letter or '-'.

Most objects in Purity//FA that can be named, including host groups, hosts, volumes, protection groups, volume and protection group suffixes, SNMP managers, and subnets, can be 1-63 characters in length.

Array names can be 1-56 characters in length. The array name length is limited to 56 characters so that the names of the individual controllers, which are assigned by Purity//FA based on the array name, do not exceed the maximum allowed by DNS.

Names are case-insensitive on input. For example, **vol1**, **Vol1**, and **VOL1** all represent the same volume. Purity//FA displays names in the case in which they were specified when created or renamed.

Pods and volume groups provide a namespace with unique naming conventions.

All objects in a pod have a fully qualified name that include the pod name and object name. The fully qualified name of a volume in a pod is **POD::VOLUME**, with double colons (::) separating the pod name and volume name. The fully qualified name of a protection group in a pod is **POD::PGROUP**, with double colons (::) separating the pod name and protection group name. For example, the fully qualified name of a volume named **vol01** in a pod named **pod01** is **pod01::vol01**, and the fully qualified name of a protection group named **pgroup01** in a pod named **pod01** is **pod01::pgroup01**.

If a protection group in a pod is configured to asynchronously replicate data to a target array, the fully qualified name of the protection group on the target array is **POD:PGROUP**, with single colons (:) separating the pod name and protection group name. For example, if protection group **pod01::pgroup01** on source array **array01** asynchronously replicates data to target array **array02**, the fully qualified name of the protection group on target array **array02** is **pod01:pgroup01**.

All objects in a volume group have a fully qualified name that includes the volume group name and the object name, separated by a forward slash (/). For example, the fully qualified name of a volume named **vol01** in a volume group named **vgroup01** is **vgroup01/vol01**.

## Volume Sizes

Volume sizes are specified as an integer, followed by one of the suffix letters **K**, **M**, **G**, **T**, **P**, representing KiB, MiB, GiB, TiB, and PiB, respectively, where "Ki" denotes  $2^{10}$ , "Mi" denotes  $2^{20}$ , and so on.

Volumes must be between one megabyte and four petabytes in size. If a volume size of less than one megabyte is specified, Purity//FA adjusts the volume size to one megabyte. If a volume size of more than four petabytes is specified, the Purity//FA command fails.

Volume sizes cannot contain digit separators. For example, **1000g** is valid, but **1,000g** is not.

## Options

**-h | --help**

Can be used with any command or subcommand to display a brief syntax description.

**--array**

In a synchronous replication configuration, breaks down performance data by the array to which the I/O is directed.

**--bw-limit *BANDWIDTH-LIMIT***

Sets the maximum QoS bandwidth limit for the volume. Whenever throughput exceeds the bandwidth limit, throttling occurs. If set, the bandwidth limit must be between 1 MB/s and 512 GB/s. Like volume sizes, the bandwidth limit is specified as an integer, optionally followed by the suffix letter **M** (for megabytes) or **G** (gigabytes).

**--historical *TIME***

Display historical performance data over the specified range of time. Valid time ranges include: 1 hour, 3 hours, 24 hours, 7 days, 30 days, 90 days, and one year.

**--interval *SECONDS***

Sets the number of seconds between displays of real-time performance data. At each interval, the system displays a point-in-time snapshot of the performance data. If omitted, the interval defaults to every 5 seconds.

**--iops-limit *IOPS-LIMIT***

Sets the maximum IOPS limit for the volume. Whenever the number of I/O operations per second exceeds the IOPS limit, throttling occurs. If set, the IOPS limit must be between 100 and 100M. The IOPS limit is specified as an integer, optionally followed by the suffix letter **K** ( $10^3$ ) or **M** ( $10^6$ ).

**--latency**

Displays real-time and historical I/O latency information for all or specified volumes.

**--mirrored**

In a synchronous replication configuration, includes performance data for mirrored writes (i.e., writes to volumes in stretched pods). If one array of a stretched pod is offline, the writes processed on the other array are reported as local writes rather than mirrored ones. Once the pod is back online, in sync, and processing each write on both sides, the writes to its volumes are reported as mirrored writes again.

**--overwrite**

Allows **purevol copy** to overwrite an existing volume. Without this option, if the target volume already exists, the command fails.

--pgroup *PGROUP*

Comma-separated list of protection groups to which the specified volumes are added or from which the specified volumes are removed. Has no effect on volumes already associated with the protection group.

--protocol-endpoint

Creates a protocol endpoint, which is used to form connections between FlashArray virtual volumes and VMware ESXi hosts and host groups. For more information about virtual volumes, including configuration steps, refer to the Pure Storage vSphere Web Client Plugin for vSphere User Guide in the Pure Storage Knowledge Base at <https://support.purestorage.com>.

--qos

Displays the QoS information for each volume, including the bandwidth limit and the IOPS limit. If the bandwidth limit is not set, the value appears as a dash (-), representing unlimited throughput. If the IOPS limit is not set, the value appears as a dash (-), representing unlimited IOPS.

--repeat *REPEAT-COUNT*

Sets the number of times to display real-time performance data. If omitted, the repeat count defaults to 1.

--size

For **purevol create**, sets the virtual capacity of the volume as perceived by hosts.

For **purevol monitor**, displays the average I/O sizes per read and write operation.

--suffix

In the **purevol snap** subcommand, an optional suffix that is appended to the names of snapped volumes to create snapshot names. If not supplied, Purity//FA constructs snapshot names of the form **VOL.NNN**, where **NNN** is a unique number assigned by Purity//FA.

--total

Follows output lines with a single line containing column totals in columns where they are meaningful.

Options that control display format:

--csv

Lists information in comma-separated value (CSV) format. The --csv output can be used for scripting purposes and imported into spreadsheet programs.

--notitle

Lists information without column titles.

--page

Turns on interactive paging.

--raw

Displays the unformatted version of column titles and data. For example, in the **purearray monitor** output, the unformatted version of column title **us/op (read)** is **usec\_per\_read\_op**. The --raw output is used to sort and filter list results.

Options that manage display results:

--filter

Displays only the rows that meet the filter criteria specified.

--limit

Limits the size of the list output to the specified maximum number of rows.

--sort

Sorts the list output in ascending or descending order by the column specified.

## Description

This page describes the management of volumes and volume snapshots.

### Volumes

#### Storage Containers

A volume can reside in one of the following types of storage containers: root of the array (""), pod, or volume group. The most simple of array configurations is one that contains volumes at the root of the array. Each pod and volume group is a separate namespace for the volumes it contains.

Pods are created and configured to store volumes and protection groups that need to be fully synchronized with other arrays.

Each volume in a pod consists of the pod namespace identifier and the volume name, separated by a double colon (::). The naming convention for a volume inside a pod is **POD::VOL**, where:

- **POD** is the name of the container pod.
- **VOL** is the name of the volume inside the pod.

For example, the fully qualified name of a volume named **vol01** inside a pod named **pod01** is **pod01::vol01**.

For more information about pods, refer to the section called “Pods” [119].

Volume groups organize volumes into logical groupings. If virtual volumes are configured, a volume group is automatically created for each virtual machine that is created.

Each volume in a volume group consists of the volume group namespace identifier and the volume name, separated by a forward slash (/). The naming convention for a volume inside a volume group is **VGROUP/VOL**, where:

- **VGROUP** is the name of the container volume group.
- **VOL** is the name of the volume in the volume group.

For example, the fully qualified name of a volume named **vol01** inside a volume group named **vgroup01** is **vgroup01/vol01**.

For more information about volume groups, refer to the section called “Volume Groups” [89].

Volumes that reside in one storage container are independent of the volumes that reside in other containers. In the following example, an array has four volumes with names that contain “**vol01**”.

Volume `vol01` represents a volume named `vol01` that resides on the root of the array, volume `vgroup01/vol01` represents a volume named `vol01` that resides in volume group `vgroup01`, volume `vgroup02/vol01` represents a volume named `vol01` that resides in volume group `vgroup02`, and volume `pod01::vol01` represents a volume named `vol01` that resides in pod `pod01`. Though all four volumes have "vol01" in their names, they are completely independent of one another.

\$ purevol list *vol01					
Name	Size	Source	Created	Serial	
pod::vol01	1T	-	2018-02-05 14:27:23 PST	23364F86CC654C2900011015	
vgroup01/vol01	1T	-	2018-02-02 16:49:18 PST	23364F86CC654C2900011011	
vgroup02/vol01	5G	-	2018-02-02 16:28:18 PST	23364F86CC654C2900011009	
vol01	1G	-	2018-02-02 10:39:32 PST	23364F86CC654C2900011010	

When administering volumes, always include the namespace identifier in the volume name.

## Creating Volumes

The `purevol create` command creates one or more FlashArray virtual storage volumes of the host-visible size specified by the `--size` option.

Volumes do not consume physical storage until data is actually written to them, so volume creation has no immediate effect on an array's physical storage consumption.

Volumes created by the `purevol create` subcommand have a null value for the source attribute.

## Connecting Volumes to Hosts or Host Groups

Volumes must be *connected* to hosts in order for the hosts to read and write data on them. The `purevol connect` command establishes either *private* connections between volumes and individual hosts or *shared* connections between volumes and host groups. The same connections can be established through the `purehost connect` and `purehgroup connect` commands. For more information about private connections, refer to `purehost-connect(1)` [302]. For more information about shared connections, refer to `purehgroup-connect(1)` [283].

The `purevol connect` command also connects volumes to apps. For more information about Pure Apps and the `pureapp` command,

## Moving Volumes

The `purevol move` command moves one or more volumes between storage containers. Run the `purevol move` command to move volumes from the root of the array to a volume group, or from a volume group to the root of the array ("").

For example, run the following command to move two volumes named `vol03` and `vol04` from the root of the array to volume group `vgroup01`:

\$ purevol move vol03 vol04 vgroup01					
Name	Size	Source	Created	Serial	
vgroup01/vol03	1T	-	2018-02-02 16:55:15 PST	23364F86CC654C2900011013	
vgroup01/vol04	1T	-	2018-02-02 16:55:15 PST	23364F86CC654C2900011013	

As another example, run the following command to move volume `vgroup01/vol03` from volume group `vgroup01` to the root of the array:

```
$ purevol move vgroup01/vol03 ""
Name      Size   Source   Created           Serial
vol03    1T     -        2018-02-02 16:55:15 PST  23364F86CC654C2900011013
```

You can move volumes between volume groups, but you cannot move volumes between pods and volume groups; To move a volume between a pod and a volume group, first move the volume to the root of the array, and then move the volume from the root of the array to the destination storage container.

### Renaming Volumes

Run the `purevol rename` subcommand to change the current (OLD-NAME) name of the volume or volume snapshot to the new name (NEW-NAME). The name change is effective immediately and the old name is no longer recognized in CLI, GUI, or REST interactions. In the Purity//FA GUI, the new name appears upon page refresh.

### Copying Volumes

The `purevol copy` command can be used to create a volume from another volume, or to replace the contents of an existing volume with those of another volume. Volumes created in this way have a `Source` attribute whose value is the name of the originating volume. An *undo snapshot* is automatically taken (providing a 24-hour window during which the previous contents can be retrieved).

### Destroying Volumes

When a volume or snapshot is no longer required, it can be *destroyed* to reclaim the physical storage occupied by its data (`purevol destroy` command). Destroying a volume implicitly destroys all of its snapshots.

Destroying an individual snapshot only reclaims space that is uniquely charged to the snapshot. Space shared with older snapshots or with the originating volume remains allocated. Before a volume can be destroyed, it must be disconnected (`purevol disconnect` command).

Destruction of volumes and snapshots is not immediate. The `purevol destroy` command places volumes and snapshots in the *eradication pending* state, making them immediately inaccessible, while leaving their data intact for 24 hours (known as the *eradication pending period*). At any time during the eradication pending period, an administrator can use the `purevol recover` command to recover the contents of a destroyed volume or destroyed snapshot.

When a destroyed volume's or snapshot's 24-hour eradication pending period has lapsed (or if an administrator eradicates the volume or snapshot during the eradication pending period), Purity//FA reclaims the physical storage occupied by its data.

If the physical storage occupied by a destroyed volume's or destroyed snapshot's data is required immediately (for example, if a large amount of new data is to be written), an administrator can use the `purevol eradicate` command to terminate the eradication pending period and initiate immediate storage reclamation. The `purevol eradicate` command only acts on previously destroyed volumes and snapshots.

Once reclamation starts, whether because an eradication pending period has lapsed or because a **purevol eradicate** command was executed, a destroyed volume's or destroyed snapshot's data can no longer be recovered.

#### Adding Volumes to Protection Groups

The **purevol add** command adds existing volumes to existing protection groups. Multiple volumes can be added to multiple protection groups. Enter multiple protection groups in comma-separated format.

If a protection group already includes other volumes, those volumes are unaffected.

Only members of the same object type can belong to a protection group. For example, a volume cannot be added to a protection group that already contains hosts or host groups.

The **purevol add** command only accepts volumes that do not already belong to any of the specified protection groups. If any of the volumes already belong to any of the protection groups, the entire command fails.

Run the **purevol list --protect** command to see a list of all protected volumes and their associated protection groups.

The **purevol remove** command removes one or more volumes from one or more protection groups. All of the specified volumes must belong to all of the specified protection groups in the command; otherwise, the command fails.

Volumes can also be added to and removed from protection groups through the **puregroup setattr** command.

#### Monitoring Volume I/O Performance

The **purevol monitor** command displays real-time and historical I/O performance information for all or specified volumes. The **purevol monitor** output includes the following data about bandwidth, IOPS, and latency:

- **Name:** Object name.
- **Time:** Current time.
- **B/s:** Bandwidth. Number of bytes read/written.
- **op/s:** IOPS. Number of read, write, or mirrored write requests processed per second.
- **us/op:** Service time. Average time, measured in microseconds, it takes the array to serve a read, write, or mirrored write I/O request. Service time does not include SAN time, queue time, or QoS rate limit time.
- **B/op:** IOPS. Average I/O size per read, write, mirrored write, and both read and write (all) operations. Must include the **--size** option to see the B/op columns.
- **Bandwidth Limit:** Maximum QoS bandwidth limit. Must include the **--qos** option to see the Bandwidth Limit column.

- **IOPS Limit:** Maximum QoS IOPS limit. Must include the `--qos` option to see the IOPS Limit column.

By default, the `purevol monitor` command displays real-time performance data. Include the `--interval` option to specify the number of seconds between each real-time update. Include the `--repeat` option to specify the number of times to repeat the real-time update. The `--interval` and `--repeat` options can be combined.

Include the `--historical` option to display historical performance data over any of the following ranges of time: 1 hour, 3 hours, 24 hours, 7 days, 30 days, 90 days, or 1 year.

Include the `--qos` option to display the bandwidth limit and the IOPS limit for each volume. If the bandwidth limit for a volume is not set, the value appears as a dash (-), representing unlimited throughput. If the IOPS limit is not set, the value appears as a dash (-), representing unlimited IOPS.

In a synchronous replication configuration, include the `--array` option to break down performance data by the array to which the I/O is directed. Include the `--mirrored` option to display performance data for mirrored writes (i.e., writes to volumes in stretched pods). If one array of a stretched pod is offline, the writes processed on the other array are reported as local writes rather than mirrored ones. Once the pod is back online, in sync, and processing each write on both sides, the writes to its volumes are reported as mirrored writes again.

### Monitoring Latency

The `purevol monitor --latency` command displays real-time and historical I/O latency information for all volumes. The `purevol monitor --latency` output includes the following data:

- **SAN us/op:** SAN time. Average time, measured in microseconds, required to transfer data between the initiator and the array. Slow data transfers will result in higher SAN times.
- **Queue us/op:** Queue time. Average time, measured in microseconds, that an I/O request spends in the array waiting to be served. The time is averaged across all I/Os of the selected types.
- **QoS Rate Limit us/op:** QoS rate limit time. Average time, measured in microseconds, that reads, writes, or mirrored writes spend in queue as a result of bandwidth limits reached on one or more volumes.
- **us/op:** Service time. Average time, measured in microseconds, it takes the array to serve a read, write, or mirrored write I/O request. Service time does not include SAN time, queue time, or QoS rate limit time.

Include the `--mirrored` option to display latency data for mirrored writes (i.e., writes to volumes in stretched pods).

### Volume Snapshots

A volume snapshot is a point-in-time image of the contents of a volume.

The volume snapshot naming convention is `VOL.NNN`, where:

- `VOL` is the name of the volume.

- **NNN** is a unique monotonically increasing number or a manually-assigned volume snapshot suffix name.

## Creating Volume Snapshots

The **purevol snap** command creates point-in-time *snapshots* of the contents of one or more volumes. FlashArray snapshots make use of Purity//FA deduplication technology; they consume space only when data is written to the volumes on which they are based.

All snapshots taken as a result of a single **purevol snap** command are atomic. They represent the specified volumes' contents as of a single instant in time.

Snapshots are *immutable*; they cannot be connected to hosts or host groups, and therefore the data they contain cannot be read or written.

## Renaming Volume Snapshots

Run the purevol rename subcommand to change the current snapshot suffix name (OLD-NAME) of the volume to the new snapshot suffix name (NEW-NAME). The name change is effective immediately and the old name is no longer recognized in CLI, GUI, or REST interactions. In the Purity//FA GUI, the new name appears upon page refresh.

## Destroying Volume Snapshots

When a snapshot is no longer required, it can be *destroyed* to reclaim the physical storage occupied by its data (**purevol destroy** command).

Destroying an individual snapshot only reclaims space that is uniquely charged to the snapshot. Space shared with older snapshots or with the originating volume remains allocated.

Destruction of snapshots is not immediate. The **purevol destroy** command places snapshots in the *eradication pending* state, making them immediately inaccessible, while leaving their data intact for 24 hours (known as the *eradication pending period*). At any time during the eradication pending period, an administrator can use the **purevol recover** command to recover the contents of a destroyed snapshot.

When a destroyed snapshot's 24-hour eradication pending period has lapsed (or if an administrator eradicates the snapshot during the eradication pending period), Purity//FA reclaims the physical storage occupied by its data.

If the physical storage occupied by a destroyed destroyed snapshot's data is required immediately (for example, if a large amount of new data is to be written), an administrator can use the **purevol eradicate** command to terminate the eradication pending period and initiate immediate storage reclamation. The **purevol eradicate** command only acts on previously destroyed snapshots.

Once reclamation starts, whether because an eradication pending period has lapsed or because a **purevol eradicate** command was executed, a destroyed snapshot's data can no longer be recovered.

## Restoring Volumes from Volume Snapshots

The `purevol copy` command can be used to restore a volume by creating the volume from a snapshot, or to replace the contents of the existing volume with those of the snapshot. Restoring a volume brings the volume back to the state it was when the snapshot was taken. When a volume is restored from a snapshot, Purity//FA overwrites the entire volume with the snapshot contents. After you restore a volume, the created date of the overwritten volume is set to the snapshot created date. Purity//FA automatically takes an undo snapshot of the overwritten volume. The undo snapshot enters a 24-hour *eradication pending* period, after which time the snapshot is destroyed. During the 24-hour pending period, the undo snapshot can be viewed, recovered, or permanently eradicated through the Destroyed Volumes folder.

Volumes created in this way have a `source` attribute whose value is the name of the originating volume.

A snapshot's `size` attribute cannot be changed. When a snapshot is restored to create new or replace an existing volume, the volume's size becomes that associated with the snapshot.

## Restoring Volume Snapshots from an Offload Target

The offload target feature enables system administrators to replicate point-in-time volume snapshots from the array to an external storage system, such as an NFS device or S3 bucket. The `purevol get --on` command restores volume snapshots from an offload target. Include the `--suffix` option to replace the default snapshot suffix with a custom string. After a volume snapshot has been restored from an offload target, run the `purevol copy` command to copy the restored volume snapshot to create a new volume. For more information about offload targets, refer to pureoffload(1) [371].

## Virtual Volumes

VMware's Virtual Volumes (VVols) storage architecture is designed to give VMware administrators the ability to perform volume operations and apply protection group snapshot and replication policies to FlashArray volumes directly through vSphere.

On the FlashArray side, virtual volumes are created and then connected to VMware ESXi hosts or host groups via a protocol endpoint (also known as a conglomerate volume). The protocol endpoint itself does not serve I/Os; instead, its job is to form connections between FlashArray volumes and ESXi hosts and host groups.

Each protocol endpoint can connect multiple virtual volumes to a single host or host group, and each host or host group can have multiple protocol-endpoint connections.

LUN IDs are automatically assigned to each protocol endpoint connection and each virtual volume connection. Specifically, each protocol endpoint connection to a host or host group creates a LUN (PE LUN), while each virtual volume connection to a host or host group creates a sub-LUN. The sub-LUN is in the format `x:y`, where `x` represents the LUN of the protocol endpoint through which the virtual volume is connected to the host or host group, and `y` represents the sub-LUN assigned to the virtual volume.

In the following example, one virtual volume named `pure_vvol` and one protocol endpoint named `pure_PE` are connected to host `host01`. The virtual volume is identified by the sub-LUN `(7:1)`.

```
$ purevol list --connect --sort lun
Name      Size  LUN  Host Group  Host
pod01::vol006  1T   1   -       host01
pod100::vol007 1T   2   -       host01
pod01::vol008  1T   3   -       host01
vol009        1T   4   -       host01
vol043        2T   5   -       host01
pure_PE        -    7   -       host01
pure_VVol     1T   7:1 -       host01
```

Note that virtual volumes are primarily configured through the vSphere Web Client plugin.

For more information about virtual volumes, including steps on how to configure them, refer to the Pure Storage vSphere Web Client Plugin for vSphere User Guide in the Pure Storage Knowledge Base at <https://support.purestorage.com>.

## Quality of Service Limits

Quality of service (QoS) limits define the maximum level of throughput and the maximum number of I/O operations per second for a volume. You can set the following QoS limits:

- QoS bandwidth limit
- QoS IOPS limit

QoS limits are not enforced on volumes that do not have the bandwidth limit or the IOPS limit set.

### QoS Bandwidth Limit

QoS bandwidth limit can be set on a volume to enforce maximum allowable throughput. Whenever throughput exceeds the bandwidth limit, throttling occurs. For example, performance caps for mission-critical workloads can be set to higher bandwidth limits relative to other volumes.

To set the absolute bandwidth limit of a volume, include the **--bw-limit** option when creating or configuring the volume. If set, the bandwidth limit must be between 1 MB/s and 512 GB/s. Like volume sizes, the bandwidth limit is specified as an integer, optionally followed by the suffix letter **M** (for megabytes) or **G** (gigabytes).

By default, the QoS bandwidth limit of a volume is unlimited. When you move a volume to a volume group that has a specified bandwidth limit, the volume inherits the bandwidth limit of the volume group regardless of whether the volume has a previously set bandwidth limit. When you move the volume out of the volume group, the bandwidth limit of the volume defaults to unlimited bandwidth.

For example, run the following command to create a volume named **vol05** of 10 gigabytes with the bandwidth limit of 1 gigabyte.

```
$ purevol create vol05 --size 10G --bw-limit 1G
Name  Size  Source  Created          Serial           Bandwidth Limit (B/s)
vol05 10G   -      2019-05-20 16:07:40 PDT  3FBF2D29EE18492000011012  1G
```

## QoS IOPS Limit

QoS IOPS limit can be set on a volume to enforce maximum I/O operations processed per second. Whenever the number of I/O operations per second exceeds the IOPS limit, throttling occurs.

To set the absolute IOPS limit of a volume, include the `--iops-limit` option when creating or configuring the volume. If set, the IOPS limit must be between 100 and 100M. The IOPS limit is specified as an integer, optionally followed by the suffix letter K (10<sup>3</sup>) or M (10<sup>6</sup>).

By default, the QoS IOPS limit of a volume is unlimited. When you move a volume to a volume group that has a specified IOPS limit, the volume inherits the IOPS limit of the volume group regardless of whether the volume has a previously set IOPS limit. When you move the volume out of the volume group, the IOPS limit of the volume defaults to unlimited IOPS.

For example, run the following command to create a volume named `vol06` of 10 gigabytes with the IOPS limit of `5K`.

```
$ purevol create vol06 --size 10G --iops-limit 5K
Name  Size   Source  Created           Serial          IOPS Limit
vol06  10G    -       2019-05-20 16:06:53 PDT  3FBF2D29EE18492000011011  5K
```

## Volume and Snapshot Space Accounting

FlashArray snapshots are inherently *space saving* in that a snapshot only consumes physical storage when data on its source volume is overwritten. Similarly, a volume created from a snapshot only consumes space when its contents change.

Output of the `purevol list --snap` command displays the amount of physical storage “charged” to specified volumes and their snapshots.

For example, if two snapshots of a volume, `s1` and `s2`, are taken at times  $t_1$  and  $t_2$  ( $t_1 < t_2$ ), space consumption is accounted as follows:

Volume data written before  $t_1$

If no snapshots exist, the volume is “charged” for space occupied by host-written data.

Volume data written after  $t_1$  but before  $t_2$

Snapshot `s1` is “charged” for space occupied by the pre-update contents of the updated volume data.

Volume data written after  $t_2$

Snapshot `s2` is “charged” for space occupied by the pre-update contents of the updated volume data (`s1` shares the pre-update contents with `s2`).

If `s1` is destroyed but `s2` remains, storage charged to `s1` is reclaimed because there is no longer a need to preserve pre-update content for updates made prior to  $t_2$ . If `s2` is destroyed, however, storage charged to it must be preserved (it represents pre-update content for updates made after  $t_1$ , which must be preserved as long as `s1` exists).

To generalize, if a volume has two or more snapshots taken at different times:

### Destroying the oldest snapshot

Space charged to the destroyed snapshot is reclaimed (after the 24-hour eradication delay period has elapsed or after a **purevol eradicate** command is executed).

### Destroying a snapshot other than the oldest

Space charged to the destroyed snapshot is charged to the next older snapshot unless it is already reflected in that snapshot (because the same volume block address was written both after the next older snapshot and after the snapshot being destroyed), in which case it is reclaimed.

## Exceptions

- Destroyed volumes cannot be truncated directly. They must be recovered to their original sizes first (provided that eradication has not begun) and then truncated. Truncation is immediate and irrevocable.
- Volumes cannot be eradicated unless (a) they have previously been destroyed, and (b) they are within their 24-hour eradication pending periods.
- The names of destroyed volumes cannot be reused to create other volumes until eradication has begun.
- When a volume is destroyed, all of its existing snapshots are destroyed as well. Similarly, when a volume is recovered, snapshots that were implicitly destroyed as a consequence of the volume's destruction are recovered as well. Snapshots that are destroyed explicitly are not recovered by a **purevol recover** command specifying the volumes from which they were created.

## Examples

### Example 1

```
purevol create --size 100G VOL5
purevol create --size 1T VGROUP01/VOL5
purevol create --size 1T VGROUP02/VOL5
purevol create --size 50G POD01::VOL5
```

Creates four volumes. One volume (**VOL5**) is on the root of the array, two volumes (**VGROUP01/VOL5** and **VGROUP02/VOL5**) are in volume groups, and one volume (**POD01::VOL5**) is in a pod. All four volumes reside in storage containers that are completely independent of one another.

### Example 2

```
purevol create --size 100G --bw-limit 10M VOL1 VOL2 VOL3
```

Creates three volumes called **VOL1**, **VOL2**, and **VOL3**, each with a host-visible capacity of 100 gigabytes ( $10^{12} * 2^{30}$  bytes), and sets the maximum QoS bandwidth limit of each volume to 10 MB/s.

### Example 3

```
purevol create --size 100G VOL5 VOL6
```

... time passes...

```
purevol destroy VOL5
```

... less than 24 hours passes...

```
purevol eradicate VOL5
```

Creates 100-gigabyte volumes **VOL5** and **VOL6**. Some time later, destroys **VOL5**, making it inaccessible to hosts and starting the 24 hour eradication pending period after which Purity//FA reclaims the physical storage occupied by its data. The **purevol eradicate** command begins reclamation of physical storage occupied by **VOL5**'s data immediately; **VOL5** can no longer be recovered after this point.

#### Example 4

```
purevol destroy VOL4.415
```

... less than 24 hours later...

```
purevol recover VOL4.415
```

Destroys volume snapshot **VOL4.415** and starts the 24-hour eradication pending period after which Purity//FA would begin to reclaim the physical storage occupied by the volume snapshot's data. During the eradication pending period, the volume snapshot is restored to active status with its data intact.

#### Example 5

```
purevol get --on nfs-target array01:pgroup01.25.vol01  
purevol copy array01:vol01.restore-of.array01-pgroup01-25-vol01 restore1
```

Restores volume snapshot **array01:pgroup01.25.vol01** from offload target **nfs-target**, resulting in a volume snapshot with the name **array01:vol01.restore-of.array01-pgroup01-25-vol01**, which is then copied to create a new volume named **restore1**.

#### Example 6

```
purevol monitor
```

Displays real-time performance data for all volumes.

#### Example 7

```
purevol monitor --repeat 60 --size
```

Displays real-time performance data for all volumes. The output includes average I/O sizes. Sixty (60) point-in-time updates are displayed, with each update taken at the default interval of every five (5) seconds.

#### Example 8

```
purevol monitor --historical 24h --csv VOL1 VOL2
```

Displays a CSV output of historical performance data for volumes **VOL1** and **VOL2** over the past 24 hours.

#### Example 9

```
purevol monitor --qos
```

Displays the QoS bandwidth limit and the QoS IOPS limit for each volume.

#### Example 10

```
purevol move VGROUP01/VOL01 ""
purevol move VOL01 POD01
```

Moves volume **VOL1** from volume group **VGROUP01** to the root of the array, and then moves the same volume from the root of the array to pod **POD01**.

#### Example 11

```
purevol rename VOL1 VOL100
purevol rename VGROUP02/VOL5 VGROUP02/VOL500
purevol rename POD01::VOL8 POD01::VOL800
```

Changes the names of volume **VOL1** to **VOL100**, volume **VGROUP02/VOL5** to **VGROUP02/VOL500**, and volume **POD01::VOL8** to **POD01::VOL800**

#### Example 12

```
purevol rename VOL1.SUFFIX1 VOL1.SUFFIX2
```

Changes the suffix of volume snapshot **VOL1.SUFFIX1** from **SUFFIX1** to **SUFFIX2**.

#### Example 13

```
purevol snap VOL1 ❶
purevol snap --suffix MONDAY1 VOL1 ❷
purevol snap --suffix MONDAY2 VOL1 VOL2 ❸
```

- ❶ Create snapshot **VOL1.1**. (Purity//FA chooses the unique number.)
- ❷ Create snapshot **VOL1.MONDAY1**, representing **VOL1** at time *t1*.
- ❸ Create two snapshots:
  - **VOL1.MONDAY2**, representing **VOL1** at time *t2*.
  - **VOL2.MONDAY2**, representing **VOL2** at time *t2*.

#### Example 14 (Using a snapshot to recover from data corruption)

```
purevol snap --suffix SNAPSHOT VOL1 ❶
...
... applications using VOL1 encounter data corruption...
```

```
purevol copy VOL1.SNAPSHOT GOODVOL1 ❷
... hosts connect to GOODVOL1, validate data, and resume...
purevol destroy VOL1.SNAPSHOT ❸
purevol eradicate VOL1.SNAPSHOT ❹
```

- ❶ Create snapshot VOL1.SNAPSHOT at a time when data is known to be correct. Some time later, applications using VOL1 encounter data corruption.
- ❷ Create volume GOODVOL1 from snapshot VOL1.SNAPSHOT. The volume contains a pre-corruption data image of VOL1. Hosts recover lost information, for example by replaying logs, and continue processing data on GOODVOL1.
- ❸ Destroy snapshot VOL1.SNAPSHOT when it is no longer required.
- ❹ Start the process of reclaiming physical storage charged to VOL1.SNAPSHOT.

#### Example 15 (Cloning multiple volumes from a snapshot)

```
purevol snap --suffix TUESDAY VOL1 VOL2 ❶
... main application continues using VOL1 and VOL2...
purevol copy Vol1.TUESDAY Vol1Backup
purevol copy Vol2.TUESDAY Vol2Backup ❷
purevol connect --host HostBackup Vol1Backup Vol2Backup ❸
... backup starts ...
purevol copy Vol1.TUESDAY Vol1Analytics ❹
purevol connect --host HostAnalytics Vol1Analytics ❺
... analytics starts, using VOL1 image only ...
... backup completes ...
purevol disconnect --host HostBackup Vol1Backup Vol2Backup
purevol destroy Vol1Backup Vol2Backup ❻
... analytics completes ...
purevol disconnect --host HostAnalytics Vol1Analytics
purevol destroy Vol1Analytics ❼
purevol eradicate Vol1Analytics Vol1Backup Vol2Backup ❾
```

- ❶ Create snapshots VOL1.TUESDAY and VOL2.TUESDAY, representing the data on the respective volumes at a single instant in time. The main application continues to operate using VOL1 and VOL2.

- ❷ Create accessible volumes **Vol1Backup** and **Vol2Backup**.
- ❸ Connect the two backup volumes to **HostBackup**, making it possible to back up the data images they contain.
- ❹ (Assuming that the analytics application only requires data from **VOL1**.) Create volume **Vol1Analytics** from the same snapshot used to create **Vol1Backup**.
- ❺ Connect **Vol1Analytics** to **HostAnalytics**, making it possible to analyze the data image from **VOL1**.
- ❻ When backup application completes, disconnect the two backup volumes from **HostBackup** and destroy them.
- ❼ When analytics application completes, disconnect **Vol1Analytics** from **HostAnalytics** and destroy the volume.
- ❽ Eradicate the three volumes used by the ancillary backup and analytics applications, starting the process of reclaiming the space charged to them.

#### **Example 16 (Destroying and recovering snapshots and volumes)**

```
purevol snap --suffix FirstSnap VOL1
purevol snap --suffix SecondSnap VOL1
purevol snap --suffix ThirdSnap VOL1
purevol snap --suffix FourthSnap VOL1 ❶
purevol destroy VOL1.FirstSnap
purevol eradicate VOL1.FirstSnap ❷
purevol destroy VOL1.SecondSnap ❸
purevol destroy VOL1 ❹
purevol recover VOL1 ❺
purevol recover VOL1.Secondsnap ❻
```

- ❶ Create snapshots of **VOL1** at four different points in time.
- ❷ Destroy and eradicate **VOL1.FirstSnap**. The snapshot is no longer recoverable.
- ❸ Destroy, but do not eradicate, **VOL1.Secondsnap**. For 24 hours after the command executes, **VOL1.Secondsnap** can be recovered.
- ❹ Destroy, but do not eradicate, **VOL1**, and implicitly, its two remaining snapshots, **VOL1.ThirdSnap** and **VOL1.FourthSnap**.
- ❺ (Executes within 24 hours of ❹ [486] execution). Recover **VOL1** and the snapshots that were implicitly destroyed along with it. Recover **VOL1.ThirdSnap** and **VOL1.FourthSnap**, but not **VOL1.Secondsnap** (because it was destroyed separately), or **VOL1.FirstSnap** (because it was destroyed and eradicated separately).
- ❻ (Executes within 24 hours of ❺ [486] execution). Recover snapshot **VOL1.Secondsnap**. Because **VOL1.FirstSnap** was eradicated, it is not possible to recover it with a similar command.

#### **Example 17 (Refreshing volume contents from snapshots and volumes)**

```
purevol snap --suffix SNAPSHOT VOL1 ❶
...
... applications continue to access VOL1...
```

```
purevol copy VOL1.SNAPSHOT VOL1Image ❶
... update data format on VOL1Image...

purevol disconnect --host HOST1 VOL1
purevol copy --overwrite VOL1Image VOL1
purevol connect --host HOST1 VOL1 ❷
... restart applications using updated data format...

... detect problem with updated format...

purevol disconnect --host HOST1 VOL1
purevol copy --overwrite VOL1.SNAPSHOT VOL1
purevol connect --host HOST1 VOL1 ❸
... resume applications using older data format...
```

- ❶ Create snapshot VOL1.SNAPSHOT of volume VOL1.
- ❷ Create volume VOL1Image. Use it to update data format for applications that use VOL1.
- ❸ Momentarily disconnect HOST1 from its data, and overwrite said data with data in the updated format from volume VOL1Image. Applications detect problems with updated data format.
- ❹ Momentarily disconnect HOST1 from its volume, revert data on VOL1 to previous format by copying from snapshot VOL1.SNAPSHOT, and reconnect VOL1 to HOST1 so that applications can resume processing using older format data.

## See Also

[purevol-list\(1\)](#) [488], [purevol-setattr\(1\)](#) [497]

[pureapp\(1\)](#) [221], [purehgroup\(1\)](#) [275], [purehgroup-connect\(1\)](#) [283], [purehost\(1\)](#) [292], [purehost-connect\(1\)](#) [302], [purepod\(1\)](#) [412]

## Author

Pure Storage Inc. <[documentfeedback@purestorage.com](mailto:documentfeedback@purestorage.com)>

## purevol-list

purevol-list, purevol-listobj — display volumes' attributes and information about the virtual and physical storage they consume

### Synopsis

```
purevol list [ --all | --connect | --protect | --qos | --space ] [ --historical { 1h | 3h | 24h | 7d | 30d | 90d | 1y } ] [--host-encryption-key] [--on OFFLOAD-TARGET] [ --pending | --pending-only ] [--pgrouplist PGROUPLIST] [ --private | --shared ] [ --protocol-endpoint ] [--remote] [ --cli | --csv | --nvp ] [--filter FILTER] [--limit LIMIT] [--notitle] [--page] [--raw] [--snap] [--sort SORT] [ --total ] [VOL...]
```

```
purevol listobj [ --csv ] [ --pending | --pending-only ] [--pgrouplist PGROUPLIST] [ --type { host | snap | vol } ] [VOL...]
```

### Arguments

**VOL**

Volume for which the information specified by options is to be displayed

### Options

Options that control information displayed:

**-h | --help**

Can be used with any command or subcommand to display a brief syntax description.

**--all**

Displays names, virtual sizes, connected hosts and host groups, and host and array IQNs, NQNs, or WWNs for the specified volumes.

**--connect**

Displays hosts and host groups associated with the specified volumes, and the LUNs used by each to address them.

**--historical *TIME***

Used with **purevol list --space** to display historical size and space consumption information over the specified range of time. Valid time ranges include: 1 hour, 3 hours, 24 hours, 7 days, 30 days, 90 days, and one year.

**--host-encryption-key**

Displays the host encryption key status for each volume or specified volumes.

**--on *OFFLOAD-TARGET***

Used with **purevol list --snap** to display a list of volume snapshots on the specified offload target.

**--pending**

Includes destroyed volumes or snapshots that are in the eradication pending state. If not specified, volumes or snapshots that are pending eradication are not shown.

**--pending-only**

Only displays destroyed volumes or snapshots that are in the eradication pending state.

**--pgrouplist**

Used with **purevol list --snap** to only display volume snapshots that have been created as part of the specified protection group or protection group snapshot. Enter multiple protection group or protection group snapshot arguments in comma-separated format.

**--private**

Used with **purevol list --connect** to restrict the display to specified volumes' privately connected hosts. Invalid when combined with other options.

**--protect**

Displays all protected volumes and their associated protection groups.

**--protocol-endpoint**

Displays the virtual volumes used to form connections between FlashArray volumes and VMware ESXi hosts and host groups. For more information about virtual volumes, including configuration steps, refer to the Pure Storage vSphere Web Client Plugin for vSphere User Guide in the Pure1 Knowledge site at <https://support.purestorage.com>.

**--qos**

Displays the QoS information for each volume, including the bandwidth limit and the IOPS limit. If the bandwidth limit is not set, the value appears as a dash (-), representing unlimited throughput. If the IOPS limit is not set, the value appears as a dash (-), representing unlimited IOPS.

**--remote**

Used with **purevol list --connect** to include a list of remote volumes.

**--shared**

Used with **purevol list --connect** to restrict the display to specified volumes' shared connections. Invalid when combined with other options.

**--snap**

Displays information for snapshots of the specified volumes rather than for the volumes themselves.

**--space**

Displays the following information about provisioned (virtual) size and physical storage consumption for each specified volume:

**Size**

Total provisioned size of the volume. Represents storage capacity reported to hosts.

#### Thin Provisioning

Percentage of volume sectors that do not contain host-written data because the hosts have not written data to them or the sectors have been explicitly trimmed.

#### Data Reduction

Ratio of mapped sectors within a volume versus the amount of physical space the data occupies after data compression and deduplication. The data reduction ratio does not include thin provisioning savings.

For example, a data reduction ratio of 5:1 means that for every 5 MB the host writes to the array, 1 MB is stored on the array's flash modules.

#### Total Reduction

Ratio of provisioned sectors within a volume versus the amount of physical space the data occupies after reduction via data compression and deduplication *and* with thin provisioning savings. Total reduction is data reduction with thin provisioning savings.

For example, a total reduction ratio of 10:1 means that for every 10 MB of provisioned space, 1 MB is stored on the array's flash modules.

#### Volume

Physical space occupied by volume data that is not shared across volumes, excluding array metadata and snapshots.

#### Snapshots

Physical space occupied by data unique to one or more snapshots.

#### Shared Space

Physical space occupied by deduplicated data, meaning that the space is shared with other volumes and snapshots as a result of data deduplication.

#### System

Physical space occupied by internal array metadata.

#### Total

Total physical space occupied by system, shared space, volume, and snapshot data.

#### --total

Follows output lines with a single line containing column totals in columns where they are meaningful.

#### --type

Specifies the type of information (connected hosts, snapshots, or echoed volume names) about specified volumes to be produced in whitespace-separated list format suitable for scripting.

#### Options that control display format:

##### --cli

Displays output in the form of CLI commands that can be issued to reproduce the current configuration. The --cli output is not meaningful when combined with immutable attributes.

--csv

Lists information in comma-separated value (CSV) format. The --csv output can be used for scripting purposes and imported into spreadsheet programs.

--notitle

Lists information without column titles.

--nvp

Lists information in name-value pair (NVP) format, in the form **ITEMNAME=VALUE**. Argument names and information items are displayed flush left. The --nvp output is designed both for convenient viewing of what might otherwise be wide listings, and for parsing individual items for scripting purposes.

--page

Turns on interactive paging.

--raw

Displays the unformatted version of column titles and data. For example, in the **purearray monitor** output, the unformatted version of column title **us/op (read)** is **usec\_per\_read\_op**. The --raw output is used to sort and filter list results.

Options that manage display results:

--filter

Displays only the rows that meet the filter criteria specified.

--limit

Limits the size of the list output to the specified maximum number of rows.

--sort

Sorts the list output in ascending or descending order by the column specified.

## Description

The **purevol list** command displays a list of Purity//FA virtual storage volumes on the array. The list includes volume attributes such as volume (virtual) size, volume creation date, and volume serial number (generated by Purity//FA when the volume is created).

The **purevol list** command includes the mutually exclusive --all, --connect, --protect, --qos, and --space options, which are used to modify the display output of **purevol list**. The --all option displays volume connection details, including volume size, LUN number, and "paths" to connected hosts (host and corresponding array port IQNs, NQNs, or WWNs). The --connect option displays the hosts and host groups to which the specified volumes are connected, and the LUNs used by each to address them. The --protect option displays all protected volumes and their associated protection groups.

The --qos option displays the bandwidth limit and the IOPS limit for each volume. If the bandwidth limit for a volume is not set, the value appears as a dash (-), representing unlimited throughput. If the IOPS limit for a volume is not set, the value appears as a dash (-), representing unlimited IOPS.

The **--space** option displays information about provisioned (virtual) size and storage consumption for each volume. The **--remote** option, which must be used with the **--connect** option, includes a list of remote volumes with host or host group connections.

The **purevol list --historical** command, which must be used with the **--space** option, displays historical size and storage consumption details for each volume.

The **purevol list --pending** command displays a list of all volumes, including ones that have been destroyed and are in the *eradication pending* state. The **purevol list --pending-only** command only displays a list of volumes that have been destroyed and are in the *eradication pending* state.

The **purevol list --total** command displays the aggregate host-visible size of the volumes.

The **--total** option cannot be run with the **--all** or **--connect** option because these options can conceivably display more than one output line per volume.

The **purevol listobj** command displays a list volume attributes, either in whitespace or comma-separated form, suitable for scripting. The **--type** option accepts the following arguments:

**--type host**

Display a list of hosts to which the specified volumes are connected. If no volumes are specified, the list contains the names of all hosts connected to any volume.

**--type snap**

Displays a list of volume snapshots, generated through volume or protection group snapshots, for the specified volumes. If no volumes are specified, the list contains the names of all volume snapshots.

**--type vol** (default if **--type** option not specified)

Lists the specified volume names. If no volume names are specified, the list contains the names of all volumes. Snapshots are not included in the list.

Lists are whitespace-separated by default. Include the **--csv** option to produce a comma-separated list.

## Displaying and Listing Snapshots

The **purevol list --snap** command displays a list of volume snapshots rather than volumes. Both volume and snapshot arguments can be included in the same command. The **--snap** option can be specified with the **--pending**, **--pending-only**, and **--space** options.

The **purevol list --snap --pgrouplist** command only displays volume snapshots that are created as part of the specified protection group or protection group snapshot. Enter multiple protection group or protection group snapshot arguments in comma-separated format.

The **--snap** option is not valid with the **--all**, **--connect**, **--private**, or **--shared** options.

The **purevol listobj --type snap** command displays the names of volume snapshots generated through volume or protection group snapshots.

## Displaying and Listing Snapshots on Offload Targets

The offload target feature enables system administrators to replicate point-in-time volume snapshots from the array to an external storage system, such as an NFS device or S3 bucket. The **purevol list --snap --on** command displays a list of volume snapshots on an offload target. Run the **purevol get --on** command to restore volume snapshots from an offload target. For more information about offload targets, refer to [pureoffload\(1\)](#) [371].

## Displaying Host Encryption Key Status

To enable deduplication on encrypted data, you need to configure the FlashArray to retrieve encryption keys from the Key Management Interoperability Protocol (KMIP) server. For more information on how to configure the KMIP server, refer to the EncryptReduce Guide in the Pure1 Knowledge site at <https://support.purestorage.com>. To show whether a volume has a host encryption key, use the **purevol list --host-encryption-key** command to display the host encryption status.

A volume can have one of the following encryption statuses:

- **none**: A volume is not encrypted.
- **detected**: A valid header (or key) is detected, but no fetched encryption key from the KMIP server.
- **fetched**: A valid header (or key) and a fetched encryption key from the KMIP server are detected.

The following example displays the host encryption key status of volumes **vol1**, **vol2**, and **vol3**.

```
$ purevol list --host-encryption-key
Name      Status
vol1      none
vol2      detected
vol3      fetched
```

## App Volumes

The Purity Run platform extends array functionality by integrating add-on services into the Purity//FA operating system. Each service that runs on the platform is provided by an app.

For each app that is installed, a boot volume is created. For some apps, a data volume is also created. Boot and data volumes are known as app volumes. Run the **purevol list** command to see a list of app volumes. Boot and data app volume names begin with a distinctive @ symbol.

The naming convention for app volumes is **@APP\_boot** for boot volumes and **@APP\_data** for data volumes, where **APP** denotes the app name.

The following example displays the boot and data volumes for the **linux** app.

```
$ purevol list @linux*
Name      Size  Source  Created          Serial
...
@linux_boot  15G   -    2016-11-09 15:13:37 MST  AC97A330F2544A3C00011010
@linux_data  16T   -    2016-11-09 15:14:17 MST  AC97A330F2544A3C00011012
```

...

The boot volume represents a copy of the boot drive of the app. Do not modify or save data to the boot volume. When an app is upgraded, the boot volume is overwritten, completely destroying its contents including any other data that is saved to it. The data volume is used by the app to store data.

For more information about Apps, refer to pureapp(1) [221].

## Exceptions

None.

## Examples

### Example 1

```
purevol list
```

Displays names, sizes, and serial numbers of all volumes. Both directly connected hosts and hosts connected by virtue of their associations with host groups are displayed.

### Example 2

```
$ purevol list @*
```

Displays a list of all app volumes on the array.

### Example 3

```
purevol list --space --csv --notitle VOL1 VOL2 VOL3
```

Displays names, sizes, data reduction ratios, and physical storage space occupied by volume data and RAID-3D check data for volumes **VOL1**, **VOL2**, and **VOL3**, in comma-separated value format. The **--notitle** option suppresses the line of column titles that would ordinarily precede the output.

### Example 4

```
purevol list --space --historical 3h --csv --notitle VOL1 VOL2 VOL3
```

Displays names, sizes, data reduction ratios, and physical storage space occupied by volume data and RAID-3D check data for volumes **VOL1**, **VOL2**, and **VOL3**, for the past 3 hours, in comma-separated value format. The **--notitle** option suppresses the line of column titles that would ordinarily precede the output.

### Example 5

```
purevol list --connect --shared --nvp VOL4 VOL5
```

For volumes **VOL4** and **VOL5**, displays names, sizes, connected host groups and their associated hosts, and LUNs used to address the volumes, all in *name-value pair (ATTRIBUTE-NAME=ATTRIBUTE-VALUE)* format. Only shared connections are included.

## Example 6

```
purevol list --connect --shared VOL1
```

Displays host groups to which **VOL1** has shared connections and the LUNs used by hosts in the groups to address it.

## Example 7

```
purevol list --connect --remote
```

Displays a list of all volumes on both the local and remote arrays, their associated hosts or host groups, and the LUNs used by each volume to address the host or host group.

## Example 8

```
purehost listobj --type vol --private $(purevol listobj --type host --private VOL1)
```

Produces a list of hosts with private connections to **VOL1**, which is input to the **purehost listobj** command to display a list of the volumes to which those hosts have private connections.

## Example 9

```
purehost list --connect $(purevol listobj --type host VOL1)
```

The inner **purevol listobj** command produces a whitespace-separated list of hosts to which **VOL1** is connected. The list becomes input to the outer **purevol list** command. The result is a display of information about hosts connected to **VOL1**.

## Example 10

```
purevol list --qos
```

Displays the QoS bandwidth limit and the QoS IOPS limit for each volume.

## Example 11

```
purevol destroy VOL2
purevol destroy VOL3
purevol list --space --pending-only
```

Places **VOL2** and **VOL3** in the *eradication pending* state for 24 hours. Produces a space report of volumes that are pending eradication (including **VOL2** and **VOL3**). **VOL2** and **VOL3** continue to occupy space until the 24-hour pending period has elapsed or until the volumes have been manually eradicated by the **purevol eradicate** command.

## Example 12

```
purevol list --snap --pgrouplist PG1.1
```

Produces a list of volume snapshots created for protection group snapshot **PG1.1**.

#### Example 13

```
purevol list --snap --pgrouplist PG1,PG2 v1
```

Displays a list of volume snapshots of **v1** that were created for all snapshots of protection groups **PG1** and **PG2**.

#### Example 14

```
purevol list --host-encryption-key vol1
```

Displays the host encryption key status of volume **vol1**.

#### Example 15

```
purevol destroy $(purevol listobj --type snap VOL1)
```

In this example:

- The **purevol listobj** (“inner”) command produces a whitespace-separated list of the snapshots of **VOL1**.
- The **purevol destroy** (“outer”) command destroys the snapshots enumerated in the inner command.

#### See Also

[purevol\(1\)](#) [469] and [purevol-setattr\(1\)](#) [497]

[pureapp\(1\)](#) [221], [purehgroup\(1\)](#) [275], [purehgroup-connect\(1\)](#) [283], [purehost\(1\)](#) [292], [purehost-connect\(1\)](#) [302]

#### Author

Pure Storage Inc. <[documentfeedback@purestorage.com](mailto:documentfeedback@purestorage.com)>

## purevol-setattr

purevol-setattr — manage QoS limits of the volumes and configure volume sizes  
purevol-truncate — increase and decrease volume sizes

### Synopsis

```
purevol setattr [--bw-limit BANDWIDTH-LIMIT] [--iops-limit IOPS-LIMIT] [--size NEW-SIZE] VOL...  
purevol truncate --size NEW-SIZE VOL...
```

### Arguments

**VOL**

Name of the volume object whose size is to be changed.

### Options

**-h | --help**

Can be used with any command or subcommand to display a brief syntax description.

**--bw-limit *BANDWIDTH-LIMIT***

Sets a QoS bandwidth limit for the specified volume. Whenever throughput exceeds the bandwidth limit, throttling occurs. If set, the bandwidth limit must be between 1 MB/s and 512 GB/s. Like volume sizes, the bandwidth limit is specified as an integer, optionally followed by the suffix letter **M** (for megabytes) or **G** (gigabytes). To clear the bandwidth limit, giving the volume unlimited throughput, set to an empty string ("").

**--iops-limit *IOPS-LIMIT***

Sets a QoS IOPS limit for the specified volume. Whenever the number of I/O operations exceeds the IOPS limit, throttling occurs. If set, the IOPS limit must be between 100 and 100**M**. The IOPS limit is specified as an integer, optionally followed by the suffix letter **K** ( $10^3$ ) or **M** ( $10^6$ ). To clear the IOPS limit, set to an empty string (""), giving the volume unlimited IOPS.

**--size**

Sets the virtual size for the specified volume.

### Volume Sizes

Volume sizes are specified as an integer, followed by one of the suffix letters **K**, **M**, **G**, **T**, **P**, representing KiB, MiB, GiB, TiB, and PiB, respectively, where "Ki" denotes  $2^{10}$ , "Mi" denotes  $2^{20}$ , and so on.

Volumes must be between one megabyte and four petabytes in size. If a volume size of less than one megabyte is specified, Purity//FA adjusts the volume size to one megabyte. If a volume size of more than four petabytes is specified, the Purity//FA command fails.

Volume sizes cannot contain digit separators. For example, **1000g** is valid, but **1,000g** is not.

## Description

The **purevol setattr** command changes the attribute values of existing volumes.

### Configuring Volume Sizes

The **purevol setattr --size** command increases the provisioned size of a volume (virtual size as perceived by hosts). The **purevol truncate** command decreases the provisioned volume size. Changes in volume size are immediate and always affect the highest-numbered sector addresses.

Changes in volume size are visible to connected hosts immediately. When a volume is truncated, Purity//FA automatically takes an undo snapshot, providing a 24-hour window during which the previous contents can be retrieved. Note that the data in truncated sectors cannot be retrieved by increasing the size of a truncated volume. When a volume's size is increased, reading data in the new virtual sector addresses before they are written returns zeros.

### Setting QoS Limits

Quality of service (QoS) bandwidth limits can be set on a per-volume basis to enforce maximum allowable throughput. The **purevol setattr --bw-limit** command sets the absolute bandwidth limit of a volume. Whenever throughput exceeds the bandwidth limit, throttling occurs. If set, the bandwidth limit must be between 1 MB/s and 512 GB/s. Like volume sizes, the bandwidth limit is specified as an integer, optionally followed by the suffix letter **M** (for megabytes) or **G** (gigabytes). To clear the bandwidth limit, giving the volume unlimited throughput, set to an empty string (""). Note that you cannot set a QoS bandwidth limit on an individual volume in a volume group.

Quality of service (QoS) IOPS limits can be set on a per-volume basis to enforce maximum allowable I/O operations per second. The **purevol setattr --iops-limit** command sets the absolute IOPS limit of a volume. Whenever the number of I/O operations per second exceeds the IOPS limit, throttling occurs. If set, the IOPS limit must be between 100 and 100M. The IOPS limit is specified as an integer, optionally followed by the suffix letter **K**( $10^3$ ) or **M** ( $10^6$ ). To clear the IOPS limit, specify an empty string (""), giving the volume unlimited IOPS.

In the following example, the **purevol create** command creates a volume named **vol01** of 10 gigabytes with the QoS bandwidth limit of 1 gigabyte and the IOPS limit of 5K; the **purevol setattr** command changes the bandwidth limit to 2 gigabytes and the IOPS limit to unlimited IOPS.

```
$ purevol create vol01 --size 10G --bw-limit 1G --iops-limit 5K
Name  Size Source Created          Serial          Bandwidth Limit(B/s)  IOPS
vol01 10G  -    2019-05-20 16:24:58 PDT 3FBF2D29EE18492000011013 1G           5K

$purevol setattr vol01 --bw-limit 2G --iops-limit ""
Name  Size  Bandwidth Limit (B/s)  IOPS Limit
vol01 10G   2G                  -
```

## Exceptions

The following exceptions apply to the **purevol setattr** command:

- The **purevol setattr** command cannot reduce the size of a volume, nor can **purevol truncate** increase the size of a volume.

- The size of a volume cannot be reduced to less than one megabyte or increased to more than 4 petabytes. If a **--size** smaller than one megabyte is specified, Purity//FA changes the volume's size to one megabyte. If a **--size** greater than 4 petabytes is specified, the command fails.

## Examples

### Example 1

```
purevol setattr --size 1T VOL1 VOL2
```

Increases the sizes of **VOL1** and **VOL2** to 1 terabyte each. If either volume is larger than 1 terabyte, an error is logged, and that volume's size does not change.

### Example 2

```
purevol setattr --size 1T VOL2
```

Increases the size of **VOL2** to 1 terabyte, assuming that the volume's current size is less than that amount.

### Example 3

```
purevol setattr --bw-limit 50G VOL2
```

Sets the maximum QoS bandwidth limit of **VOL2** to 50 gigabytes.

### Example 4

```
purevol setattr --iops-limit 5000 VOL2
```

Sets the maximum QoS IOPS limit of **VOL2** to 5000 IOPS.

### Example 5

```
purevol setattr --iops-limit "" VOL2
```

Clears the maximum QoS IOPS limit set on **VOL2**.

## See Also

[purevol\(1\)](#) [469] and [purevol-list\(1\)](#) [488],

[purehgroup\(1\)](#) [275], [purehgroup-connect\(1\)](#) [283], [purehost\(1\)](#) [292], [purehost-connect\(1\)](#) [302]

## Author

Pure Storage Inc. <[documentfeedback@purestorage.com](mailto:documentfeedback@purestorage.com)>



## Chapter 11. CLI Getting Started

FlashArrays present disk-like volumes to hosts. From the Purity//FA standpoint, volumes and hosts are abstract objects which must be created by an administrator and connected to each other before actual host computers can write and read data on volumes.

### Creating Volumes

Once a FlashArray is installed and initially configured, an administrator's typical first task is creation of volumes that hosts will subsequently format and use to store and access data. Example 11.1 illustrates three examples of volume creation using the CLI.

#### Example 11.1: Creating Volumes

```
$ purevol create vol1 ❶
usage: purevol create [-h] --size SIZE VOL ...
purevol create: error: argument --size is required
$ purevol create vol1 --size 100g ❷
Name Size Serial
vol1 100G 7CBFCE0B2660CC3000010001
$ purevol create vol2 vol3 vol4 --size 200g ❸
Name Size Serial
vol2 200G 7CBFCE0B2660CC3000010002
vol3 200G 7CBFCE0B2660CC3000010003
vol4 200G 7CBFCE0B2660CC3000010004
```

- ❶ Command fails, because no virtual size is specified for the volume.
- ❷ Creates a volume that hosts will perceive as containing 100 gigabytes of storage, although it occupies almost no physical space until hosts write data to it.
- ❸ Creates three volumes in a single command. There is no practical limit to the number of volumes that can be created in a single command, but only one **--size** option can be specified in the command, so all volumes initially have the same virtual size.

The output of the **purevol create** command lists the name, size, and Purity//FA-generated serial number of each volume created.

### Creating Hosts

Internally, Purity//FA represents host computers that use FlashArray storage services as host objects (usually just called "hosts"). A Purity//FA host object has a name and one or more iSCSI Qualified Names (IQNs), NVMe Qualified Names (NQNs), or Fibre Channel World Wide Names (WWNs). IQNs, NQNs, and WWNs correspond to initiators, typically host computers or host computer ports with which a FlashArray exchanges commands, data, and messages. Purity//FA represents a host's IQNs, NQNs, and WWNs as a multi-valued attribute, specified via either the **--iqnlist** (alias **--iqn**), **--nqnlist** (alias **--nqn**), or **--wwnlist** (alias **--wwn**) option when hosts are created.

iSCSI IQNs conform to Internet fully-qualified domain name (FQDN) standards. NVMe NQNs conform to the naming standards set by NVM Express. Fibre Channel WWNs consist of exactly 16 hexadecimal digits, and may be specified either as continuous strings or as eight digit pairs separated by colons. Example 11.2 illustrates examples of host creation.

### Example 11.2: Creating Hosts

```
$ purehost create host1 ❶
Name  WWN
host1 -
$ purehost create --wwnlist 0123456789abcde2 host2 ❷
Name  WWN
host2 01:23:45:67:89:AB:CD:E2
$ purehost create --wwnlist 0123456789abcde3,01:23:45:67:89:ab:cd:e4 host3 ❸
Name  WWN
host3 01:23:45:67:89:AB:CD:E3
    01:23:45:67:89:AB:CD:E4
$ purehost create --wwn 01:23:45:67:89:ab:cd:e4 host4 ❹
Error on host4: The specified WWN is already in use.
```

- ❶ Creates host object **host1**. At the time of creation, **host1** cannot connect to volumes because no IQNs, NQNs, or WWNs are associated with it.
- ❷ Creates **host2** and assigns WWN **0123456789abcde2** to it. Volumes can be connected to **host2** immediately.
- ❸ Creates **host3** and associates two WWNs with it (typically because the host is equipped with two Fibre Channel ports). This example also illustrates the two forms in which world wide names can be entered--continuous digit string and colon-separated two-digit groups.
- ❹ Attempts to create **host4** and fails, because the WWN specified is already associated with another host. Because they identify host ports on the storage network, WWNs must be unique within an array.

## Connecting Hosts and Volumes

For a host to read and write data on a volume, a connection between the two must be established. A connection is effectively permission for an array to respond to I/O commands from a given set of storage network addresses (host IQNs, NQNs, or WWNs). In SCSI terminology, a connection is an *Initiator-Target-LUN (I\_T\_L)* nexus, a completely specified path for message and data exchange. By default, Purity//FA chooses a LUN based on availability at both the initiator (host) and target (array) when a connection is established. Alternatively, an administrator can specify an unused LUN when establishing a connection.

The **purehost connect** command establishes private connections between a single volume and one or more hosts. The **purevol connect** command establishes connections between a single host and one or more volumes. The two commands are functionally identical. Both commands are provided for administrative convenience. Example 11.3 illustrates the use of both commands.

### Example 11.3: Establishing Host-Volume Connections

```
$ purehost connect --vol vol1 host1 host2 host3 ①
Name   Vol   LUN
host1  vol1  1
host2  vol1  1
host3  vol1  1
$ purevol connect --host host1 vol2 vol3 vol4 ②
Name  Host Group  Host   LUN
vol2  -          host1  2
vol3  -          host1  3
vol4  -          host1  4
$ purevol connect --lun 249 --host host1 vol5 ③
Name  Host Group  Host   LUN
vol5  -          host1  249
$ purehost disconnect --vol vol1 host2 ④
Name   Vol
host2  vol1
$ purevol disconnect --host host3 vol1 ⑤
Name Hgroup Host
vol1      host3
```

- ① Connects **vol1** to hosts **host1**, **host2**, and **host3**.
- ② Connects **host1** to volumes **vol2**, **vol3**, and **vol4**.
- ③ Connects **host1** to **vol5**, manually associating LUN **249** with the connection. Any available LUNs in the range [1...4095] can be used for private or shared connections.
- ④⑤ The **purehost disconnect** or **purevol disconnect** commands in the example break the private connections between **host2** and **vol1**, and between **host3** and **vol1** respectively. Either can be used to break a private connection between a host and a volume, regardless of how the connection was established. As with the connect subcommands, **purehost disconnect** can disconnect multiple hosts from a single volume, and **purevol disconnect** can disconnect multiple volumes from a single host. The two subcommands are provided for administrative convenience.

Whichever way a connection is made, the host administrator can immediately rescan the host's flash modules (if necessary) to discover the logical units represented by newly-connected volumes, create file systems, and mount them for storing and retrieving data.

## Resizing Volumes

FlashArray administrators also perform routine management tasks on objects within an array. Typically the most frequent of these are operations on volumes. You can change the size (host-visible storage capacity) of a volume, disconnect it from hosts, rename it, destroy it, eradicate all or part of the data it contained, and under certain circumstances, restore the contents of a destroyed volume or truncated sector range.

To minimize the possibility for inadvertent errors, the CLI uses different subcommands for increasing and decreasing the host-perceived sizes of volumes. The **purevol setattr** command increases the

sizes of one or more volumes to a given level; the `purevol truncate` command decreases volumes' sizes. Both commands can operate on multiple volumes, but `purevol setattr` can only increase volume sizes, and `purevol truncate` can only decrease them.

Example 11.4 illustrates the use of these commands to change volumes' sizes.

#### Example 11.4: Resizing Volumes

```
$ purevol list vol1 vol2 vol3 vol4
Name  Size  Serial
vol1  100G  7CBFCE0B2660CC3000010001
vol2  200G  7CBFCE0B2660CC3000010002
vol3  200G  7CBFCE0B2660CC3000010003
vol4  200G  7CBFCE0B2660CC3000010004
$ purevol setattr --size 300g vol1 vol2 ①
Name  Size  Serial
vol1  300G  7CBFCE0B2660CC3000010001
vol2  300G  7CBFCE0B2660CC3000010002
$ purevol truncate --size 100g vol3 vol4 ②
Name  Size  Serial
vol3  100G  7CBFCE0B2660CC3000010003
vol4  100G  7CBFCE0B2660CC3000010004
$ purevol setattr --size 250g vol2 vol3 ③
Name  Size  Serial
vol3  250G  7CBFCE0B2660CC3000010003
Error on vol2: Implicit truncation not permitted.
$ purevol list vol2 vol3
Name  Size  Serial
vol2  300G  7CBFCE0B2660CC3000010002
vol3  250G  7CBFCE0B2660CC3000010003
$ purevol truncate --size 275g vol2 vol3 ④
Name  Size  Serial
vol2  275G  7CBFCE0B2660CC3000010002
Error on vol3: Target size exceeds volume size.
$ purevol list vol2 vol3
Name  Size  Serial
vol2  275G  7CBFCE0B2660CC3000010002
vol3  250G  7CBFCE0B2660CC3000010003
```

- ① Increases the sizes of `vol1` and `vol2` from 100 gigabytes and 200 gigabytes respectively to 300 gigabytes.
- ② Reduces the sizes of `vol3` and `vol4` from 200 gigabytes to 100 gigabytes.
- ③ Attempts to change the sizes of `vol2` and `vol3` to 250 gigabytes. The command succeeds for `vol3`, but fails for `vol2` because it is already larger than 250 gigabytes.
- ④ Attempts to change the sizes of `vol2` and `vol3` to 275 gigabytes. The command succeeds for `vol2`, but fails for `vol3` because it is already smaller than 275 gigabytes (the "target size").

Increasing a volume's host-perceived size does not increase its consumption of physical storage. Volumes only consume storage when hosts write data to previously unused volume block addresses or overwrite blocks with less-compressible data.

When a volume is truncated, Purity//FA automatically takes an *undo snapshot* (providing a 24-hour window during which the previous contents can be retrieved). If the physical storage occupied by data in truncated sectors is urgently required, the eradication pending period can be terminated, and space reclamation initiated immediately by calling **purevol eradicate** on the undo snapshot.

## Destroying Volumes

When the data stored in a volume is no longer of interest, the **purevol destroy** command obliterates the data it contains and frees the storage it occupies. Destroyed volumes become invisible to hosts and to most CLI and GUI displays immediately, but their data is not obliterated, nor is the storage it occupies reclaimed until a 24-hour *eradication pending* period has elapsed. During this period, a destroyed volume can be recovered with its data intact. The **purevol eradicate** command terminates the eradication pending period and begins storage reclamation immediately. Once eradication has begun, a volume can no longer be recovered.

Example 11.5 illustrates the use of the **purevol destroy** command.

### Example 11.5: Destroying Volumes

```
$ purevol list --connect vol1 ①
Name Size LUN Hgroup Host
vol1 100G 1          host1
$ purevol destroy vol1 ②
Error on vol1: Cannot destroy volume because it is currently connected.
$ purevol disconnect --host host1 vol1
Name Hgroup Host
vol1          host1
root@gt2-ct0:~# purevol destroy vol1 ③
Name
vol1
$ purevol list ④
Name  Size  Serial
vol2  200G  27CA508DA38AF9F400010001
vol3  300G  27CA508DA38AF9F400010002
vol4  300G  27CA508DA38AF9F400010003
$ purevol list --pending ⑤
Name  Size  Time Remaining  Source  Created           Serial
vol1  100G  1 day, 0:00:00 -      2013-04-01 15:45 PDT  27CA508DA38AF9F400010000
vol2  100G  -          -      2013-04-01 15:46 PDT  27CA508DA38AF9F400010001
vol3  100G  -          -      2013-04-01 15:47 PDT  27CA508DA38AF9F400010002
vol4  100G  -          -      2013-04-01 15:48 PDT  27CA508DA38AF9F400010003
```

- ① As background, illustrates that **vol1** is connected to **host1**
- ② Attempt to destroy **vol1** fails, because it has a connection.

- ❸ After **vol1**'s connection to **host1** is broken, the **purevol destroy** command succeeds.
- ❹ Illustrates that a volume in the eradication pending state (**vol1**) is only displayed by the **purevol list** command, when the **--pending** option is specified.

## Recovering and Eradicating Destroyed Volumes

To provide a margin of error for administrators, Purity//FA retains a volume's contents and structure for 24 hours after it is destroyed. The **purevol recover** command restores a destroyed volume to its size at the time of destruction with its contents intact, as Example 11.6 illustrates.

### Example 11.6: Recovering and Eradicating Volumes

```
$ purevol destroy vol1
Name
vol1
$ purevol list --pending vol1
Name  Size  Source  Time Remaining  Created          Serial
vol1  100G -        1 day, 0:00:00  2013-04-12 15:45 PDT  27CA508DA38AF9F400010004
$ purevol recover vol1 ❶
Name
vol1
$ purevol list vol1
Name  Size  Source  Created          Serial
vol1  100G -        2013-04-12 15:45 PDT  27CA508DA38AF9F400010004
$ purevol destroy vol2
Name
vol2
$ purevol eradicate vol2 ❷
Name
vol2
$ purevol recover vol2 ❸
Error on vol2: Volume does not exist.
```

- ❶ Restores destroyed volume **vol1** (provided that the command executes within the volume's eradication pending period).
- ❷ Obliterates the data from destroyed volume **vol2**, and begins reclamation of the storage it had occupied.
- ❸ Fails to recover **vol2**, illustrating that once it is eradicated, a volume can no longer be recovered.

## Renaming Volumes

Volume names identify volumes in CLI and GUI commands and displays. Volume names are assigned at creation, and, as with other objects can be changed at any time by the **purevol rename** command, as Example 11.7 illustrates. Renaming a volume has no effect on its attributes or connections.

### Example 11.7: Renaming Volumes

```
$ purevol rename vol4 NewVol4
```

```
Name
NewVol4
$ purevol list vol4
Error on vol4: Volume does not exist.
$ purevol list NewVol4
Name      Size   Source   Created           Serial
NewVol4   300G   -        2013-04-12 15:48 PDT  27CA508DA38AF9F400010003
```

The name of **vol4** is changed to **NewVol4**. The **purevol list** commands that follow illustrate that after renaming, a volume is immediately visible under its new name, and is no longer visible under its former name. All attributes, including the volume serial number, which is visible to hosts, remain unchanged.

## Ongoing Administration of Hosts

Purity//FA host administration includes the following tasks:

- Creation, deletion and renaming of host objects
- Connecting hosts to and disconnecting them from volumes
- Changing the IQNs, NQNs, or WWNs associated with a host
- Adding hosts to and removing them from host groups.

Example 11.2 illustrates host creation.

Example 11.3 illustrates both connection of multiple volumes to a single host (**purevol connect**) and multiple hosts to a single volume (**purehost connect**).

Example 11.8 illustrates the remaining host operations.

### Example 11.8: Administrative Operations on Hosts

```
$ purehgroup listobj --type host hgroup1 ❶
host1
host2

$ purehost list --connect host1 ❷
Name  LUN  Vol  Host Group
host1  1    vol1

$ purehost delete host1 ❸
Error on host1: Could not delete host.
$ purehgroup setattr --remhost host1 hgroup1
Name      Hosts
hgroup1  host1

$ purehost delete host1 ❹
Error on host1: Could not delete host.
$ purehost disconnect --vol vol1 host1 ❺
Name      Vol
host1    vol1
```

```
$ purehost delete host1
Name
host1
```

- ❶ These commands illustrate that **host1** is associated with host group **hgroup1**, and has a private connection to **voll**.
- ❷ Attempt to delete **host1** fails because it is associated with a host group.
- ❸ After **host1** is removed from **hgroup1**, the attempt to delete it still fails because it is still connected to volume **voll**.
- ❹ Deletion of **host1** succeeds because it is not associated with a host group and has no private connections to volumes.

## Monitoring I/O Performance

The **purevol monitor** command produces periodic displays of I/O performance information for some or all of an array's volumes, as Example 11.9 illustrates. Use **purehost monitor**, **purehgroup monitor**, and **purearray monitor** commands to display I/O performance information for host, host group, or the whole array.

### Example 11.9: Monitoring I/O Performance

```
$ purevol monitor --interval 1 --repeat 3 --total
Name      Time          B/s (read)  B/s (write)  op/s (read)  op/s (write)  us/op (read)
voll     2014-08-15 13:23:19 PDT  3.16M       1.51M       808         386        119
vol2     2014-08-15 13:23:19 PDT  1.53M       1.58M       392         405        93
(total)  2014-08-15 13:23:19 PDT  4.69M       3.09M       1K          791        110
Name      Time          B/s (read)  B/s (write)  op/s (read)  op/s (write)  us/op (read)
voll     2014-08-15 13:23:20 PDT  3.23M       1.47M       828         376        117
vol2     2014-08-15 13:23:20 PDT  1.57M       1.63M       401         417        92
(total)  2014-08-15 13:23:20 PDT  4.80M       3.10M       1K          793        109
Name      Time          B/s (read)  B/s (write)  op/s (read)  op/s (write)  us/op (read)
voll     2014-08-15 13:23:21 PDT  3.05M       1.55M       780         398        119
vol2     2014-08-15 13:23:21 PDT  1.57M       1.53M       401         391        91
(total)  2014-08-15 13:23:21 PDT  4.61M       3.08M       1K          789        109
```

The **--interval** option specifies the number of seconds between displays of data. The **--repeat** option specifies the number of displays that the command produces. The default interval is 5 seconds; the default repetition count is 1.

Displays read and write operations and data transfer performance for volumes **voll** and **vol2** with totals. Three sets of results are produced (**--repeat 3**) at intervals of one second (**--interval 1**).

## Using listobj Output

The shell in which the Purity//FA CLI runs supports *command substitution*, the embedding of a list-type *inner command* within an *outer command*, so that the output of the inner command becomes the argument list for the outer one. The **listobj** subcommands of the

`purehost`, `purehgroup`, `purevol`, and `purealert` commands are included in the CLI primarily for this purpose.

The output of an embedded command becomes input to the command in which it is embedded. This feature can be used, for example, to operate on selected sets of objects, as Example 11.10 illustrates.

### Example 11.10: Using the Output of the `listobj` Subcommand

```
purevol list $(purehost listobj --type vol HOST1 HOST2)
Name      Size  Source  Created          Serial
vol35     4T    -       2013-09-03 16:47:18 PDT 64939A0301530F5400010050
vol57     6T    -       2013-11-04 16:42:23 PST 64939A0301530F5400010054
vol58     3T    -       2013-02-25 13:42:16 PST 64939A0301530F5400010045
vol634    4000G -       -                  64939A0301530F540001001B
vol659    3T    -       2013-02-25 13:48:46 PST 64939A0301530F5400010046
vol1324   4000G -       -                  64939A0301530F540001001A
vol256    4T    -       -                  64939A0301530F5400010024
```

In the above example, the inner `purehost listobj` command produces a whitespace-separated list of the volumes connected to HOST1 and HOST2. The outer `purevol list` command displays the names, sizes, and serial numbers of the volumes specified in the inner command.

Administrators can also use shell scripting to construct sequences of commands dynamically. Example 11.11 illustrates one use of scripting.

### Example 11.11: Shell Scripting with the `listobj` Subcommand

```
$ BASENAME=VOL
$ for I in 100 150 200
> do
> purevol create --size $I\g $BASENAME$I\GB
> purevol connect --host host1 $BASENAME$I\GB
> done
Name      Size  Source  Created          Serial
VOL100GB  100G  -       2013-04-12 15:57 PDT 1B4B44F7DEAEF5000010010
Name      Host Group  Host   LUN
VOL100GB  -        host1 3
Name      Size  Source  Created          Serial
VOL150GB  150G  -       2013-04-12 15:57 PDT 1B4B44F7DEAEF5000010011
Name      Host Group  Host   LUN
VOL150GB  -        host1 4
Name      Size  Source  Created Serial
VOL200GB  200G  -       2013-04-12 15:57 PDT 1B4B44F7DEAEF5000010012
Name      Host Group  Host   LUN
VOL200GB  -        host1 5
```

The `purevol create` command within the loop creates three volumes whose names are keyed to their sizes: `VOL100GB` has a size of 100 gigabytes, etc. The `purevol connect` command, also within the loop, connects each newly-created volume to `host1` in turn.

Example 11.12 demonstrates another way to use the `listobj` subcommand to build an argument list.

### Example 11.12: Building Arguments with the `listobj` Subcommand

```
$ for VOLNAME in $(purehost listobj --type vol host1)
> do
> purevol setattr --size 1T $VOLNAME
> done
Name      Size
vol046    1T
Name      Size
vol050    1T
Name      Size
vol033    1T
Name      Size
vol045    1T
Name      Size
vol032    1T
Name      Size
vol037    1T
```

The `purehost listobj` command creates a whitespace-separated list of volumes that have private connections to `host1`. The list becomes the set of objects for which the do loop iterates. The `purevol setattr --size` command within the loop uses the volume names to resize each of the volumes in succession.

# Part 4: Supplementary Information



## Appendix A. The Pure Storage Glossary

<b>administrator</b>	An individual who administers a FlashArray system. Administrators may be authorized to control an array or may be limited to a monitor role, in which they can observe the array's configuration, state, and performance, but cannot alter its operating parameters.
<b>allocation unit</b>	The unit in which Purity//FA manages flash storage. Each allocation unit consists of several consecutively numbered erase blocks on a flash module. Purity//FA allocates storage in groups of allocation units on different flash modules.
<b>array management port</b>	An Ethernet interface in each FlashArray controller that provides administrative access to the array from browser or virtual terminal-equipped workstations.
<b>back-end controller interconnect network</b>	The dual-path network that interconnects FlashArray controllers. FlashArray FA-300 and FA-400 series uses InfiniBand ports.
<b>back-end controller interconnect port</b>	A FlashArray back-end controller port used for interconnecting with other controllers in the array.
<b>back-end storage interconnection port</b>	A Serial Attached SCSI (SAS) port in a FlashArray controller or storage shelf used to connect controllers to storage shelves and the flash modules within them.
<b>basic object</b>	A manageable object in a FlashArray system. Basic objects may be hardware (controllers, storage shelves, flash modules, NVRAM modules, front-end ports) or virtual (volumes, hosts, and host groups).
<b>client, client computer</b>	Synonym for host.
<b>client block map</b>	A set of persistent data structures in which Purity//FA maintains the correspondence between volume block addresses exported to hosts and physical storage locations of data.
<b>command line interface (CLI)</b>	The Purity//FA virtual console-based tool used by administrators to monitor and control FlashArray systems. Administrators launch the Purity//FA CLI via a secure shell (ssh) connection (for Windows administrative workstations, typically through a terminal emulator such as PuTTY).
<b>continuous storage reclamation</b>	The Purity//FA process that continually frees lightly occupied allocation units by moving data from them to more densely populated ones. Purity//FA reuses reclaimed allocation units to store host-written data and data consolidated from reclaimed storage.

<b>controller</b>	The FlashArray component that contains a processor complex, DRAM cache, front-end and back-end ports, and an administrative port. The two controllers in a highly-available array cooperate with each other to provide a failure tolerant, high-performing, single-image storage system.
<b>controller pair</b>	Synonym for partner controllers.
<b>data reduction</b>	Reduction in the size of a set of data by elimination of duplication. Purity//FA reduces blocks of data written by hosts by eliminating repeating patterns, deduplicating entire blocks against each other, and compressing non-duplicate blocks, continuously throughout their lives in an array.
<b>disk block</b>	Synonym for sector. The atomic unit in which magnetic disk drives read, write, and ECC-protect data. Analogous to a flash page in a flash module.
<b>eradication pending state</b>	A state in which an administrator places a volume by executing a <b>purevol destroy</b> [469] command against it. Purity//FA does not export volumes in the eradication pending state; they are invisible to most administrative operations. Volumes remain in the eradication pending state for 24 hours, after which Purity//FA <i>eradicates</i> them, deleting their data and freeing the storage capacity they occupy.
<b>erase block</b>	The unit of memory in which flash memory erases data prior to overwriting. Each erase block consists of multiple consecutively addressed flash pages.
<b>export</b>	(v.) To present one or more objects to clients. A FlashArray system exports disk-like volumes to its clients (host computers).
<b>Fibre Channel front-end port</b>	A FlashArray controller front-end port that implements Fibre Channel physical signaling and low-level protocol.
<b>flash module</b>	A data storage device based on a persistent flash technology. Solid-state drives are the physical data storage elements in FlashArray systems.
<b>flash page</b>	The atomic unit in which data is read from flash memory.
<b>FlashArray</b>	A scalable enterprise-class high-performance data storage system based entirely on flash storage technology.
<b>front-end host connection port</b>	A FlashArray controller port used to connect host(s) to the array, usually via a storage network. Host connection ports implement the Fibre Channel, iSCSI, or NVMe-oF protocol.
<b>graphical user interface (GUI)</b>	A user interface to a computer that represents objects and methods graphically rather than with character strings. The Purity//FA GUI runs within the context of a browser.
<b>group object</b>	A Purity//FA construct for managing collections of basic objects (hosts and volumes). Administrators can create group objects for management convenience, particularly in arrays that support large numbers of volumes.

<b>high availability</b>	The ability of a system to sustain a major component failure and continue to perform its function. Typically achieved by configuring and integrating redundant components and connections and implementing software that detects and isolates failures, “heals” the functions they affect, and assists in repair, replacement, and reassimilation of components.
<b>host</b>	A server or desktop computer connected to a FlashArray system via a storage network that makes use of the array's data storage services. Hosts may be physical or virtual, and are identified to Purity//FA by the port names used to connect them to the FlashArray system.
<b>host capacity</b>	Synonym for size (q.v.).
<b>host occupancy</b>	The amount of host-visible storage occupied by host-written data: the number of unique volume sector addresses written by hosts (and not trimmed) multiplied by the size of a sector (512 bytes).
<b>host port (H-port)</b>	An I/O port in a host used to connect to a FlashArray system, usually via a storage network.
<b>host-written data</b>	Data written by a host to one or more volume sector addresses in a FlashArray volume.
<b>InfiniBand (IB)</b>	An interconnect standard created by server manufacturers. Used primarily to connect servers at high speeds over short distances. The FlashArray FA-300 and FA-400 series use InfiniBand to interconnect a highly-available array's controllers. FlashArray//X and //M use InfiniBand ports for upgrades.
<b>initial data reduction</b>	Purity//FA's first-pass reduction of data as it enters an array. Initial data reduction includes elimination of patterned blocks and may also include deduplication and compression of non-duplicates.
<b>interposer</b>	A device that provides dual-channel access to a flash module and also converts between the SATA protocol native to FlashArray flash modules and the SAS protocol used to communicate with controllers. Interposers are mounted in the storage carriers that hold flash modules.
<b>I/O card</b>	A PCI Express interface module in a FlashArray controller containing either SAS ports that connect to storage shelves, InfiniBand or NTB ports used to interconnect controllers, or Fibre Channel, iSCSI, or NVMe ports that provide access to the external environment. The FlashArray FA-300 and FA-400 series use InfiniBand ports.
<b>I/O module</b>	A FlashArray storage shelf component containing a SAS expander that fans out an incoming SAS bus to the flash modules and NVRAM modules within the shelf and to other shelves downstream from it. Each storage shelf contains two I/O modules, providing redundant access paths to the flash modules and NVRAM modules within it.

<b>I/O performance density</b>	The number of I/O operations per second of which a storage system is capable divided by the effective data storage capacity of the system. For example, a storage system that presents 10 terabytes of usable storage and is capable of 100,000 IOPS has an I/O performance density of 10,000 IOPS per terabyte.
<b>I/O port</b>	A connection point (socket) on an I/O card. I/O ports interconnect array components, and provide access to the external environment for I/O. The I/O ports in storage shelves are SAS ports. Those in controllers are back-end SAS, InfiniBand ports, or Fibre Channel, iSCSI, or NVMe-oF front-end ports.
<b>IOPS</b>	I/O operations per second. The preferred measure of I/O performance with random (non-sequential) I/O workloads.
<b>logical block</b>	Synonym for sector.
<b>logical block address (LBA)</b>	The mechanism by which logical blocks are addressed in read and write commands to disk drives and flash modules.
<b>logical block number (LBN)</b>	Synonym for logical block address.
<b>logical page</b>	The atomic unit in which Purity//FA reads data from flash modules. Consists of one or more flash pages in which data and metadata are stored. Purity//FA computes and stores a checksum in each logical page, and recalculates it when the page is read to detect read errors that are not discovered by flash devices' mechanisms.
<b>logical page checksum</b>	A checksum that Purity//FA calculates and appends to each logical page it writes for the purpose of detecting read errors that are not detected by flash devices' ECC mechanisms.
<b>logical unit</b>	Synonym for volume.
<b>logical unit number (LUN)</b>	The mechanism for addressing a logical unit or volume.
<b>management port (M-port)</b>	A FlashArray controller port used for connecting a management workstation to the array, usually via a data center network.
<b>micro-provisioning</b>	Purity//FA's advanced implementation of thin provisioning, in which only the storage required to contain reduced data blocks and the metadata that describes them is allocated.
<b>non-volatile random access memory (NVRAM)</b>	Randomly addressable memory whose contents survive power outages intact. FlashArray controllers store host-written data temporarily in NVRAMs mounted in storage shelves prior to writing it to flash modules, so that power outages and system resets do not result in data loss.

<b>object database</b>	A persistent database used internally by Purity//FA to maintain records related to both the intrinsic and administrator-defined virtual objects in an array.
<b>occupancy</b>	A measure of the physical or host-visible storage occupied by host-written data.
<b>page</b>	In this guide, a synonym for flash page (q.v.).
<b>partner controllers</b>	Two Pure Storage controllers connected to the same (two or more) storage shelves. Partner controllers maintain synchronized copies of each others' NVRAM contents so that if one of them should fail, its partner can control all flash modules and export volumes seamlessly to hosts on its front-end host connection ports.
<b>patterned block</b>	A block of data received from a host whose contents consist of a repeated data pattern of between 1 and 8 bytes. Purity//FA does not allocate storage for patterned block data, but records their existence in its internal metadata.
<b>physical occupancy</b>	The amount of physical storage occupied by host-written data after reduction by Purity//FA.
<b>port</b>	An interface between FlashArray components (controllers and storage shelves), or between controllers and a network or host. Array controllers contain front-end ports for connecting to hosts via storage networks, back-end ports that interconnect the controllers in a highly available array, back-end SAS ports that connect controllers to storage shelves, and GbE administrative ports that connect controllers to a network for administration. Storage shelves contain SAS ports that connect to controllers and interconnect storage shelves with each other.
<b>Purity//FA CLI</b>	The CLI used to administer a FlashArray system. The Purity//FA CLI runs on an array and is accessed via a virtual terminal emulator such as PUTTY.
<b>Purity//FA GUI</b>	The browser-based graphical user interface used to administer FlashArray systems.
<b>Purity//FA Operating Environment (Purity//FA)</b>	The operating system and array logic that run in each FlashArray controller.
<b>RAID-3D</b>	FlashArray's dynamic multi-level scheme for protecting against data loss due to uncorrectable read errors and device failures. RAID-3D minimizes the impact of read error recovery, and automatically adjusts protection parameters based on the nature of stored data and conditions within an array.

<b>reduction</b>	Synonym for data reduction.
<b>reduction ratio</b>	Any of several ratios of the amount of physical storage occupied by data and the data's host-visible size.
<b>sector</b>	The common 512-byte unit in which data is written to, read from, and ECC-protected on disk drives and flash modules. Pure Storage volumes present virtual sectors of storage in which hosts can read and write data.
<b>segment</b>	A Purity//FA data structure consisting of groups of allocation units, each on a separate flash module. The segment is the unit in which Purity//FA allocates storage capacity for writing data and metadata, as well as the unit of RAID protection. Purity//FA assigns each segment's RAID protection scheme at allocation time.
<b>Serial-Attached SCSI (SAS)</b>	The industry standard technology that connects storage shelves (and the flash modules within them) to controllers in FlashArray systems.
<b>size</b>	The storage capacity of a volume as it appears to hosts' storage administration tools.
<b>snapshot</b>	A host-accessible virtual image of the contents of a set of data as they appeared at some point in time. Purity//FA snapshots capture virtual images of volume contents, and appear to hosts as read-only volumes.
<b>storage area network (SAN), storage network</b>	A network whose primary purpose is to enable communication between hosts and storage devices or systems. FlashArray systems support iSCSI, NVMe, or Fibre Channel storage networks.
<b>storage shelf</b>	The FlashArray component that houses flash modules. In a highly available array, each storage shelf connects to a pair of controllers to provide redundant access and performance scaling.
<b>thin provisioning</b>	A storage virtualization technique in which allocation of physical storage to volume blocks is deferred until a host writes data to the volumes' block addresses.
<b>trim</b>	(v.) Deallocation of storage capacity that holds host-written blocks that a host has indicated are no longer in use (via a SCSI trim command).
<b>virtual space</b>	The amount of virtual space in a volume or array that is occupied by host-written data.
<b>volume</b>	A disk-like random access virtual storage device that a FlashArray system exports to hosts via a logical unit number (LUN). To a host, a FlashArray volume contains a number of 512-byte sectors in which data can be written and from which it can be read.
<b>write amplification</b>	The generation of additional I/O internal to a conventional array or flash storage as a consequence of a single write request. A write to a RAID

group in an array that is smaller than the group's data stripe size results in one or more internally generated log-read-modify-write sequences in order to synchronize RAID check data. A write to a flash module may require block erasure, resulting in a read of an erase block to preserve existing data, a block erasure, and an overwrite of the block with a combination of new and existing data. Write amplification is functionally invisible to the writer, but takes time and consumes internal bandwidth.

**write buffer**

A DRAM buffer in which Purity//FA stages data for writing to persistent flash module storage. The size of a write buffer is the same as that of a write stripe unit. Purity//FA writes all buffers in a write stripe in sequence.

© 2018 Pure Storage, Inc. All rights reserved. Pure Storage, Pure1, and the Pure Storage logo are trademarks or registered trademarks of Pure Storage, Inc. in the U.S. and other countries. Other company, product, or service names may be trademarks or service marks of their respective owners.

The Pure Storage products described in this documentation are distributed under a license agreement restricting the use, copying, distribution, and decompilation/reverse engineering of the products. The Pure Storage products described in this documentation may only be used in accordance with the terms of the license agreement. No part of this documentation may be reproduced in any form by any means without prior written authorization from Pure Storage, Inc. and its licensors, if any. Pure Storage may make improvements and/or changes in the Pure Storage products and/or the programs described in this documentation at any time without notice.

THIS DOCUMENTATION IS PROVIDED "AS IS" AND ALL EXPRESS OR IMPLIED CONDITIONS, REPRESENTATIONS AND WARRANTIES, INCLUDING ANY IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NONINFRINGEMENT, ARE DISCLAIMED, EXCEPT TO THE EXTENT THAT SUCH DISCLAIMERS ARE HELD TO BE LEGALLY INVALID. PURE STORAGE SHALL NOT BE LIABLE FOR INCIDENTAL OR CONSEQUENTIAL DAMAGES IN CONNECTION WITH THE FURNISHING, PERFORMANCE, OR USE OF THIS DOCUMENTATION. THE INFORMATION CONTAINED IN THIS DOCUMENTATION IS SUBJECT TO CHANGE WITHOUT NOTICE.



Pure Storage, Inc.  
Twitter: @purestorage

650 Castro Street, Suite 260  
Mountain View, CA 94041  
800-379-7873

Sales: sales@purestorage.com  
Support: support@purestorage.com  
Media: pr@purestorage.com  
General: info@purestorage.com