PURESTORAGE

Pure Storage OpenStack (2023.2 [Bobcat]) Cinder Driver Best Practices

Simon Dodsley, Director

Version 1.0, October 2, 2023

Contents

Notices

Pure Storage, Inc.
650 Castro Street
Mountain View, CA 94041

Executive Summary

OpenStack® is an open source cloud computing platform designed as an Infrastructure as a Service (IaaS) implementation. It has the ability to control large numbers of compute, network and storage resources.

The block storage portion of OpenStack is referred to as the Cinder project and aims to provide persistent block level storage to the OpenStack compute (Nova) instances. The Pure Storage Cinder driver is a Python®-based module integrated into the main OpenStack distributions.

This document describes the best practices for using the Pure Storage® OpenStack™ 2023.2 (Bobcat) Cinder driver and details specific configuration recommendations to get the best from the Pure Storage Cinder driver.

This document assumes the use of OpenStack 2023.2 (Bobcat).

Audience

This document describes how to configure an OpenStack Cinder node with a Pure Storage FlashArray providing the shared storage. The intended audiences for this document include the following:

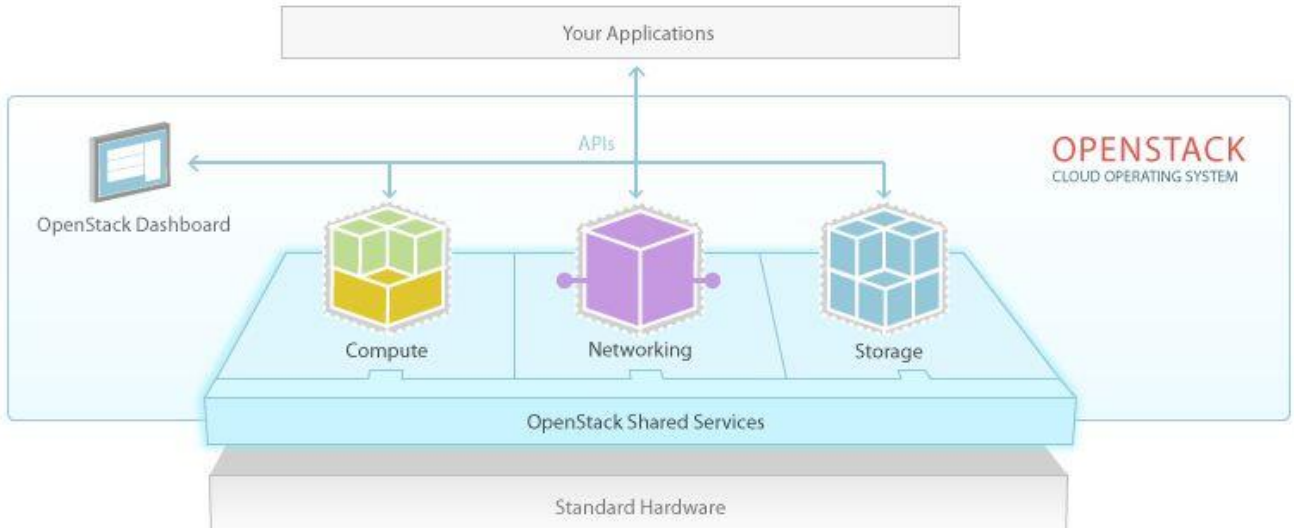
- OpenStack administrators, who configure and use OpenStack Cinder
- Pure Storage system administrators, who configure and use the Purity Operating Environment
- Pure Storage partners and solutions engineers, who support the Pure Storage FlashArray Volume Driver for OpenStack Cinder

This guide assumes familiarity with storage and networking concepts, OpenStack administration, Cinder administration, the Purity Operating Environment, and Pure Storage's iSCSI, Fibre Channel and NVMe best practices.

OpenStack Architecture

OpenStack provides services to allow storage, compute and network resources to be connected together to form a cloud using independent programs.

The high-level architecture is shown here.



At the time of writing these are the OpenStack services and associated project names:

Services	OpenStack Project
Object Storage	Swift
Compute	Nova
Image	Glance
Dashboard	Horizon
Identity	Keystone
Network	Neutron
Block Storage	Cinder
Telemetry	Ceilometer
Orchestration	Heat
Database	Trove
Data Processing	Sahara
Bare Metal	Ironic

Cinder Architecture

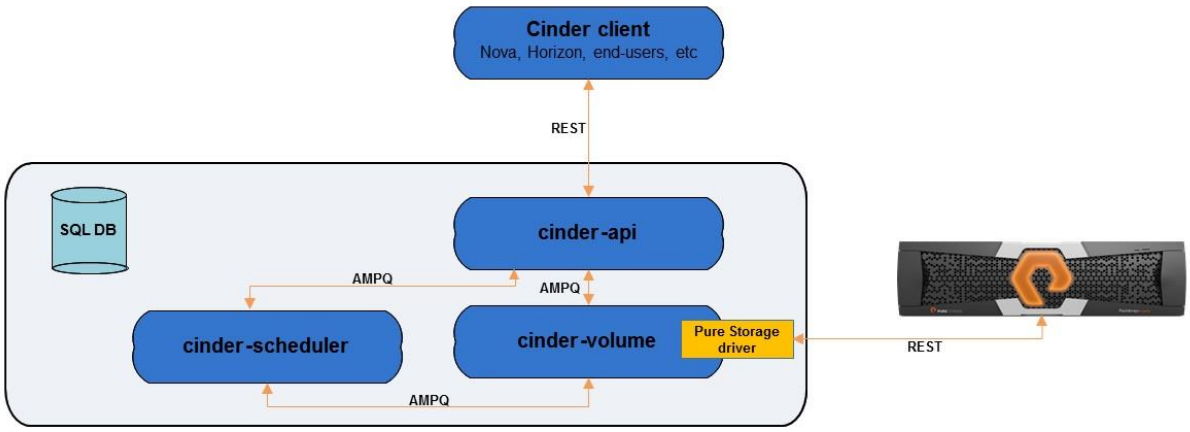
Cinder is the Block Storage provider to OpenStack and relies on 3 main components: Cinder-API, Cinder-Scheduler, and Cinder-Volume. Communication between these components is achieved using an advanced message queue (AMQP). Other projects within OpenStack communicate with Cinder using REST protocols. Cinder communicates with other projects using their own REST protocol.

The Cinder-API service provides the RESTful API for the major volume operations, such as create, delete, snapshot, expand, etc.

The API service sends requests using AMQP to the Scheduler service. This then ensures that sufficient resources are available to service before dispatching the requests to the Volume service. There will be a Volume service instance for each Cinder backend configured in the environment.

In the case of a backend array from Pure Storage, the Cinder-Volume instance will load the Pure Storage Cinder driver to control and communicate with the backend array.

A high-level view of this can be seen below:



Pure Storage Cinder Driver

Supported Versions

The Pure Storage Cinder driver is supported for all Pure Storage FlashArray models running Purity 5.3.0 or higher.

Getting the Pure Storage Cinder driver

Pure Storage has contributed a Cinder driver since the official Juno release of the OpenStack project.

Development of the driver has added support for multiple data transport layers at different OpenStack versions. These are detailed below.

Transport Layer	Support from…
iSCSI	Juno
Fibre Channel	Liberty
NVMe-RoCE	Zed
NVMe-TCP	2023.2 (Bobcat)

Installation

The Pure Storage Cinder driver is available as a standard driver under the Cinder Volume service. Please refer to the installation guide of your specific OpenStack deployment, or the main OpenStack website (http://www.openstack.org), to ensure you have the Cinder project correctly installed.

After ensuring that your OpenStack deployment has the Cinder Volume service correctly installed, it is necessary to additionally install the Pure Storage Python SDK kit. This is publicly available and is installed using the following command:

```
pip3 install purestorage
```

Ensure that you have the *python-pip* package installed prior to running this command, using the appropriate package installation procedures for your Linux distribution. The Pure Storage Python SDK has a prerequisite of the *requests* Python package, therefore ensure that this is available.

If you are using the NVMe driver then you must also have the NVMe CLI software installed for your operating system. This is usually called `nvme-cli`, but may vary depending on operating system flavour.

NTP Configuration

It is recommended that the Pure Storage FlashArray and all your OpenStack controller and compute nodes are configured to use NTP.

If the array and OpenStack nodes are significantly out of sync, you may experience error messages in the Cinder volume controller and REST logs that show as 401 errors.

Kernel module configuration (NVMe-only)

If you are an NVMe driver then you must ensure that the appropriate kernel modules have been installed.

For example, in Ubuntu-based deployments use the following commands:

```
modprobe nvme-core
modprobe nvme-fabrics
modprobe nvme-rdma  (for NVME-RoCE driver)
modprobe nvme-tcp  (for NVMe-TCP driver)
```

MPIO Configuration (iSCSI or FC)

To ensure multipath volume accessibility for all volumes presented from the Pure Storage FlashArray, it is essential to perform the following steps:

- Install the `device-mapper-multipath` or `multipath-tools` package (depending on your Linux distribution) on nodes running the Cinder Volume service and the Nova Compute service
- Install `sysfsutils` and `sg3_utils` on nodes running both the Cinder Volume service and the Nova Compute service
- Set `volume_use_multipath = True` in the `[libvirt]` section of `/etc/nova/nova.conf` for any nodes using iSCSI as the Cinder block protocol driver. Certain OpenStack deployments may set this value through different methods. Please refer to your deployments configuration guide for specific details.
- Set `use_multipath_for_image_xfer = True` in `/etc/cinder/cinder.conf` for each required backend array stanza on the Cinder volume service node(s)

Modify the multipathing software configuration file which contains the information required for the Pure Storage FlashArray. Example entries in the `/etc/multipath.conf` configuration file is shown here:

It is recommended to check the latest Pure Storage Linux Recommended Settings to ensure these parameters have not been updated since publication of this document

```
defaults {
    polling_interval          10
    user_friendly_names       no
}

devices {
    device {
        vendor                "PURE"
        path_selector         "queue-length 0"
        path_grouping_policy  multibus
```

```
        path_checker            tur
        fast_io_fail_tmo        10
        dev_loss_tmo            60
        no_path_retry           0
        features                0
    }
}
```

Finally, ensure that the multi-pathing software service is enabled and running.

In newer versions of Linux `multipath-tools`, the Pure Storage FlashArray information is included in the default settings, therefore, there is no requirement to modify the `/etc/multipath.conf` file. For example, Red Hat Enterprise Linux v7.3. To check if the parameters are included in your Linux release use the `multipathd show config` command.

MPIO Configuration (NVMe)

OpenStack currently only supports native NVMe multipathing, therefore it is not necessary to configure any multipath configuration files.

However, `multipathd` must still be installed and running due to an architectural issue within OpenStack.

To ensure that native NVMe multipathing is enabled issue the following command:

```
cat /sys/module/nvme_core/parameters/multipath
```

and ensure the response is **Y.**

Native NVMe multipath is a drastic simplification of the host I/O stack, and Pure Storage looks forward to it maturing enough for us to recommend it over DM multipath. However, in its current state, we believe the existence of high-severity bugs and the lack of an adaptive path selection policy mean it is not ready for production deployments.

Modify the Cinder Configuration File

The `/etc/cinder/cinder.conf` file must include settings to allow the Cinder Volume service to communicate with and control the Pure Storage FlashArray. To enable this to happen, create a new stanza for the Pure Storage FlashArray, such as `[PURE]`, and add the following parameters:

```
volume_driver=cinder.volume.drivers.pure.PureFCDriver
or
volume_driver=cinder.volume.drivers.pure.PureISCSIDriver
or
volume_driver=cinder.volume.drivers.pure.PureNVMEDriver

san_ip=<MANAGEMENT-INTERFACE>
pure_api_token=<API-TOKEN>
```

where

MANAGEMENT-INTERFACE: The IP address for the FlashArray's management interface virtual IP address or the fully qualified domain name assigned to both array controllers. This is required to maintain high availability and support non-disruptive firmware updates to the array.

API-TOKEN: The Purity API token that the volume driver uses to perform volume management on this FlashArray. More details on how to obtain the API Token are given in the Pure Storage FlashArray documentation which can be accessed through the FlashArray GUI.

NOTE: If you are using the Fibre Channel driver, then it will also be necessary to add the following line to the `[DEFAULT]` section of the cinder configuration file:

```
zoning_mode=fabric
```

and add `[fc-zone-manager]` and `[fabric-x]` sections to the configuration file as well. For more details on this, please refer to the OpenStack documentation from your fabric switch vendor.

NOTE: If you are using the NVMe driver, then it will also be necessary to add the following line to the individual backend configuration stanza section of the cinder configuration file:

```
nvme_transport_type = <type>
```

where `type` supports `roce` or `tcp` as options. The default value should this line be missing from the configuration file for an NMVe backend is `roce`.

Restarting OpenStack Services

After modifying any OpenStack configuration file, it is required to restart the associated service to ensure your modifications are accepted and implemented.

Other Configuration Requirements

Pure Storage also recommends additional configuration changes to HBA and Queue setting to optimise for high-performance flash-based storage.

For more details on these settings and also more details on the multipath configuration mentioned above, please refer to the **Linux® Recommended Settings** article available in the Pure1® Support website (http://support.purestorage.com).

Additional Volume Metadata

Volume metadata is added to every volume created on a Pure backend describing the real-world volume name (`array_volume_name`) and array name (`array_name`) the OpenStack volume resides on.

The additional volume metadata can be obtained from the OpenStack GUI by selecting the volume, or by using the `openstack volume show` command, as shown here:

```
$ openstack volume show b2205d05-4a2e-4818-b95c-59974ffdf417
+------------------------------+-------------------------------------------------------------+
| Field                        | Value                                                       |
+------------------------------+-------------------------------------------------------------+
| attachments                  | []                                                          |
```

```
| availability_zone              | nova                                                                |
| bootable                       | false                                                               |
| consistencygroup_id            | None                                                                |
| created_at                     | 2022-03-22T19:00:48.000000                                          |
| description                    | None                                                                |
| encrypted                      | False                                                               |
| id                             | b2205d05-4a2e-4818-b95c-59974ffdf417                                |
| migration_status               | None                                                                |
| multiattach                    | False                                                               |
| name                           | None                                                                |
| os-vol-host-attr:host          | db-10@puredriver-1#puredriver-1                                     |
| os-vol-mig-status-attr:migstat | None                                                                |
| os-vol-mig-status-attr:name_id | None                                                                |
| os-vol-tenant-attr:tenant_id   | 0df76fc99df744328c4783a84ca4423f                                    |
| properties                     | array_name='pure-fa1', array_volume_name='roce-b2205d05-
4a2e-4818-b95c-59974ffdf417-cinder' |
| replication_status             | disabled                                                            |
| size                           | 2                                                                   |
| snapshot_id                    | None                                                                |
| source_volid                   | None                                                                |
| status                         | available                                                           |
| type                           | puredriver-1                                                        |
| updated_at                     | 2022-03-22T19:00:48.000000                                          |
| user_id                        | 6d14ae58c5834670a1d253f95d9cf326                                    |
+--------------------------------+---------------------------------------------------------------------+
```

Advanced Functionality

Thin and Over-Provisioning Support

As the Pure Storage FlashArray is fully thin-aware and uses industry-leading data reduction technologies we can allow for massive over-provisioning of the array.

Due to the architectural design of the Pure Storage FlashArray, we recommend leaving the `reserved_percentage` at the default value of 0 (zero) in the `/etc/cinder/cinder.conf` file.

Also recommended is the use of the following two parameters in the required backend stanzas:

- `max_over_subscription_ratio`
- `pure_automatic_max_oversubscription_ratio`

SSL Certification

By default SSL certificate validation is disabled in the Cinder driver and therefore log files will contain many warning messages containing the following text:

```
InsecureRequestWarning: Unverified HTTPS request is being made. Adding
certificate verification is strongly advised.
```

To remove these errors from the Cinder log files it is required to set the following option in the backend stanza for your FlashArray:

```
driver_ssl_cert_verify = true
```

If you are using a non-default path to the CA_Bundle file or directory with certificates of trusted Cas then provide this information using the following option:

```
driver_ssl_cert_path = <Certificate path>
```

Using Multiple Cinder Backends

The previous sections have assumed that there is only one Cinder backend. In reality, there may be a number of backends available to the OpenStack administrator, either of the same type or from different vendors.

If the multiple backends are exclusively using the same Pure Storage Cinder driver, then only the following modifications would need to be made to the above recommendations:

Each backend requires its own stanza with the `/etc/cinder/cinder.conf` file. The following example shows two backend arrays, each with its own stanza.

```
[Pure-1]
volume_backend_name=pure-1
volume_driver=cinder.volume.drivers.pure.PureISCSIDriver
```

```
san_ip=10.209.112.203
pure_api_token=960e185a-255f-4bdb-3b03-3b9eb1e5cf1f

[Pure-2]
volume_backend_name=pure-2
volume_driver=cinder.volume.drivers.pure.PureFCDriver
san_ip=10.209.112.201
pure_api_token=12de56fa-04d5-fde4-22ed-23ed409fab65
```

Notice that each section has a new parameter, `volume_backend_name`. These are unique to each backend and are used to tell Cinder which backends are enabled (you may have backends listed that are not enabled for use if required).

To inform Cinder which backends are actually enabled, modify the `/etc/cinder/cinder.conf` file and set the parameter `enabled_backends` with comma-separated backend names. For example:

```
enabled_backends=pure-1, pure-2
```

If you have backends using different drivers, i.e. different vendor arrays, you need to check whether certain configuration options are supported or are going to be implemented across one or all backends.

Enabling TRIM / SCSI UNMAP

OpenStack can support TRIM / SCSI UNMAP on Pure Storage hosted Cinder boot devices created with appropriately configured Glance images.

To enable this functionality, it is necessary to modify Glance images to use the virtio-scsi block storage driver and thereby provide the discard support required. This is performed by issuing the following command against each Glance image:

```
glance image-update --property hw_scsi_model=virtio-scsi --property
hw_disk_bus=scsi <glance_image_id>
```

The glance image ID can be obtained from the command `glance image-list`.

It is important to note that these settings will enable the ability to use the discard feature for the boot device and for all subsequent devices attached to the Nova instance.

Glance Image Cache for Cinder

The Glance Image Cache for Cinder is by default disabled; therefore it is recommended to enable this when using a Pure Storage array as a Cinder backend. There are significant improvements in instance creation times that can be achieved by using this functionality.

A pre-requisite for this feature is the creation of an OpenStack project/tenant and user to act as the internal owner of the image cache volumes. This is done with the following commands:

```
# openstack project create –domain default –description "Cinder Internal
Tenant Project" cinder_internal_project
# openstack user create –domain default –password-prompt
cinder_internal_project
```

```
# openstack role add –project cinder_internal_project  –user
cinder_internal_project _member_
```

Once these are created, use the created project/tenant and user IDs to add the following (example) entries to the `cinder.conf` file

```
[DEFAULT]
...
cinder_internal_tenant_project_id = PROJECT_ID
cinder_internal_tenant_user_id = USER_ID

[PURE]
...
image_volume_cache_enabled = True
image_volume_cache_max_size_gb = 200
image_volume_cache_max_count = 50
```

Note that a value of 0 (zero) for these parameters means unlimited.

More details on the benefits and use of this feature with a Pure Storage array are detailed in a whitepaper specifically dealing with this subject entitled **OpenStack Liberty: A Look at the Glance Image-Cache for Cinder** that is available from the Pure Storage website (http://www.purestorage.com/resources/).

Generic Volume Groups

A generic volume group is a number of disks that are linked together for administrative purposes, such as linking all application disks under the same umbrella. These disks may not require I/O consistency, but this feature (the equivalent of consistency groups) can be provided using extra_specs when defining the volume group. Pure Storage FlashArrays fully support I/O consistency snapshot support for generic volumes.

It should be noted that generic volume groups can only have one group type, but a group type can support multiple volume types. However, if you which to perform I/O consistent operations the group can only contain volumes on the same backend.

A generic volume group created with the appropriate extra_spec value (example below) will retain the functionality of the deprecated consistency group construct, including the ability to perform I/O consistent snapshots. The group needs to be associated with a volume type(s) that is directing the generic volume group to a specific backend that supports consistent snapshot capabilities.

```
# cinder group-type-create test_cg_group
# cinder group-type-key test_cg_group set consistent_group_snapshot_enabled="<is> True"
# cinder group-create --name cg_group <group type uuid> -
                            <volume type uuid>,<volume type uuid>
```

From the 2023.2 (Bobcat) release, the Pure Storage Cinder drivers also support Replication Enabled Consistency Groups.

Storage Quality of Service

Whilst Pure Storage arrays inherently provide volumes with very high performance in respect of IOPs and bandwidth capabilities, there are certain circumstances where an administrator may wish to apply Quality of

Service limitations to specific volumes presented from a Pure Storage array to an instance. This can be achieved using both Frontend and Backend Quality of Service settings.

Frontend Storage Quality of Service functionality can be leveraged to achieve a desired result, although this currently only works when using KVM or QEMU as your hypervisor. The options available for QoS limits are:

- `total_bytes_sec`: the total allowed bandwidth for the volume per second
- `read_bytes_sec`: sequential read limit
- `write_bytes_sec`: sequential write limit
- `total_iops_sec`: the total allowed IOPS for the volume per second
- `read_iops_sec`: random read limit
- `write_iops_sec`: random write limit
- `read_bytes_sec_max`: burst sequential read bandwidth limit
- `write_bytes_sec_max`: burst sequential write bandwidth limit
- `total_bytes_sec_max`: burst total bandwidth limitation
- `read_iops_sec_max`: burst random read limit
- `write_iops_sec_max`: burst random write limit
- `total_iops_sec_max`: burst total IOPs limit
- `size_iops_sec`: IO size limit

Backend Quality of Service utilizes the native QoS settings available on a per volume basis provided by the Pure Storage FlashArray. There are only two settings available, these being:

- `maxBWS`: the total allowed bandwidth for the volume per second, Range 100 – 100 million
- `maxIOPS`: the total number of IOPs for a volume per second in MB/s. Range 1 -524288 (512GB/s)

To apply QoS to Pure Storage volumes perform the following steps:

1. Create a Frontend Cinder Quality of Service for each QoS level you require

```
# cinder qos-create fe-limit consumer="front-end" read_iops_sec=2000
write_iops_sec=1000
+----------+------------------------------------------------------------+
| Property |                          Value                             |
+----------+------------------------------------------------------------+
| consumer |                       front-end                            |
|    id    |          bd9c4200-d216-4320-a3f0-675f92a7b7c7               |
|   name   |                          fe-limit                          |
|  specs   | {u'write_iops_sec': u'1000', u'read_iops_sec': u'2000'}    |
+----------+------------------------------------------------------------+
```

   Or create a Backend Cinder Quality of Service for each QoS level you require

```
# cinder qos-create be-limit consumer="back-end" maxBWS=2000 maxIOPS=1000
+----------+------------------------------------------------------------+
| Property |                          Value                             |
+----------+------------------------------------------------------------+
| consumer |                       back-end                             |
|    id    |          bd9c4200-d216-4320-a3f0-675f92a7b7c7               |
|   name   |                          be-limit                          |
```

```
|  specs   |       {u'maxBWS': u'2000', u'maxIOPS': u'1000'}         |
+----------+----------------------------------------------------------+
```

2. Create a new volume type for each QoS level

```
# cinder type-create fe-limit
+--------------------------------------+----------+-------------+-----------+
|                  ID                  |   Name   | Description | Is_Public |
+--------------------------------------+----------+-------------+-----------+
| 26baaa59-24ca-43ad-b3c7-94cab636f36b | fe-limit |      -      |    True   |
+--------------------------------------+----------+-------------+-----------+
# cinder type-create be-limit
+--------------------------------------+----------+-------------+-----------+
|                  ID                  |   Name   | Description | Is_Public |
+--------------------------------------+----------+-------------+-----------+
| 834dca98-11cd-3de5-6657-27583de9cd55 | be-limit |      -      |    True   |
+--------------------------------------+----------+-------------+-----------+
```

3. Associate the new volume type to the appropriate QoS level. In this case the Frontend based level

```
# cinder qos-associate bd9c4200-d216-4320-a3f0-675f92a7b7c7 26baaa59-24ca-43ad-b3c7-
94cab636f36b
# cinder create --display-name fe-limit --volume-type fe-limit 1
+---------------------------------------+--------------------------------------+
|               Property                |                Value                 |
+---------------------------------------+--------------------------------------+
|              attachments              |                  []                  |
|           availability_zone           |                 nova                 |
|               bootable                |                false                 |
|          consistencygroup_id          |                 None                 |
|              created_at               |      2015-11-30T14:48:36.000000      |
|              description              |                 None                 |
|               encrypted               |                False                 |
|                   id                  | 89f3d3d4-e498-46f3-8dc1-358923457481 |
|               metadata                |                  {}                  |
|              multiattach              |                False                 |
|                  name                 |              fe-limit                |
|         os-vol-host-attr:host         |                 None                 |
|     os-vol-mig-status-attr:migstat    |                 None                 |
|     os-vol-mig-status-attr:name_id    |                 None                 |
|      os-vol-tenant-attr:tenant_id     |   e6b9d4b0a58e4586af88cbfff6d5b02c   |
|   os-volume-replication:driver_data   |                 None                 |
| os-volume-replication:extended_status |                 None                 |
|           replication_status          |               disabled               |
|                  size                 |                  1                   |
|              snapshot_id              |                 None                 |
|              source_volid             |                 None                 |
|                status                 |               creating               |
|                user_id                |   f1528c46b89a43ea80414d7b63932bc5   |
```



```
|              volume_type              |               fe-limit               |
+---------------------------------------+--------------------------------------+
```

When this volume is now attached to a Nova instance, the hypervisor or the FlashArray will control the IOPs and bandwidth to this specific disk, depending on the type of QoS created

You can have multiple disks, each with a different QoS level attached to a single instance if required.

If the administrator needs to change the QoS level of a volume, then this can be done using the `cinder retype` command available within OpenStack.

Host Personality

The host personality determines how the Purity 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`, `hpux`, `oracle-vm-server`, `solaris`, and `vms`. If your Nova compute nodes personality is not listed as one of the valid host personalities, do not set the option. By default, the host personality is not set.

The only known use of `host_personality` parameter is to use `aix` in an IBM PowerVC deployment where AIX is the Nova operating system.

The setting of `vms` is for use with **OpenVMS** systems. This is not to be used for VMware virtual machines.

To set the host personality, modify the following option in the `cinder.conf` file in the appropriate backend stanza:

```
pure_host_personality = <personality>
```

Note that this setting is only valid when the FlashArray is running Purity v5.0 or higher and only affects newly created host objects in the array.

VLAN configuration

Using a FlashArray in a multi-workload environment may lead to situations where the iSCSI or NVMe-RoCE/TCP targets on the FlashArray are configured across multiple networks, or VLANs. To ensure that Cinder does not try to use and connect to iSCSI targets on a FlashArray that are not in the same subnet as the OpenStack storage network VLAN you must configure the parameter `pure_iscsi_cidr` or `pure_nvme_cidr` (depending on the driver being used) for the required IP ranges. To support multiple CIDR ranges the parameter `pure_iscsi_cidr_list`, or `pure_nvme_cidr_list`, can be used. This is a comma-separated list of IPv4 and/or IPv6 CIDR ranges.

The provided CIDR ranges provided should be the network range that your OpenStack storage VLAN is configured to use in Neutron.

The default setting for `pure_iscsi_cidr` or `pure_nvme_cidr` is the IPv4 CIDR range **0.0.0.0/0**.

If a list is supplied for `pure_iscsi_cidr_list` or `pure_nvme_cidr_list` this will override any setting in `pure_iscsi_cidr` or `pure_nvme_cidr` respectively.

Should these parameters not be set correctly then OpenStack will attempt to connect to inaccessible target ports leading to iSCSI or NVMe timeouts.

These parameters can also be used in the use case where multiple OpenStack clouds are connected to the same FlashArray but use different storage VLANs for multi-tenancy purposes.

Multi-Attach Support

Attaching a volume created by the Pure Storage Cinder driver is supported. This can be achieved through a volume type where a type key of `multiattach ="is <True>"` has been set.

For example you can set a volume type with multiattach capability like this:

```
# cinder type-create multiattach

# cinder type-key multiattach set multiattach="is <True>"
```

Cinder Volume Active/Active Support

The Pure Storage Cinder driver has been tested and certified to work with the Cinder volume service is in Active/Active HA mode. This also includes support for replication in Active/Active environments.

To enable Active/Active mode you must set the `cluster` option in the `DEFAULT` stanza of the Cinder configuration file and configure the Distributed Lock Manager in the `coordination` stanza. More details can be found in the OpenStack documentation.

Replication

To enable the replication functionality in the Pure Storage Cinder driver it necessary to provide a `replication_device` parameter in the Pure Storage backend stanza of the primary/source array. This parameter should be provided with three key/value pairs defining the remote/target array, these key/value pairs being `backend_name`, `san_ip`, `api_token` and `type`. The values allowed for the `type` key are `sync` or `async`.

An example of this parameter is:

```
replication_device = backend_id:pure2,san_ip:10.209.112.203,
api_token:3d8fe4ef-8d01-bb2c-8dcb-9adcf4465fdf,type:sync
```

The `replication_device` value must be a single line; the above example only being split over two lines for documentation formatting purposes only.

If the `type` is `sync`, volumes will be created in a stretched Pod. This requires the two arrays to be pre-configured with Active Cluster enabled. You can optionally specify `uniform` as `true` or `false`. This will

instruct the driver that data paths are uniform between arrays in the cluster and data connections should be made to both when attaching.

From the 2023.1 release, an additional capability of three-site replication, i.e. sync-sync-async, is supported. To enable this functionality exactly two `replication_device` parameters must be provided; one to define the sync replication target and another to provide the async replication target. These, together with setting `pure_trisync_enabled = True` will enable the trisync replication capability. More details on how these volumes can be implemented can be found later in this section.

Additional parameters that must also be defined in the backend stanza of the primary/source provide the schedule details of the replication and are as follows:

- `pure_replica_interval_default`: Interval between snapshots (in seconds), defaults to 3600.
- `pure_replica_retention_short_term_default`: How long to retain all snapshots on replication target array (in seconds), defaults to 14400
- `pure_replica_retention_long_term_per_day_default`: How many per day to retain on the replication target array, defaults to 3
- `pure_replica_retention_long_term_default`: How long to retain snapshots (the per-day ones) on the replication target array (in days), defaults to 7
- `pure_replication_pg_name`: Name of the Protection Group name for Asynchronous replication, defaults to `cinder-group`
- `pure_replication_pod_name`: Name of the ActiveCluster pod name for Synchronous replication, defaults to `cinder-pod`

The replication functionality allows for a multi-target replication configuration should this be required. To enable this, there need to be multiple `replication_device` parameters defined in the backend stanza with different key/value pairs for each target/remote array.

An example of a Pure Storage backend array configuration with two replication targets is as follows:

```
[PURE1]
volume_backend_name = pure1
volume_driver = cinder.volume.drivers.pure.PureISCSIDriver
san_ip = 10.209.112.201
pure_api_token = fad8ec88-9592-39fb-d6d2-0cde59b79dc6
replication_device = backend_id:pure2,san_ip:10.209.112.203,
api_token:3d8fe4ef-8d01-bb2c-8dcb-9adcf4465fdf,type:sync
replication_device = backend_id:pure3,san_ip:10.209.112.205,
api_token:de1405b9-c103-ab47-7967-a350b5eef551,type:async
pure_replica_interval_default = 3600
pure_replica_retention_short_term_default = 15000
pure_replica_retention_long_term_per_day_default = 5
pure_replica_retention_long_term_default = 7
```

For the current release of replication, each of these multi-target arrays will be replicated using the same schedule.

Any volume that is required to be replicated must be created with a volume type that has an extra_spec set

```
replication_enabled="<is> True"
```

You can also specify the `replication_type` key to specify `"<in> sync"` or `"<in> async"` to choose the replication for that volume. If this is not specified, the default replication will be `async`.

If trisync replication has been enabled a volume type must be created, as above, but with the replication_`type` key set to `"<in> trisync"`.

It is essential that this extra_spec is written exactly as above including the capitalization.

Whilst the target arrays for replication can be managed by OpenStack and have their own backend stanzas in the Cinder configuration file, in the current release of Cinder Replication it is assumed they are not and the `replication_device` key/value pairs must still be provided.

From the 2023.1 release it is possible to determine what level of replication capability is available for a configured backend. This is achieved by using the `get_pools` command, where an additional field called **Replication Capability** is provided. The value of `trisync` implies support for sync and async, and a value of `sync` implies a support for async. A value of `async` informs that only asynchronous replication is possible with the current configuration of the backend. This response is dynamic and if the base replication capabilities of the array are changed, this value will reflect that change.

Asynchronous Replication Protection Group Name

By default, when an asynchronous relationship is set up between two Pure Storage FlashArrays in Cinder the automatically created protection group name is `cinder-group`. This can be overridden by using the following parameter in the source arrays stanza in the Cinder configuration file:

```
pure_replication_pg_name = <PG name>
```

Synchronous Replication ActiveCluster Pod Name

By default, when a synchronous relationship is set up between two Pure Storage FlashArrays in Cinder the automatically created ActiveCluster pod name is `cinder-pod`. This can be overridden by using the following parameter in the source arrays stanza in the Cinder configuration file:

```
pure_replication_pod_name = <pod name>
```

Trisync Replication ActiveCluster Pod Protection Group Name

By default, when a tri-synchronous relationship is set up between three Pure Storage FlashArrays in Cinder the automatically created Protection Group in the synchronous replication ActiveCluster pod is `cinder-trisync`. This can be overridden by using the following parameter in the source arrays stanza in the Cinder configuration file:

```
pure_trisync_pg_name = <AC-PG name>
```

Volume Eradication

By default, when a Pure Storage volume is deleted by OpenStack, the volume is placed in the 'Destroyed Volumes' group within the Pure Storage array where it will remain for 24 hours before it is finally eradicated. Prior to final eradication, the destroyed volume can be recovered and all the data on the volume will be recovered and made available for reattachment to hosts, or to be imported back into the OpenStack management framework.

In cases where there are very large numbers of volumes being created and deleted, it is possible that the Pure Storage volume count limit will be reached, or the limit set in the volume scheduler may be reached. Should this happen, subsequent requests to create new volumes will not be honoured.

You may set the Pure Storage Cinder driver to eradicate volumes upon deletion without waiting for the 24-hour window to complete. This setting is array specific and should be added to individual backend stanzas for the arrays this feature is required.

The setting to enable immediate volume eradication is:

```
pure_eradicate_on_delete = True
```

Troubleshooting

If you are experiencing any issues in OpenStack, whether it be with a Cinder driver or another module with the project, the first and best method of root cause analysis is by investigating the OpenStack log files.

Enhanced logging is available in all modules of OpenStack by setting the `verbose` and `debug` parameters to `true` in the module configuration files.

Examples of configuration files that can be relevant to issues with Cinder drivers and their interaction with other modules are `/etc/cinder/cinder.conf`, `/etc/nova/nova.conf`, `/etc/glance/glance-api.conf` and `/etc/glance/glance-registry.conf`. You must restart these services if you modify the configuration files.

All logs will be created in a module-specific subdirectory from `/var/log`. For example, the Cinder log files will be found in `/var/log/cinder`.

Additional troubleshooting documentation can be found on the Pure Storage Support website.

References

The following references were used as part of the development of this document.

- Pure Storage Linux Recommended Settings [https://support.purestorage.com/Solutions/Operating_Systems/Linux/Linux_Recommended_Settings]
- OpenStack Liberty: A Look at the Glance image-Cache for Cinder [http://support.purestorage.com/Solutions/Cloud/OpenStack/OpenStack%C2%AE_Liberty%3A_A_Look_at_the_Glance_Image-Cache_for_Cinder]

About the Author

As Director of Open Source Integrations, Simon is helping direct and implement Open Source technologies in cloud, automation and orchestration technologies within Pure Storage. Core items include best practices, reference architectures and configuration guides.

With over 35 years of storage experience across all aspects of the discipline, from administration to architectural design, Simon has worked with all major storage vendors' technologies and organizations, large and small, across Europe and the USA as both customer and service provider. He also specializes in Data Migration methodologies assisting customers in their Pure Storage transition.

Blog: http://blog.purestorage.com/author/simon

PURESTORAGE

Pure Storage, Inc.
Twitter: @purestorage
www.purestorage.com

650 Castro Street, Suite #260
Mountain View, CA 94041

T: 650-290-6088
F: 650-625-9667

Sales: sales@purestorage.com
Support: support@purestorage.com
Media: pr@purestorage.com
General: info@purestorage.com