Redbooks
ibm.com/redbooks

Using IBM DS8000 in an OpenStack Environment

Bert Dufrasne

Axel Westphal

Cloud

Storage

IBM

Redpaper

International Technical Support Organization

Using IBM DS8000 in an OpenStack Environment

December 2017

REDP-5220-01

Note: Before using this information and the product it supports, read the information in “Notices” on page v.

Second Edition (December 2017)

This edition applies to DS8000 License Machine Code Release 8.3 with the OpenStack Pike release.

This document was created or updated on December 28, 2017.

Contents

Notices

This information was developed for products and services offered in the US. This material might be available from IBM in other languages. However, you may be required to own a copy of the product or product version in that language in order to access it.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:
IBM Director of Licensing, IBM Corporation, North Castle Drive, MD-NC119, Armonk, NY 10504-1785, US

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some jurisdictions do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM websites are provided for convenience only and do not in any manner serve as an endorsement of those websites. The materials at those websites are not part of the materials for this IBM product and use of those websites is at your own risk.

IBM may use or distribute any of the information you provide in any way it believes appropriate without incurring any obligation to you.

The performance data and client examples cited are presented for illustrative purposes only. Actual performance results may vary depending on specific configurations and operating conditions.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

Statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to actual people or business enterprises is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. The sample programs are provided "AS IS", without warranty of any kind. IBM shall not be liable for any damages arising out of your use of the sample programs.

Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corporation, registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the web at "Copyright and trademark information" at http://www.ibm.com/legal/copytrade.shtml

The following terms are trademarks or registered trademarks of International Business Machines Corporation, and might also be trademarks or registered trademarks in other countries.

DS8000®
Enterprise Storage Server®
FlashCopy®
IBM®
IBM Spectrum™
Redbooks®
Redpaper™
Redbooks (logo) ®
System Storage®
XIV®

The following terms are trademarks of other companies:

Linux is a trademark of Linus Torvalds in the United States, other countries, or both.

Windows, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

Java, and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

Other company, product, or service names may be trademarks or service marks of others.

Preface

With the availability of the IBM® Storage Driver for OpenStack, the IBM DS8000® can offer a range of capabilities that enable more effective storage automation deployments to private or public clouds. Enabling OpenStack with DS8000 allows storage to be made available whenever it is needed without the traditional associated cost of highly skilled administrators and infrastructure.

This IBM Redpaper™ publication explains how to integrate the DS8000 in an OpenStack environment, first from the DS8000 Storage Administrator perspective and then from a cloud administrator standpoint.

This paper also contains practical examples and illustrations of DS8000 functions that can be used with OpenStack, as it applies for DS8880 Release 8.3 and the OpenStack Pike release.

Authors

This paper was produced by a team of specialists from around the world working at the International Technical Support Organization (ITSO), San Jose Center.

Bert Dufrasne is an IBM Certified IT Specialist and Project Leader for IBM System Storage® disk products at the ITSO, San Jose Center. He has worked at IBM in various IT areas. He has written many IBM Redbooks® publications and has developed and taught technical workshops. Before joining the ITSO, he worked for IBM Global Services as an Application Architect. He holds a Master's degree in Electrical Engineering.

Axel Westphal is working as certified IT Specialist at the IBM EMEA Storage Competence Center (ESCC) in Kelsterbach, Germany. He joined IBM in 1996, working for Global Services as a System Engineer. His areas of expertise include setup and demonstration of IBM storage products and solutions in various environments. He wrote several storage white papers and co-authored several IBM publications about DS8000, IBM Spectrum™ Virtualize, and Spectrum Accelerate.

Thanks to the following people for their contributions to this project:

Justin Cripps, Eddie Lin, Adrian Orban, Harald Seipp.
IBM

Now you can become a published author, too!

Here's an opportunity to spotlight your skills, grow your career, and become a published author—all at the same time! Join an ITSO residency project and help write a book in your area of expertise, while honing your experience using leading-edge technologies. Your efforts will help to increase product acceptance and customer satisfaction, as you expand your network of technical contacts and relationships. Residencies run from two to six weeks in length, and you can participate either in person or as a remote resident working from your home base.

Find out more about the residency program, browse the residency index, and apply online at:

ibm.com/redbooks/residencies.html

Comments welcome

Your comments are important to us!

We want our papers to be as helpful as possible. Send us your comments about this paper or other IBM Redbooks publications in one of the following ways:

- Use the online **Contact us** review Redbooks form found at:

  ibm.com/redbooks

- Send your comments in an email to:

  redbooks@us.ibm.com

- Mail your comments to:

  IBM Corporation, International Technical Support Organization
  Dept. HYTD Mail Station P099
  2455 South Road
  Poughkeepsie, NY 12601-5400

Stay connected to IBM Redbooks

- Find us on Facebook:

  http://www.facebook.com/IBMRedbooks

- Follow us on Twitter:

  https://twitter.com/ibmredbooks

- Look for us on LinkedIn:

  http://www.linkedin.com/groups?home=&gid=2130806

- Explore new Redbooks publications, residencies, and workshops with the IBM Redbooks weekly newsletter:

  https://www.redbooks.ibm.com/Redbooks.nsf/subscribe?OpenForm

- Stay current on recent Redbooks publications with RSS Feeds:

  http://www.redbooks.ibm.com/rss.html

1

OpenStack

This chapter gives a short overview of OpenStack, its architecture, and components.

If you are already familiar with OpenStack, you can proceed to Chapter 2, “DS8000 integration in an OpenStack environment” on page 11.

This chapter includes the following sections:

- OpenStack overview
- OpenStack terminology
- OpenStack components
- OpenStack infrastructure and architecture
- OpenStack and storage

1.1 OpenStack overview

OpenStack is being developed as a *cloud computing* project. One of the ideas behind cloud computing is that IT departments should be able to fast react on requirements imposed by the business. Further, it allows the more economical usage of the data center hardware and software, and entrusts remote services with a user's data, software, and computation.

The National Institute of Standards and Technology (NIST) provides the following definition of cloud computing:

> "*Cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction*."[1]

Although not directly mentioned in this definition, one of the key enablers of cloud technology is *virtualization*. Virtualization is not something new. The idea behind it goes back to the 1960s. It can be described as a resource pooling concept. Computing resources are seen in one shared package, managed as a whole, and can be allocated on demand. But in typical virtualization solutions of today, there are still some manual tasks required. For example, creating virtual machines and configuring storage and network devices. In this respect, *automation*, a second key enabler of cloud technology, allows the scalable and elastic use of available resources. So in addition to virtualization, many other technologies are required to complete a cloud infrastructure.

According to a NIST definition, there are three different cloud service models:

- Software as a Service (SaaS) provides applications running on a cloud infrastructure.
- Platform as a Service (PaaS) provides a platform for deploying applications created using programming languages, libraries, services, and tools supported by the provider.
- Infrastructure as a Service (IaaS) provides the capability to provision processing, storage, networks, and other fundamental computing resources.

OpenStack is deployed as an IaaS solution.

The development is managed by the OpenStack Foundation, which is a non-profit organization that was started by Rackspace Hosting and NASA in 2010. Including IBM, more than 200 companies contributed to OpenStack. This enables a global collaboration of developers to develop an open source cloud-computing platform for public and private clouds under the terms of the Apache license.

The main modules of OpenStack are developed with the programming language Python. The following link provides more information about the community developing OpenStack:

http://www.openstack.org/community

[1] See Special Publication 800-145 The NIST Definition of Cloud Computing
http://csrc.nist.gov/publications/nistpubs/800-145/SP800-145.pdf

OpenStack consists of many interrelated services that control resources for compute, storage, and networking. The modular architecture of OpenStack (see Figure 1-1) makes cloud deployment and operations flexible and easy.

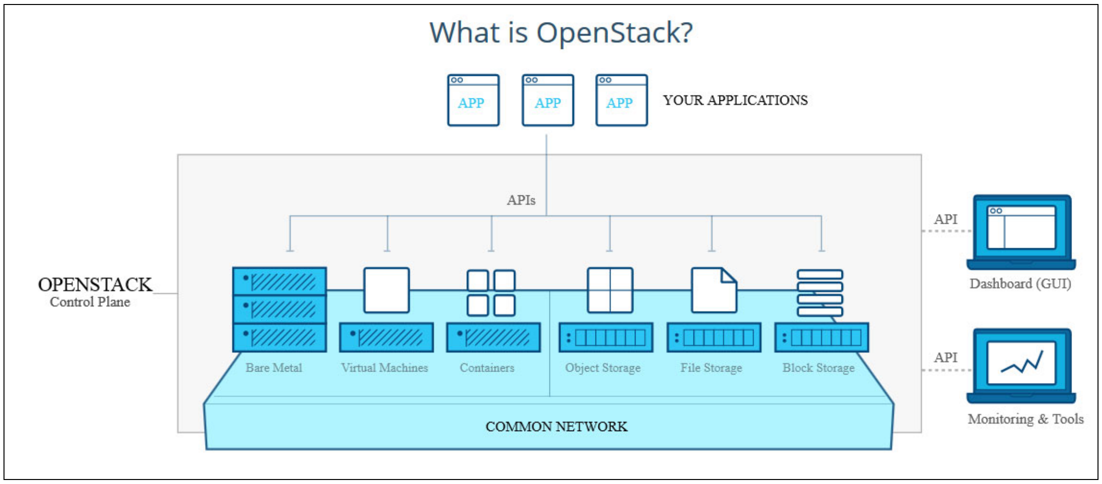

Figure 1-1 OpenStack cloud computing (source: http://www.openstack.org/software)

OpenStack is a cloud operating system that controls compute, storage, and networking resources throughout a data center, all managed through a dashboard that gives administrators control while empowering their users to provision resources through a web interface.

Figure 1-2 shows some of the services that OpenStack can automate.

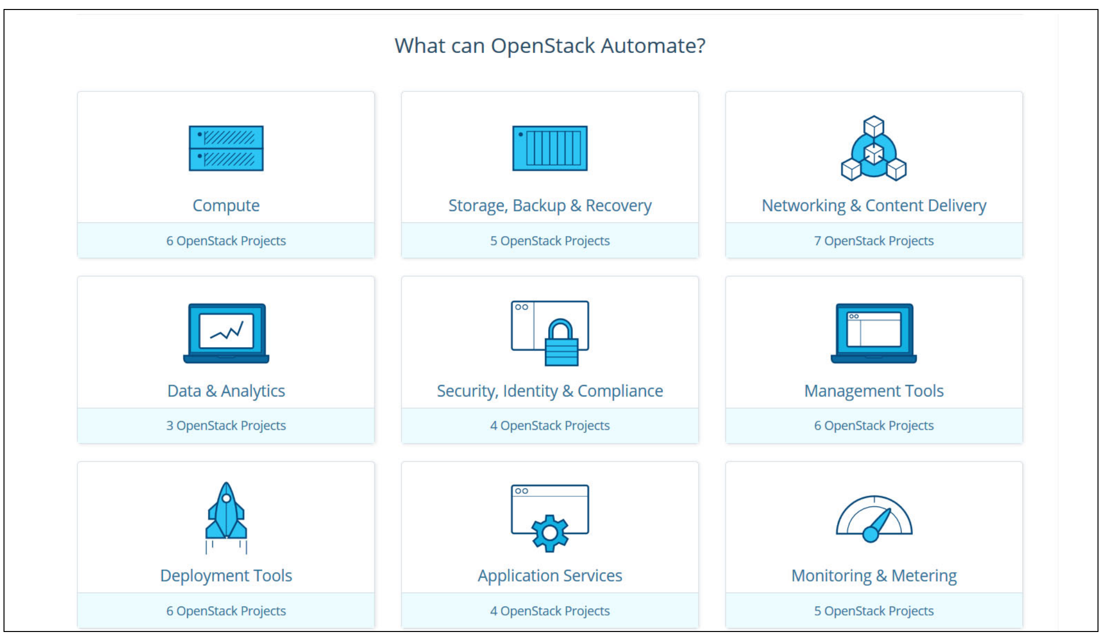

Figure 1-2 What OpenStack can automate

The different modular core services have different functions and responsibilities. Each service has a different code name. For the IBM DS8000, this document covers the following services and their code names include the following core components:

- Compute (Nova)
- Networking (Neutron)

- Object Storage (Swift)
- Block Storage (Cinder)
- Dashboard (Horizon)

As an example, the Horizon dashboard, an OpenStack web interface, provides the capability to create and manage the different resources in a cloud environment. See Figure 1-3.

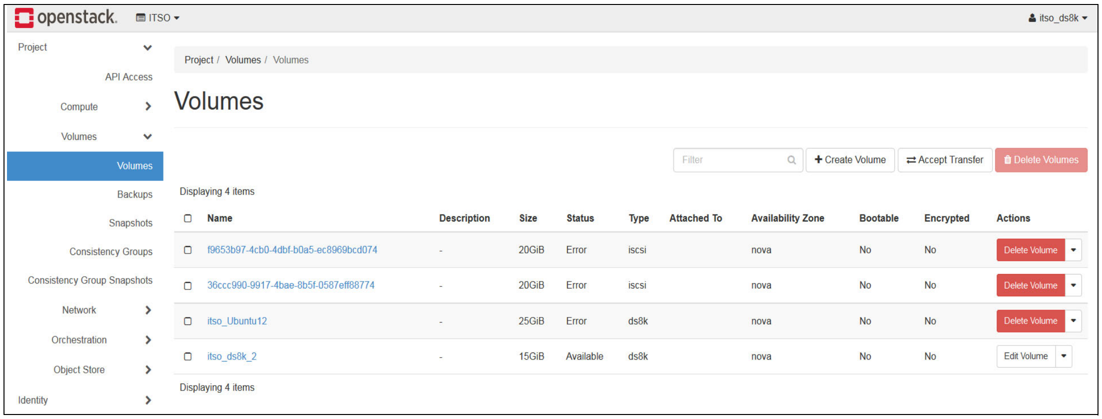

Figure 1-3 OpenStack Horizon dashboard - Volumes

The mentioned services are interconnected through application programming interfaces (APIs). This configuration enables users to deploy virtual machines or attach storage to them over a single dashboard, which can use the APIs of different OpenStack services. For instance, Horizon hands over the request to deploy virtual machines to the compute service Nova, which is responsible for managing the different virtual machines. Section 1.3, “OpenStack components” on page 5 provides a further explanation about the different OpenStack components, the OpenStack functions, and interfaces.

The OpenStack release cycle is mostly every six months. For information about former, current, and future releases, see the following website:

https://wiki.openstack.org/wiki/Releases

OpenStack releases have specific code names that follow the alphabet sequence. The release that was used for this paper has the code name *Pike* (August 2017).

1.2 OpenStack terminology

OpenStack uses specific terms to describe its features and functions. The following list provides those important terms, which are also used in this paper:

Tenant/Project OpenStack supports the concept of multi-tenancy for an organizational structure. By this approach, resources are managed within a *tenant* or *project* and can share services. This architecture offers the ability to correlate usage tracking, auditing, authorization, and so on.

Service OpenStack and Linux services have some similarities. A Linux service (also called a *daemon*) is a program that runs in the background of the operating system and listens to a specific port to respond to service requests. An OpenStack service consists of Linux services that are working together. For example, nova-compute and nova-scheduler are two of the Linux services that implement the OpenStack Compute service.

> **Service:** This document uses the term *service* to refer both to Linux and OpenStack services.

Hypervisor A hypervisor is a type of software that allows you to run one or more virtual machines on a single physical machine. OpenStack Compute requires a hypervisor and controls the hypervisors through an API server. Because OpenStack uses an abstraction layer for compute drivers, it supports different hypervisors with different features. Most OpenStack installations are based on the kernel-based virtual machine (KVM) and Xen hypervisors. For a detailed list of features and support across the hypervisors, see this website:

http://wiki.openstack.org/HypervisorSupportMatrix

Image Images or disk images are single files that contain a virtual disk with a bootable operating system installed on it. Users can create virtual machines out of this virtual machine image.

Instance Instances or compute instance are the actual virtual machines running on physical compute nodes. It is possible to run more than one instance from the same image. In this case, OpenStack uses a copy of the base image. It is possible to create snapshots of a running instance, which create a new image based on the running version. Later it can be deployed as a new instance.

Node A node is the physical component that OpenStack uses. As an example, the compute nodes, which contain the resources of the cloud environment like processing, memory, network, and storage to create and run virtual machines.

Code names Every OpenStack service has a code name. These code names are reflected in the names of command-line utility programs and configuration files. For example, the OpenStack Compute service Nova has a configuration file that is called `nova.conf`. Section 1.3, “OpenStack components” on page 5 describes which OpenStack service has which specific code name.

Further, we use a different terminology of user roles in this paper. On one hand, we use the term *storage admin* for administrators who have access to the different management tools of the DS8000, like the Storage Management GUI or DS CLI. However, we use the term *user* or *cloud admin* for those who can run OpenStack-related tasks by using the Horizon dashboard or the command-line interface (CLI) of the different OpenStack services.

1.3 OpenStack components

As mentioned, OpenStack includes several components like Compute, Networking, and Storage. Further, it contains several shared services for authentication functions or providing images for virtual machines. In addition, other specific higher-level services, like scalable database-functional lists, are available. The following list gives a short overview about the core services of OpenStack in the following format: **Name of Service** (code name):

- **OpenStack Compute** (Nova) is responsible for managing the pool of compute instances in an OpenStack environment. It is the major part of an IaaS system. APIs allow Nova to interact with other OpenStack services like the Horizon dashboard or the Glance Image Service. Nova is the most complex and distributed component of OpenStack. Many processes are involved to handle user requests and launch compute instances.

The following list is a summary of Nova processes:

- The nova-compute process is a daemon that launches and terminates VM instances by using the hypervisor's APIs like XenAPI for XenServer/XCP, libvirt for KVM, and QEMU or VMwareAPI for VMware.
- The nova-api process handles user requests through API calls.
- The nova-database process stores several items and states of resources in the cloud environment. For example, an item can be an instance, available network, or a project.
- The nova-schedule process determines where instances run based on requests that are in the message queue. This configuration is implemented with RabbitMQ or Qpid.
- The nova-console process enables the outside access to an instance.
- The nova-consoleauth process offers a proxy to access the compute instances to users. It shares service with nova-console.

Some other special services are part of OpenStack Compute, such as nova-cert, which is a service for X509 certificates, and nova-objectstore for registering S3 (Amazon Simple Storage Service) interfaces.

- **OpenStack Dashboard** (Horizon) provides a modular web-based user interface for all the OpenStack services. Through this self-service portal, the user can perform most operations and configurations on their OpenStack environment.
- **OpenStack Networking** (Neutron) provides "network connectivity as a service" between interface devices that are managed by other OpenStack services (most likely, Nova). The service works by allowing users to create their own networks and then attach interfaces to them.
- **Storage**
  - **OpenStack Block Storage** (Cinder) provides persistent block storage to instances. It came out of code that was originally in Nova (the nova-volume service). Cinder replaces the nova-volume starting with the Folsom release. Cinder virtualizes pools of block storage devices and provides users with a self-service API to request and consume storage resources.
  - **OpenStack Object Storage** (Swift) provides object storage and allows users to store or retrieve files in a blob mode. It is built for scale-out networks and it is ideal for storing unstructured data that can group without bounds.

  Section 1.5, "OpenStack and storage" on page 8 provides more information about the different storage concepts in OpenStack.
- **Shared services**
  - **OpenStack Image Service** (Glance) provides a catalog and repository for disk operating system images for instances that are mostly used by OpenStack Compute.
  - **OpenStack Identity** (Keystone) provides authentication and authorization for all the OpenStack services. It also provides a service catalog of services within a particular OpenStack cloud.
  - **OpenStack Telemetry** (Ceilometer) provides a mechanism to collect and configure the necessary data for measuring the utilization of the physical and virtual resources in the OpenStack environment. It can calculate CPU and network costs or custom usage data by additional plug-ins.

- **Higher-level services**
  - **OpenStack Orchestration** (Heat) provides a template for creating most OpenStack resource types (instances, floating IP addresses, volumes, security, groups, and users). This service enables cloud deployments to integrate with the Orchestration module directly or through custom plug-ins.
  - **OpenStack Database Service** (Trove) provides a scalable cloud provisioning function for relational and non-relational database engines. This function enables users to use the database features without dealing with the administrative tasks. It also provides resource isolation and automates tasks, such as deployment, configuration, patching, backups, restores, and monitoring.
  - **OpenStack Data Processing Service** (Sahara) provides functionality to automate the provision and scale of Hadoop clusters in OpenStack.

In addition to these core services, there are a number of other services that are included in the OpenStack core. Information about these services and enhancements to OpenStack is available at the following web page:

https://docs.openstack.org/pike/

1.4 OpenStack infrastructure and architecture

Due to the modular architecture of OpenStack and its services, there are several ways to design an OpenStack infrastructure. This modularity enables OpenStack to meet different needs, several operational scenarios, and distribute the different services across several nodes. Figure 1-4 shows an example architecture that contains three nodes.

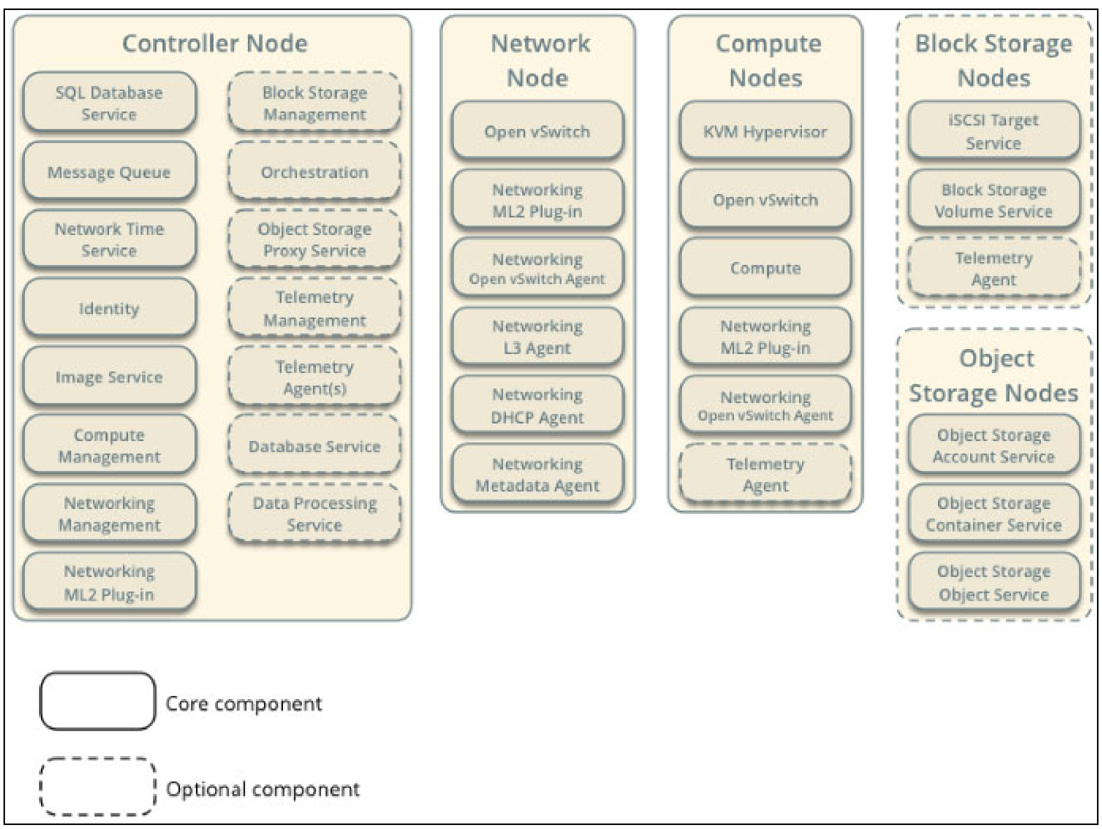

Figure 1-4 OpenStack three-node architecture with neutron (source: docs.openstack.org)

The following are the three node types:

- The *controller node* holds services like the Horizon dashboard, the Keystone identity service, and the Glance image service. Several additional services for compute and networking management are included.
- The *network node* is responsible for agents that manage network provisioning and switching.
- The *compute node* consists of the hypervisor to run virtual machines or instances on it.

In this configuration example, nodes for Swift block storage services and Cinder object storage services can optionally be added to complete the OpenStack infrastructure.

According to the OpenStack documentation, the following minimum requirements must be met. See the OpenStack.org website for further details.

- CPU recommendations:
  - Compute hosts with multi-core processors
  - Infrastructure hosts with multi-core processors
- Disk recommendations:
  - Deployment Hosts 10 GB
  - Compute Hosts 1 TB
  - Storage Hosts 1 TB
  - Infrastructure Hosts 100 GB
  - Logging Hosts 50 GB
- Network recommendations:
  - Gigabit or 10-Gigabit Ethernet

In typical production environments, it is a good practice to have more than one compute node. Multiple nodes allow you to distribute the compute resources across several systems to deliver a better performance of the entire cloud environment.

The following link provides more information about the installation and architecture of OpenStack and the different ways that you can set up an OpenStack infrastructure:

https://docs.openstack.org/pike/

1.5 OpenStack and storage

OpenStack stores the metadata of the cloud environment, usually in a database, mostly MySQL. This is the configuration data from the different services like Keystone (Identity Service), Neutron (Networking), Ceilometer (Monitoring), or Heat (Orchestration). The user data of the virtual machines is stored in another way. Therefore, OpenStack uses different kinds of storage in its stack. This section briefly describes the differences between the storage types and how they are implemented in OpenStack.

In general, cloud administrators need to take care about the following two types of storage:

- *Ephemeral storage* means that the volume will not save the changes made to it and the data will disappear when the instance is terminated. By default, if the OpenStack compute service (nova) is deployed only, the storage will be volatile.
- *Persistent storage* means that changes to the data will be saved to the volume. In OpenStack, this service is separated from the single instances and will be available, regardless of the state of a running instance. OpenStack supports two types of persistent storage: Object storage and block storage (or volume storage).

  These two different kinds of persistent storage capabilities, object and block storage, are managed by the following services:

  – The *OpenStack Object Store* service is also known as *Swift*. It is a distributed storage service for storing objects and metadata, like images, videos, or different files. The Object Storage API allows you to read, modify, or create those objects and object containers. Object containers provide storage for objects comparable to a Windows folder or Linux file directory. The following link provides more information about Swift:

    https://docs.openstack.org/swift/latest/

  – The *OpenStack Block Storage* service is also known as *Cinder*. The service allows the integration of traditional block storage devices into OpenStack Compute. The Cinder driver manages the creation of, and attaching, and detaching of volumes, to the compute resources. It is responsible for control path only. The I/O to the devices runs directly from the compute resources and is mounted over a controlled protocol (for example, iSCSI).

    Users can request or consume storage resources using an API. The Horizon dashboard is also interconnected over this service, and offers different operations against the storage devices. This feature enables the users to perform configuration tasks without requiring deep skills about the backend storage system. The following link provides more information about Cinder:

    https://docs.openstack.org/cinder/latest/

  Through an API, Swift and Cinder provided block and object storage services. The following website provides more information about storage in OpenStack and especially how block storage and object can be integrated:

  https://developer.openstack.org/api-guide/quick-start/index.html

2

DS8000 integration in an OpenStack environment

This chapter describes how to integrate the IBM DS8000 in an OpenStack environment. It describes the tasks to be performed by the DS8000 storage administrator and those subsequently involving the cloud administrator.

This chapter includes the following sections:

- IBM Storage Driver for OpenStack
- IBM Storage Driver for OpenStack capabilities
- IBM Storage Driver configuration

2.1 IBM Storage Driver for OpenStack

This section focuses on the Cinder block storage and the integration of traditional block storage resources, in our example DS8000 volumes.

Integration of storage systems requires an OpenStack Block Storage driver on the OpenStack Cinder nodes. For DS8000, OpenStack includes the IBM Storage Driver for OpenStack, which is available from the Open Source Community. Pike (2.1.0) is the latest release of OpenStack and includes additional functions for Snapshot and Replication capabilities. With Ocata or Pike, you no longer need to acquire a proprietary IBM driver supporting OpenStack Block Storage and Cinder technologies.

Figure 2-1 illustrates the control path of one Cinder node. It is secured by SSL and uses XSF, a gateway into the DS8000 IBM Enterprise Storage Server® Network Interface (ESSNI) service. ESSNI is the logical server that runs on the DS8000 Management Console and interacts with the two central processor complexes inside the DS8000. Because it interfaces through the ESSNI, the Cinder node can, if necessary, be connected to multiple DS8000s.

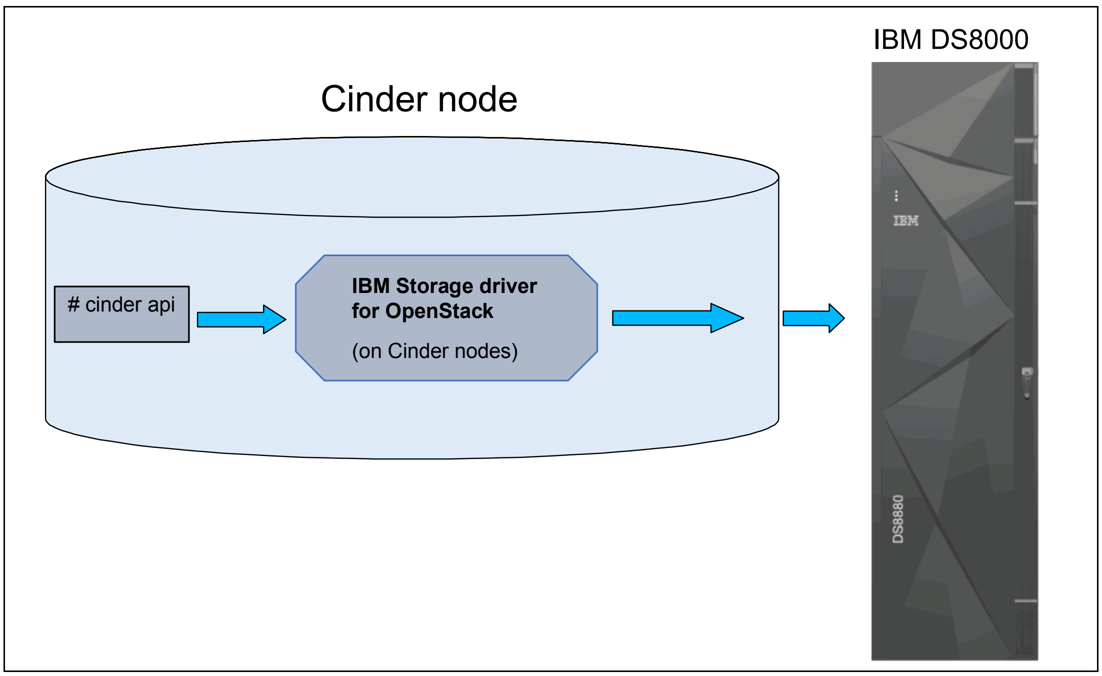

Figure 2-1 IBM Storage Driver for OpenStack flow with DS8000

The following software is also required on the Cinder node:

- Java Runtime Environment
- Py4J (Python-to-Java access library)

The IBM Storage Driver for OpenStack is also used for connecting to the IBM XIV® Storage System. Therefore, IBM XIV Storage and IBM DS8000 can be used together on the same IBM Storage Driver for OpenStack installation.

Figure 2-2 illustrates how the Horizon (Dashboard) and Nova (Compute) interact with Cinder over the Ethernet control-path. It also illustrates the data-path from Nova to the DS8000 storage system.

In general, the IBM Storage Driver for OpenStack supports Fibre Channel and iSCSI as the data-path between the compute resources and the IBM storage systems. With DS8000, the data path is supported over Fibre Channel only.

IBM has contributed to the development of Cinder and enabled Cinder and Horizon automation to support the DS8000 in the Horizon dashboard, allowing several DS8000 actions, such as volume and host creation, to be performed directly from Horizon. OpenStack cloud administrators can thus configure and provision storage in the DS8000 by themselves. The requests are transparently handled by the IBM Storage Driver for OpenStack.

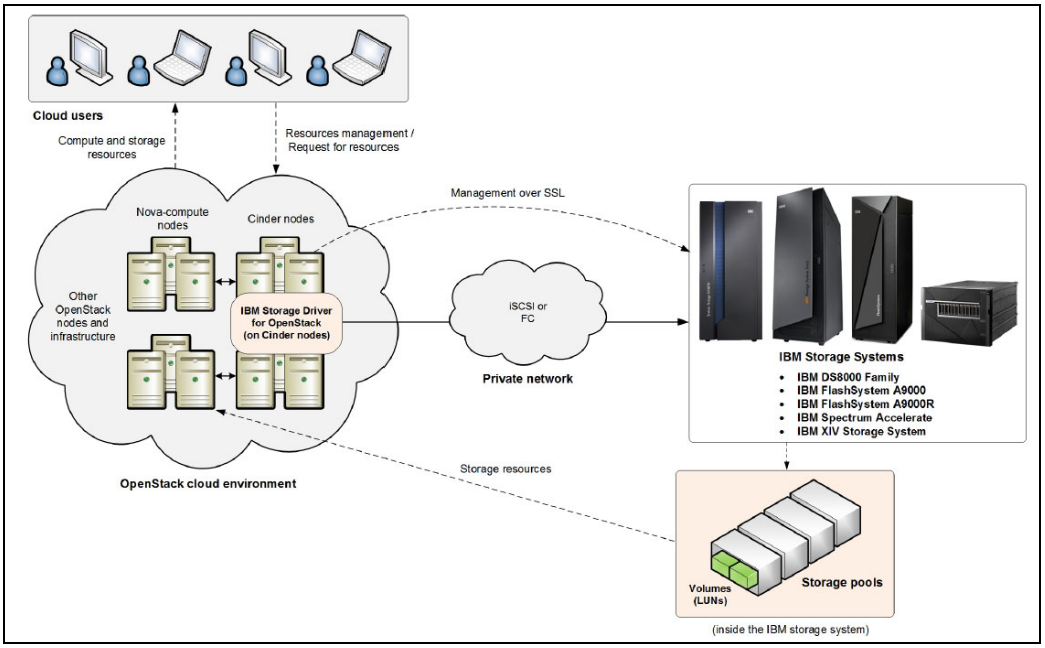

Figure 2-2 OpenStack and DS8000 infrastructure (IBM Storage Driver for OpenStack User Guide)

The following list shows the Cinder functions that were available at the time of writing this paper, and that can be performed against block storage devices:

- Create Volume
- Delete Volume
- Attach Volume
- Detach Volume
- Extend Volume
- Create Snapshot
- Delete Snapshot
- List Snapshots
- Create Volume from Snapshot
- Create Volume from Image
- Create Volume from Volume (Clone)
- Create Image from Volume
- Volume Migration (host assisted)
- QoS (quality of service)
- Volume Replication

- Consistency Group
- Enable, disable, and failover replication
- Replication consistency group support

However, not all functions that Cinder provides are necessarily supported by the different backend storage systems. Which functions are actually supported depends on the capabilities of the OpenStack Block Storage driver required by the attached storage system. The following link shows a matrix of storage systems and their corresponding drivers with the current Cinder functions and indicates whether a particular feature is supported, and the first release of OpenStack that it is supported in:

https://wiki.openstack.org/wiki/CinderSupportMatrix

Some Cinder functions, like volume replication, require connectivity to more than one storage system. The IBM Storage Driver for OpenStack enables Cinder to connect to two different DS8000 systems. Figure 2-3 shows two Cinder driver back-ends at one Cinder node. One is responsible for the primary DS8000, and the other for the secondary site. This setup allows Cinder volume operations against logical unit numbers (LUNs) on two DS8000 systems, enabling tasks like replication.

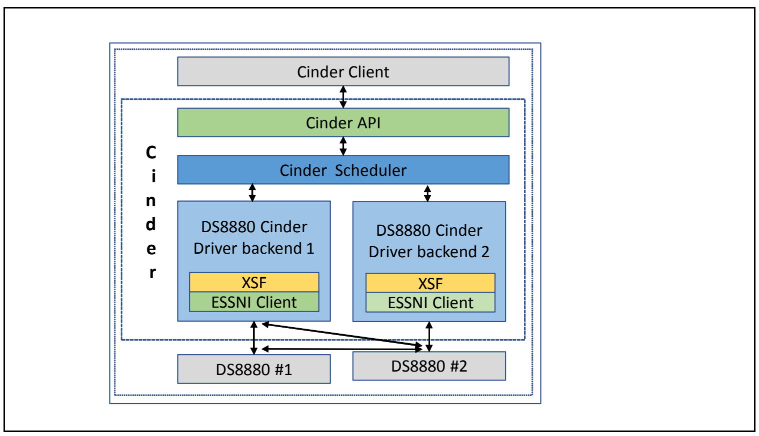

Figure 2-3 IBM Storage Driver for OpenStack with 2 connected DS8000

The release of the IBM OpenStack cinder driver that is used for this paper is 2.1.0. It is no longer required that you download the IBM OpenStack cinder driver for DS8000 from IBM Fix Central. The IBM OpenStack cinder driver is included into OpenStack since the Ocata release.

Important: This paper focuses on the OpenStack Pike release and IBM Storage Driver for OpenStack 2.1.0.

2.2 IBM Storage Driver for OpenStack capabilities

The IBM Storage Driver for OpenStack makes the DS8000 storage system available to cloud administrators and users. In addition, the OpenStack cloud environment itself can benefit from different internal DS8000 functions. With the release of Pike (2.1.0), volume snapshots and mirroring functions are now implemented in Cinder, which manages the related DS8000 operations. Such capabilities bring enhanced data resiliency levels to the OpenStack environment.

The following functions are supported in Cinder with the IBM Storage Driver for OpenStack and DS8000 at the time of writing this paper:

- Create Volume to create a DS8000 volume.
- Delete Volume to delete a DS8000 volume.
- Attach Volume to attach a DS8000 volume to an instance (host), which includes the management of the internal DS8000 LUN mapping.
- Detach Volume to detach a DS8000 volume from an instance (host).
- Create Snapshot to create a snapshot of a DS8000 volume. Creating snapshots uses the IBM FlashCopy® functionality.
- Delete Snapshot to delete a snapshot of a DS8000 volume.
- List Snapshots to list snapshots of DS8000 volumes.
- Create Volume from Snapshot to create a DS8000 volume from an existing snapshot by using the IBM FlashCopy functionality.
- Create Volume from Image to create a DS8000 volume out of an image. DS8000 creates a volume and the OpenStack Image Service (Glance) provides the data of the image.
- Create Image from Volume to create an image out of a DS8000 volume. This action is also performed by the OpenStack Image Service (Glance) service.
- Create Volume from Volume (Clone) to create a DS8000 volume from an existing volume. This process is also supported by the DS8000 internal IBM FlashCopy function when the source volume already is on the same DS8000.
- Volume Replication to replicate DS8000 volumes. In OpenStack, replication can be either host-based replication or it can be performed by the backend storage. IBM DS8000, as storage backend for OpenStack, supports the synchronous replication method, IBM Metro Mirror.

  For more information about volume replication, see the following OpenStack specifications:

  `https://blueprints.launchpad.net/cinder/+spec/volume-mirroring`

- Volume Retype to change the volume type of an existing DS8000 volume. Retype of a volume from non-replicated to replicated creates the IBM Metro Mirror relationship. The opposite process drops this relationship.
- Consistency Groups for Snapshots to create a consistent snapshot if data is spread over multiple DS8000 volumes. Therefore, internal IBM FlashCopy operations are also used.

See Chapter 3, “Using IBM DS8000 in an OpenStack environment” on page 25 for more information and practical illustrations of these capabilities.

For details about IBM DS8000 Copy Services functions and features, such as FlashCopy, Metro Mirror, and Consistency Groups, see *DS8000 Copy Services*, SG24-8367.

2.3 IBM Storage Driver configuration

This section describes how to configure the IBM Storage Driver for OpenStack for use with the DS8000.

2.3.1 Environment

For our testing purposes, we used a simple OpenStack environment with just one physical node, rather than multiple nodes, which would be more typical for a real environment as shown in Figure 1-4 on page 7.

Figure 2-4 depicts our test environment with only one node running the compute, networking, and all other required services. The node is connected over Fibre Channel to two DS8000s: DS8000#1 and DS8000#2.

Cinder and IBM Storage Driver for OpenStack are also installed on the same node, which has Ethernet connectivity to both DS8000 Management Consoles for the required ESSNI communication.

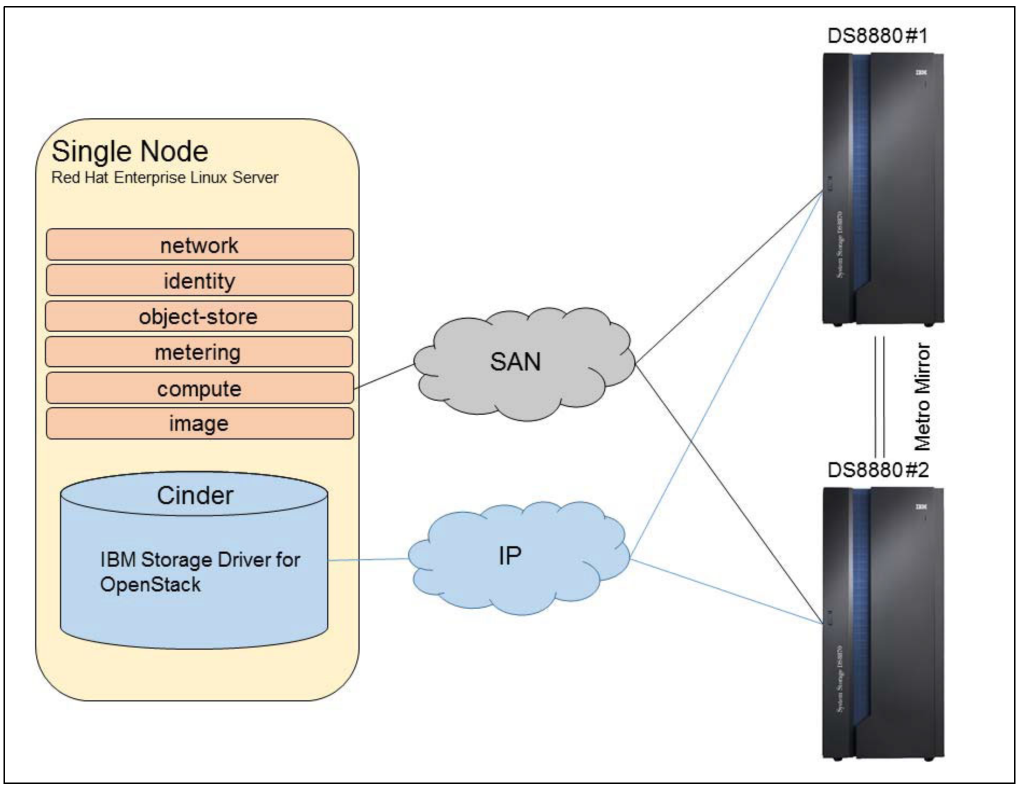

Figure 2-4 Test environment setup

The single node runs Red Hat Enterprise Linux Server release 7.4 and OpenStack Pike release. The version of the IBM Storage Driver for OpenStack is 2.1.0.

OpenStack support was introduced with DS8000 R7.3. However, many specific functions like volume replication or volume retype are only supported in the later releases in combination with the matching IBM Storage Driver for OpenStack. The two DS8000s in our environment have release 8.3 installed and support volume replication and volume retype, for example.

The following website provides more information about the DS8000 code bundle and their functions:

https://www.ibm.com/support/docview.wss?uid=ssg1S1005392

Important: Illustrations of commands in this paper are based on the OpenStack Pike release and DS8000 release 8.3 with IBM Storage Driver for OpenStack v2.1.0.

2.3.2 Preparing the DS8000 for OpenStack

To effectively use the DS8000 with OpenStack, a first requirement is to have a predefined storage pool on the DS8000 storage system before installing the driver. If you have two different back-end storage systems connected to Cinder, for instance to use replication, additional steps are required, such as preparing the Metro Mirror connectivity and creating the Metro Mirror paths. Also, when using functions like FlashCopy or Metro Mirror, the particular DS8000 license feature needs to be activated on the DS8000.

To perform those tasks on the DS8000s in our test environment, we used the DS8000 command-line interface (DS CLI).

For more details about DS CLI commands and their syntax, see the IBM DS8000 Command-Line Interface User's Guide, available at the following link:

http://www.ibm.com/support/docview.wss?uid=ssg1S7002620

You must have Storage Administrator authority on the DS8000 systems to perform the steps listed here:

Note: You can skip steps 6-7 if you will not use replication.

1. Write down the Storage Unit identifier.

   Use the **lssi** DS CLI command as marked in bold in Example 2-1. If you use replication, write down the identifiers for both DS8000 storage systems.

Example 2-1 DS CLI: Verify Storage Unit identifier

```
DS8000 #1
dscli> lssi
Name             ID               Storage Unit     Model WWNN             State  ESSNet
========================================================================================
IBM.2107-75LR811 IBM.2107-75LR811 IBM.2107-75LR810 961   5005076308FFC54C Online Enabled

DS8000 #2
dscli> lssi
Name             ID               Storage Unit     Model WWNN             State  ESSNet
========================================================================================
IBM.2107-75LT671 IBM.2107-75LT671 IBM.2107-75LT670 961   5005076308FFC56F Online Enabled
```

2. Select (or create) a storage pool OpenStack use.

   Before installation of the IBM Driver for OpenStack, a DS8000 storage pool must be created or reserved for OpenStack use. OpenStack uses this pool to create volumes, snapshots, and so on.

Example 2-2 shows an output of the **lsextpool** command to list all configured storage pools. In our case, we used already-defined storage pools (DS8000 #1 P0 and DS8000 #1 P6). You need to write down the ID of the pool because it is required during the IBM Driver for OpenStack installation.

Example 2-2 DS CLI: Verify ID of the storage pool

```
DS8000 #2
dscli> lsextpool
Name           ID  stgtype rankgrp status   availstor (2^30B) %allocated available reserved numvols
===================================================================================================
ckd_et         P0  ckd           0    below             12072         52     13702        0     710
ckd_et         P1  ckd           1    below              2321         84      2634        0     704
ckd_ssd        P2  ckd           0 exceeded               288         86       327        0      76
ckd_ssd        P3  ckd           1 exceeded               193         90       219        0      80
ckd_r10        P4  ckd           0    below              3146         31      3571        0      62
ckd_r10        P5  ckd           1    below              3099         32      3517        0      64
fb_et        > P6  fb            0    below             16898         48     16898        0     198
fb_et          P7  fb            1    below             14892         51     14892        0     193
fb_flash       P8  fb            0    below              1082         49      1082        0       2
fb_ssd         P9  fb            1    below              5868         25      5868        0       2
fb_r10         P10 fb            0    below              1188         57      1188        0      32
fb_r10         P11 fb            1    below              1287         55      1287        0      32
fb_r6          P12 fb            0    below              9614         30      9614        0      65
fb_r6          P13 fb            1    below             10742         23     10742        0      65
```

For information about how to create a new storage pool, or other information and illustrated examples of the DS88870 logical configuration tasks, see *IBM DS8880 Architecture and Implementation (Release 8.3)*, SG24-8323.

The IBM Storage Driver for OpenStack also enables the creation of thin provisioned volumes and therefore uses Extent Space Efficient (ESE) DS8000 volumes. For more information about thin provisioning and ESE volumes, see *DS8000 Thin Provisioning*, REDP-4554.

3. Configure Fibre Channel connectivity.

   As described in 2.1, "IBM Storage Driver for OpenStack" on page 12, the data path between the DS8000 and to the Nova compute node is supported through Fibre Channel only. Because the IBM Storage Driver for OpenStack will not configure the Fibre Channel attachment, it is necessary to perform that configuration before driver installation.

   Check that OpenStack compute nodes are physically connected and logically attached to the IBM DS8000 storage system.

   First, locate the worldwide port names (WWPNs) of the Nova computing node. As shown in Example 2-3, we used the systool utility for determining the WWPNs of our Red Hat server.

   Example 2-3 RHEL systool

```
[root@server ~]# systool -c fc_host -A "port_name"
Class = "fc_host"
  Class Device = "host3"
    port_name           = "0x10000000c99478a6"
    Device = "host3"
  Class Device = "host4"
    port_name           = "0x10000000c99478a7"
    Device = "host4"
```

Next, perform the logical host connection on the DS8000. Example 2-4 shows the DS CLI configuration tasks performed for our test environment. We created a volume group and a host attachment for one host adapter port to one DS8000 host adapter port, respectively.

Example 2-4 DS CLI creating volume group and host connection

```
DS8000 #1
dscli> mkvolgrp -dev IBM.2107-75LR811 nova_vg

CMUC00030I mkvolgrp: Volume group V4 successfully created.

dscli> mkhostconnect -dev IBM.2107-75LR811 -wwname 10000000c99478a6  -hosttype
"LinuxRHEL" -volgrp V4 nova

CMUC00012I mkhostconnect: Host connection 0019 successfully created.

DS8000 #2
dscli> mkvolgrp -dev IBM.2107-75LT671 nova_vg

CMUC00030I mkvolgrp: Volume group V6 successfully created.

dscli> mkhostconnect -dev IBM.2107-75LT671 -wwname 10000000c99478a7 -hosttype
"LinuxRHEL" -volgrp V6 nova

CMUC00012I mkhostconnect: Host connection 001B successfully created.
```

Of course, for a production environment, you should define a redundant host setup. For more information about the host attachment for Linux and DS8000, see this link:

http://www.ibm.com/support/knowledgecenter/HW213_7.4.0/com.ibm.storage.ssic.help.doc/f2c_attchngintlrunlnx_1tm120.html

4. Check the network configuration and write down the DS8000 user credentials.

   To verify the control-path, check that Cinder is able to communicate through Ethernet with the DS8000 Management Console. Because the Storage Driver for OpenStack must connect through ESSNI, you must provide credentials for DS8000 access to OpenStack or the cloud administrator.

   > **Note:** The DS8000 ESSNI implements the role-based user management. The credentials that you provide for OpenStack access must have a role with permissions to perform the required functions on the DS8000, such as create logical volumes or run copy services-related tasks.

   For more information about DS8000 user management and the specific user roles, see *IBM DS8880 Architecture and Implementation (Release 8.3)*, SG24-8323.

5. Check for required license keys.

   To take advantage of some IBM DS8000 functions in an OpenStack environment, DS8000 license keys for those functions must be in place. For example, the Cinder snapshot function uses the DS8000 FlashCopy operations in the background, and thus requires a FlashCopy license.

Use the **lskey** command as shown in Example 2-5 for verification.

Example 2-5 DS CLI Verify the key for Metro Mirror

```
dscli> lskey IBM.2107-75LT671

Activation Key                              Authorization Level (TB) Scope
==========================================================================
Easy Tier Server                            on                       All
Encryption Authorization                    on                       All
Global mirror (GM)                          120.9                    All
High Performance FICON for System z (zHPF)  on                       CKD
IBM FlashCopy SE                            120.9                    All
IBM System Storage DS8000 Thin Provisioning on                       All
IBM System Storage Easy Tier                on                       All
Metro mirror (MM)                           120.9                    All
Metro/Global mirror (MGM)                   120.9                    All
Operating environment (OEL)                 2779.6                   All
Point in time copy (PTC)                    120.9                    All
```

For more information about DS8000 and licensing, see *IBM DS8880 Architecture and Implementation (Release 8.3)*, SG24-8323.

6. Verify that Metro Mirror (MM) is enabled.

   This step is only required if you plan to use Cinder replication functions.

   Metro Mirror must be enabled on both DS8000 storage systems. Example 2-5 shows the entry for Metro Mirror, which means this DS8000 allows for synchronous mirroring to another DS8000 storage system.

7. Verify that a reliable communication link is established between the primary and secondary DS8000.

   Again, this step is only required if you use Cinder replication functions.

   The link is an effective communication to allow the IBM Storage Driver for OpenStack to establish and manage the Metro Mirror operations. It includes the physical connection and the logical PPRCs path between the two storage systems.

   The creation of the logical PPRC paths requires a background knowledge of the DS8000 logical architecture, with at least an understanding of the DS8000 volume addressing. The internal construct for addressing volumes inside a DS8000 is called logical subsystem (LSS). A maximum of 255 LSSs can exist in the DS8000. They each have an identifier from 00 - FE. The volumes also have an identifier that ranges from 00 - FF. The combination of those numbers indicates the LUN, as illustrated below:

   `LUN 1000 = LSS 10 and volume id 00`

   For more information about the DS8000 logical architecture, see *IBM DS8870 Architecture and Implementation (Release 7.5)*, SG24-8085.

   By default, the IBM Storage Driver for OpenStack uses LSS10 and LSS11 depending on the storage pool. An even storage pool uses LSS10 and an odd storage pool uses LSS11. If those LSSs are already fully used, the IBM Storage Driver for OpenStack increases the LSS number by one and uses LSS12 for even pools, LSS 13 for odd pools, and so on.

> **Important:** By default, the IBM Storage Driver for OpenStack uses LSS10 for even storage pools and LSS11 for odd storage pools.

Therefore, before creating the PPRC paths, check whether the LSS is still available because PPRC paths are created at an LSS level. You must also create at least one volume within the LSS. Note that, when there is no LSS defined yet, creating a volume also defines the LSS. For instance, create LUN 1000 to create LSS 10.

Example 2-6 shows the creation of LUN 1000 with the **mkfbvol** DS CLI command on DS8000 #2 where LSS10 was not previously defined. Use the **lslss** command to list each LSS defined.

Example 2-6 Create a volume and list LSS

```
DS8000 #2
dscli> mkfbvol -extpool P6 -cap 10 -type ds -name test -sam standard 1000

CMUC00025I mkfbvol: FB volume 1000 successfully created.

dscli> lslss

ID Group addrgrp stgtype confgvols
==================================
10     0       1 fb            1
```

Example 2-7 shows the use of the **lsavailpprcport** DS CLI command to find the available ports for PPRC and **lspprcpath** to create the PPRC pates.

Because in our example, both DS8000 systems provide an even-numbered storage pool for OpenStack (DS8000 #1 P0 and DS8000 #1 P6), we use two PPRC paths for LSS10.

This particular example also shows a scenario in which DS8000 #2 was already configured for Metro Mirror while DS8000 #1 was not.

For more information about Metro Mirror configuration setup, see *IBM DS8870 Copy Services for Open Systems*, SG24-6788.

Example 2-7 Verify and create the communication link between DS8000 systems

```
DS8000 #1 (not configured)
dscli> lsavailpprcport -l -remotewwnn 5005076308FFC56F 10:10
Local Port Attached Port Type Switch ID Switch Port
===================================================
I0033      I0131         FCP  NA        NA
I0033      I0333         FCP  NA        NA
I0301      I0131         FCP  NA        NA
I0301      I0333         FCP  NA        NA

dscli> lspprcpath -l 10
CMUC00234I lspprcpath: No Remote Mirror and Copy Path found

dscli> mkpprcpath -srclss 10 -tgtlss 10 -remotewwnn 5005076308FFC56F
I0033:I0333 I0301:I0131
CMUC00149I mkpprcpath: Remote Mirror and Copy path 10:10 successfully
established.

dscli> lspprcpath 10
Src Tgt State   SS   Port  Attached Port Tgt WWNN
=========================================================
10  10  Success FF10 I0033 I0333         5005076308FFC56F
10  10  Success FF10 I0301 I0131         5005076308FFC56F
```

```
DS8000 #2 (already configured)
dscli> lspprcpath 10
Src Tgt State   SS   Port  Attached Port Tgt WWNN
=========================================================
10  10  Success FF10 I0333 I0033         5005076308FFC54C
10  10  Success FF10 I0131 I0301         5005076308FFC54C
```

2.3.3 Installing IBM Storage Driver for OpenStack

This section describes steps normally performed by a cloud administrator to install the IBM Storage Driver for OpenStack on the Cinder node:

> **Important:** You must have root user privileges on the Cinder node.

1. Check that you have a supported Java version installed on the Cinder node that needs to connect to a DS8000. At the time of writing this paper, Java Runtime Environment 6 or later was required. See Example 2-8 for an illustration of how to check your Java version.

   Example 2-8 Verify Java version

   ```
   [root@server ~]# java -version
   java version "1.7.0_75"
   OpenJDK Runtime Environment (rhel-2.5.4.2.el7_0-x86_64 u75-b13)
   OpenJDK 64-Bit Server VM (build 24.75-b04, mixed mode)
   ```

2. Check that the Py4J library is installed on the Cinder node that will connect to a DS8000. Py4J is a bridge between Python and Java and is required for the IBM Storage Driver for OpenStack in DS8000 environments. For more details about Py4J and especially how to install it, see the following website:

   http://py4j.sourceforge.net/install.html

3. Extract the IBM Storage Driver for OpenStack installation package onto the Cinder node. Example 2-9 shows the command that we used to extract the package and lists the extracted files afterward.

   Example 2-9 Unpack IBM Storage Driver for OpenStack installation package

   ```
   [root@server ~]# tar xzvf IBM_Storage_Driver_for_OpenStack_1.5.0-b601.tar.gz

   [root@server ~]# ls IBM_Storage_Driver_for_OpenStack_1.5.0-b601
   certs  configure.sh  create_volume_types.sh  EXAMPLE.ini  install.sh  license
   runtime  uninstall.sh
   ```

4. Install the IBM Storage Driver for OpenStack by running the **install.sh** script (see Example 2-10).

   Example 2-10 IBM Storage Driver for OpenStack installation script

   ```
   [root@server IBM_Storage_Driver_for_OpenStack_1.5.0-b601]# ./install.sh
   Welcome to the IBM Storage Driver for OpenStack (v1.5.0-b601) installation.
   Press [ENTER] to proceed.

   Installing IBM Storage Driver for OpenStack Python eggs...
   Patching open source driver...
   Configuring OpenStack with IBM XIV|DS8000 Storage System information...
   ```

```
Installation of the IBM Storage Driver for OpenStack (v1.5.0-b601) is complete.
Press [ENTER] to exit.
```

> **Tip:** You can also use the silent mode with **#./install.sh -s**, which avoids prompts during installation. The **./install.sh -h** command provides help and usage information.

5. To configure the IBM Storage Driver for OpenStack at the Cinder node, you can use the **configure.sh** script, which is part of the installation package. This script helps you customize the Cinder config file in an interactive mode. The script first checks if some back-end storage is configured. Back-end storage is any individual connection between the Cinder node and a storage system. Example 2-11 shows a new configuration with no back-end storage configured.

   At this stage, the cloud administrator installing the driver must enter the information provided by the storage admin and gathered as described in 2.3.2, "Preparing the DS8000 for OpenStack" on page 17. Example 2-11 shows the workflow to add two DS8000 systems that will use the replication functionality. During the interactive configuration, the script asks you if you want to use replication and configure two storage back-ends.

Example 2-11 IBM Storage Driver for OpenStack configuration script

```
[root@server IBM_Storage_Driver_for_OpenStack_1.5.0-b601]# ./configure.sh
Welcome to the IBM Storage Driver for OpenStack (v1.5.0b603) configuration.
Press [ENTER] to proceed.

No backends were configured yet. Would you like to configure a new backend?
[Default: Yes ]:Yes
Enter the storage system type [x]iv or [d]s8k: [Default: x ]: d
Enter the storage system IP address or hostname: smoker1h.tuc.stglabs.ibm.com
Enter the username: [Default: admin ]:  admin
Enter the password:

Enter the IBM DS8000 Storage unit identifier: IBM.2107-75LR811
Please enter the full path of the Java executable: [Default: /usr/bin/java ]:
/usr/bin/java
Enter the name of the storage pool to be used: P0

Would you like to configure replication? [Default: n ]: y
Enter the storage system IP address or hostname: mtc036h.tuc.stglabs.ibm.com
Enter the username: [Default: admin ]:  admin
Enter the password:

Enter the name of the storage pool to be used: P6
Enter the IBM DS8000 Storage unit identifier: IBM.2107-75LT671
Copying DS8K Runtime Jars
Would you like to use CHAP authentication (relevant only to iSCSI connections)
for all iSCSI backends? [Default: N ]: n

Two replica backends are successfully configured, and the primary name is
IBM-DS8K_smoker1h.tuc.stglabs.ibm.com_P0_fibre_channel.

Please choose an action: [a]dd backend, [r]emove backend, [l]ist, [c]hange
password, [e]xit: l
Index   Storage System   IP Address or Hostname        Storage Pool
Connectivity
```

```
1       DS8K            smoker1h.tuc.stglabs.ibm.com   P0
fibre_channel
2       DS8K            mtc036h.tuc.stglabs.ibm.com    P6
fibre_channel
password, [e]xit: e
Stopping OpenStack Volume service...
Starting OpenStack Volume service...

Press [ENTER] to exit.
```

If there is already a configuration, the backends can be listed, removed, or their passwords can be changed by running the same **configure.sh** script again.

```
[a]dd backend, [r]emove backend, [l]ist, [c]hange password
```

6. After performing the configuration tasks, you can double-check the Cinder configuration file, which is located in `/etc/cinder/cinder.conf` for the submitted settings.

3

Using IBM DS8000 in an OpenStack environment

This chapter contains several examples that illustrate the use of the IBM Storage Driver for OpenStack with the DS8000. Examples cover volume creation and host attachment, snapshots, and volume replication. Each example details the related tasks performed by the user or the cloud administrator and shows which DS8000 functions are exercised through OpenStack. In addition, examples also illustrate the results of those actions as might be observed by the storage administrator on the DS8000.

This chapter includes the following sections:

- OpenStack volumes overview
- Volume creation
- Volume attach/detach
- Volume replication
- Volume retype
- Volume snapshots
- Volume Type specifications
- Snapshot consistency groups

3.1 OpenStack volumes overview

In OpenStack, volumes are detachable block storage devices. You can run several tasks in OpenStack for creating or attaching volumes to an instance.

This chapter describes tasks performed using the Horizon dashboard against connected DS8000 volumes. You can also use the OpenStackClient or Cinder command-line interface (CLI) for managing volumes. The following link provides more details about the specific Cinder CLI commands:

http://docs.openstack.org/user-guide-admin/common/cli_manage_volumes.html

OpenStackClient is a command-line client for OpenStack that brings the command set for Compute, Identity, Image, Object Storage, and Block Storage APIs together in a single shell with a uniform command structure. The following link provides more details about the specific OpenStackClient commands:

https://docs.openstack.org/python-openstackclient/latest/

3.2 Volume creation

The IBM Storage Driver for OpenStack allows you to create volumes from the Horizon dashboard. Figure 3-1 shows the volume list, which can be accessed from the main Horizon menu by selecting **Project** → **Compute** → **Volumes**.

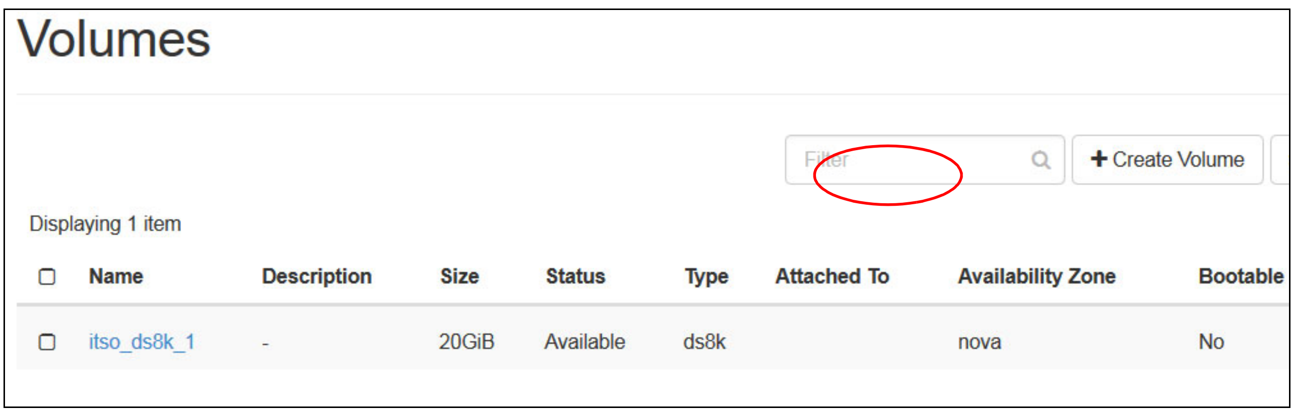

Figure 3-1 OpenStack Horizon volume list

To create a volume, click **Create Volume**. The Create Volume dialog is displayed. You must provide a **Volume Name**, **Description** (optional), **Volume Source**, **Volume type**, **Size**, and **Availability Zone**, as shown in Figure 3-2 on page 27.

In this OpenStack setup we defined the following DS8000 volume types:

- ds8k
- ds8k_thin
- ds8k_replication
- ds8k_thin_replication

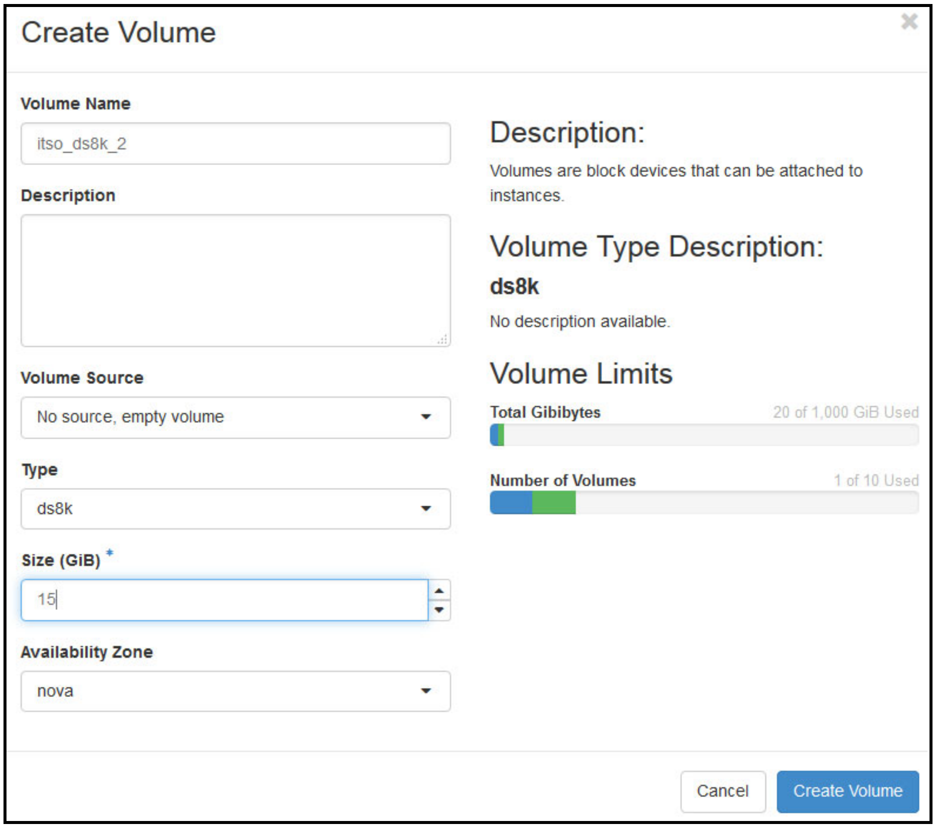

Figure 3-2 OpenStack Horizon create volume

With one DS8000 backend connected and no replication function configured, two different volume types are available:

- **ds8k**, which is used to create a standard logical unit number (LUN) by allocating all required extents on the DS8000.
- **ds8k_thin**, which is used to create a thin provisioned LUN that will be created as an Extent Space Efficient (ESE) LUN in the DS8000.

Important: To create ESE logical volumes, the thin provisioning licensed feature must be installed on the DS8000. For more information, see *DS8000 Thin Provisioning*, REDP-4554.

If a second DS8000 backend was created and replication was enabled during the configuration of IBM Storage Driver for OpenStack, the following volume types are visible:

- **ds8k_replication**
- **ds8k_thin_replication**

For more details about the volume types for replication, see 3.4, “Volume replication” on page 32.

In the example shown in Figure 3-3, we chose the volume type **ds8k** to create a volume on DS8000 #1.

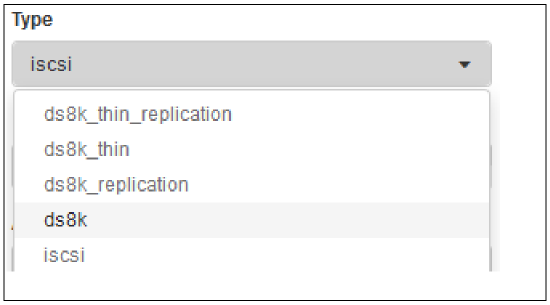

Figure 3-3 OpenStack Horizon volume types

After a successful volume creation, the new volume is listed and marked as available in the Volumes window (Figure 3-4).

Name	Description	Size	Status	Type	Attached To	Availability Zone	Bootable	Encrypted	Actions
itso_ds8k_2	-	15GiB	Available	ds8k		nova	No	No	Edit Volume

Figure 3-4 OpenStack Horizon volume list

Tip: In case of an error during volume creation, you can check `/var/log/cinder/volume.log` on your Cinder node for more details.

Click the volume name to display its settings and metadata, as shown in Figure 3-5.

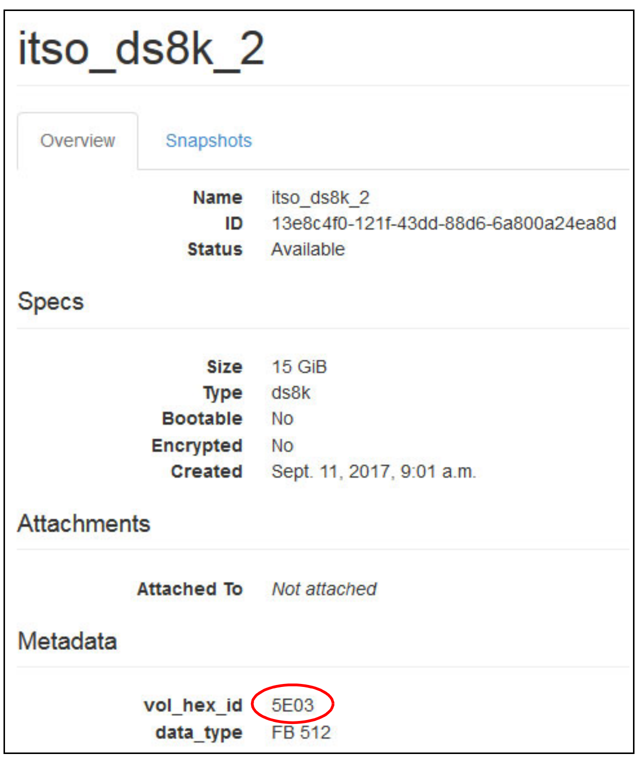

Figure 3-5 OpenStack Horizon volume details

The *vol_hex_id* in the Horizon dashboard is the LUN that was created on the DS8000 #1. In this case, it is LUN 5E03.

Example 3-1 illustrates the **showfbvol** DS CLI command that can be used to verify the settings of the corresponding LUN in the DS8000. In this example, you can see that, in this state, the volume is available in storage pool P14, but is not mapped to a volume group. In other words, the volume is not connected to a host.

Example 3-1 DS CLI showfbvol before mapping

```
dscli> showfbvol 5E03
Date/Time: September 22, 2017 10:46:16 AM CEST IBM DSCLI Version: 7.8.30.397 DS:
IBM.2107-75ACA91
Name                itso_ds8k_2
ID                  5400
accstate            Online
datastate           Normal
configstate         Normal
deviceMTM           2107-900
datatype            FB 512
addrgrp             5
extpool             P14
exts                15
cap (MiB)           15360
captype             DS
cap (2^30B)         15.0
cap (10^9B)         -
cap (blocks)        31457280
volgrp              - //no mapped volume group
ranks               1
dbexts              0
sam                 Standard
repcapalloc         -
eam                 managed
reqcap (blocks)     31457280
realextents         15
virtualextents      3
realcap (MiB)       15360
migrating           0
migratingcap (MiB)  0
perfgrp             PG0
migratingfrom       -
resgrp              RG0
tierassignstatus    -
tierassignerror     -
tierassignorder     -
tierassigntarget    -
%tierassigned       0
etmonpauseremain    -
etmonitorreset      unknown
GUID                6005076303FFD13E0000000000005400
```

3.3 Volume attach/detach

Before trying to attach a volume to an instance using the Horizon dashboard, verify that the nodes are physically connected and logically attached to the IBM DS8000. This verification was already part of the driver installation (see 2.1, “IBM Storage Driver for OpenStack” on page 12).

Users can configure the volume attachment out of the volume list in the Horizon dashboard by selecting **Manage Attachments** for the particular volume (see Figure 3-6).

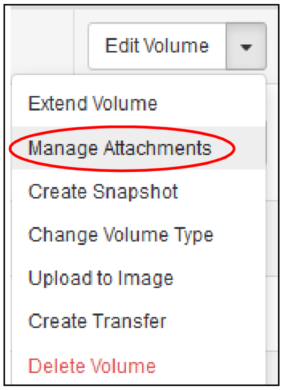

Figure 3-6 OpenStack Horizon manage attachments

Figure 3-7 shows the window where an instance to which the volume should be attached can be selected.

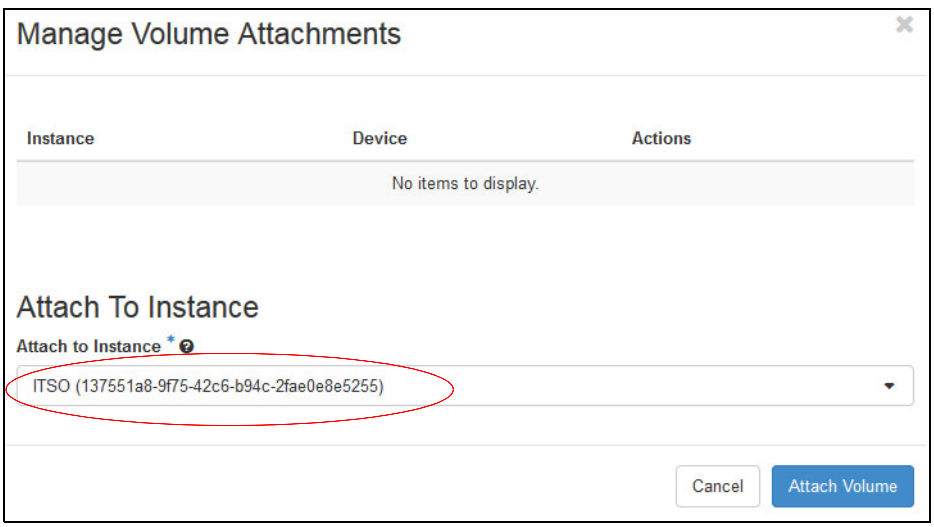

Figure 3-7 OpenStack Horizon attach a volume to an instance

When the user assigns a volume to an instance, Cinder and the IBM Storage Driver for OpenStack perform the required changes in the DS8000 logical configuration by adding the LUN to the corresponding volume group.

The DS8000 storage admin can check the mapping by issuing the DS CLI **showfbvol** command, as illustrated in Example 3-2.

Example 3-2 DS CLI showfbvol after mapping

```
dscli> showfbvol 5E03
Date/Time: September 22, 2017 10:46:16 AM CEST IBM DSCLI Version: 7.8.30.397 DS:
IBM.2107-75ACA91
Name                itso_ds8k_2
ID                  5400
accstate            Online
datastate           Normal
configstate         Normal
deviceMTM           2107-900
datatype            FB 512
addrgrp             5
extpool             P14
exts                15
cap (MiB)           15360
captype             DS
cap (2^30B)         15.0
cap (10^9B)         -
cap (blocks)        31457280
volgrp              -
ranks               V27 // mapped to volumegroup V27
dbexts              0
sam                 Standard
repcapalloc         -
eam                 managed
reqcap (blocks)     31457280
realextents         15
virtualextents      3
realcap (MiB)       15360
migrating           0
migratingcap (MiB)  0
perfgrp             PG0
migratingfrom       -
resgrp              RG0
tierassignstatus    -
tierassignerror     -
tierassignorder     -
tierassigntarget    -
%tierassigned       0
etmonpauseremain    -
etmonitorreset      unknown
GUID                6005076303FFD13E0000000000005400
```

Detaching a volume that was previously attached to an instance can be done from the same Horizon Manage Volume Attachment configuration window that is shown in Figure 3-7 on page 30.

3.4 Volume replication

This section illustrates a synchronous replication of Cinder volumes between two DS8000 systems. To that end, you must select one of the volume types with the “replication” suffix in its name. For example:

```
ds8k_replication
```

Note: Up to now, the IBM Storage Driver for OpenStack and DS8000 supports only synchronous volume replication. On DS8000, this is called *Metro Mirror*.

When a volume of the replica type is created, the IBM Storage Driver for OpenStack automatically creates the required LUNs on each of the DS8000s that has physical and logical Metro Mirror connections, as explained in 2.3.2, “Preparing the DS8000 for OpenStack” on page 17.

Figure 3-8 shows the details of the newly created volume `itso_ds8_repl1`. The metadata area shows `vol_hex_id = 0200`, which is the primary volume. At the secondary site, a corresponding LUN is also created, which is `replication_vol_hex_id = 0200` in our example.

Figure 3-8 OpenStack Horizon volume details replica

In addition to creating the LUNs, the replication (Metro Mirror relationship) is created and activated automatically between those two DS8000 LUNs. See Figure 3-9.

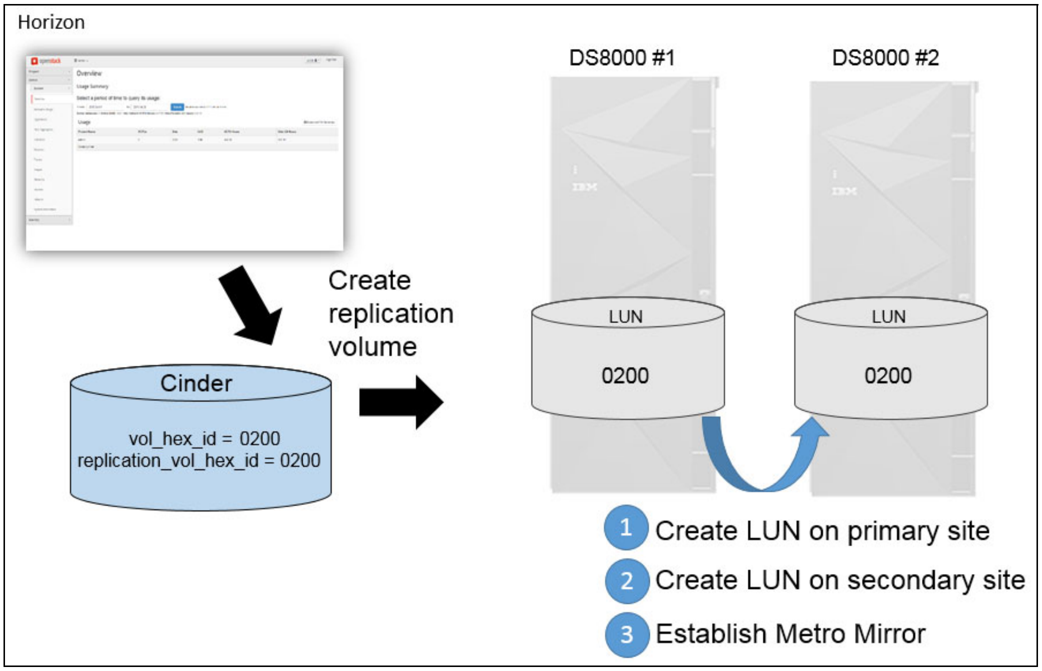

Figure 3-9 DS8000 volume replication in OpenStack

Example 3-3 shows the output of the **lspprc** DS CLI command when querying the DS8000 Metro Mirror relationship on both volumes after a single replication volume over the Horizon dashboard was created.

Example 3-3 DS CLI lspprc after creating a replication volume

```
DS8000 #1
dscli> lspprc 0200
Date/Time: September 21, 2017 2:20:48 PM CEST IBM DSCLI Version: 7.8.30.397 DS: IBM.2107-75ACA91
ID        State       Reason Type         SourceLSS Timeout (secs) Critical Mode First Pass Status
==========================================================================================
0200:0200 Full Duplex -      Metro Mirror 02        60             Disabled      Invalid

DS8000 #2
dscli> lspprc 0200
Date/Time: September 21, 2017 2:26:37 PM CEST IBM DSCLI Version: 7.8.30.397 DS: IBM.2107-75HGX91
ID        State              Reason Type         SourceLSS Timeout (secs) Critical Mode First Pass Status
================================================================================================
0200:0200 Target Full Duplex -      Metro Mirror 02        unknown        Disabled      Invalid
```

Example 3-4 shows the section in `cinder.conf` that is required to be able to replicate between two DS8000 systems.

Example 3-4 DS8000 configuration in cinder.conf for replication

```
[ds8k-r8-03_P13_P14_replicated]
volume_backend_name=ds8k-r8-03_P13_P14_replicated
volume_driver=cinder.volume.drivers.ibm.ibm_storage.ibm_storage.IBMStorageDriver
proxy=cinder.volume.drivers.ibm.ibm_storage.ds8k_proxy.DS8KProxy
san_ip=x.x.x.x
san_login=itso_admin
```

```
# base64-encoded password
san_password=cmVkYjBca3M=
san_clustername=P13,P14
#ds8k_devadd_unitadd_mapping =
replication_device=backend_id: ds8k_75HGX91, san_ip: x.x.x.x,
san_login:itso_admin, san_password: cmVkYjDwv3A=, san_clustername: P4,
connection_type: fibre_channel
#ds8k_host_type=auto
#ds8k_ssid_prefix=FF
connection_type=fibre_channel
chap=disabled
```

3.4.1 Managing volume replication

IBM Storage Driver for OpenStack also enables Cinder to manage the different replication tasks through the Cinder API.

> **Note:** Not all tasks that Cinder offers are implemented in the Horizon dashboard. In this regard, the following sections show several illustrations of the Cinder CLI instead of screen captures from the Horizon dashboard.

For details about the Cinder CLI commands, see the following link:

https://docs.openstack.org/python-cinderclient/latest/cli/details.html

Reenabling replication

Example 3-5 illustrates a problem that could occur in the connectivity between two mirrored LUNs, which leads to a Metro Mirror suspended state.

Example 3-5 DS CLI lspprc in suspended mode

```
DS8000 #1
dscli> lspprc 0200
Date/Time: September 22, 2017 12:39:34 PM CEST IBM DSCLI Version: 7.8.30.397 DS: IBM.2107-75ACA91
ID        State     Reason      Type         SourceLSS Timeout (secs) Critical Mode First Pass Status
====================================================================================================
0200:0200 Suspended Host Source Metro Mirror 02        60             Disabled      Invalid
```

Cinder recognizes this state and shows the `replication_status` of the affected volume as `inactive`, as illustrated in Example 3-6.

Example 3-6 Cinder CLI cinder show

```
cinder show 5a976ab3-7784-47cb-b3e1-b9882462c6df
+ ------------------+-----------------------------------------------------------------+
|     Property      |                             Value                               |
+ ------------------+-----------------------------------------------------------------+
|        id         |                04662df6-0ad4-46ce-bbb6-6a9e68ab1afd |
|     metadata      | {u'vol_hex_id': u'0200', u'replication_vol_hex_id': u'0200'}    |
|       name        |                            itso_ds8_repl1 |
| replication_status |                          inactive                               |
|       size        |                              1                                  |
|      status       |                          available                              |
|    volume_type    |                       ds8k_replication |
+ ------------------+-----------------------------------------------------------------+
(output modified for this paper)
```

To resynchronize the secondary volume with the primary after the connectivity issue has been fixed, you can use the **replication-reenable** Cinder CLI command:

```
cinder replication-reenable 04662df6-0ad4-46ce-bbb6-6a9e68ab1afd
```

Together with the IBM Storage Driver for OpenStack, Cinder re-establishes the Metro Mirror connectivity between both DS8000 LUNs.

Example 3-7 shows the **lspprc** output after reenabling the replication with the Cinder CLI command.

Example 3-7 DS CLI lspprc after replication-reenable

```
DS8000 #1
dscli> lspprc 1002

ID        State       Reason Type        SourceLSS Timeout (secs) Critical Mode First Pass Status
=================================================================================================
1002:10A6 Full Duplex -      Metro Mirror 10       60             Disabled      Invalid

DS8000 #2
dscli> lspprc 10a6

ID        State              Reason Type        SourceLSS Timeout (secs) Critical Mode First Pass Status
========================================================================================================
1002:10A6 Target Full Duplex -      Metro Mirror 10       unknown        Disabled      Invalid
```

The replication_status of the Cinder CLI command cinder show also reflects the new state (Example 3-8).

Example 3-8 Cinder CLI Cinder showing state after replication-reenable

```
cinder show 5a976ab3-7784-47cb-b3e1-b9882462c6df
+ -------------------+-------------------------------------------------------------+
|      Property      |                            Value                            |
+ -------------------+-------------------------------------------------------------+
|         id         |             5a976ab3-7784-47cb-b3e1-b9882462c6df            |
|      metadata      | {u'vol_hex_id': u'1002', u'replication_vol_hex_id': u'10A6'} |
|        name        |                        ds8_1_replica                        |
| replication_status |                           active                            |
|        size        |                              1                              |
|       status       |                          available                          |
|    volume_type     |                 fibre_channel_not_thin_replica              |
+ -------------------+-------------------------------------------------------------+
(output modified for this paper)
```

Replication promotion

Another replication feature enabled in Cinder when used with the DS8000 is the promotion of a replicated volume. The command is illustrated below:

```
cinder replication-promote 5a976ab3-7784-47cb-b3e1-b9882462c6df
```

This command promotes a secondary volume to become the primary. This particular action is usually performed during a planned or unplanned outage at the primary site.

Example 3-9 shows that issuing the above Cinder CLI command causes the Metro Mirror relationship to go to a suspended state at both sites. Furthermore, this command initiates a Metro Mirror failover at the secondary site with LUN 10A6. This LUN is now also marked as a source LUN.

Example 3-9 DS CLI lspprc after replication-promotion

```
DS8000 #1
dscli> lspprc 1002

ID        State     Reason      Type         SourceLSS Timeout (secs) Critical Mode First Pass Status
=====================================================================================================
1002:10A6 Suspended Host Source Metro Mirror 10        60             Disabled      Invalid

DS8000 #2
dscli> lspprc 10A6

ID        State     Reason      Type         SourceLSS Timeout (secs) Critical Mode First Pass Status
=====================================================================================================
10A6:1002 Suspended Host Source Metro Mirror 10        60             Disabled      Invalid
```

The Cinder **show** command illustrates the promotion in the metadata column by swapping `vol_hex_id` and `replication_vol_hex_id`. For the user, the volume ID of the Cinder volume remains (see Example 3-10).

Example 3-10 Cinder CLI cinder show after replication-promotion

```
+ -------------------+-----------------------------------------------------------------+
|      Property      |                              Value                              |
+ -------------------+-----------------------------------------------------------------+
|         id         |              5a976ab3-7784-47cb-b3e1-b9882462c6df               |
|      metadata      | {u'vol_hex_id': u'10A6', u'replication_vol_hex_id': u'1002'}    |
|        name        |                          ds8_1_replica                          |
| replication_status |                            inactive                             |
|        size        |                                1                                |
|       status       |                            available                            |
|    volume_type     |                   fibre_channel_not_thin_replica                |
+ -------------------+-----------------------------------------------------------------+
(output modified for this paper)
```

Another re-enable establishes a Metro Mirror fallback from LUN10A6 in DS8000 #2 to LUN 1002 in DS8000 #1. Example 3-11 shows that 10A6 is now PPRC source and 1002 is the PPRC target.

Example 3-11 DS CLI lspprc after second reenable-replication

```
DS8000 #1
dscli> lspprc 1002

ID        State              Reason Type         SourceLSS Timeout (secs) Critical Mode First Pass Status
=========================================================================================================
10A6:1002 Target Full Duplex -      Metro Mirror 10        unknown        Disabled      Invalid

DS8000 #2
dscli> lspprc 10A6
ID        State       Reason Type         SourceLSS Timeout (secs) Critical Mode First Pass Status
==================================================================================================
10A6:1002 Full Duplex -      Metro Mirror 10        60             Disabled      Invalid
```

The Cinder CLI now also shows the replication_status as `active` (Example 3-12).

Example 3-12 Cinder CLI cinder show after replication-promotion

```
cinder show 5a976ab3-7784-47cb-b3e1-b9882462c6df
+ ------------------+---------------------------------------------------------------+
|     Property      |                             Value                             |
+ ------------------+---------------------------------------------------------------+
|        id         |             5a976ab3-7784-47cb-b3e1-b9882462c6df              |
|     metadata      | {u'vol_hex_id': u'10A6', u'replication_vol_hex_id': u'1002'}  |
|       name        |                         ds8_1_replica                         |
| replication_status |                           active                              |
|       size        |                               1                               |
|      status       |                           available                           |
|    volume_type    |                fibre_channel_not_thin_replica                 |
+ ------------------+---------------------------------------------------------------+
(output modified for this paper)
```

3.4.2 Methods for consistency group (replication)

The OpenStack Pike release has the following new methods for consistency groups with replication:

- Create Consistency Group Type:

  `cinder group-type-create cg-replica-type`

- Enable Snapshot Capability on CG Type:

  ```
  cinder group-type-key cg-replica-type set
  consistent_group_replication_enabled="<is> True”
  ```

- Create a Consistency Group:

  `cinder group-create --name cg2 cg-replica-type`

- Create a volume in a CG or add an existing volume to a CG:

  ```
  cinder create --group-id --volume-type --name
  cinder group-update
  --add-volumes=09d119b2-c5de-4b53-ae35-992cf1825acd,11ab8cf8-2e26-4e8f-b249-54b1
  3daf3dc5,33f87199-df02-41db-a46d-dc8d697f6447
  7c91d2f0-c1ca-48f1-9e23-421d969550ec9
  ```

3.5 Volume retype

With volume retyping, users can change the volume type after its creation. The Horizon action menu allows this action by selecting **Change Volume Type** for a specific volume, as illustrated in Figure 3-10.

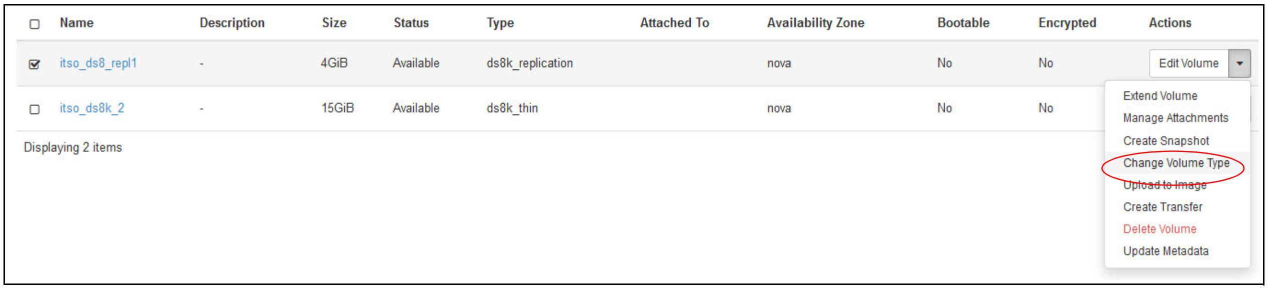

Figure 3-10   OpenStack Horizon volume retype

In particular, a volume retype can be used to drop and enable replication for a volume. In this scenario, you drop the replication by changing a replicated volume (created in Section 3.4, “Volume replication” on page 32) to a non-replicated volume type.

Figure 3-11 shows changing the volume type from **ds8k_replication** to **ds8k**.

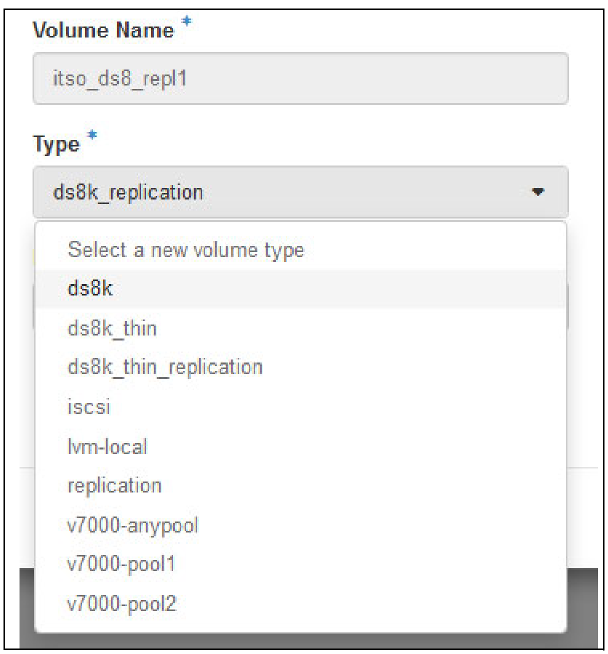

Figure 3-11   OpenStack Horizon change volume type

This action causes two changes on the storage subsystems. Example 3-13 shows that the Metro Mirror relation (PPRC pair) and LUN 10a6 of DS8000 #2 are deleted by this task.

Example 3-13   DS CLI lspprc and lsfbvol after volume retype

```
DS8000 #1
dscli> lspprc 0200
Date/Time: September 27, 2017 11:23:40 AM CEST IBM DSCLI Version: 7.8.30.397 DS:
IBM.2107-75ACA91
CMUC00234I lspprc: No Remote Mirror and Copy found.
```

```
DS8000 #2
dscli> lsfbvol 0200
Date/Time: September 27, 2017 11:25:40 AM CEST IBM DSCLI Version: 7.8.30.397 DS:
IBM.2107-75HGX91
CMUC00234I lsfbvol: No FB Volume found.
```

Of course, it is also possible to retype a volume from **ds8k_thin_replication** to **ds8k_replication**. Instead of dropping the copy relationship between the LUNs, it creates a new one.

3.6 Volume snapshots

OpenStack offers multiple actions that are all called *snapshots*. The following list shows the different types of snapshot actions:

- Images (single file that contains a virtual disk)
- Instance snapshots (which are similar to images)
- Volume snapshots

The differences between these actions sometimes causes confusion.

Here, we focus on volume snapshots. This action can be invoked from the volumes list window in the Horizon dashboard. Figure 3-12 shows the invocation of **Create Snapshot** for the volume *ds8_1_nothin*.

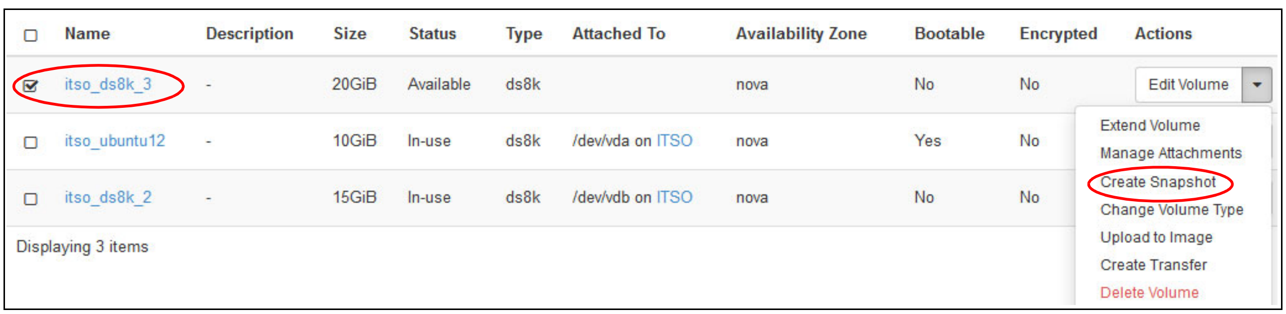

Figure 3-12 OpenStack Horizon create snapshot

When creating a volume snapshot, you only need to type in a name for the snapshot. Cinder passes that request to the DS8000 where this action automatically creates a new LUN. It then performs a DS8000 internal FlashCopy operation with background copy between the source LUN (1000) and the newly created target LUN (1001), as depicted in Figure 3-13.

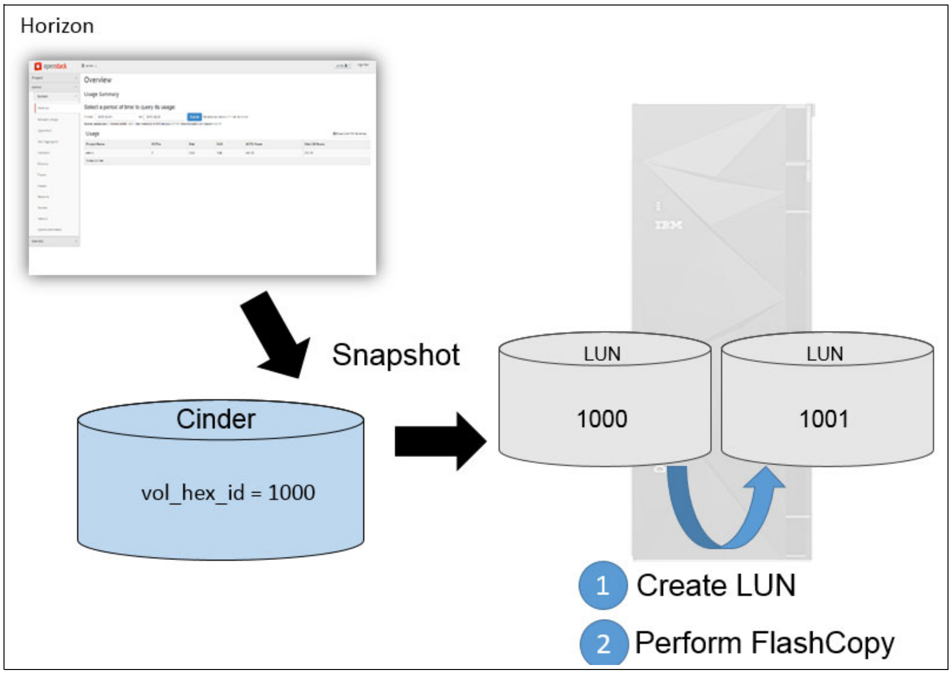

Figure 3-13 DS8000 snapshots in OpenStack

There are several usage scenarios for snapshots. For example, you can use a snapshot to create a volume by selecting the snapshot as a source when creating a LUN (Figure 3-14).

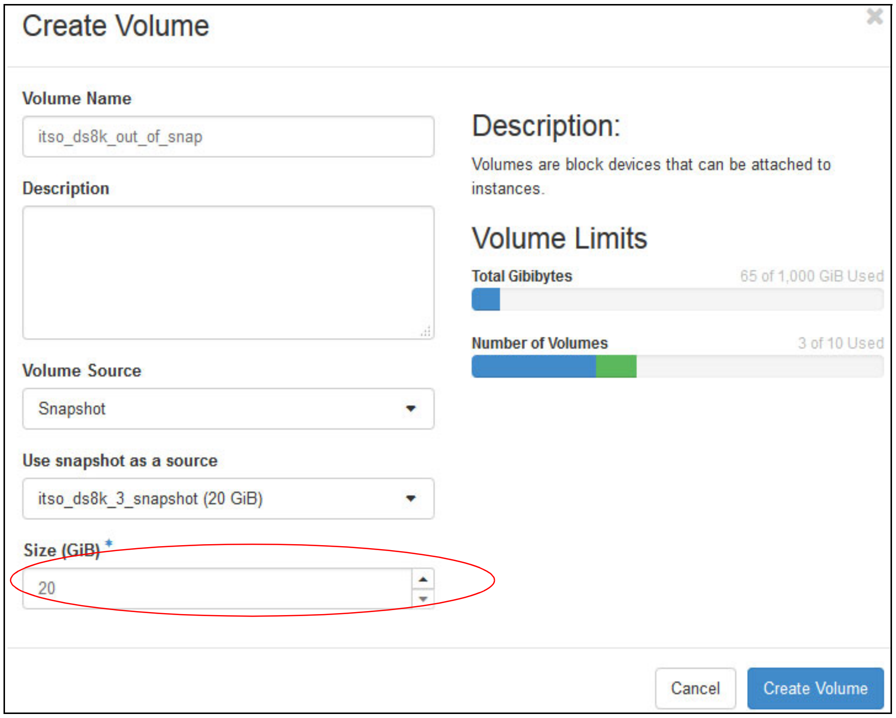

Figure 3-14 OpenStack Horizon

3.7 Volume Type specifications

In OpenStack, the different volume types are stored in an internal database table that contains their different specifications. Cloud administrators can check and change those specifications on their own.

Figure 3-15 shows this capability in the Horizon dashboard. From the main dashboard window, select **Admin** → **Volume** → **Volume Types** to obtain the list of all volume types that the user can select when creating a volume.

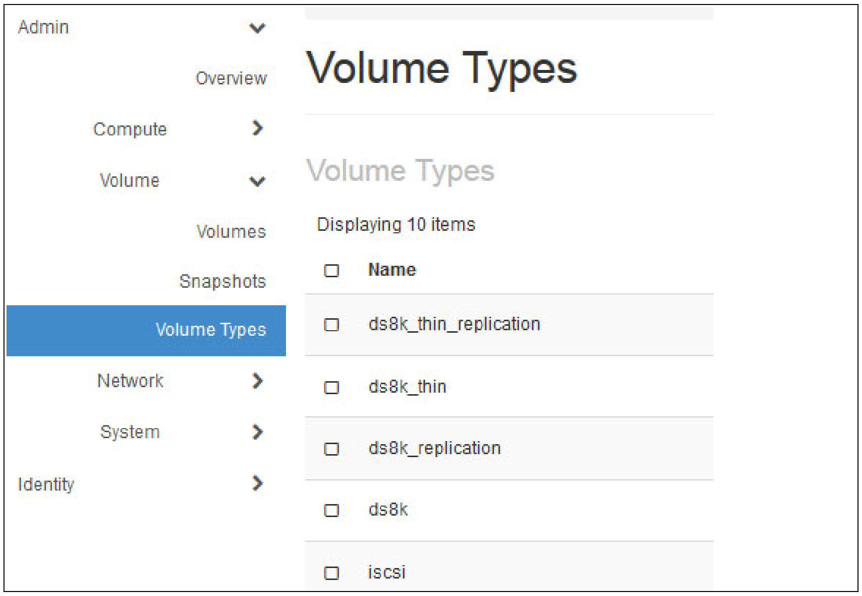

Figure 3-15 OpenStack Horizon list volume types

Each specific volume type characteristic can be edited through the **Actions** menu. Figure 3-16 shows the actual specs about the volume type `ds8k_thin_replication`.

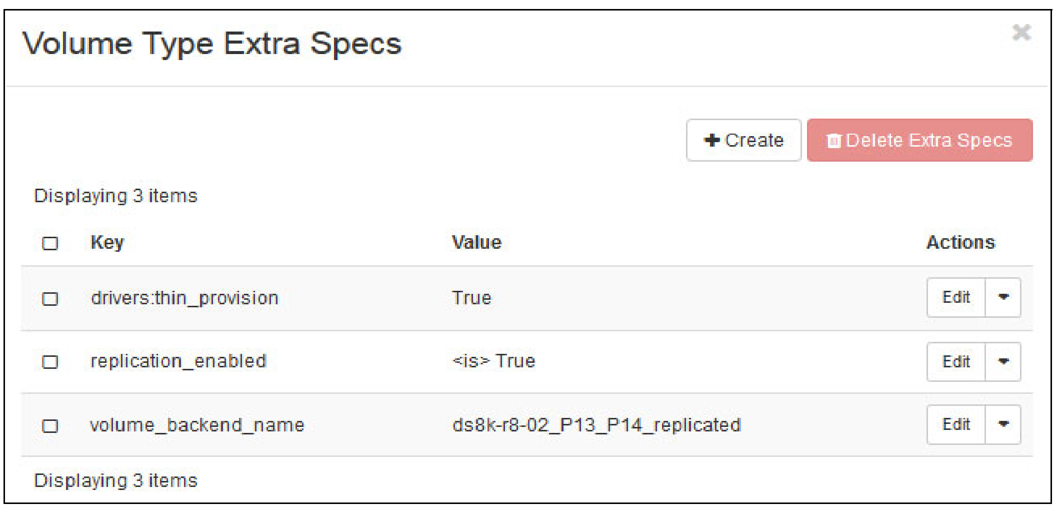

Figure 3-16 OpenStack list volume type extra specs

For example, setting the `replication_enabled` value to `<is> True` allows you to create a mirror relationship with this volume type.

3.8 Snapshot consistency groups

When applications have their data spread over multiple volumes and those volumes must remain consistent, a consistent snapshot of all the volumes can be created by using a consistency group snapshot. Cinder and the IBM Storage Driver for OpenStack support this capability. A consistency group (CG) is a collection of relationships that can be treated as one entity, usually to preserve write order consistency across a group of volumes that pertain to one application, for example, a database volume and a database log file volume.

3.8.1 Enabling the consistency group support in OpenStack policy

By default, the consistency group functions are disabled. You must enable them before running consistency group operations by changing policies in the `/etc/cinder/policy.json` file.

The default policies for consistency groups are shown in Example 3-14.

Example 3-14 Default policy entries for consistency groups

```
"consistencygroup:create": "group:nobody",
"consistencygroup:delete": "group:nobody",
"consistencygroup:get": "group:nobody",
"consistencygroup:get_all": "group:nobody",
"consistencygroup:create_cgsnapshot" : "group:nobody",
"consistencygroup:delete_cgsnapshot": "group:nobody",
"consistencygroup:get_cgsnapshot": "group:nobody",
"consistencygroup:get_all_cgsnapshots": "group:nobody",
```

To enable consistency groups, remove `group:nobody` as shown in Example 3-15.

Example 3-15 Policy entries to enable consistency group operations

```
"consistencygroup:create": "",
"consistencygroup:delete": "",
"consistencygroup:update": "",
"consistencygroup:get": "",
"consistencygroup:get_all": "",
"consistencygroup:create_cgsnapshot" : "",
"consistencygroup:delete_cgsnapshot": "",
"consistencygroup:get_cgsnapshot": "",
"consistencygroup:get_all_cgsnapshots": "",
```

Restart the Cinder API service after changing policies.

The following consistency group operations are supported in the OpenStackClient:

- openstack consistencygroup list
- openstack consistency group add volume
- openstack consistency group create
- openstack consistency group delete
- openstack consistency group list
- openstack consistency group remove volume
- openstack consistency group set
- openstack consistency group show
- openstack consistency group snapshot create

- openstack consistency group snapshot delete
- openstack consistency group snapshot list
- openstack consistency group snapshot show

3.8.2 Create a consistency group type

The **cinder group-type-create cg-snap-type** creates a consistency group type and enables it for FlashCopy, as shown in Example 3-16.

At the time of writing only Block Storage V3 API supports groups. You can specify `--os-volume-api-version 3.x` when using the Cinder command line for group operations where 3.x contains a microversion value for that command.

Example 3-16 Cinder creates consistency group type and enable for FlashCopy

```
cinder --os-volume-api-version 3.13 group-type-create itso-cg-type
+--------------------------------------+--------------+-------------+
| ID                                   | Name         | Description |
+--------------------------------------+--------------+-------------+
| 5642d7d4-861f-4911-838f-79031711a114 | itso-cg-type | -           |
+--------------------------------------+--------------+-------------+

cinder --os-volume-api-version 3.13 group-type-key itso-cg-type  set
consistent_group_snapshot_enabled='<is> True'
```

3.8.3 Create a consistency group

The **openstack consistency group create** command creates a new group. The syntax of the command is:

```
openstack consistency group create --volume_type <type1,type2,...> <name>
```

A consistency group can support more than one volume type. A consistency group is empty upon its creation. Volumes need to be created and added to it later.

Example 3-17 shows the creation of consistency group `ds8cg1`. We specify a name and the volume type that can be supported by this consistency group (see section 3.7, “Volume Type specifications” on page 41). We then create two volumes and add them to the consistency group `ds8cg1`.

Example 3-17 Cinder CLI consistency group and volume create

```
cinder --os-volume-api-version 3.13 group-create --name ds8cg1 itso-cg-type ds8k
+-------------------+------------------------------------------+
| Property          | Value                                    |
+-------------------+------------------------------------------+
| availability_zone | nova                                     |
| created_at        | 2017-09-20T10:47:45.000000               |
| description       | None                                     |
| id                | 7636a3f2-f3b1-4c9d-8c23-7ac47cafcdbb     |
| name              | ds8cg1                                   |
| status            | creating                                 |
| volume_types      | [u'192b3f67-7b2f-4c56-b2bb-7a18895406ce'] |
+-------------------+------------------------------------------+
```

```
cinder --os-volume-api-version 3.13 group-update --add-volumes=itso_ds8_cg1_1,
itso_ds8_cg1_2 ds8cg1

or create volumes with consistency group
cinder --os-volume-api-version 3.13 create --group-id
7636a3f2-f3b1-4c9d-8c23-7ac47cafcdbb --volume-type ds8k --name itso_ds8_cg1_1 5
```

Figure 3-17 shows a list of the two volumes created.

Figure 3-17 OpenStack Horizon volume list

3.8.4 Create a consistency group snapshot

To initiate a consistent snapshot, use the **group-snapshot-create** Cinder CLI command, as shown in Example 3-18.

The following are methods you can use for Consistency group snapshot:

- Create Consistency group type:

  ```
  cinder group-type-create
  ```

- Enable snapshot capability on consistency group type:

  ```
  cinder group-type-key cg-snap-type set consistent_group_snapshot_enabled="<is>
  True”
  ```

Example 3-18 Cinder CLI goup-snapshot-create

```
# cinder --os-volume-api-version 3.14 group-snapshot-create --name ds8_consnap
+--------------------+--------------------------------------+
|      Property      |                Value                 |
+--------------------+--------------------------------------+
| consistencygroup_id | 0afcb6fb-c21d-4820-9d85-69a269ed4693 |
|     created_at     |      2017-09-29T16:52:29.000000      |
|    description     |                 None                 |
|         id         | 098e087b-0194-43ba-99df-a19f78886b47 |
|        name        |             ds8_consnap              |
|       status       |               creating               |
+--------------------+--------------------------------------+
```

The **group-snapshot-create** command creates a snapshot of every volume that is part of the consistency group (see Figure 3-18).

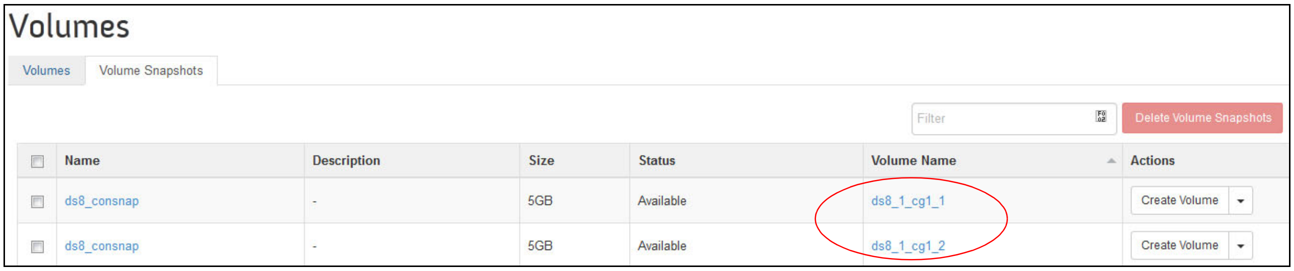

Figure 3-18 OpenStack Horizon volume snapshot list

The DS8000 FlashCopy operation accounts for data consistency when creating the snapshots. As described in Figure 3-19, after creating the required LUNs (FlashCopy targets), the FlashCopy operation includes a freeze option. This option holds off I/O activity to the involved source LUNs for a specific time. If a consistent state is given and the FlashCopy operation has been started, an unfreeze will follow to resume I/O activity.

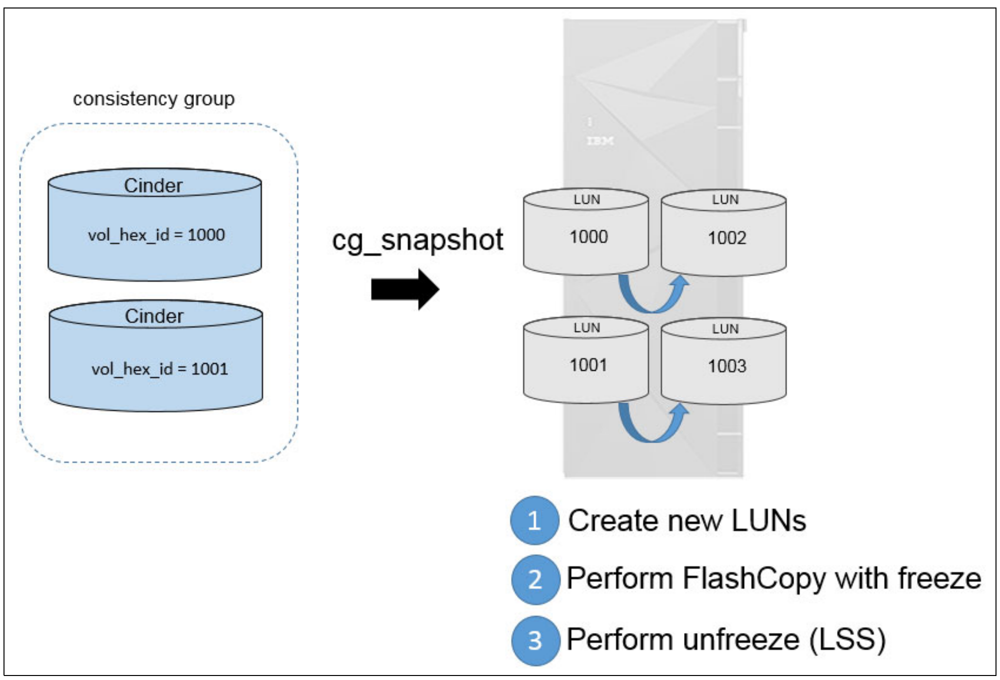

Figure 3-19 DS8000 consistency group snapshots with OpenStack

3.8.5 Deleting volumes in a consistency group

To delete the volumes after they have been created in a consistency group, you must delete that consistency group and all its volumes by using the following command:

```
cinder --os-volume-api-version 3.13 group-update --remove-volumes=<uuid1,uuid2>
<consistency group>
```

Related publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this paper.

IBM Redbooks

The following IBM Redbooks publications provide additional information about the topic in this document. Note that some publications referenced in this list might be available in softcopy only.

- *IBM DS8870 Architecture and Implementation (Release 7.5)*, SG24-8085
- *DS8000 Copy Services*, SG24-8367

You can search for, view, download or order these documents and other Redbooks, Redpapers, Web Docs, draft and additional materials, at the following website:

ibm.com/redbooks

Online resources

These websites are also relevant as further information sources:

- IBM Support Portal

  http://www.ibm.com/storage/support
- OpenStack Storage reference page

  http://www.openstack.org/software/openstack-storage

Help from IBM

IBM Support and downloads

ibm.com/support

IBM Global Services

ibm.com/services

REDP-5220-01

ISBN 0738456535

Printed in U.S.A.

Get connected

Redbooks®

ibm.com/redbooks