Abstract

The use of OpenStack as an orchestration and operating platform for on-premise and private cloud infrastructure deployments continues to grow at a rapid rate. There are several ways to implement OpenStack. Methods vary widely depending on the use case, the type of workload and the back-end infrastructure being deployed on. This Technical Report (TR) provides guidance for an approach for implementing OpenStack (Liberty release) with NetApp's ONTAP 9 data management operating system. The target audience for this TR is systems or storage administrators tasked with implementing OpenStack with ONTAP 9 in a data center or on a private cloud environment. This TR represents one of several implementation approaches for OpenStack in combination with ONTAP 9. The method described in this TR uses Packstack with Puppet modules. The benefits of this approach are simplicity (provides a single interface for connecting private networks to the internet and storage), it allows OpenStack to take advantage of ONTAP 9's storage efficiency features, and it provides a consistent data management platform for current users of ONTAP 9 that want to understand how to deploy ONTAP 9 in an OpenStack Environment. These benefits are not unique to this approach. The intent is to present an additional implementation option for consideration.

NetApp was an early supporter of OpenStack. In addition to participating on board of the OpenStack Foundation, NetApp has been an active contributor to the Cinder, Manila, and Swift drivers for blocks, files, and object stores.
# TABLE OF CONTENTS

1. Introduction .................................................................................................................. 4
2. Target Audience .......................................................................................................... 5
3. DevOps Test Environment ............................................................................................. 5
4. OpenStack Liberty Installation Prerequisites ............................................................... 6
   4.1 Controller and Compute Node .................................................................................. 6
   4.2 NetApp ONTAP 9 ..................................................................................................... 7
5. Deploy OpenStack ....................................................................................................... 7
   5.1 Subscribing to the Repositories ............................................................................ 7
   5.2 Running the Puppet Modules and Packstack Script ........................................... 8
   5.3 Initial Setup for OpenStack .................................................................................... 9
   5.3.1 Setup Neutron Networking .............................................................................. 9
   5.3.2 Create Floating IPs .......................................................................................... 12
   5.3.3 Set up Security Groups ..................................................................................... 13
   5.3.4 Creating a Key Pair ......................................................................................... 13
   5.3.5 Setup Glance with an NFS Share ..................................................................... 13
   5.3.6 Setup Nova with an NFS Share .......................................................................... 15
   5.3.7 Setup Cinder with an NFS Share ....................................................................... 16
6. Configure Additional Compute Nodes ....................................................................... 17
7. Launch an Instance .................................................................................................... 18
   7.1 Launch an Instance from the Dashboard ............................................................. 18
   7.2 Launch an instance from CLI ................................................................................ 21
   7.3 Login to an Instance .............................................................................................. 22
   7.3.1 Associate a Floating IP from the Dashboard .................................................. 22
   7.3.2 Associate a Floating IP from the CLI .............................................................. 22
   7.3.3 Change the password for a Linux Instance .................................................... 23
8. Attaching Cinder Volumes to an Instance ................................................................ 23
   8.1 Create a Cinder Volume from CLI ......................................................................... 23
   8.2 Create a Cinder Volume from the Dashboard ...................................................... 23
   8.3 Attach cinder volume to an Instance ...................................................................... 24
Appendix .......................................................................................................................... 26
   Troubleshoot Installation ............................................................................................ 26
   Troubleshoot NFS Shares ........................................................................................... 27
   Troubleshoot the Glance Service ............................................................................... 28
   Troubleshoot the Nova Service .................................................................................. 28
   Troubleshoot the Neutron Service .............................................................................. 31
Troubleshoot the Cinder Service........................................................................................................................................32
Reset the Default Password........................................................................................................................................34
Configure Network Interfaces.....................................................................................................................................34

Acknowledgments ..........................................................................................................................................................35
References .................................................................................................................................................................35
Version History ..........................................................................................................................................................36

LIST OF TABLES
Table 1) Pre-requisites for OpenStack Controller and Compute Nodes .................................................................7

LIST OF FIGURES
Figure 1) OpenStack Components ..............................................................................................................................4
Figure 2) OpenStack Liberty Setup on NetApp ONTAP 9...........................................................................................5
Figure 3) Basic Architecture of a Two-node Deployment .........................................................................................6
1 Introduction

OpenStack as an open-source cloud operating system is what Linux was to servers years ago. Open-source Linux has rapidly matured and evolved over the years through different distributions, and OpenStack is quickly heading down that path, becoming more popular as infrastructure as a service (IaaS) in the cloud.

OpenStack provides a wide variety of software-defined tools or components that abstract the underlying infrastructure into a large resource pool. With this feature, users have efficient on-demand provisioning of resources, and heterogeneous platforms can be supported in the cloud environment.

Figure 1 lists all the different components that are part of the OpenStack modular architecture. Although some of the components, such as Glance, Nova, Neutron, and Cinder, are commonly used, Manila and Swift are out of incubation and are part of the Liberty release. Other modules such as database and data processing are quickly evolving to address some of the key business requirements.

OpenStack is commonly used as IaaS by many infrastructure developers who have the task of adopting the DevOps culture for the new generation of application development. Infrastructure automation and on-demand provisioning by using APIs are a major requirement in the development environment. However, the various OpenStack components in Figure 1 that are relevant to the DevOps practice might be incomplete and fragmented. This document provides a sample OpenStack installation and the configuration steps that are required for DevOps on the NetApp clustered Data ONTAP operating system.

NetApp has a strong storage presence in the many organizations that are developing software. The traditional form of monolithic software development is rapidly shifting to a more modular application development process that can speed time to market and is cost-efficient yet maintains a better code quality. Development workflows are changing and infrastructure automation and provision are quickly ramping up, but storing and managing data on NetApp storage with its native technologies has been a huge asset to business owners and developers. The RESTful APIs from NetApp clustered Data ONTAP enable many operations teams to automate numerous storage operations that use OpenStack for IaaS or any other platform-as-a-service integrations.

The scalable clustered file system in the NetApp ONTAP 9 operating system provides thin-provisioned NetApp FlexVol® volumes and Snapshot® copies to store and protect code repositories. It also provides user workspaces through NetApp FlexClone® technology, which are used by developers to modify, unit-test, and build code. The workspaces are instantaneous and transient in nature, and they take up almost no storage space. The workspaces are prepackaged with all the necessary files, including compiled...
system images, libraries, tools, RPM, and so on. For better manageability and storage efficiency, when developers have completed the code changes, these workspaces can be deleted.

NetApp also addresses the growing storage needs of customers by providing different service-level objectives (SLOs). These SLOs efficiently tier data based on the different workloads that are generated by different classes of application developers during the entire application lifecycle in a DevOps practice. NetApp clustered ONTAP 9 provides some of the key features, such as deduplication, compression, and compaction, to further improve the storage efficiency for builds and after the application is deployed into production. Integrating the development process with NetApp and OpenStack can empower business owners with the following benefits:

- Agile infrastructure
- Risk mitigation
- Improved return on investment

2 Target Audience

This document is primarily to provide guidance to infrastructure administrators who are charged with configuring OpenStack Liberty on NetApp clustered ONTAP 9 and Red Hat Enterprise Linux (RHEL) 7.2 clients. The information in this report includes:

- Prerequisites for installation of the different OpenStack components on the compute, network, and storage units
- Best practices for storage, network, and compute nodes, and tips and tricks for configuring the different OpenStack components on NetApp storage with Packstack

3 DevOps Test Environment

Depending on the business requirements, every DevOps environment can be different. This document illustrates a sample DevOps environment on NetApp storage that uses OpenStack to provision resources for the application development workflow. There might be environments that are similar to this setup, or this architecture can also be used as a reference in production.

Figure 2) OpenStack Liberty Setup on NetApp ONTAP 9.

Figure 2 depicts a simple architecture that uses OpenStack components such as Glance, Nova, Neutron, Cinder, and Horizon on NetApp All Flash FAS (AFF) 8080 EX and RHEL 7.2 running on the physical compute nodes. The instructions found on https://www.rdoproject.org/install/quickstart/ were used to install Liberty with Packstack utility. For the purposes of this Technical Report, it is recommended to use
the Red Hat OpenStack Platform (RHOSP) 8 director to install Red Hat OpenStack Platform while setting up an OpenStack environment with Red Hat.

One of the physical compute nodes has Nova, Neutron, and Cinder installed for configuring that component as the main OpenStack controller node. The other physical compute nodes that are added to the compute farm will run only Nova and Neutron on them to spin up the Nova virtual machines (VMs) and the communication between the nodes in the network.

All the flexible volumes in the NetApp storage are mounted over Network File System (NFS) v3 on the physical RHEL 7.2 compute nodes. Glance uses a flexible volume to store the ISO images and the snapshot copies of the Nova VM root partitions. All the Nova VM root partitions are placed into a single flexible volume over NFS. The additional data partitions on the VMs are assigned to separate Cinder volumes that are part of a single flexible volume on NetApp. Apart from these volumes this, there are other flexible volumes that are created on the NetApp storage for GIT repository and code builds.

The details on the prerequisites, best practices, and installation tips and tricks appear later in this report.

4 OpenStack Liberty Installation Prerequisites

This section lists the prerequisites for configuring the network and the type of hardware that is used for the OpenStack controller. It also lists the compute nodes and the NetApp storage controller in the NetApp designed DevOps environment that uses OpenStack Liberty. The system is deployed by using Packstack. Figure 3 shows a two-node deployment.

Figure 39) Basic Architecture of a Two-node Deployment.

4.1 Controller and Compute Node

This section describes the pre-requisites for the OpenStack Controller and Compute node.
Table 1) Pre-requisites for OpenStack Controller and Compute Nodes

<table>
<thead>
<tr>
<th>OpenStack Controller Node</th>
<th>OpenStack Compute Node</th>
</tr>
</thead>
<tbody>
<tr>
<td>RHEL 7.2</td>
<td>RHEL 7.2</td>
</tr>
<tr>
<td>Three network interfaces:</td>
<td>Two network interfaces:</td>
</tr>
<tr>
<td>• Interface 1 for Internet access</td>
<td>• Interface 1 for OpenStack nodes to communicate</td>
</tr>
<tr>
<td>• Interface 2 for OpenStack nodes to communicate</td>
<td>• Interface 2 for access to the storage network.</td>
</tr>
<tr>
<td>• Interface 3 for access to the storage network</td>
<td>(NetApp recommends a third interface to get Internet access to the compute node to update or install various packages.)</td>
</tr>
<tr>
<td>Minimum of 64GB RAM with 8 processors</td>
<td>Minimum of 64GB RAM with 8 processors</td>
</tr>
<tr>
<td>To configure the interfaces, see the <a href="https://netapp.github.io/openstack-deploy-ops-guide/attachments/Instructions_and_Script_to_Provision_OpenStack_Liberty_with_PackStack.zip">appendix</a></td>
<td>To configure the interfaces, see the <a href="https://netapp.github.io/openstack-deploy-ops-guide/attachments/Instructions_and_Script_to_Provision_OpenStack_Liberty_with_PackStack.zip">appendix</a></td>
</tr>
</tbody>
</table>

### 4.2 NetApp ONTAP 9

Configure the following on ONTAP 9 in a single storage virtual machine (SVM):

1. One NFS share for the Glance service.
2. One NFS share for the Nova service.
3. At least one NFS share for the Cinder service.
4. Appropriate export policies to enable exporting the shares.

### 5 Deploy OpenStack

This section covers the entire installation process of all the nodes and their respective services in an OpenStack environment.

OpenStack is a fast-paced and massive project. There are three different ways to install OpenStack Liberty. One of the best ways is to rely on the downstream distributions, which handle various dependencies and minute details for your environment, such as the Red Hat OpenStack Platform 8 Director. Another option is to use scripts to deploy the various nodes and services of OpenStack. The third option is to provision using Packstack.

In this paper, we use the third option, Packstack, to provision OpenStack Liberty. Packstack uses Puppet modules to deploy OpenStack on multiple servers over SSH.


**Note:** The setup process is similar regardless of the mode of installation.

### 5.1 Subscribing to the Repositories

Perform the following steps on the controller node and on all the compute nodes.

1. Enable the repos to install various dependencies.

   ```bash
   subscription-manager repos --enable rhel-7server-extras-rpms
   subscription-manager repos --enable rhel-7-server-optional-rpms
   ```
2. Install the following packages after the repositories have been enabled.

   ```bash
   sudo yum install --y python-werkzeug
   ```

   ```bash
   sudo yum install --y python-cheetah
   ```

5.2 Running the Puppet Modules and Packstack Script

This section describes how to run the script to install various services on to the nodes:

1. Download the script from the link provided at the start of section 5.
2. Unzip the package.
3. Run the following command as a root user or create another user to run the script on the controller node.

   ```bash
   ./provision-packstack-liberty.sh <answer_file> <cluster-management-ip-address> <vserver-name> <cDOT-username> <cDOT-password> <external-ethernet-interface> <private-ethernet-interface> <storage-ethernet-interface>
   ```

   Following is a sample command.

   ```bash
   [root@Linux-node3 rduser1]# ls
   [root@Linux-node3 rduser1]# ./provision-packstack-liberty.sh ans.txt 10.192.39.10 devops-wanlla admin netappl23 eno2 enp7s0f1 enp7s0f6
   ```

4. You should see the following, indicating that the installation was successful.

   ```bash
   Applying 10.192.39.23_neutron.pp
   10.192.39.23_samp.pp: [ DONE ]
   Applying 10.192.39.23_waaspp.pp: [ DONE ]
   Applying 10.192.39.23_nginx.pp
   Applying 10.192.39.23_nginx.pp:
   Applying 10.192.39.23_keystone.pp
   Applying 10.192.39.23_glance.pp
   Applying 10.192.39.23_cinder.pp
   Applying 10.192.39.23_mongodb.pp
   Applying 10.192.39.23_manila.pp
   Applying 10.192.39.23_manila.pp:
   Applying 10.192.39.23_manila.pp:
   Applying 10.192.39.23_sqla.pp
   Applying 10.192.39.23_sqla.pp:
   Applying 10.192.39.23_zabbix.pp
   Applying 10.192.39.23_zabbix.pp:
   Applying 10.192.39.23_nova.pp
   Applying 10.192.39.23_nova.pp:
   Applying 10.192.39.23_nova.pp:
   Applying 10.192.39.23_neutron.pp
   Applying 10.192.39.23_neutron.pp:
   Applying 10.192.39.23_ceilometer.pp
   Applying 10.192.39.23_ceilometer.pp:
   Applying 10.192.39.23_horizon.pp
   Applying 10.192.39.23_horizon.pp:
   Applying 10.192.39.23_postscript.pp
   Applying 10.192.39.23_postscript.pp:
   Applying Puppet manifests
   Finalizing
   **** Installation completed successfully *******
   ```

   Additional information:
   * The file /root/keystonerc_admin has been created on OpenStack Client host 10.192.39.23. To use the command line tools you need to source the file.
   * To access the OpenStack Dashboard browse to http://10.192.39.29/dashboard .
   * Please, find your login credentials stored in the keystonerc_admin in your home directory.
   * Because of the normal updates the node 10.192.39.23 requires reboot.
   * The installation log file is available at: /var/log/packstack/20180216-138825-5jtw/openstack-setup.log
   * The generated manifests are available at: /var/log/packstack/20180216-138825-5jtw/handtests

5. If you encounter an error during this step, see Troubleshoot Installation in the appendix.
6. To add more compute nodes, see section Configure Additional Compute Nodes.
5.3 Initial Setup for OpenStack

When the installation is complete, we proceed with the initial setup of OpenStack. The setup can be performed both from the terminal by using APIs and commands and from the Horizon dashboard, which is the GUI for OpenStack:

1. The credentials and the user name are stored in the file `keystonerc_admin`, which is in the home directory.
2. Open the `keystonerc_admin` file.

   ```
   unset OS_SERVICE_TOKEN
   export OS_USERNAME=admin
   export OS_PASSWORD=1373deb64178a1691
   export OS_AUTH_URL=http://10.192.39.231:5000/v2.0
   export PSI='\\{\\d\\h \#(keystore_admin)\\}\\$
   export OS_TENANT_NAME=Devops NetApp
   export OS_REGION_NAME=RegionOne
   ```

3. Log in to the dashboard with the URL `http://<CONTROLLER IP>/dashboard` and enter the credentials from the `keystonerc_admin` file.

4. Optional: If you want to change the default password, see Reset Default Password in the appendix.

5.3.1 Setup Neutron Networking

This section describes setting up of networking in OpenStack using the Neutron Services.

1. We can use APIs from the terminal on the OpenStack controller node. But before this we need to source the `keystonerc_admin` file.

   ```
   source <Path to keystonerc_admin file>
   ```
2. Create networks in OpenStack either from the terminal or from the dashboard. In this step, we create an external network by using the APIs.

   ```
   neutron net-create external-net --router:external --provider:physical_network exnet --provider:network_type flat
   ```

   ```
   ```

3. Create a private network for the VMs to communicate with each other. Log in as admin and navigate to System > Networks > Create Network.

4. Click on the network that was created in the previous step under the Network Name tab.

5. Click on Create Subnet.

6. Enter the details of the subnet in the form.
7. Click on **Next**. Provide the DNS IP addresses to enable Internet access to the VMs.

8. Create an additional storage network through which the hosts and VMs can access shares from the dedicated storage node.

```
neutron net-create storage-net --router:external --provider:physical_network stgnet --provider:network_type flat
neutron subnet-create storage-net 192.168.50.0/24 --name storage-subnet --allocation-pool start=192.168.50.100, end=192.168.50.200
```
9. Create a new router. Navigate to **Project > Network > Routers > Create Router.**

![Create Router](image1)

10. Click on the router that was created in the previous step and navigate to the Interfaces tab. Click on **Add Interface.** Select the private subnet and provide the gateway IP for the subnet.

![Router Details](image2)

11. Repeat step 10 to create an interface for storage network.

**Tip**

The advantage of attaching a storage network interface to the Neutron router is that the instances need just a single interface from the private network to gain access to the Internet and the storage. With this step, you can mount shares that are created on your storage directly to your instance and you can give Internet access to the VMs. This interface does not provide SSH or RDP connection/access to the instances.

5.3.2 **Create Floating IPs**

Floating IPs can be created on the external network and can be attached to any instance. For Linux-based instances, this floating IP can be used to SSH from an external network, and for Windows
instances, this IP can be used to RDP. These IPs can be released and reassociated with another instance.

```
nova floating-ip-create <NAME OF THE EXTERNAL NETWORK>
```

For example:

```
nova floating-ip-create external-net
```

It can also be created using the Neutron Service.

```
neuton floatingip-create <EXTERNAL NETWORK>
```

### 5.3.3 Set up Security Groups

Security groups are necessary to enable or disable access to a particular type of protocol or ports. In the following examples, we add the rules for TCP, ICMP, and UDP packets, respectively.

```
nova secgroup-add-rule default tcp 1 65535 172.20.124.0/24
```

```
nova secgroup-add-rule default icmp -1 -1 172.20.124.0/24
```

```
nova secgroup-add-rule default udp 1 65535 172.20.124.0/24
```

**Best Practice**

It is not advisable to use 0.0.0.0/0 in enterprise deployments. Instead, provide the subnet IP address for which the following protocols must be enabled. If there are multiple subnets for which the following access might be required, then add separate rules for each subnet, similar to the ones in the examples.

### 5.3.4 Creating a Key Pair

The following commands show the steps to create a key pair to use for SSH to the VMs.

1. Create a directory.

```
mkdir /home/USER/.ssh
```

2. Create keys for SSH.

```
ssh-keygen -t rsa -N "" --f id_rsa
```

3. Move the keys to the folder created in step 1.

```
mv id_rsa /home/USER/.ssh
```

```
mv id_rsa.pub /home/USER/.ssh
```

4. Add the keypair using Nova services.

```
nova keypair-add myNewKey --pub-key /home/USER/.ssh/id_rsa.pub
```

### 5.3.5 Setup Glance with an NFS Share

Glance is the OpenStack image service that enables users to upload images and metadata definitions that are meant to be used with other services.

By default, Glance uses local storage of the server to store images and their metadata. Usually the local storage that is available on these servers might not be sufficient for larger images or for storing multiple snapshots (OpenStack snapshots) of the instances.
Best Practice

In such scenarios, NetApp recommends using an NFS share to store these data assets. To save more space, enable deduplication (volume efficiency) on a Glance share.

1. Create an NFS share on the SVM that you created in the earlier steps for the Glance service. NFS shares can be created from the NetApp OnCommand® System Manager or by SSH to the storage controller. For more information about creating and using an NFS share, see TR 4067: Clustered Data ONTAP NFS Best Practice and Implementation Guide.

2. Create a directory for these images at /var/lib/glance/images.

   ```bash
   mkdir Devops
   ```


   ```bash
   mount <SVM_LIF_IP>:/<GLANCE_SHARE_NAME> <PATH_TO_MOUNT_THE_SHARE>
   ```
   
   For example:

   ```bash
   mount 192.168.50.35:/GlanceVol /var/lib/glance/images/Devops
   ```

4. If you receive an error here, see Troubleshoot the Glance Service in the appendix.

5. Edit the glance-api.conf and specify the path to the mount from step 3.

   ```bash
   sudo vi /etc/glance/glance-api.conf
   ```
   
   Add the following line in the [glance_store] stanza and in the [DEFAULT] stanza.

   ```bash
   filesystem_store_datadir=<PATH_TO_THE_FOLDER_WHERE_SHARE_WAS_MOUNTED>/Devops
   ```
   
   For example:

   ```bash
   filesystem_store_datadir=/var/lib/glance/images/Devops
   ```

6. Make sure that the ownership of the mount point is with Glance. To change ownership, type `ls -l` and check whether the ownership has changed.

   ```bash
   chown glance:glance <PATH_TO_MOUNT_POINT>
   ```
   
   For example:

   ```bash
   chown glance:glance /var/lib/glance/images/Devops
   ```

7. Restart the Glance Service.

   ```bash
   chown glance:glance <PATH_TO_MOUNT_POINT>
   ```
8. Now you can upload an image to the Glance store.

```
curl http://cloud-images.ubuntu.com/trusty/current/trusty-server-cloudimg-amd64-disk1.img |
glance image-create --name=ubuntu --container-format=bare --disk-format=qcow2
```

9. If you receive an error here, see Troubleshoot the Glance Service in the appendix.

### 5.3.6 Setup Nova with an NFS Share

Nova is the OpenStack compute service that provides on-demand, scalable self-service access to the compute resources. Similar to the Glance service, described in section 5.3.5, the Nova service also uses local storage for storing disks of the instances (VMs).

#### Best Practice

To have a scalable infrastructure, NetApp recommends using an NFS share to store the root disks of instances. These shares can be expanded at any time as the need arises.

1. Mount the Nova share at this location: /var/lib/nova/instances.

   ```
   mount <SVM_LIF_IP>:/<NOVA_SHARE_NAME> <PATH_TO_MOUNT_THE_SHARE>
   ```

2. Edit the `nova.conf`.

   ```
   sudo vi /etc/nova/nova.conf
   ```

3. Make sure that the following lines are uncommented.

   ```
   instances_path=$state_path/instances
   state_path=/var/lib/nova
   ```

4. Make sure that the ownership of the mount point is with Nova. To change ownership, use:

   ```
   chown nova:nova /var/lib/nova/instances
   Type ls -1 and check whether the ownership has changed.
   ```

5. Restart the Nova services.

   ```
   openstack-service restart nova
   ```
5.3.7 Setup Cinder with an NFS Share

Cinder is the OpenStack block storage service that provides the running instances with persistent block storage. It has a pluggable driver architecture that eases the process creating and managing block storage devices.

This service also runs on the controller node in our architecture. Cinder can be used to create volumes that can be attached to any running instance. It can also be used to store root disks for instances instead of storing them by using the Nova service, as mentioned in section 5.3.6.

Best Practice

To have a scalable infrastructure, NetApp recommends using an NFS share to store the Cinder volumes. These shares can be expanded at any time as the need arises.

1. Create a new file, `nfs_share.conf`.
   ```bash
   vi /etc/cinder/nfs_share.conf
   ```

2. Define the NFS share in `/etc/cinder/nfs_share.conf` by using the following format:
   ```bash
   <SVM LIF IP>:/<ABSOLUTE PATH TO THE NFS SHARE>
   ```
   For example:
   ```bash
   192.168.0.10:/CinderShare
   ```

3. Set the root user and Cinder group as the owner of the file `shares.conf`.
   ```bash
   chown root:cinder /etc/cinder/nfs_share.conf
   ```

4. Set the file to be readable by members of the Cinder group.
   ```bash
   chmod 0640 /etc/cinder/nfs_share.conf
   ```

5. Now create a new definition for the NFS back end in `cinder.conf`.
   ```bash
   vi /etc/cinder/cinder.conf
   ```

6. Add it at the end of the file.
   ```bash
   [myNfsBackend]
   volume_backend_name=myNfsBackend
   volume_driver=Cinder.volume.drivers.netapp.common.NetAppDriver
   netapp_server_hostname=hostname
   netapp_server_port=80
   netapp_storage_protocol=nfs
   netapp_storage_family=ontap_cluster
   netapp_login=admin_username
   netapp_password=admin_password
   netapp_vserver=svm_name
   nfs_shares_config=/etc/cinder/nfs_share.conf
   ```

Best Practice

To have a scalable infrastructure, NetApp recommends using an NFS share to store the Cinder volumes. These shares can be expanded at any time as the need arises.

1. Create a new file, `nfs_share.conf`.
   ```bash
   vi /etc/cinder/nfs_share.conf
   ```

2. Define the NFS share in `/etc/cinder/nfs_share.conf` by using the following format:
   ```bash
   <SVM LIF IP>:/<ABSOLUTE PATH TO THE NFS SHARE>
   ```
   For example:
   ```bash
   192.168.0.10:/CinderShare
   ```

3. Set the root user and Cinder group as the owner of the file `shares.conf`.
   ```bash
   chown root:cinder /etc/cinder/nfs_share.conf
   ```

4. Set the file to be readable by members of the Cinder group.
   ```bash
   chmod 0640 /etc/cinder/nfs_share.conf
   ```

5. Now create a new definition for the NFS back end in `cinder.conf`.
   ```bash
   vi /etc/cinder/cinder.conf
   ```

6. Add it at the end of the file.
   ```bash
   [myNfsBackend]
   volume_backend_name=myNfsBackend
   volume_driver=Cinder.volume.drivers.netapp.common.NetAppDriver
   netapp_server_hostname=hostname
   netapp_server_port=80
   netapp_storage_protocol=nfs
   netapp_storage_family=ontap_cluster
   netapp_login=admin_username
   netapp_password=admin_password
   netapp_vserver=svm_name
   nfs_shares_config=/etc/cinder/nfs_share.conf
   ```
7. Add the NFS back end to the list of enabled back ends.

    enabled_backends=lvm,nfs

8. Restart the `cinder-volume` service.

    openstack-service restart cinder-volume

**Tip**

The name of the back end that is specified in this step should be the same as the name of the stanza in step 6. In this case, it’s `[nfs]`. Also, if you want to have multiple shares for Cinder, then insert one stanza as mentioned in step 6 for every share that you want to use.

9. View the existing file types.

    cinder type-list

10. Create a new volume type for the NFS back end.

    cinder type-create nfstype

11. Configure the `nfstype` volume type to use the NFS back end.

    cinder type-create nfstype set volume_backend_name=nfsbackend

12. Verify that the type was created and configured correctly.

    cinder type-list

    cinder extra-specs-list

13. Test whether the back end is working by creating a Cinder volume of size 1GB.

    cinder create --volume_type nfstype --display_name nfsVolume 1

14. If you receive an error, see *Troubleshoot the Cinder Service* in the appendix.

### 6 Configure Additional Compute Nodes

In a two-node installation, you can configure additional compute nodes by using the following steps:

1. Edit the Packstack answer file, which is usually stored in the home directory.

    vi /home/<answer_file>

2. If the Packstack answer file doesn’t exist, then generate the answer file by using the following command.

    sudo packstack --gen-answer-file=<ANSWER_FILE>

3. Within the answer file, provide the IP address of the node where you do not want to run Packstack. It’s usually the controller’s IP address, because we have already installed OpenStack on it.

    EXCLUDE_SERVERS=<LIST OF SERVERS TO BE EXCLUDED FROM INSTALLATION>

4. Provide the IP address of the nodes on which compute services are to be installed.

    CONFIG_COMPUTE_HOSTS=<COMMA SEPARATED LIST OF HOSTS TO INSTALL COMPUTE SERVICES>
5. Specify the interfaces for Nova compute and Nova network for the compute hosts.

```
CONFIG_NOVA_COMPUTE_PRIVIF=<INTERFACE NAME>
CONFIG_NOVA_NETWORK_PRIVIF=<INTERFACE NAME>
```

6. Run the Packstack script.

```
sudo packstack --answer-file=<PATH TO ANSWER FILE>
```

7. When the installation is complete, follow the steps in section 5.3.6 for every compute node.

### 7 Launch an Instance

An OpenStack VM is also known as an instance. Instances are provisioned on compute nodes and can be created by using either the CLI or the dashboard. We review both methods in this section.

#### 7.1 Launch an Instance from the Dashboard

1. Login to the dashboard.
2. Select the appropriate project from the top-left drop down menu.
3. On the Project tab, open the Compute tab and click on Instances. It now displays all the details pertaining to the instances. Click on Launch instance.

**Best Practice**

Configure one compute host at a time.

**Note:** This process of adding more compute nodes might vary if you are using some other mode of deployment. For more details about deployment options, see section 5.
4. Specify the details for the instance in the dialog box. In this example, we create an **ubuntu** instance.

- **Details tab:**
  a. **Availability Zone:** By default, the value is set to **nova**. If you have created additional availability zones, then choose the appropriate one.
  
  b. **Instance Name:** Assign a name to the VM.
  
  c. **Flavor:** Select the appropriate type for the instance.
  
  d. **Instance Count:** Specify the number of instances that should be created.
  
  e. **Instance Boot Source:** Various options are available to the user:
     - Boot from Image
     - Boot from Snapshot
     - Boot from Volume
     - Boot from Image (creates a new volume)
     - Boot from Volume Snapshot (creates a new volume)

   To learn more about these options, see the [OpenStack guide](#).
  
  f. **Image Name:** Select the image name from the drop-down list. Recall that we had initially uploaded an Ubuntu image.
- **Access & Security tab:**

  ![Launch Instance](image)

  a. Key Pair: Select the key pair that was created in [section 5.3.4](#).
  
  b. Security Groups: Select the appropriate security group. In this case, we have only Default.

- **Networking tab:**

  Select the network interfaces for the instance.

  ![Launch Instance](image)

- **Post-Creation tab:**

  If you want to run a script after the instance is launched, then use this tab.

- **Advanced Options tab:**

  You can select the type of disk partition, Automatic or Manual.
5. Now click on Launch. Your instance starts on the compute node.

6. If you encounter an error, see Troubleshoot the Nova Service in the appendix.

### 7.2 Launch an instance from CLI

1. SSH to the OpenStack controller node.

2. Source the keystonerc_admin file.

   ```
   Source <Path to Keystonerc_admin file>
   ```

3. Get the list of available images.

   ```
   glance image-list
   ```

4. Get the list of available networks.

   ```
   neutron net-list
   ```

5. Get the flavor list.

   ```
   nova flavor-list
   ```

6. Get the list of available hypervisors (this step is optional).

   ```
   nova hypervisor-list
   ```

7. Boot an instance.

   ```
   nova boot --image <IMAGE ID> --nic net-id=<NETWORK ID> --flavor <FLAVOR ID> --key-name <NAME OF THE KEY> --availability-zone <ZONE>:<HYPERVISOR NAME> <INSTANCE NAME>
   ```

Tip

The `--availability-zone` parameter is optional. This option forces the instance to be created on a specific compute node.

For the key name, use the key created in section 5.3.4.
8. If you receive an error here, see Troubleshoot the Nova service in appendix.

7.3 Login to an Instance
To log in to a newly created instance, associate a floating IP to the instance. You can do it either through the dashboard or through the CLI.

7.3.1 Associate a Floating IP from the Dashboard
1. Log in to the dashboard.
2. From the Project tab, open the Compute tab and click on Instances.
3. In the Actions column on the dashboard, open the drop-down list that is specific to the instance.
4. Click on Associate Floating IP. If a floating IP has already been created, select it from the drop-down list. If not, click the plus sign (+) to create a new floating IP.
5. Click on Associate. If you encounter an error, see Troubleshoot the Neutron Service in the appendix.

7.3.2 Associate a Floating IP from the CLI
1. Determine whether a floating IP exists.

```bash
neutron floatingip-list
```
2. If no floating IP exists, create a new floating IP.

```bash
neutron floatingip-create <EXTERNAL NETWORK>
```
3. If a floating IP does exist, attach it to the instance. If you receive an error, see Troubleshoot the Neutron Service in the appendix.

```
nova floating-ip-associate <INSTANCE NAME> <FLOATING IP>
```

For example:
```
nova floating-ip-associate myInstance 10.192.20.101
```

### 7.3.3 Change the password for a Linux Instance

After you SSH into the Linux instance, set the root password and the user account password, if they were not already set.

1. To set the root password, use the following:

```
sudo su
```

2. Type `passwd` and press Enter. When prompted, enter the new password.

```
passwd
```

3. To set password to a specific user created on the instance, use the following.

```
passwd <USER NAME>
```

In this example, `ubuntu` is the user name.

```
ubuntu@myinstance:~$ sudo su
sudo: unable to resolve host myinstance
root@myinstance:/home/ubuntu# passwd
Enter new UNIX password:
Retype new UNIX password:
passwd: password updated successfully
root@myinstance:/home/ubuntu# passwd ubuntu
Enter new UNIX password:
Retype new UNIX password:
passwd: password updated successfully
root@myinstance:/home/ubuntu# 
```

### 8 Attaching Cinder Volumes to an Instance

Cinder volumes can be attached to any instance. These volumes can also be extended even after they are created. Cinder Volumes can be created from the CLI or the dashboard.

#### 8.1 Create a Cinder Volume from CLI

Refer to section 5.3.7 to create a Cinder volume by using the CLI.

#### 8.2 Create a Cinder Volume from the Dashboard

1. From the Project tab, open the Compute tab and click on Volumes.
2. On the dashboard, click on Create Volume.
3. Fill in all the necessary details on the form.

4. Click on Create Volume.

8.3 Attach cinder volume to an Instance

1. Open the drop down list and click on Manage Attachments.
2. Select the instance to which the volume must be attached and then click on Attach Volume.
Appendix

This section will cover various troubleshooting techniques as well as some more tips and tricks to get OpenStack environment up and running.

We will begin with troubleshooting a few installation errors.

Troubleshoot Installation

In this section, we discuss a few errors that you might encounter during installation, and we present their possible solutions.

1. **Error:** Requires python-docutils
   
   ```
   Error: Package python-docutils is missing
   Requires: python-docutils
   
   sudo yum install python-docutils
   ```

2. **Error:** Requires dnsmasq-utils
   
   ```
   Error: Package: dnsmasq-utils is missing
   Requires: dnsmasq-utils
   
   sudo yum install dnsmasq-utils
   ```

Open the file to find the full trace at the location that is displayed on the terminal. Search for the error in the file.
In this case, we notice the following error.

```
Error: Execution of /usr/bin/yum -d 0 -y install openstack-neutron returned 1: Error: Package: openstack-neutron-7.0.1-1.i386 (openstack-liberty)
```

Solution: Follow the same procedure as for the preceding error.

Then use the following.

```
sudo yum install dnsmasq-utils
```

### Troubleshoot NFS Shares

1. **Error:** Access denied to the share  
   
   **Probable cause:** The export policies are not set correctly in ONTAP 9.  
   
   **Solution:** Set the export policies so that the controller node has access to the share:

   a. Log in to OnCommand System Manager.  
   b. Open the Storage Virtual Machine tab and click on your SVM.  
   c. Go to Policies > Export Policies.  
   d. Select the policy that was applied to the NFS share and edit it.  
   e. Specify the subnet in the Client Specification on which the LIF for SVM exists. Use `0.0.0.0/0` to allow share access to all the subnets.  
   f. Make sure that this subnet is accessible by the controller.

### Best Practice

One out of the three interfaces on the controller node must be on the same network as that of the SVM LIF. This interface is part of the storage network that was created earlier.

g. Optional: Enable Kerberos 5 and Kerberos 5i.
Troubleshoot the Glance Service
To check the log files for the Glance service, go to /var/log/glance/.

1. **Error:** StoreAddDisabled

   ![Error Log]

   **Probable cause:** The path to the Glance store was set incorrectly. The ownership of the folder where the NFS share was mounted is not set to the Glance user.

   **Solution:**
   b. Add the following lines in the `[DEFAULT]` stanza and uncomment the same line from the `[glance_store]` stanza.

   ```
   filesystem_store_datadir=<PATH TO THE GLANCE SHARE/MOUNT POINT>
   ```
   c. Restart the Glance service.
   d. Set the ownership of the mount point to `glance`.

   ```
   chown glance:glance <PATH TO THE GLANCE SHARE>
   ```

Troubleshoot the Nova Service
To check the log files for the Nova service, go to /var/log/nova/.

Check the Nova log files in the node on which the instance was installed.

1. **Error:** libvirt:Error: Cannot access storage file `/var/lib/instances/<disk id>/disk':No such file or directory.

   ![Error Log]

   **Probable cause:** The ownership of the folder where the NFS share was mounted is not set to the Nova user.
**Solution:** Set the ownership of the mount point to `nova`. (See [section 5.3.6](#).)

```bash
chown nova:nova /var/lib/nova/instances
```

2. **Error:** Cannot delete a VM that is running or is in the shutoff state.

```
ERROR: Cannot 'forceDelete' while instance is in vm_state error (HTTP 409)
```

In this scenario, the instances cannot be deleted even with the `force-delete` parameter.

**Probable cause:** We observed that if the resource usage has reached the set limit or quota, the Nova service fails to delete a VM in the shutoff or the running state.

**Solution:** Increase the resource usage limits or quotas to slightly higher than the previously set values so that you can delete VMs in such a state:

a. Login to the dashboard.

b. Go to **Identity > Projects**.

c. Click on the project under which the instance was created.

d. Open the drop-down list on the top-right corner and select **Modify Quotas**.

e. Modify the values in the following fields: **Instances and Volumes**.

   Also modify any other value that you believe might be required in the future.
f. Restart the Nova services and try to delete the instance again.

3. **Error:** Nova-Compute state down.
   
   State of the nova compute service is XXX.

   **Probable cause:** The rabbitmq server has not been updated yet. It is also possible that the NFS share that was mounted for the Nova service is not accessible.

   **Solution:**
   
   a. Check the state of all the Nova services.

   ```
   nova service-list
   ```

   b. Try to ping the IP of the LIF, where the NFS Share exists.

   c. Try to open the folder where NFS share was mounted on the compute/controller node.

   ```
   cd /var/lib/nova/instances
   ```

   d. If you are unable to cd (change directory) to the folder or if its hung up then it implies that the share is not accessible.

   e. Check your storage network for any problems. Also, check whether the compute nodes are able to access the Nova NFS Share.

   f. After it has been resolved restart all the Nova Services.
4. **Error:** Unable to connect to the vnc console using dashboard.

   <Instance Name> failed to connect to server code:1006.

   **Probable cause:** The Nova service is not configured properly for the VNC server.

   **Solution:**
   a. SSH to the controller.
   b. Source the keystonerc_admin file.
   c. Edit the nova.conf file at /var/lib/nova/nova.conf.
   d. Edit the following values in the [DEFAULT] stanza.

   ```
   vncserver_proxyclient_address=<CONTROLLER MANAGEMENT IP>
vncserver_listen=0.0.0.0
novncproxy_base_url=http://<CONTROLLER MANAGEMENT IP>:6080/vnc_auto.html
vnc_enabled=True
   ```
   e. Save the file and restart the Nova services.
   f. Try opening the console again.

5. **Error:** Instance creation fails with **ERROR** status.

   Instance creation fails with No valid host being found.

   **Probable cause:** The Nova scheduler was unable to find a host with sufficient resources to provision the instance.

   It is also possible that while creating the instance from the CLI by using the `nova boot` command, an incorrect hypervisor name was provided with the `--availability-zone` parameter.

   Another possibility is that the credentials in the keystonerc_admin file are incorrect, and so it fails to authenticate against Keystone.

   **Solution:**
   a. Check for available resources on your compute nodes. Verify that the flavor size selected for the instance is within the available resources.
   b. Check the CLI command for an incorrect hypervisor name. Use `nova hypervisor-list` to get the correct names for the hypervisor.
   c. Edit the keystonerc_admin file and check the credentials. If you have modified the credentials by using the dashboard, be sure to change the credentials in the keystonerc_admin file, too.

**Troubleshoot the Neutron Service**

To check the log files for the Neutron service, go to `/var/log/neutron/`.

Check the Neutron log files in the node on which the instance was installed.

1. **Error:** No ports are available to associate floating IPs.

   **Probable cause:** Sometimes the Horizon dashboard fails to fetch all the ports that are available for the instance.

   This error might also occur if the interface that is connected to the external network on the router is not accessible or is unavailable.

   **Solution:**
   a. Try associating by using the CLI:
      i. SSH to the controller.
      ii. Source the keystonerc_admin file.
iii. Get the ID of the instance to which the floating IP should be associated.

```
nova list
```

iv. Get the list of ports that are present on the instance.

```
neutron port-list --device-id <ID OF THE VM FROM STEP (iii)>
```

v. Create a new floating IP.

```
neutron floatingip-create <EXTERNAL NETWORK>
```

vi. Get the ID of the floating IP that you created in step v.

```
neutron floatingip-list
```

vii. Associate the floating IP to the instance.

```
neutron floatingip-associate <ID OF THE FLOATING IP FROM STEP (v)> <ID OF THE PORT FROM STEP (iv)>
```

b. If you receive the following error, proceed to the next step:

i. Go to Admin > System > Routers. Click on the router and delete the router’s external interface.

```
ERROR: External Network <ID OF EXTERNAL NETWORK> is not reachable from <SUBNET ID>. Therefore cannot associate port <PORT ID> with a floating IP.
```

ii. Go to Project > Compute > Access & Security.

iii. Click on the Floating IPs tab and release all the floating IPs.

iv. Go to Project > Network > Routers.

v. Click on Set Gateway in the Actions Column. This action creates a new interface on the router.

vi. Try to Ping this gateway IP. If you are able to ping the IP, try associating the floating IP again.

vii. If you are still unable to ping, verify that your external network (not the OpenStack network) is configured appropriately.

**Troubleshoot the Cinder Service**

To check the log files for the Cinder service, go to `/var/log/cinder/`.  

1. **Error:** Cannot delete or force-delete the Cinder volume.  
   The Cinder volume is Attached to None.
Probable cause: The instance was deleted by Nova before the Cinder volume could be detached.

Solution:

a. SSH to the controller node.
b. Source the `keystonerc_admin` file.
c. Get the list of Cinder volumes.

cinder list
d. Try to reset the state of the Cinder volume from the `Error_deleting` or `Detaching state` to `available`.
cinder reset-state --state available <VOLUME ID FROM STEP(c)>
e. If the state is reset, try deleting the volume. If step d fails, proceed to the next step.
f. Login to the mysql db for Cinder.

mysql cinder
g. Set the volume state by using the SQL query.

```
UPDATE volumes SET attach_status='detached',status="available" WHERE id='"<VOLUME ID FROM STEP(3)>'
```
h. Try to delete the volume again.
i. If you still cannot delete it, use the following query. NetApp does not recommend this step in all scenarios.

```
UPDATE volumes SET deleted=1,status='deleted',deleted_at=now(),updated_at=now() WHERE deleted=0 and id='"<VOLUME ID>'
```
2. **Error**: Cinder Volumes are created with the **ERROR** status.

   **Probable cause**: The NFS back end for the Cinder service was not configured properly.

   **Solution**:
   a. SSH into the controller node.
   b. Source the `keystonerc_admin` file.
   c. Find out which Cinder services are running.

   ```
   cinder service-list
   ```
   d. If the `cinder-volume` service for the back end is **disabled** or **down**, edit the `cinder.conf` file.

   ```
   vi /etc/cinder/cinder.conf
   ```
   e. Add the back-end name to the `enabled_backends` list. In this example, `nfs` is the back-end name.

   ```
   enabled_backends=lvm,nfs
   ```
   f. Check whether the stanza for the back end is configured appropriately. See section 5.3.7.
   g. Also check whether the `nfs_shares.conf` file that was created in section 5.3.7 has the correct details.
   h. Restart the Cinder services.

**Reset the Default Password**

The default password for logging in to the dashboard is found in the `keystonerc_admin` file. To modify this password, follow these steps:

a. Log in to the dashboard by using the default user name and password. (See section 5.3)

b. Click on **Admin** on the top right corner of the dashboard.

c. Click on **Settings**. Then click on **Change Password**.

d. Enter your new password and **save** it.

e. SSH to the controller node.

f. Edit the `keystonerc_admin` file, remove the old password, and type the new password.

g. Save this file. From now on, use this file to “source” when you use the CLI.

**Configure Network Interfaces**

a. Edit the configuration file for the interface that is connected to your LAN

   ```
   vi /etc/sysconfig/network-scripts/<INTERFACE NAME>
   ```
For example:

```
vi /etc/sysconfig/network-scripts/ifcfg-eno2
```

b. Set the following details for the external interface:

```
TYPE=Ethernet
BOOTPROTO=static
IPADDR=<IP ADDRESS>
NETMASK=255.255.255.0
GATEWAY=<GATEWAY>
DEFROUTE=yes
PEERDNS=yes
PEERROUTES=yes
NAME=eno2
UUID= 6da51dbd-fe53-4a7c-90da-27a9dd8cb373
DEVICE=eno2
ONBOOT=yes
DNS1=<DNS IP ADDRESS FOR LAN>
DNS2=<ADDITIONAL IP ADDRESS FOR LAN>
```

c. Similarly, set the following information for private and storage network interfaces:

```
TYPE=Ethernet
BOOTPROTO=static
IPADDR=<IP ADDR FOR STORAGE/PRIVATE NETWORK>
NETMASK=255.255.255.0
DEFROUTE=no
NAME=enp17s0f1
UUID=6fa51dbd-fe53-4a7c-5f2b-57a9db8cd30db
DEVICE=enp17s0f1
ONBOOT=yes
```

d. Restart the network services.

```
service network restart
```

Acknowledgments

The authors of this document would like to thank and acknowledge the following persons, who had a positive impact on the quality of this document:

- Akshai Parthasarathy, Technical Marketing Engineer
- Igor Feoktistov, Hosting Architect
- Manasi Prabhavalkar, Systems Architect

References

The following references were used in this TR:

- Theory of Operation and Deployment Choices
  

- Red Hat Image Service
  

- OpenStack Installation Guide for Red Hat Enterprise Linux and CentOS
  
  [http://docs.openstack.org/liberty/install-guide-rdo/](http://docs.openstack.org/liberty/install-guide-rdo/)

- Red Hat Enterprise Linux OpenStack Platform product documentation

- Red Hat Enterprise Linux OpenStack Platform 5 on NetApp Clustered Data ONTAP
- Multiple store locations for Glance images
  http://egonzalez.org/multiple-store-locations-for-glance-images/

## Version History

<table>
<thead>
<tr>
<th>Version</th>
<th>Date</th>
<th>Document Version History</th>
</tr>
</thead>
<tbody>
<tr>
<td>Version 1.0</td>
<td>July 2016</td>
<td>Initial release</td>
</tr>
</tbody>
</table>