Reference Architecture for Deploying CDH 5.x On Red Hat OSP 11

Important Notice

Cloudera, Inc.
1001 Page Mill Road, Building 2
Palo Alto, CA 94304-1008
info@cloudera.com
US: 1-888-789-1488
Intl: 1-650-843-0595
www.cloudera.com

Release Information

Date: 7/10/17

Version: 5.12

Executive Summary

This document provides a reference architecture for deploying Cloudera Enterprise including CDH on Red Hat's OSP 11. Much like the Hadoop platform, OpenStack is comprised of a number of related projects to control pools of storage, processing, and networking resources within a data center, and to build a multi-datacenter private cloud infrastructure. The following OpenStack projects are in scope for this release of the reference architecture:

- Compute (Nova): on-demand computing resources from a large network of virtual machines
- Storage Service (Cinder): storage management and provisioning for Cloudera Instances while maintaining data locality
- Networking (Neutron): flexible models for managing networks and IP addresses (includes Open vSwitch)
- Image service (Glance): discovery, registration, and delivery for disk and virtual machine images
- Identity Management service (Keystone): Manage identity and authorizations for various system users, projects and end-users who will use the OpenStack self-service infrastructure

This release of the reference architecture is for deploying Cloudera's Distribution of Apache Hadoop (CDH) 5.11 on Red Hat OSP 11. This reference architecture articulates a specific design pattern which is recommended to be administrator-driven as opposed to end-user self-service based. The RA will also be applicable for all 5.x releases of CDH subsequent to C 5.11.

Other OpenStack projects, such as telemetry and alerting (Ceilometer), monitoring (Horizon), elastic mapreduce (Sahara), Orchestration (Heat), and bare metal (Ironic), are considered out of scope for this release of the reference architecture.

Also out of scope are Object Storage services (Swift and Ceph) and the Software Defined distributed storage platform (Ceph).

Future editions may include more information on these OpenStack projects.

Business Objectives

The objective of this Reference architecture is to provide safe and reliable design patterns that customers can use to leverage OpenStack to deploy Cloudera EDH IaaS clusters in private cloud environments.

Cloudera Enterprise

Cloudera is an active contributor to the Apache Hadoop project and provides an enterprise-ready, 100% open-source distribution that includes Hadoop and related projects. The Cloudera distribution bundles the innovative work of a global open-source community, including critical bug fixes and important new features from the public development repository, and applies it to a stable version of the source code. In short, Cloudera integrates the most popular projects related to Hadoop into a single package that is rigorously tested to ensure reliability during production.

Cloudera Enterprise is a revolutionary data-management platform designed specifically to address the opportunities and challenges of big data. The Cloudera subscription offering enables data-driven enterprises to run Apache Hadoop production environments cost-effectively with repeatable success. Cloudera Enterprise combines Hadoop with other open-source projects to create a single, massively scalable system in which you can unite storage with an array of powerful processing and analytic frameworks—the Enterprise Data Hub. By uniting flexible storage and processing under a single management framework and set of system resources, Cloudera delivers the versatility and agility required for modern data management. You can ingest, store, process, explore, and analyze data of any type or quantity without migrating it between multiple specialized systems.

Cloudera Enterprise makes it easy to run open-source Hadoop in production:

Accelerate Time-to-Value

- Speed up your applications with HDFS caching
- Innovate faster with pre-built and custom analytic functions for Cloudera Impala

Maximize Efficiency

- Enable multi-tenant environments with advanced resource management (Cloudera Manager + YARN)
- Centrally deploy and manage third-party applications with Cloudera Manager

Simplify Data Management

- Data discovery and data lineage with Cloudera Navigator
- Protect data with HDFS and HBase snapshots
- Easily migrate data with NFSv3 support

See Cloudera Enterprise for more detailed information.

Cloudera Enterprise can be deployed in a Red Hat OpenStack Platform based infrastructure using the reference architecture described in this document.

About Red Hat

Red Hat is the world's leading provider of open source software solutions, using a community-powered approach to reliable and high-performing cloud, Linux, middleware, storage, and virtualization technologies. Red Hat also offers award-winning support, training, and consulting services. As a connective hub in a global network of enterprises, partners, and open source communities, Red Hat helps create relevant, innovative technologies that liberate resources for growth and prepare customers for the future of IT.

About Red Hat OpenStack Platform

Red Hat OpenStack Platform allows customers to deploy and scale a secure and reliable private or public OpenStack cloud. By choosing Red Hat OpenStack Platform, companies can concentrate on delivering their cloud applications and benefit from innovation in the OpenStack community, while Red Hat maintains a stable OpenStack and Linux platform for production deployment.

Red Hat OpenStack Platform is based on OpenStack community releases, co-engineered with Red Hat Enterprise Linux 7. It draws on the upstream OpenStack technology and includes enhanced capabilities for a more reliable and dependable cloud platform, including:

- Red Hat OpenStack Platform director, which provides installation, day-to-day management and orchestration, and automated health-check tools, to ensure ease of deployment, long-term stability, and live system upgrades for both core OpenStack services, as well as the director itself.
- High availability for traditional business-critical applications via integrated, automated monitoring and failover services.
- Stronger network security and greater network flexibility with OpenStack Neutron modular layer 2 (ML2), OpenvSwitch (OVS) port security, and IPv6 support.
- Integrated scale-out storage with automated installation and setup of Red Hat Ceph Storage.
- A large OpenStack ecosystem, which offers broad support and compatibility, with more than 350 certified partners for OpenStack compute, storage, networking, and independent software vendor (ISV) applications and services.

Audience and scope

This reference architecture is aimed at Datacenter, Cloud, and Hadoop architects who will be deploying Cloudera's Hadoop stack on private OpenStack cloud infrastructure.

This release of the reference architecture is for deploying Cloudera's Distribution of Apache Hadoop (CDH) 5.11 on Red Hat OSP 11. This reference architecture articulates a specific design pattern which is recommended to be administrator-driven as opposed to end-user self-service based. The RA will also be applicable for all 5.x releases of CDH subsequent to C 5.11.

Other OpenStack projects, such as telemetry and alerting (Ceilometer), monitoring (Horizon), elastic mapreduce (Sahara), Orchestration (Heat), and bare metal (Ironic), are considered out of scope for this release of the reference architecture.

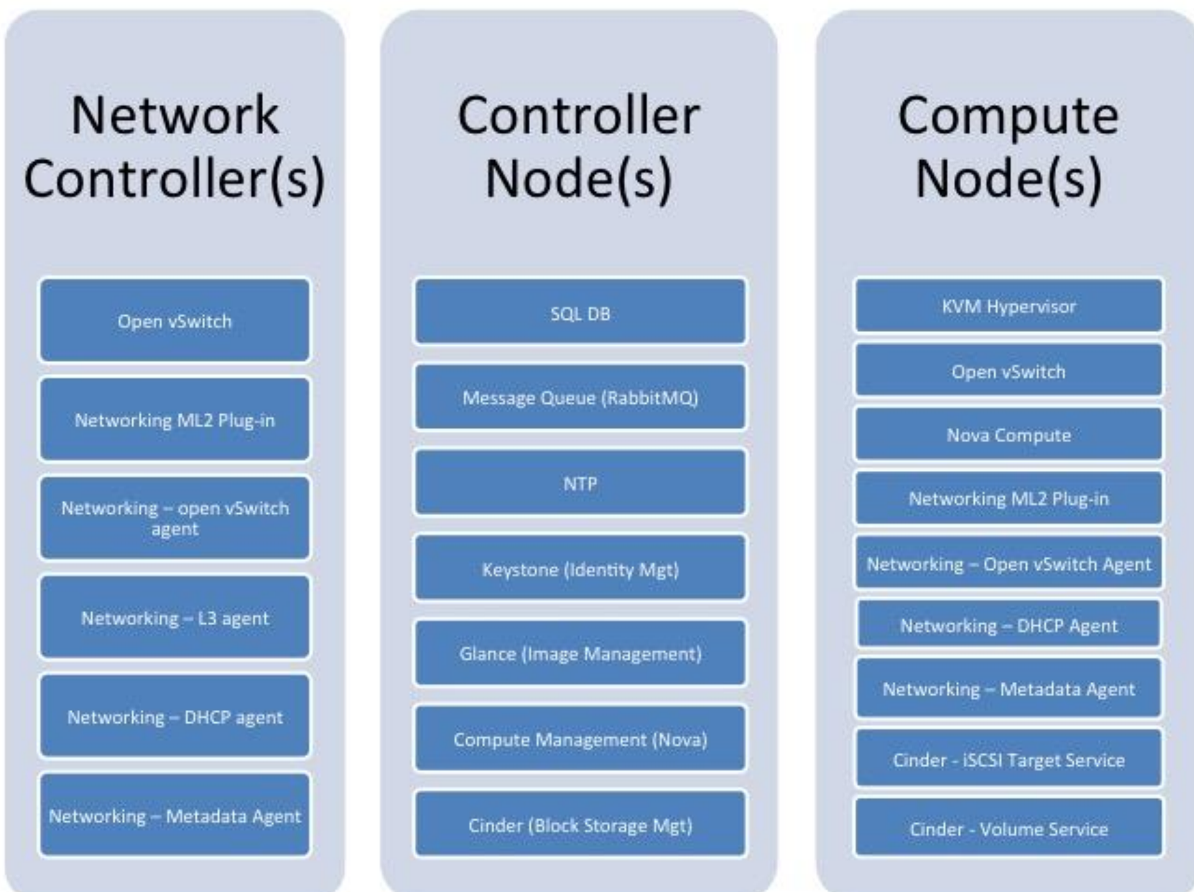
Also out of scope are Object Storage services (Swift and Ceph) and the Software Defined distributed storage platform (Ceph).

Future editions may include more information on these OpenStack projects.

Reference Architecture

Component design

The following diagram illustrates the various components of the OpenStack deployment. Not all the components shown in this high level diagram are covered in this reference architecture document. Please refer to the **Audience and Scope** section - it highlights which components are considered in scope and which are considered out of scope for this revision.



High level block diagram 1

Component Table

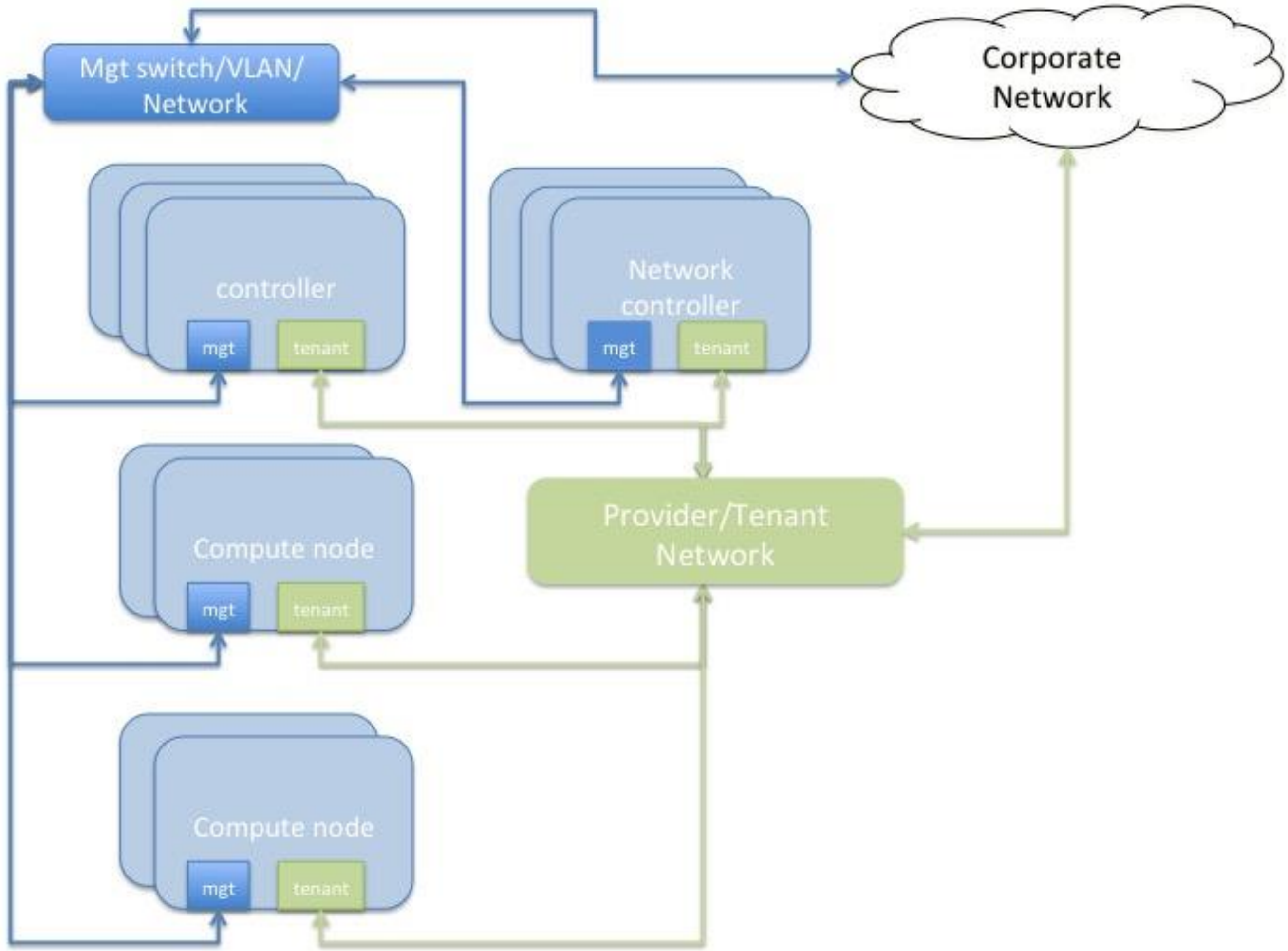
Component Role	Quantity	Component Details	Description

OpenStack Controller	3	• 2-sockets with 6-10 cores per socket
		• 128GB RAM
		• 2 x 10GbE NICs
		  ○ 1 x 10GbE for Compute/Tenant network
		  ○ 1x 10GbE for Mgmt network
		• 6 x 2TB+ internal HDDs
		  ○ 4 x drives in RAID-10 configuration for various Databases
		  ○ 2 x drives in RAID-1 for OS bits
			Set up 3 controller nodes in HA configuration. This will ensure that the various key components of the OpenStack deployment will continue to run in case of a hardware failure

Compute Node	Minimum 8, max Depends on use-case.	• 2-sockets with 6-10 cores per socket
		• At least 256GB RAM
		• 2 x 10GbE NICs
		  ○ 1 x 10GbE Tenant network interfaces
		  ○ 1 x 10GbE Management network interface
		• 12-24 2TB+ internal HDDs
		  ○ 2 x HDDs in RAID-1 for OS bits
		  ○ All other spindles in JBOD mode, to be presented as Cinder LVM backends. Details provided in Storage configuration section of this document.
			A minimum of 3 Master and 5 worker nodes (CDH) are needed to ensure that when HDFS blocks are placed within VMs running on these nodes, we have physical disparity to match the 3x replication factor of HDFS. We will use HVE to ensure that duplicate copies of any HDFS block are not placed on the same compute node. But there need to be at least the physical availability of 8 compute nodes.

Network

This section covers the network topology used in development of this reference architecture, as well as a brief summary of options available in the OpenStack ecosystem in general. A generic guideline for networking would be to advise the customers to pick a model that yields highest network throughput, or at least sufficient network throughput to match the theoretical throughput capabilities of the disks being presented to the VMs on each physical node.



Network topology diagram 2

a. Controller and compute nodes have 2 x 10GbE NICs each - one will provide the tenant network, the other is the management network which is used for OS provisioning of the physical infrastructure, as well as provide data path for other OpenStack management traffic.

b. There are two general flavors of network topology that can be used in an OpenStack based private cloud.

1. Provider Networks -- Provider Networks are essentially physical networks (with physical routers) and are managed by the OpenStack administrators. End-users cannot manage and make changes to these networks. They are the simplest and also the most performant. They entail connecting directly to the physical network infrastructure with minimal SDN (Software Defined Networking) functionality being used.
2. Self-Service networks -- These are networks that can be created and managed by the OpenStack end-users. The underlying physical infrastructure can be provider networks, but there would a virtualized overlay using VXLAN or GRE tunneling. These would typically be private

networks which will be routed through a software router hosted on a network controller node.

NOTE:

- In our labs, we have used a provider-network based deployment, which provides better network performance for Hadoop workloads, wherein each compute node is able to directly access the physical network infrastructure. This model is however limiting in terms of flexibility in scenarios where self-service capabilities are needed.
- For best network performance, consider using SR-IOV. This will allow the VMs to directly access previously defined virtual functions created on physical NICs. This option is further limiting in terms of flexibility, and the NIC hardware is subject to supportability on Red Hat OSP.
- For a more detailed understanding of the various networking options available in Red Hat OSP 11, refer to the Networking Guide.

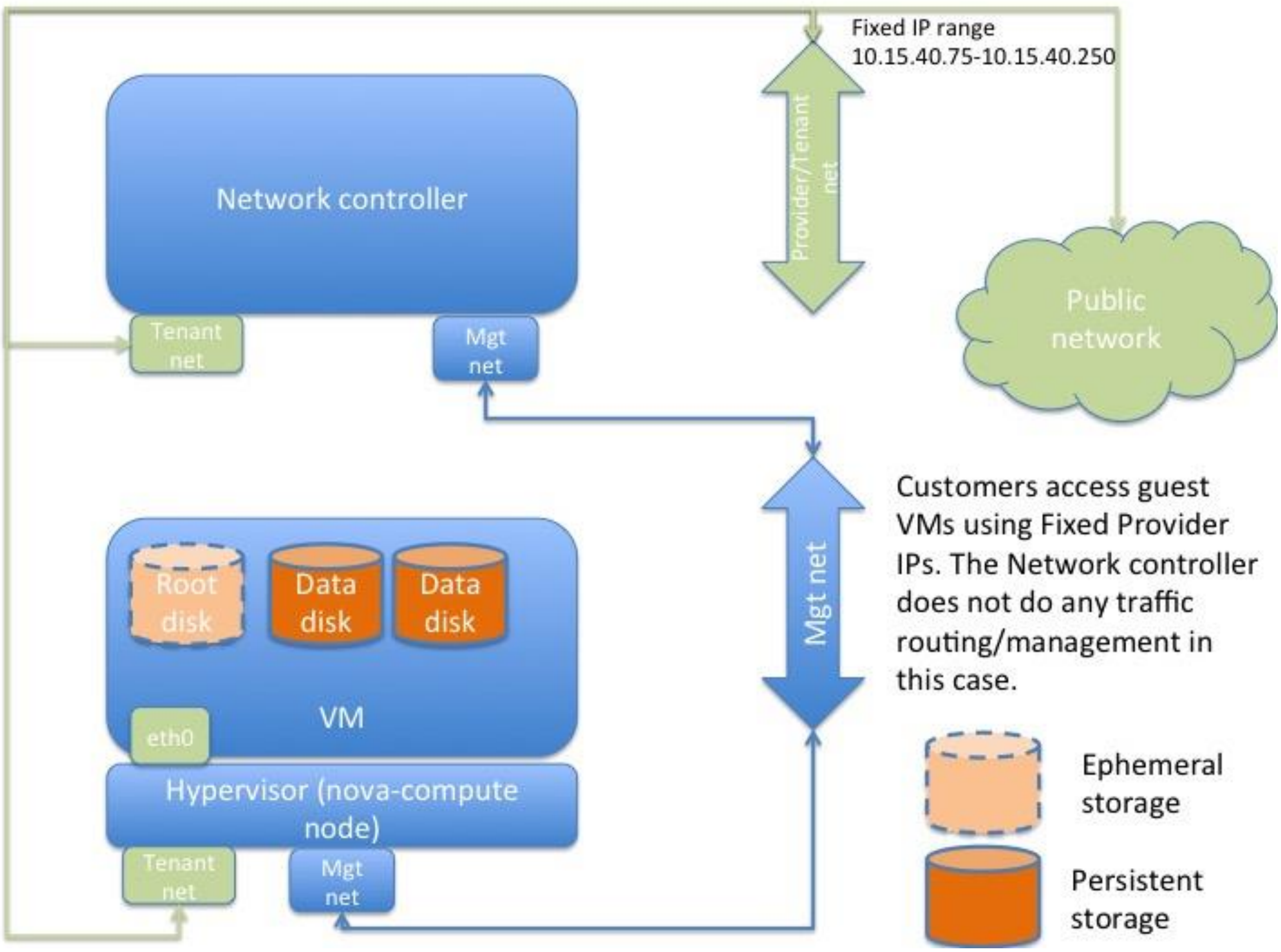
Compute (Nova)

The compute nodes' design considerations are as follows -

a. The hypervisor (KVM/QEMU)

b. The instance storage location - Let there be sufficient storage capacity in /var/lib/nova/ to house the ephemeral root disks

c. other considerations if applicable - such as appropriate drivers for network and storage, etc for optimal performance.



Logical instance diagram 3

Over Commitment Ratio

OpenStack's default over-subscription ratio (OSR) of CPU is 16:1 and Memory is 1.5:1. For Hadoop workloads we recommend setting the CPU OSR to 1:1 and Memory OSR to 1:1. Do not over-commit either of the resources. Hadoop workloads are very CPU and memory heavy, besides being IO and Network intensive; they will push the boundaries on all the subcomponents of your infrastructure.

Set the following in /etc/nova/nova.conf on all nodes running Nova-compute --

```
cpu_allocation_ratio = 1
ram_allocation_ratio = 1
```

Instance Types/Flavors

Red Hat OSP 11 does not have instance flavors defined out of the box. Therefore, consider crafting some custom ones that make sense for Hadoop workloads.

We have provided some guidance towards reasonable flavors. These are dependent on the workloads being run on Cloudera EDH.

Instance Flavors Table

Name	RAM (MB)	Disk (GB)	Ephemeral (GB)	VCPUs
cdh-tiny	1024	10	10	1
cdh-quartersize	56320	100	0	9
cdh-halfsize	122880	100	0	18
cdh-fullsize	225280	100	0	36

The number of vCPUs to allocate will depend on the number of cores per Socket.

NOTE:

- The flavor configurations are provided here as guidelines. Depending on the use case, the customer should adjust the size of CPUs and Memory. Typically it is recommended to make the instances larger in size and along CPU socket boundaries. Memory sizes will be predicated by the number of applications and types of services that will be running in the cluster.
- The general guidance for CPU allocation is to maintain 1:1 HT core to vCPU ratio. Similarly for RAM, guidance is to maintain 1:1 Physical to Virtual Memory allocation ratio. However, 1-2 cores and about 32GB of RAM should be left reserved for the hypervisor OS.
- Customers are advised to work with their Cloudera Account teams to determine the best instance flavors applicable to their environments, based on their existing or proposed workloads.
    - It is a good idea to keep minimum supportable configurations in mind while defining these flavors. For instance, Cloudera's MPP component - Impala has a minimum requirement for 128GB, and ideally at least 256GB of RAM.

The root disk should be at least 100GB, preferably > 200GB, such that we have sufficient logging space in the "/var" mountpoint/directory.

Warning:

It is better to have larger root disks or mount a cinder volume with sufficient storage capacity to handle multiple copies of the system and various CDH component logs under the /var/log mount point. Cloudera recommends larger root disks and separating /var/log to a dedicated mountpoint.

Orchestration

We do not have any specific orchestration rules or recommendations for Hadoop instances. There is no benefit to migration/live migration or storage migration of the instances. Moreover, the design patterns presented in this reference architecture will naturally prevent the VMs from being mobile. Each VM will reside in a dedicated availability zone.

Red Hat OSP Director provides automation for the OpenStack platform build (aka the Overcloud), and Red Hat OSP deployments in production are supported only when deployed with the Director.

In order to automate the deployment of Guest VMs and associated virtual infrastructure (such as networks, subnets, etc) as well as the application, various tools can be leveraged.

- The openstack ecosystem includes Heat, which allows for templating VMs and automating OSP infrastructure and guest VM deployment.
- There are guides available in the public domain that articulate how to leverage popular tools such as Ansible, Vagrant, Foreman, Chef, Puppet, to fully automate lifecycle management of OpenStack infrastructure as well.

For the Cloudera Enterprise application deployment, the Cloudera Manager API is a very popular option and all Cloudera build automation is done using the CM API. Some relevant URLs are provided in the References section of this document.

A more detailed discussion on this topic is out of scope for the current version of this document.

Storage

Following table summarizes what might be considered reasonable performance characteristics of the various storage components

Storage Performance Profile Table

Storage Component	Response time/latency (ms)	Minimum Acceptable Throughput (MB/s)	Comments
Ephemeral	< 20	30-40	Since ephemeral storage is on the local disk of the compute nodes, the storage subsystem needs to be tuned such that we can attain the acceptable minimum performance numbers as shown.
Cinder Local-Storage over iSCSI	< 30	40-60	

Ephemeral Storage

Ephemeral storage in OpenStack allows you to associate disks to a Nova compute instance. These disks are ephemeral, meaning that they are effectively deleted when the Nova instance is terminated.

There is only one type of workload that will run on ephemeral storage -- the Operating System disks of the Nova instances, which are predominantly random read/write.

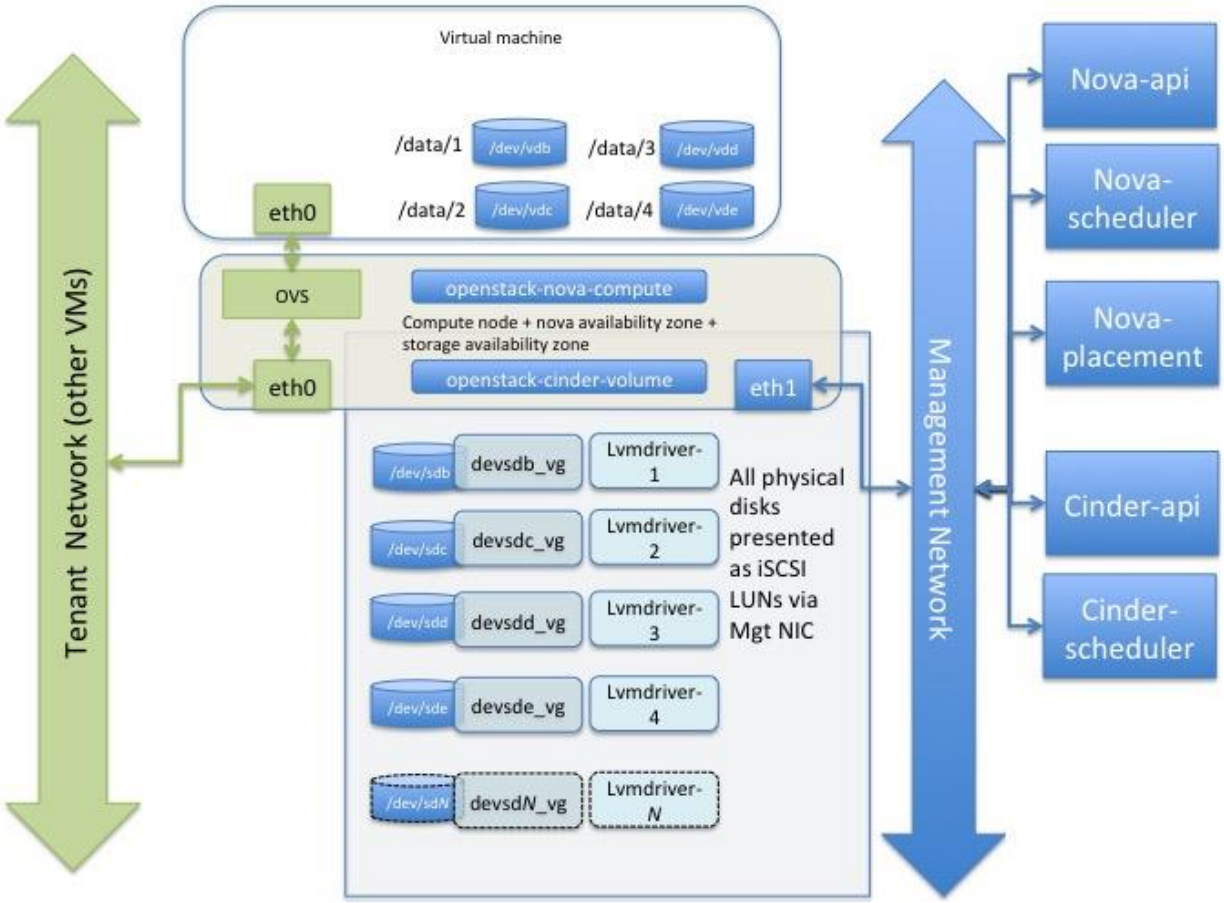
Ephemeral storage is created in the /var/lib/nova/instances directory for Nova VM instances. This directory can be created as a separate mount point that is backed by a RAID volume and configured to provide a certain level of throughput and latency (see above table).

Storage Management Strategy

Cinder with Local Disks presented as LVM backends

This involves setting up cinder-volume services on each of the compute nodes and using the LVM Volume driver (cinder.volume.drivers.lvm.LVMVolumeDriver) of cinder to present entire disks to the guests.

The diagram below illustrates the mechanism --



VM Instance with Storage diagram 4

NOTE:

- While Red Hat's documentation states that the LVM-driver based implementation is not suitable for enterprise workloads, that is with reference to majority of use-cases where Cinder + LVM is used as a truly network based storage option (with limited High availability capabilities).
- The solution provided in this document solves the storage locality problem without which Hadoop workloads would suffer from poor performance. Since Cloudera storage solutions like HDFS and Kudu have fault tolerance already built in, the concerns wrt. Fault Tolerance and High Availability are already mitigated at the application layer.
- Red Hat fully supports this configuration.

The following sequences of actions have to be performed -

1. Set up the cinder-volume and target service on each of the compute nodes

2. Create an aggregation and availability zone corresponding to each compute node in the cluster. This would imply that each nova-compute + cinder-volume node is its own nova availability zone as well as its own storage availability zone.
3. Create cinder volumes spanning the entire capacity of each disk in each availability zone
4. Instantiate a VM instance of chosen flavor in the specific availability zone
5. Attach the volumes to the VM
6. Format the volumes presented as virtual disks and mount them with appropriate mount options, such that Cloudera EDH can consume them.

For example:

```
t0311 ~(keystone_admin)]$ openstack availability zone list
+-----------+-------------+
| Zone Name | Zone Status |
+-----------+-------------+
| internal  | available   |
| t0212     | available   |
| t0211     | available   |
| t0315     | available   |
| t0213     | available   |
| t0207     | available   |
| t0208     | available   |
| t0317     | available   |
| t0314     | available   |
| t0316     | available   |
| t0209     | available   |
| t0215     | available   |
| t0210     | available   |
| t0214     | available   |
| t0316     | available   |
| t0212     | available   |
| t0207     | available   |
| t0214     | available   |
| t0215     | available   |
| t0209     | available   |
| t0208     | available   |
| t0210     | available   |
| t0314     | available   |
| t0315     | available   |
| t0211     | available   |
| t0317     | available   |
| t0213     | available   |
| nova      | available   |
| nova      | available   |
+-----------+-------------+
```

```
t0311 ~(keystone_admin)]$ cinder availability-zone-list
+-------+-----------+
| Name  | Status    |
+-------+-----------+
| t0207 | available |
| t0208 | available |
| t0209 | available |
| t0210 | available |
| t0211 | available |
| t0212 | available |
| t0213 | available |
| t0214 | available |
| t0215 | available |
| t0314 | available |
| t0315 | available |
| t0316 | available |
| t0317 | available |
+-------+-----------+
```

The premise is that, we will deploy instances localized to each compute node and present Networked volume devices local to each compute node to the VM instances therein.

These would look like --

```
t0311 ~(keystone_admin)]$ openstack volume list
+--------------------------------------+---------------+-----------+------+------------------------------------
+
| ID                                   | Display Name  | Status    | Size | Attached to                        |
+--------------------------------------+---------------+-----------+------+------------------------------------
+
| 347863f8-dbd0-4d7b-b85e-aa3f36d78eac | t0213-vol-07  | in-use    | 1800 | Attached to cdh-t0213 on
/dev/vdh  |
| 87df68d4-f468-4bf4-9a3f-d4125f69aef4 | t0213-vol-06  | in-use    | 1800 | Attached to cdh-t0213 on
/dev/vdg  |
| 71356df4-df9a-48ff-a7d0-89cb562ad38b | t0213-vol-05  | in-use    | 1800 | Attached to cdh-t0213 on
/dev/vdf  |
| 8f753d53-e773-488a-bb1c-132de82fda51 | t0213-vol-04  | in-use    | 1800 | Attached to cdh-t0213 on
/dev/vde  |
| 6140a7ad-60e0-45e5-bd0b-4055169a37c4 | t0213-vol-03  | in-use    | 1800 | Attached to cdh-t0213 on
/dev/vdd  |
| ff191b27-94fb-4b11-bf90-45ff3800c3bc | t0213-vol-02  | available | 1800
|                                      |
| 16a25661-f2ba-4998-a9c6-d818be1899d4 | t0213-vol-01  | available | 1800
|                                      |
| 5e938c6c-d999-428a-80e7-1c2b9c79e442 | t0212-vol-07  | in-use    | 1800 | Attached to cdh-t0212 on
/dev/vdi  |
| c709cf81-191e-441a-a894-d94b19f35f58 | t0212-vol-06  | in-use    | 1800 | Attached to cdh-t0212 on
/dev/vdh  |
| a2a586e0-551c-45de-8436-62a21b35573b | t0212-vol-05  | in-use    | 1800 | Attached to cdh-t0212 on
/dev/vdg  |
| 6534f0e8-0f2e-4274-8035-b3c91742c1ee | t0212-vol-04  | in-use    | 1800 | Attached to cdh-t0212 on
/dev/vdf  |
| ed575008-76fa-4035-917a-eb03f61ef386 | t0212-vol-03  | in-use    | 1800 | Attached to cdh-t0212 on
/dev/vde  |
| 436df962-101e-46c3-9828-0e7366343bed | t0212-vol-02  | in-use    | 1800 | Attached to cdh-t0212 on
/dev/vdd  |
| 7cbed019-df9b-43a1-a4fa-5226296520bd | t0212-vol-01  | in-use    | 1800 | Attached to cdh-t0212 on
/dev/vdc  |
```

In our labs, we have 14 x 2 TB drives installed on each Nova compute node. We present each of these as single-disk backends to LVM volume groups.

```
[t0212 ~]$ sudo pvs
 PV         VG         Fmt  Attr PSize PFree
 /dev/sdb1  dev-sdb1vg lvm2 a--  1.82t 63.01g
 /dev/sdc1  dev-sdc1vg lvm2 a--  1.82t  1.82t
 /dev/sdd1  dev-sdd1vg lvm2 a--  1.82t 63.01g
 /dev/sde1  dev-sde1vg lvm2 a--  1.82t 63.01g
 /dev/sdf1  dev-sdf1vg lvm2 a--  1.82t 63.01g
 /dev/sdg1  dev-sdg1vg lvm2 a--  1.82t  1.82t
 /dev/sdh1  dev-sdh1vg lvm2 a--  1.82t  1.82t
 /dev/sdi1  dev-sdi1vg lvm2 a--  1.82t  1.82t
 /dev/sdj1  dev-sdj1vg lvm2 a--  1.82t 63.01g
 /dev/sdk1  dev-sdk1vg lvm2 a--  1.82t 63.01g
 /dev/sdl1  dev-sdl1vg lvm2 a--  1.82t  1.82t
 /dev/sdm1  dev-sdm1vg lvm2 a--  1.82t  1.82t
 /dev/sdn1  dev-sdn1vg lvm2 a--  1.82t  1.82t
 /dev/sdo1  dev-sdo1vg lvm2 a--  1.82t 63.01g
```

The underlying volumes must be created to span as close to the full size of the entire physical spindle as possible.

```
[t0212 ~]$ sudo lvs
 LV                                        VG         Attr       LSize Pool Origin
Data%  Meta%  Move Log Cpy%Sync Convert
```

```
 volume-5e938c6c-d999-428a-80e7-1c2b9c79e442 dev-sdb1vg -wi-ao----
1.76t
 volume-7cbed019-df9b-43a1-a4fa-5226296520bd dev-sdd1vg -wi-ao----
1.76t
 volume-c709cf81-191e-441a-a894-d94b19f35f58 dev-sde1vg -wi-ao----
1.76t
 volume-ed575008-76fa-4035-917a-eb03f61ef386 dev-sdf1vg -wi-ao----
1.76t
 volume-a2a586e0-551c-45de-8436-62a21b35573b dev-sdj1vg -wi-ao----
1.76t
 volume-6534f0e8-0f2e-4274-8035-b3c91742c1ee dev-sdk1vg -wi-ao----
1.76t
 volume-436df962-101e-46c3-9828-0e7366343bed dev-sdo1vg -wi-ao---- 1.76t
```

When these volumes (created as shown below) are presented to the VM, they will show up as regular spindles to the VM.

```
$ openstack volume create --size=1800 --availability-zone=t0211 t0211-vol-01
$ openstack server add volume cdh-t0211 t0211-vol-01
```

```
# sudo fdisk -l|grep 1932
Disk /dev/vdc: 1932.7 GB, 1932735283200 bytes, 3774873600 sectors
Disk /dev/vdd: 1932.7 GB, 1932735283200 bytes, 3774873600 sectors
Disk /dev/vde: 1932.7 GB, 1932735283200 bytes, 3774873600 sectors
Disk /dev/vdf: 1932.7 GB, 1932735283200 bytes, 3774873600 sectors
Disk /dev/vdg: 1932.7 GB, 1932735283200 bytes, 3774873600 sectors
Disk /dev/vdh: 1932.7 GB, 1932735283200 bytes, 3774873600 sectors
Disk /dev/vdi: 1932.7 GB, 1932735283200 bytes, 3774873600 sectors
```

These are then formatted (using mkfs) and mounted for Cloudera EDH to consume --

```
# df -h
Filesystem      Size  Used Avail Use% Mounted on
/dev/vda2        97G  8.4G   89G   9% /
devtmpfs        109G     0  109G   0% /dev
tmpfs           109G  8.0K  109G   1% /dev/shm
tmpfs           109G   41M  109G   1% /run
tmpfs           109G     0  109G   0% /sys/fs/cgroup
/dev/vdi        1.8T  943G  857G  53% /data/vdi
/dev/vdh        1.8T  947G  853G  53% /data/vdh
/dev/vdg        1.8T  940G  860G  53% /data/vdg
/dev/vde        1.8T  945G  855G  53% /data/vde
/dev/vdf        1.8T  944G  856G  53% /data/vdf
/dev/vdd        1.8T  944G  856G  53% /data/vdd
/dev/vdc        1.8T  940G  860G  53% /data/vdc
/dev/vda1      1014M  172M  843M  17% /boot
tmpfs            22G     0   22G   0% /run/user/0
cm_processes    109G   59M  109G   1% /run/cloudera-scm-agent/process
tmpfs            22G     0   22G   0% /run/user/1000
```

Warning:

Do not use a single spindle to create more than one volume in this design, or there will be performance penalties incurred. In Bare-metal deployments, a single NL-SAS/SATA drive (as is the popular choice for storage media in Cloudera installations) is capable of throughput between 120-140MB/s. HDFS is able to drive multiple such spindles simultaneously to their full practical throughput capabilities. If we consider a hypothetical scenario where a single 2TB spindle is split between 4 x 500GB volumes, we run the risk of severely affecting performance.

Install cinder-volume on the compute nodes

The cinder volume service can be installed by running -

```
# yum install -y openstack-cinder-volume target
```

Once the software is installed, create/update the configuration file /etc/cinder/cinder.conf as follows --

```
[DEFAULT]
enabled_backends = lvmdriver-1, lvmdriver-2, lvmdriver-3, lvmdriver-4, lvmdriver-5,
lvmdriver-6, lvmdriver-7, lvmdriver-8, lvmdriver-9, lvmdriver-10, lvmdriver-11,
lvmdriver-12, lvmdriver-13, lvmdriver-14
storage_availability_zone = t0212
default_availability_zone = t0212
iscsi_protocol = iscsi
iscsi_helper = lioadm
iscsi_ip_address = 10.10.1.16


[lvmdriver-1]
volume_driver = cinder.volume.drivers.lvm.LVMVolumeDriver
volume_group = dev-sdb1vg
iscsi_prototol = iscsi
iscsi_helper = lioadm
storage_availability_zone = t0212
iscsi_ip_address = 10.10.1.16
[lvmdriver-2]
volume_driver = cinder.volume.drivers.lvm.LVMVolumeDriver
volume_group = dev-sdc1vg
iscsi_prototol = iscsi
iscsi_helper = lioadm
storage_availability_zone = t0212
iscsi_ip_address = 10.10.1.16
[lvmdriver-3]
volume_driver = cinder.volume.drivers.lvm.LVMVolumeDriver
volume_group = dev-sdd1vg
iscsi_prototol = iscsi
iscsi_helper = lioadm
storage_availability_zone = t0212
iscsi_ip_address = 10.10.1.16
[lvmdriver-4]
volume_driver = cinder.volume.drivers.lvm.LVMVolumeDriver
volume_group = dev-sde1vg
iscsi_prototol = iscsi
iscsi_helper = lioadm
storage_availability_zone = t0212
iscsi_ip_address = 10.10.1.16
[lvmdriver-5]
volume_driver = cinder.volume.drivers.lvm.LVMVolumeDriver
volume_group = dev-sdf1vg
iscsi_prototol = iscsi
iscsi_helper = lioadm
storage_availability_zone = t0212
iscsi_ip_address = 10.10.1.16
[lvmdriver-6]
volume_driver = cinder.volume.drivers.lvm.LVMVolumeDriver
volume_group = dev-sdg1vg
iscsi_prototol = iscsi
```

```
iscsi_helper = lioadm
storage_availability_zone = t0212
iscsi_ip_address = 10.10.1.16
[lvmdriver-7]
volume_driver = cinder.volume.drivers.lvm.LVMVolumeDriver
volume_group = dev-sdh1vg
iscsi_prototol = iscsi
iscsi_helper = lioadm
storage_availability_zone = t0212
iscsi_ip_address = 10.10.1.16
[lvmdriver-8]
volume_driver = cinder.volume.drivers.lvm.LVMVolumeDriver
volume_group = dev-sdi1vg
iscsi_prototol = iscsi
iscsi_helper = lioadm
storage_availability_zone = t0212
iscsi_ip_address = 10.10.1.16
[lvmdriver-9]
volume_driver = cinder.volume.drivers.lvm.LVMVolumeDriver
volume_group = dev-sdj1vg
iscsi_prototol = iscsi
iscsi_helper = lioadm
storage_availability_zone = t0212
iscsi_ip_address = 10.10.1.16
[lvmdriver-10]
volume_driver = cinder.volume.drivers.lvm.LVMVolumeDriver
volume_group = dev-sdk1vg
iscsi_prototol = iscsi
iscsi_helper = lioadm
storage_availability_zone = t0212
iscsi_ip_address = 10.10.1.16
[lvmdriver-11]
volume_driver = cinder.volume.drivers.lvm.LVMVolumeDriver
volume_group = dev-sdl1vg
iscsi_prototol = iscsi
iscsi_helper = lioadm
storage_availability_zone = t0212
iscsi_ip_address = 10.10.1.16
[lvmdriver-12]
volume_driver = cinder.volume.drivers.lvm.LVMVolumeDriver
volume_group = dev-sdm1vg
iscsi_prototol = iscsi
iscsi_helper = lioadm
storage_availability_zone = t0212
iscsi_ip_address = 10.10.1.16
[lvmdriver-13]
volume_driver = cinder.volume.drivers.lvm.LVMVolumeDriver
volume_group = dev-sdn1vg
iscsi_prototol = iscsi
iscsi_helper = lioadm
storage_availability_zone = t0212
iscsi_ip_address = 10.10.1.16
[lvmdriver-14]
volume_driver = cinder.volume.drivers.lvm.LVMVolumeDriver
volume_group = dev-sdo1vg
iscsi_prototol = iscsi
iscsi_helper = lioadm
storage_availability_zone = t0212
iscsi_ip_address = 10.10.1.16
```

Set up Aggregations and availability zones

For each of the compute nodes, run the following --

```
[~(keystone_admin)]# nova aggregate-create t0211 t0211
```

```
[~(keystone_admin)]# nova aggregate-add-host 1 t0211.mydomain.com
```

Now, after doing this for each compute node, you should see something like this --

```
0311 ~(keystone_admin)]$ nova availability-zone-list

+----------------------------+----------------------------------------+
| Name                       | Status                                 |
+----------------------------+----------------------------------------+
| internal                   | available                              |
| |- t0311.mydomain.com |                                        |
| | |- nova-conductor        | enabled :-) 2017-06-13T16:08:54.000000 |
| | |- nova-scheduler        | enabled :-) 2017-06-13T16:08:52.000000 |
| | |- nova-consoleauth      | enabled :-) 2017-06-13T16:08:46.000000 |
| t0212                      | available                              |
| |- t0212.mydomain.com |                                        |
| | |- nova-compute          | enabled :-) 2017-06-13T16:08:44.000000 |
| t0211                      | available                              |
| |- t0211.mydomain.com |                                        |
| | |- nova-compute          | enabled :-) 2017-06-13T16:08:46.000000 |
| t0315                      | available                              |
| |- t0315.mydomain.com |                                        |
| | |- nova-compute          | enabled :-) 2017-06-13T16:08:53.000000 |
| t0213                      | available                              |
| |- t0213.mydomain.com |                                        |
| | |- nova-compute          | enabled :-) 2017-06-13T16:08:47.000000 |
| t0207                      | available                              |
| |- t0207.mydomain.com |                                        |
| | |- nova-compute          | enabled :-) 2017-06-13T16:08:49.000000 |
| t0208                      | available                              |
| |- t0208.mydomain.com |                                        |
| | |- nova-compute          | enabled :-) 2017-06-13T16:08:46.000000 |
| t0317                      | available                              |
| |- t0317.mydomain.com |                                        |
| | |- nova-compute          | enabled :-) 2017-06-13T16:08:44.000000 |
| t0314                      | available                              |
| |- t0314.mydomain.com |                                        |
| | |- nova-compute          | enabled :-) 2017-06-13T16:08:49.000000 |
| t0316                      | available                              |
| |- t0316.mydomain.com |                                        |
| | |- nova-compute          | enabled :-) 2017-06-13T16:08:46.000000 |
| t0209                      | available                              |
| |- t0209.mydomain.com |                                        |
| | |- nova-compute          | enabled :-) 2017-06-13T16:08:51.000000 |
| t0215                      | available                              |
| |- t0215.mydomain.com |                                        |
| | |- nova-compute          | enabled :-) 2017-06-13T16:08:45.000000 |
| t0210                      | available                              |
| |- t0210.mydomain.com |                                        |
| | |- nova-compute          | enabled :-) 2017-06-13T16:08:46.000000 |
| t0214                      | available                              |
| |- t0214.mydomain.com |                                        |
| | |- nova-compute          | enabled :-) 2017-06-13T16:08:44.000000 |
```

Restart openstack-cinder-volume services on the compute nodes and the controller node. After this, you should see the following availability zones visible from cinder as well --

```
t0311 ~(keystone_admin)]$ cinder availability-zone-list
+-------+-----------+
| Name  | Status    |
+-------+-----------+
| t0207 | available |
```

```
| t0208 | available |
| t0209 | available |
| t0210 | available |
| t0211 | available |
| t0212 | available |
| t0213 | available |
| t0214 | available |
| t0215 | available |
| t0314 | available |
| t0315 | available |
| t0316 | available |
| t0317 | available |
+-------+-----------+
```

Create instances corresponding to each availability zone.

From the cli or WebUI, create volumes corresponding to each of the availability zones and block devices on the compute nodes. After this is done, you can attach the block devices to the instances on each of the compute nodes.

From the nova-compute node, you should be able to see the LUN presentation via iscsi as follows --

```
[t0212 ~]$ sudo targetcli ls
o- / ......................................................................................................... [...]
 o- backstores ............................................................................................. [...]
 | o- block ..................................................................................... [Storage Objects: 7]
 | | o- iqn.2010-10.org.openstack:volume-436df962-101e-46c3-9828-0e7366343bed  [/dev/dev-sdo1vg/volume-436df962-101e-46c3-9828-0e7366343bed (1.8TiB) write-thru activated]
 | | o- iqn.2010-10.org.openstack:volume-5e938c6c-d999-428a-80e7-1c2b9c79e442  [/dev/dev-sdb1vg/volume-5e938c6c-d999-428a-80e7-1c2b9c79e442 (1.8TiB) write-thru activated]
 | | o- iqn.2010-10.org.openstack:volume-6534f0e8-0f2e-4274-8035-b3c91742c1ee  [/dev/dev-sdk1vg/volume-6534f0e8-0f2e-4274-8035-b3c91742c1ee (1.8TiB) write-thru activated]
 | | o- iqn.2010-10.org.openstack:volume-7cbed019-df9b-43a1-a4fa-5226296520bd  [/dev/dev-sdd1vg/volume-7cbed019-df9b-43a1-a4fa-5226296520bd (1.8TiB) write-thru activated]
 | | o- iqn.2010-10.org.openstack:volume-a2a586e0-551c-45de-8436-62a21b35573b  [/dev/dev-sdj1vg/volume-a2a586e0-551c-45de-8436-62a21b35573b (1.8TiB) write-thru activated]
 | | o- iqn.2010-10.org.openstack:volume-c709cf81-191e-441a-a894-d94b19f35f58  [/dev/dev-sde1vg/volume-c709cf81-191e-441a-a894-d94b19f35f58 (1.8TiB) write-thru activated]
 | | o- iqn.2010-10.org.openstack:volume-ed575008-76fa-4035-917a-eb03f61ef386  [/dev/dev-sdf1vg/volume-ed575008-76fa-4035-917a-eb03f61ef386 (1.8TiB) write-thru activated]
 | o- fileio .................................................................................... [Storage Objects: 0]
 | o- pscsi ..................................................................................... [Storage Objects: 0]
 | o- ramdisk ................................................................................... [Storage Objects: 0]
 o- iscsi ............................................................................................. [Targets: 7]
 | o- iqn.2010-10.org.openstack:volume-436df962-101e-46c3-9828-0e7366343bed ........................................... [TPGs: 1]
 | | o- tpg1 ....................................................................................... [no-gen-acls, auth per-acl]
 | |   o- acls .................................................................................................. [ACLs: 1]
 | |   | o- iqn.1994-05.com.redhat:a47695191 ..................................................... [1-way auth, Mapped LUNs: 1]
 | |   |   o- mapped_lun0 ................ [lun0 block/iqn.2010-10.org.openstack:volume-436df962-101e-46c3-9828-0e7366343bed (rw)]
 | |   o- luns .................................................................................................. [LUNs: 1]
 | |   | o- lun0  [block/iqn.2010-10.org.openstack:volume-436df962-101e-46c3-9828-0e7366343bed (/dev/dev-sdo1vg/volume-436df962-101e-46c3-9828-0e7366343bed)]
 | |   o- portals ............................................................................................ [Portals: 1]
 | |     o- 10.10.1.16:3260 ........................................................................................... [OK]
 | o- iqn.2010-10.org.openstack:volume-5e938c6c-d999-428a-80e7-1c2b9c79e442 ........................................... [TPGs: 1]
 | | o- tpg1 ....................................................................................... [no-gen-acls, auth per-acl]
 | |   o- acls .................................................................................................. [ACLs: 1]
 | |   | o- iqn.1994-05.com.redhat:a47695191 ..................................................... [1-way auth, Mapped LUNs: 1]
 | |   |   o- mapped_lun0 ................ [lun0 block/iqn.2010-10.org.openstack:volume-5e938c6c-d999-428a-80e7-1c2b9c79e442 (rw)]
 | |   o- luns .................................................................................................. [LUNs: 1]
 | |   | o- lun0  [block/iqn.2010-10.org.openstack:volume-5e938c6c-d999-428a-80e7-1c2b9c79e442 (/dev/dev-sdb1vg/volume-5e938c6c-d999-428a-80e7-1c2b9c79e442)]
 | |   o- portals ............................................................................................ [Portals: 1]
 | |     o- 10.10.1.16:3260 ........................................................................................... [OK]
 | o- iqn.2010-10.org.openstack:volume-6534f0e8-0f2e-4274-8035-b3c91742c1ee ........................................... [TPGs: 1]
 | | o- tpg1 ....................................................................................... [no-gen-acls, auth per-acl]
 | |   o- acls .................................................................................................. [ACLs: 1]
 | |   | o- iqn.1994-05.com.redhat:a47695191 ..................................................... [1-way auth, Mapped LUNs: 1]
 | |   |   o- mapped_lun0 ................ [lun0 block/iqn.2010-10.org.openstack:volume-6534f0e8-0f2e-4274-8035-b3c91742c1ee (rw)]
 | |   o- luns .................................................................................................. [LUNs: 1]
 | |   | o- lun0  [block/iqn.2010-10.org.openstack:volume-6534f0e8-0f2e-4274-8035-b3c91742c1ee (/dev/dev-sdk1vg/volume-6534f0e8-0f2e-4274-8035-b3c91742c1ee)]
 | |   o- portals ............................................................................................ [Portals: 1]
 | |     o- 10.10.1.16:3260 ........................................................................................... [OK]
 | o- iqn.2010-10.org.openstack:volume-7cbed019-df9b-43a1-a4fa-5226296520bd ........................................... [TPGs: 1]
 | | o- tpg1 ....................................................................................... [no-gen-acls, auth per-acl]
```

```
| |   o- acls ................................................................................................ [ACLs: 1]
| |   | o- iqn.1994-05.com.redhat:a47695191 ....................................................... [1-way auth, Mapped LUNs: 1]
| |   |   o- mapped_lun0 ................ [lun0 block/iqn.2010-10.org.openstack:volume-7cbed019-df9b-43a1-a4fa-5226296520bd (rw)]
| |   o- luns ................................................................................................ [LUNs: 1]
| |   | o- lun0  [block/iqn.2010-10.org.openstack:volume-7cbed019-df9b-43a1-a4fa-5226296520bd (/dev/dev-sdd1vg/volume-7cbed019-df9b-43a1-a4fa-5226296520bd)]
| |   o- portals ........................................................................................... [Portals: 1]
| |     o- 10.10.1.16:3260 ...................................................................................... [OK]
| o- iqn.2010-10.org.openstack:volume-a2a586e0-551c-45de-8436-62a21b35573b ........................................... [TPGs: 1]
| | o- tpg1 ........................................................................................ [no-gen-acls, auth per-acl]
| |   o- acls ................................................................................................ [ACLs: 1]
| |   | o- iqn.1994-05.com.redhat:a47695191 ....................................................... [1-way auth, Mapped LUNs: 1]
| |   |   o- mapped_lun0 ................ [lun0 block/iqn.2010-10.org.openstack:volume-a2a586e0-551c-45de-8436-62a21b35573b (rw)]
| |   o- luns ................................................................................................ [LUNs: 1]
| |   | o- lun0  [block/iqn.2010-10.org.openstack:volume-a2a586e0-551c-45de-8436-62a21b35573b (/dev/dev-sdj1vg/volume-a2a586e0-551c-45de-8436-62a21b35573b)]
| |   o- portals ........................................................................................... [Portals: 1]
| |     o- 10.10.1.16:3260 ...................................................................................... [OK]
| o- iqn.2010-10.org.openstack:volume-c709cf81-191e-441a-a894-d94b19f35f58 ........................................... [TPGs: 1]
| | o- tpg1 ........................................................................................ [no-gen-acls, auth per-acl]
| |   o- acls ................................................................................................ [ACLs: 1]
| |   | o- iqn.1994-05.com.redhat:a47695191 ....................................................... [1-way auth, Mapped LUNs: 1]
| |   |   o- mapped_lun0 ................ [lun0 block/iqn.2010-10.org.openstack:volume-c709cf81-191e-441a-a894-d94b19f35f58 (rw)]
| |   o- luns ................................................................................................ [LUNs: 1]
| |   | o- lun0  [block/iqn.2010-10.org.openstack:volume-c709cf81-191e-441a-a894-d94b19f35f58 (/dev/dev-sde1vg/volume-c709cf81-191e-441a-a894-d94b19f35f58)]
| |   o- portals ........................................................................................... [Portals: 1]
| |     o- 10.10.1.16:3260 ...................................................................................... [OK]
| o- iqn.2010-10.org.openstack:volume-ed575008-76fa-4035-917a-eb03f61ef386 ........................................... [TPGs: 1]
|   o- tpg1 ........................................................................................ [no-gen-acls, auth per-acl]
|     o- acls ................................................................................................ [ACLs: 1]
|     | o- iqn.1994-05.com.redhat:a47695191 ....................................................... [1-way auth, Mapped LUNs: 1]
|     |   o- mapped_lun0 ................ [lun0 block/iqn.2010-10.org.openstack:volume-ed575008-76fa-4035-917a-eb03f61ef386 (rw)]
|     o- luns ................................................................................................ [LUNs: 1]
|     | o- lun0  [block/iqn.2010-10.org.openstack:volume-ed575008-76fa-4035-917a-eb03f61ef386 (/dev/dev-sdf1vg/volume-ed575008-76fa-4035-917a-eb03f61ef386)]
|     o- portals ........................................................................................... [Portals: 1]
|       o- 10.10.1.16:3260 ...................................................................................... [OK]
o- loopback ........................................................................................... [Targets: 0]
```

And once the Cloudera Enterprise cluster is installed, you should be able to see the capacity in hdfs --

```
[root@host-ip-10-15-40-87 ~]# hdfs dfs -df -h
Filesystem                                         Size   Used  Available  Use%
hdfs://host-ip-10-15-40-83.MYDOMAIN.COM:8020  608.7 G  1.5 G    607.2 G    0%
```

In order to run some quick sanity checks, run teragens/terasorts against the cluster as follows --

```
[root@host-ip-10-15-40-87 ~]# /usr/bin/hadoop jar /opt/cloudera/parcels/CDH/lib/hadoop-0.20-mapreduce/hadoop-examples.jar teragen -Ddfs.replication=1 -Ddfs.client.block.write.locateFollowingBlock.retries=15 -Dyarn.app.mapreduce.am.job.cbd-mode.enable=false -Dyarn.app.mapreduce.am.job.map.pushdown=false -Dmapreduce.job.maps=15 -Dmapreduce.map.memory.mb=1024 10000000 ts_in
17/06/16 16:26:03 INFO client.RMProxy: Connecting to ResourceManager at host-ip-10-15-40-83.MYDOMAIN.COM/10.15.40.83:8032
17/06/16 16:26:03 INFO hdfs.DFSClient: Created token for systest: HDFS_DELEGATION_TOKEN owner=systest@MYDOMAIN.COM, renewer=yarn, realUser=, issueDate=1497644763914, maxDate=1498249563914, sequenceNumber=1, masterKeyId=4 on 10.15.40.83:8020
17/06/16 16:26:05 INFO security.TokenCache: Got dt for hdfs://host-ip-10-15-40-83.MYDOMAIN.COM:8020; Kind: HDFS_DELEGATION_TOKEN, Service: 10.15.40.83:8020, Ident: (token for systest: HDFS_DELEGATION_TOKEN owner=systest@MYDOMAIN.COM, renewer=yarn, realUser=, issueDate=1497644763914, maxDate=1498249563914, sequenceNumber=1, masterKeyId=4)
17/06/16 16:26:05 INFO security.TokenCache: Got dt for hdfs://host-ip-10-15-40-83.MYDOMAIN.COM:8020; Kind: kms-dt, Service: 10.15.40.93:16000, Ident: (kms-dt owner=systest, renewer=yarn, realUser=, issueDate=1497644764703, maxDate=1498249564703, sequenceNumber=1, masterKeyId=118)
17/06/16 16:26:05 INFO terasort.TeraGen: Generating 10000000 using 15
17/06/16 16:26:06 INFO mapreduce.JobSubmitter: number of splits:15
17/06/16 16:26:06 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1497627345482_0001
```

```
17/06/16 16:26:06 INFO mapreduce.JobSubmitter: Kind: HDFS_DELEGATION_TOKEN, Service:
10.15.40.83:8020, Ident: (token for systest: HDFS_DELEGATION_TOKEN
owner=systest@MYDOMAIN.COM, renewer=yarn, realUser=, issueDate=1497644763914,
maxDate=1498249563914, sequenceNumber=1, masterKeyId=4)
17/06/16 16:26:06 INFO mapreduce.JobSubmitter: Kind: kms-dt, Service: 10.15.40.93:16000,
Ident: (kms-dt owner=systest, renewer=yarn, realUser=, issueDate=1497644764703,
maxDate=1498249564703, sequenceNumber=1, masterKeyId=118)
17/06/16 16:26:08 INFO impl.YarnClientImpl: Submitted application
application_1497627345482_0001
17/06/16 16:26:08 INFO mapreduce.Job: The url to track the job: https://host-ip-10-15-40-
83.MYDOMAIN.COM:8090/proxy/application_1497627345482_0001/
17/06/16 16:26:08 INFO mapreduce.Job: Running job: job_1497627345482_0001
17/06/16 16:26:18 INFO mapreduce.Job: Job job_1497627345482_0001 running in uber mode :
false
17/06/16 16:26:18 INFO mapreduce.Job:  map 0% reduce 0%
17/06/16 16:26:28 INFO mapreduce.Job:  map 33% reduce 0%
17/06/16 16:26:32 INFO mapreduce.Job:  map 73% reduce 0%
17/06/16 16:26:33 INFO mapreduce.Job:  map 100% reduce 0%
17/06/16 16:26:36 INFO mapreduce.Job: Job job_1497627345482_0001 completed successfully
17/06/16 16:26:36 INFO mapreduce.Job: Counters: 33
File System Counters
FILE: Number of bytes read=0
FILE: Number of bytes written=1942955
FILE: Number of read operations=0
FILE: Number of large read operations=0
FILE: Number of write operations=0
HDFS: Number of bytes read=1272
HDFS: Number of bytes written=1000000000
HDFS: Number of read operations=60
HDFS: Number of large read operations=0
HDFS: Number of write operations=30
Job Counters
Launched map tasks=15
Other local map tasks=15
Total time spent by all maps in occupied slots (ms)=166113
Total time spent by all reduces in occupied slots (ms)=0
Total time spent by all map tasks (ms)=166113
Total vcore-milliseconds taken by all map tasks=166113
Total megabyte-milliseconds taken by all map tasks=170099712
Map-Reduce Framework
Map input records=10000000
Map output records=10000000
Input split bytes=1272
Spilled Records=0
Failed Shuffles=0
Merged Map outputs=0
GC time elapsed (ms)=4913
CPU time spent (ms)=71190
Physical memory (bytes) snapshot=5264490496
Virtual memory (bytes) snapshot=25258631168
Total committed heap usage (bytes)=12362711040
Peak Map Physical memory (bytes)=364609536
Peak Map Virtual memory (bytes)=1693241344
org.apache.hadoop.examples.terasort.TeraGen$Counters
CHECKSUM=21472776955442690
File Input Format Counters
Bytes Read=0
File Output Format Counters
Bytes Written=1000000000
```

One should be able to observe the performance of the storage subsystem using the iostat tool (part of the sysstat package) -

```
06/16/2017 04:39:39 PM
avg-cpu:  %user   %nice %system %iowait  %steal   %idle
           3.30    0.00   10.09    3.31    0.04   83.25
```

```
Device:         rrqm/s   wrqm/s     r/s     w/s    rMB/s    wMB/s avgrq-sz avgqu-sz   await r_await w_await  svctm  %util
vda               0.00     1.00    0.00   76.80     0.00    37.24   993.06   125.03 1603.91    0.00 1603.91  13.02 100.00
vdc               0.00     0.00    0.00  149.00     0.00    70.41   967.74     6.98   46.85    0.00   46.85   3.88  57.86
vdd               0.00     0.00    0.00  145.00     0.00    68.83   972.09     8.69   59.94    0.00   59.94   4.61  66.84
vde               0.00     0.00    0.00  135.40     0.00    64.02   968.32     6.28   46.38    0.00   46.38   3.94  53.30
vdf               0.00     0.00    0.00  144.80     0.00    68.99   975.81     7.04   48.23    0.00   48.23   4.07  59.00
vdg               0.00     0.00    0.00  144.20     0.00    68.80   977.07     6.95   48.64    0.00   48.64   4.05  58.34
vdh               0.00     0.00    0.00  142.20     0.00    67.24   968.35     6.65   46.67    0.00   46.67   3.88  55.20
vdi               0.00     0.00    0.00  147.60     0.00    70.07   972.23     6.73   46.02    0.00   46.02   3.85  56.80
```

For read-only workloads, it should yield read throughput as follows -

```
06/16/2017 04:41:34 PM
avg-cpu:  %user   %nice %system %iowait  %steal   %idle
           2.92    0.00    0.75   20.11    0.01   76.22


Device:         rrqm/s   wrqm/s     r/s     w/s    rMB/s    wMB/s avgrq-sz avgqu-sz   await r_await w_await  svctm  %util
vda               0.00     0.00    0.00  129.20     0.00    63.46  1005.92   125.17  955.19    0.00  955.19   7.74 100.00
vdc               0.00     0.00 2629.80    0.00   164.36     0.00   128.00     0.97    0.37    0.37    0.00   0.37  96.78
vdd               0.00     0.00 2539.00    0.00   158.69     0.00   128.00     0.96    0.38    0.38    0.00   0.38  96.34
vde               0.00     0.00 2622.80    0.00   163.93     0.00   128.00     0.96    0.37    0.37    0.00   0.36  95.60
vdf               0.00     0.00 2675.20    0.00   167.20     0.00   128.00     0.97    0.36    0.36    0.00   0.36  96.70
vdg               0.00     0.00 2621.60    0.00   163.85     0.00   128.00     0.97    0.37    0.37    0.00   0.37  96.66
vdh               0.00     0.00 2595.80    0.40   162.24     0.00   127.98     0.98    0.38    0.37   18.00   0.37  96.82
vdi               0.00     0.00 2588.20    0.00   161.76     0.00   128.00     0.97    0.37    0.37    0.00   0.37  96.72
```

Cloudera Software stack

Guidelines for installing the Cloudera stack on this platform are nearly identical to those for bare-metal. This is addressed in various documents on Cloudera's website.

Do not allow more than one replica of an HDFS block on any particular physical node. This is enabled with configuring the Hadoop Virtualization Extensions (HVE).

The minimum requirements to build out the cluster are:

- **3x Master Nodes (VMs)**
- **5x DataNodes (VMs)**

The DataNode count depends on the size of HDFS storage to deploy. For simplicity, ensure that DataNodes cohabitate with YARN NodeManager roles. The following table identifies service roles for different node types.

Follow the guidelines in the Compute section of this document to provision instance types.

- Care must be taken to ensure that CPU and Memory resources are not overcommitted while provisioning these node instances on the virtualized infrastructure.
- Care should also be taken to ensure automated movement of VMs is disabled. There **should be no** Migration/Live Migration of VMs allowed in this deployment model.
- Care must be taken to ensure that the Master Nodes are provisioned on disparate physical hardware, if possible in separate racks (provision Master nodes in separate Availability zones).

Logical Component Layout Tables

General Component Layout

Service/Role	Master Node	Master Node2	Master Node3	Worker Node 1..n
ZooKeeper	ZK	ZK	ZK	
HDFS	NN,QJN	NN,QJN	QJN	Data Node
Kudu	Master	Master	Master	Tablet Server
YARN	RM	RM	History Server	Node Manager
Hive			MetaStore, WebHCat, HiveServer2	
Management(misc)	Oozie, CMA	Oozie, CM (standby), Management Services (standby), CMA	Oozie, CM (active), Management Services (active),CMA	CMA
Database		Standby DB	Active DB	

Navigator			Navigator	
HUE			HUE (place on Edge node if exists)	
SEARCH				Solr
Spark	History Server			Runs on YARN
impala			statestore, catalog	impalad
HBASE	HMaster	HMaster	HMaster	Region Servers
Sentry		Sentry	Sentry	

Additional Services Component Layout

Service/Role	Kafka ZooKeeper (1..5)	KeyTrustee Server 1	KeyTrustee Server 2	KMS Proxy 1	KMS Proxy 2	Kafka Brokers	Edge Node (1 per 20 Workers)
Management(misc)	CMA	CMA	CMA	CMA	CMA	CMA	CMA
Database							
Navigator							
Kafka (Separate Cluster if doing > 100,000 transactions/second, 3-5 ZK nodes in separate ZK ensemble)	ZK					Kafka Broker	

KeyTrustee Server (Separate Cluster)	KTS (active)	KTS (standby)			
KMS (dedicated Nodes)			KMS (active)	KMS (standby)	
Sentry					
Flume					Flume Agent

> **NOTE:**
> - For the various abbreviations used in these tables, please refer to the Glossary of Terms section.

Instance-type Table

Instance Role	Instance Type/Flavor	Comments
Master Nodes	cdh-fullsize	The master instances will house components of the cloudera stack as shown in the tables above
Worker Nodes	cdh-fullsize	These will have sufficient Compute resources.

NOTE:

- It is advisable to work with a Cloudera SE to determine appropriate instance sizes based on the workloads as well physical resource parameters.
- In our testing, we have found that sharing a physical network interface between multiple VMs results in the performance getting bottlenecked at the network layer. Proper precautions have to be taken when instantiating multiple VMs per physical hypervisor.

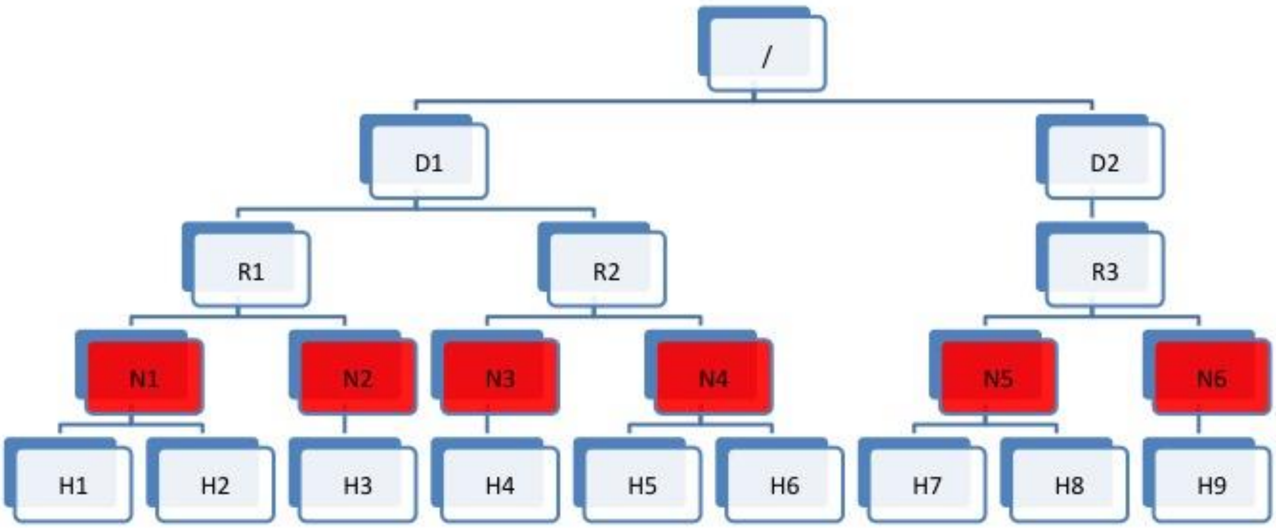
Enabling Hadoop Virtualization Extensions (HVE)

NOTE: While this document refers to hypervisors and virtual machines, this methodology is applicable to all any scenario where a "shared" something is involved. This is a strategy to help mitigate single points of failure, be it a shared power supply, a shared chassis, a shared storage tray, and so on.

Referring to the HDFS-side HVE JIRA (HADOOP-8468), following are considerations for HVE:

> *1. VMs on the same physical host are affected by the same hardware failure. In order to match the reliability of a physical deployment, replication of data across two virtual machines on the same host should be avoided.*
>
> *2. The network between VMs on the same physical host has higher throughput and lower latency and does not consume any physical switch bandwidth.*
>
> *Thus, we propose to make Hadoop network topology extendable and introduce a new level in the hierarchical topology, a node group level, which maps well onto an infrastructure that is based on a virtualized environment.*

The following diagram illustrates the addition of a new layer of abstraction (in red) called NodeGroups. The NodeGroups represent the physical hypervisor on which the nodes (VMs) reside.



HVE Topology diagram 6

All VMs under the same node group run on the same physical host. With awareness of the node group layer, HVE refines the following policies for Hadoop on virtualization:

Replica Placement Policy

- No duplicated replicas are on the same node or nodes under the same node group.
- First replica is on the local node or local node group of the writer.
- Second replica is on a remote rack of the first replica.
- Third replica is on the same rack as the second replica.
- The remaining replicas are located randomly across rack and node group for minimum restriction.

Replica Choosing Policy

The HDFS client obtains a list of replicas for a specific block sorted by distance, from nearest to farthest: local node, local node group, local rack, off rack.

Balancer Policy

- At the node level, the target and source for balancing follows this sequence: local node group, local rack, off rack.
- At the block level, a replica block is not a good candidate for balancing between source and target node if another replica is on the target node or on the same node group of the target node.

HVE typically supports failure and locality topologies defined from the perspective of virtualization. However, you can use the new extensions to support other failure and locality changes, such as those relating to power supplies, arbitrary sets of physical servers, or collections of servers from the same hardware purchase cycle.

Using Cloudera Manager, configure the following in safety valves:

- HDFS
  - hdfs core-site.xml (Cluster-wide Advanced Configuration Snippet (Safety Valve) for core-site.xml/core_site_safety_valve):

```
<property>
    <name>net.topology.impl</name>
      <value>org.apache.hadoop.net.NetworkTopologyWithNodeGroup</value>
</property>
<property>
    <name>net.topology.nodegroup.aware</name>
    <value>true</value>
</property>
<property>
    <name>dfs.block.replicator.classname</name>
      <value>org.apache.hadoop.hdfs.server.blockmanagement.BlockPlacemen
      tPolicyWithNodeGroup</value>
</property>
```

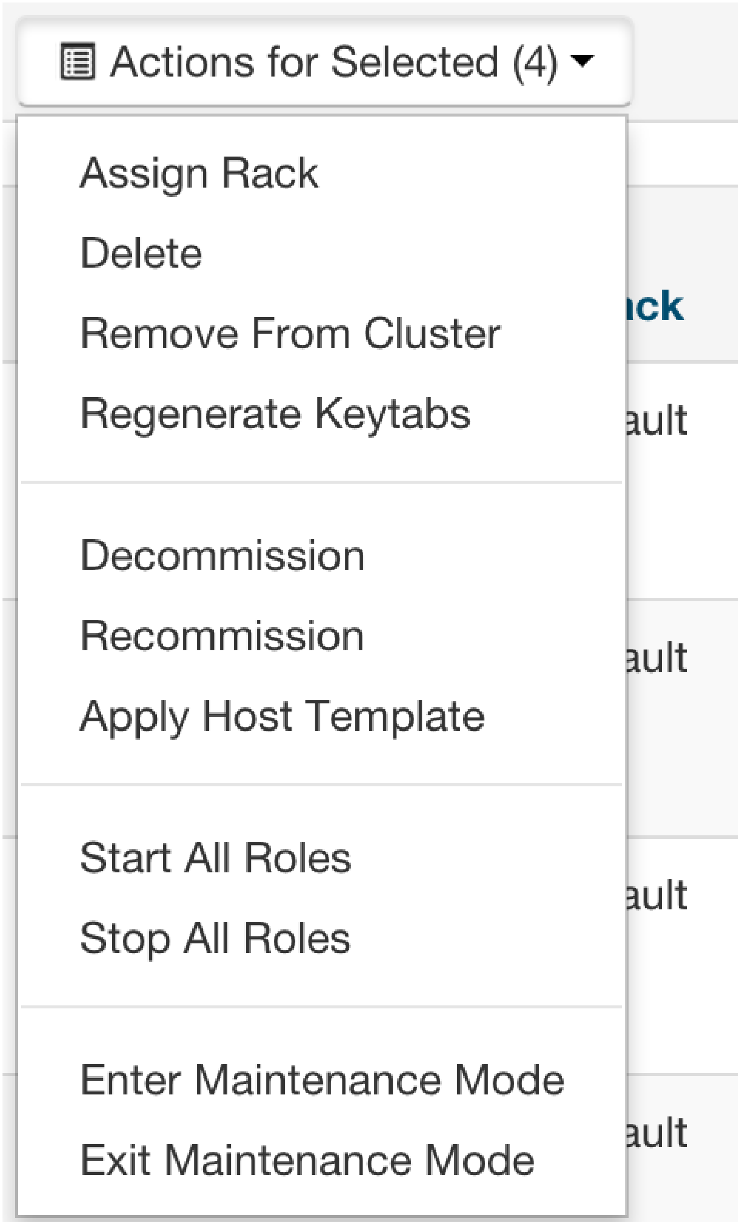
- In mapred-site.xml, add the following properties and values (this is set using the HDFS Replication Advanced configuration snippet (safety valve) mapred-site.xml (mapreduce_service_replication_config_safety_valve)):

```
<property>
      <name>mapred.jobtracker.nodegroup.aware</name>
      <value>true</value>
</property>
<property>
      <name>mapred.task.cache.levels </name>
<value>3</value>
</property>
```

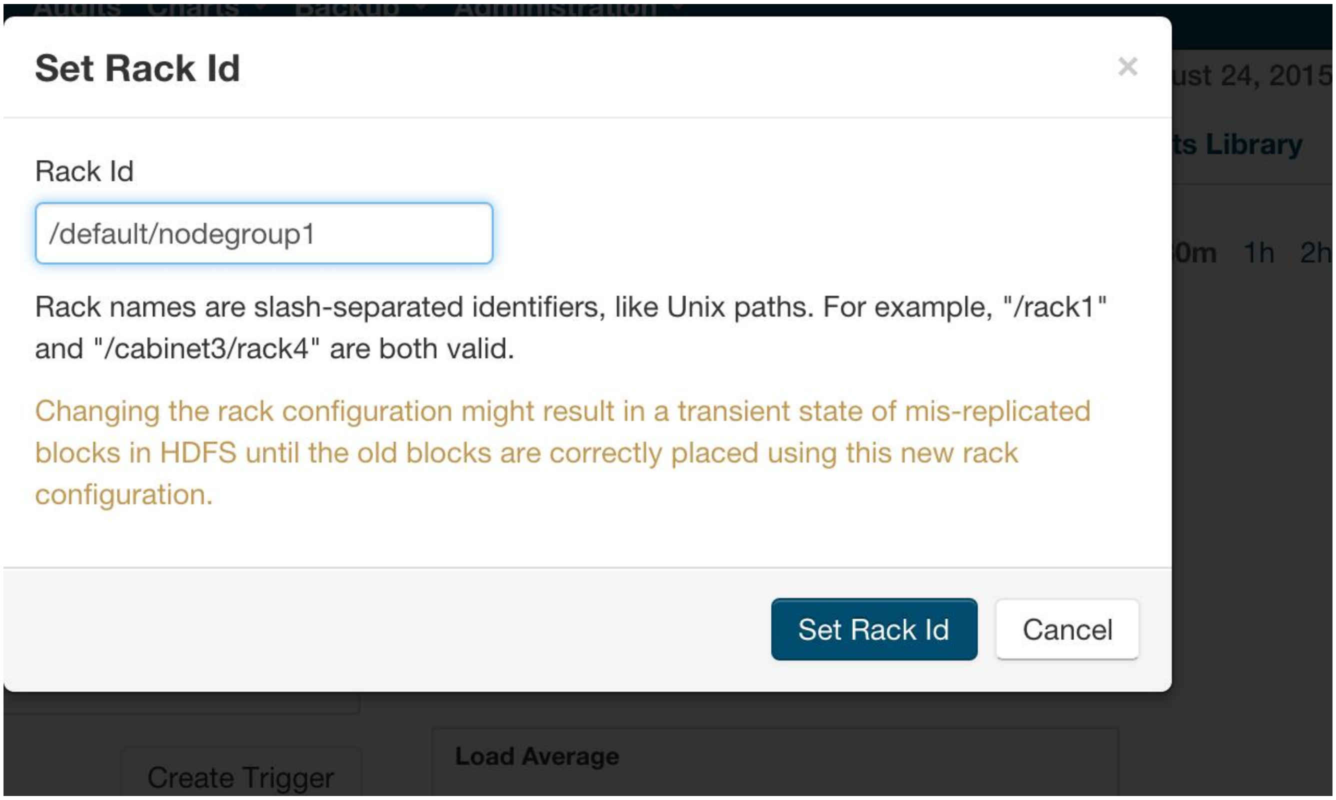
Establish the Topology:

Follow the instructions to set rack location of hosts here -- Specifying Racks for Hosts.

Select all multiple hosts from the Hosts page and then assign rack.



Alternately, In Cloudera manager, you can specify the topology by going into the Hosts/Status page and editing the Rack assignment from /default to /default/nodegroup<id>.



[Instructions]()

The following safety valves need to be applied --

1. HDFS -- Cluster-wide Advanced Configuration Snippet (Safety Valve) for core-site.xml
2. YARN - YARN Service MapReduce Advanced Configuration Snippet (Safety Valve) - mapred.xml

Follow this sequence of actions to enable HVE --

- Apply the safety valves
- Assign the rack topology to the nodes
- Stop the cluster
- Deploy client config
- Start ZooKeeper
- Start HDFS
- Start all other services

References

1. Product Documentation for Red Hat OpenStack Platform
2. SR-IOV on Red Hat OSP
3. Red Hat OSP Networking Guide
4. Understanding Red Hat OSP High Availability
5. Cloudera Product Documentation
6. Cloudera Manager API Documentation
7. Cloudera Manager CM API Python End-to-end Guide
8. HVE - HADOOP-8468

Glossary of Terms

Term	Description
CDH	Cloudera Distributed Hadoop
Ceph	An open-source distributed storage framework (RADOS or Reliable Autonomic Distributed Object Store) that allows a network of commodity hardware to be turned into a shared, distributed storage platform. Ceph natively provides Block Storage (RBD or RADOS Block Device) that are striped across the entire storage cluster, an Object Store as well as a shared filesystem.
Cinder	Cinder is the Storage provisioning and management component of the OpenStack framework.
CM	Cloudera Manager
CMA	Cloudera Manager Agent
DataNode	Worker nodes of the cluster to which the HDFS data is written.
Cloudera EDH	Cloudera Enterprise Data Hub
Ephemeral storage	Storage devices that are locally attached to Nova instances. They persist guest operating system reboots, but are removed when a Nova instance is terminated.
Glance	This is the imaging services component of the OpenStack framework. This maintains images that are used to instantiate Virtual machines in an OpenStack cluster.
HBA	Host bus adapter. An I/O controller that is used to interface a host with storage devices.
HDD	Hard disk drive.
HDFS	Hadoop Distributed File System.
HA/High	Configuration that addresses availability issues in a cluster. In a standard

Availability	configuration, the NameNode is a single point of failure (SPOF). Each cluster has a single NameNode, and if that machine or process became unavailable, the cluster as a whole is unavailable until the NameNode is either restarted or brought up on a new host. The secondary NameNode does not provide failover capability. High availability enables running two NameNodes in the same cluster: the active NameNode and the standby NameNode. The standby NameNode allows a fast failover to a new NameNode in case of machine crash or planned maintenance.
HVE	Hadoop Virtualization Extensions - this is what enables proper placement of data blocks and scheduling of YARN jobs in a Virtualized Environment wherein, multiple copies of a single block of data or YARN jobs (don't get placed/scheduled on VMs that reside on the same hypervisor host). The YARN component of HVE is still work in progress and won't be supported in CDH 5.4 (YARN-18). The HDFS component is supported in CDH 5.4.
Ironic	Ironic is an OpenStack project which provisions bare metal (as opposed to virtual) machines by leveraging common technologies such as PXE boot and IPMI to cover a wide range of hardware, while supporting pluggable drivers to allow vendor-specific functionality to be added
JBOD	Just a Bunch of Disks (this is in contrast to Disks configured via software or hardware RAID with striping and redundancy mechanisms for data protection)
JHS/Job History Server	Process that archives job metrics and metadata. One per cluster.
LUN	Logical unit number. Logical units allocated from a storage array to a host. This looks like a SCSI disk to the host, but it is only a logical volume on the storage array side.
NN/NameNode	The metadata master of HDFS essential for the integrity and proper functioning of the distributed filesystem.
NIC	Network interface card.
Nova	The Compute Scheduling and resource management component of OpenStack.
NodeManager	The process that starts application processes and manages resources on the DataNodes.
Neutron	Neutron is the Network management layer of OpenStack - it incorporates/supports SDN (Software Defined Networking) features, advanced overlay features such as VxLAN and GRE tunneling, and provides a plugin-in architecture to enable support for different technologies.
Open vSwitch/OVS	Open vSwitch is a production quality, multilayer virtual switch licensed under the open source Apache 2.0 license. It is designed to enable massive network automation through programmatic extension, while still supporting standard management interfaces and protocols (e.g. NetFlow, sFlow, IPFIX, RSPAN, CLI, LACP, 802.1ag). In addition, it is designed to support distribution across multiple physical servers.
PDU	Power distribution unit.
QJM QJN	Quorum Journal Manager. Provides a fencing mechanism for high availability in a Hadoop cluster. This service is used to distribute HDFS edit logs to multiple hosts (at least three are required) from the active NameNode. The

	standby NameNode reads the edits from the JournalNodes and constantly applies them to its own namespace. In case of a failover, the standby NameNode applies all of the edits from the JournalNodes before promoting itself to the active state. Quorum JournalNodes. Nodes on which the journal services are installed.
RM	ResourceManager. The resource management component of YARN. This initiates application startup and controls scheduling on the DataNodes of the cluster (one instance per cluster).
SAN	Storage area network.
Sahara	Sahara project aims to provide users with simple means to provision a Hadoop cluster at OpenStack by specifying several parameters like Hadoop version, cluster topology, nodes hardware details and a few more.
SPOF	Single Point of Failure
ToR	Top of rack.
VM/instance	Virtual machine.
ZK/ZooKeeper	ZooKeeper. A centralized service for maintaining configuration information, naming, and providing distributed synchronization and group services.