Generic Reference Architecture for Cloudera Enterprise running in a Private Cloud

Important Notice

Cloudera, Inc.
395 Page Mill Road
Palo Alto, CA 94306
info@cloudera.com

US: 1-888-789-1488
Intl: 1-650-362-0488
www.cloudera.com

Release Information
Date: 08/30/2018
Version: 5.x, 6.x

Table of Contents

About 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 Apache HBase snapshots
- Easily migrate data with NFSv3 support

See Cloudera Enterprise for more detailed information.

Executive Summary

This document is intended to capture guidelines towards leveraging virtualized infrastructure (IaaS) upon which Cloudera Enterprise Clusters can be deployed. The focus of this document is around sizing and design patterns, independent of the underlying technology. This document is agnostic of specifics such as hypervisor, private cloud software, storage or networking infrastructure.

After reading this document, it is our expectation that customers and partners can build infrastructure following the guidelines provided to support deployment of Cloudera Enterprise on virtualized infrastructure.

NOTE:

This document should be considered a superset and update to the following reference architectures already published, each pertaining to vendor-specific technologies --

- Cloudera Reference Architecture for VMware vSphere with locally attached storage
- Cloudera Reference Architecture for RedHat OSP with Locally attached storage
- Cloudera Reference Architecture for RedHat OSP with Ceph Storage

.

Target Audience and Scope

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

Specifically, this document articulates design patterns that involve two distinct flavors of virtualized CDH deployment --

- Virtualized with Direct Attached storage (Shared-nothing)
- Virtualized with Remote Block storage (Converged)

Reference Architecture

Why virtualize?

There are many reasons to virtualize infrastructure in the data center. This is a trend that has been ongoing since at least the past ten years or so. Most enterprise applications have specific and intermittent peaks in terms of utilization, and that results in extended periods of time when the hardware on which these applications run remains idle. In many cases, even during peak utilization one finds that only 15-20% of the resources of a node are actually being used. As a result, in order to improve resource utilization and provide better return on investment, virtualization took off as a technology.

However, as the community and practice matured, more benefits became apparent.

- Increased Agility
- Higher Flexibility
- Faster Time to market
- Higher Multi-tenancy

Cloudera's customers too have been looking for these advantages, in order to better serve their internal lines of businesses, bringing public cloud-like agility to their data centers. While in the past, our reference architectures have either been in collaboration with specific partner products or technologies that had significantly large footprints, we now have enough empirical data to provide customers and partners with this generic guide that addresses topics such as sizing, network design and so on, agnostic of the underlying platform or technology.

A key factor in enabling this has been the Cloudera Enterprise Storage Device Acceptance Criteria Guide, which provides customers and partners with a baseline for acceptable storage performance criteria.

Design Patterns

In context of virtualized infrastructure (IaaS) based deployments, this document covers the two design patterns listed below..

- Virtualized with Direct attached storage - This involves deploying Virtual nodes with storage physically located on its hypervisor/host OS.
- Virtualized with remote storage - This involves deploying virtual nodes with storage located remote to its hypervisor/host OS.

Before delving into the specific design patterns, some basic concepts should be explored.

What is the purpose of a Hadoop cluster? The Hadoop ecosystem has many components that are designed to do functions varying from querying complex data structures to machine learning, but the underlying workload, when taking a reductionist approach boils down to two aspects --

- IO -- Write and read large volumes of data to and from storage.

- Compute -- Perform computations based on the data.

Two patterns emerge from the IO component of the workload.

- Write lots of data intermittently to the underlying storage and read that data with far greater frequency. The data is written and read in an ordered fashion. This type of workload is categorized as sequential IO.
- Read and write random, small chunks of data very fast. This type of workload is categorized as random IO.

The two storage products within the Cloudera ecosystem that address these two workloads are HDFS for Sequential IO and Apache Kudu which provides a balance between reasonable sequential and random IO.

Kudu is designed to operate very well within the parameters that govern HDFS[1], with the caveat that faster storage (NVMe or SSDs) to accelerate writes (Write Logging) be leveraged.

With that in mind, we should look at sizing a cluster with HDFS capacity and throughput in mind. The majority of Hadoop ecosystem components will fit into that model.

The topic of sizing the clusters is a complex one, and while we can provide generic outlines towards sizing, the process is something that should be explored with systems engineers during an actual customer engagement. Accurate sizing requires an understanding of the specific workloads and use-cases. The following section will provide an outline that will provide scientific estimates in terms of infrastructure requirements, which can then be fine-tuned during a pre-sales engagement with Cloudera Systems Engineers.

For a more detailed discussion of the various components within the Cloudera Enterprise stack, please review the Cloudera Generic Bare-metal Reference Architecture.

Cluster sizing

Minimum and Recommended Performance characteristics

This section is augmented by the Sizing methodology section presented later in this document. This can be used to derive network throughput requirements, following a different approach than that taken in the sizing section. Per the Cloudera Enterprise Storage Device Acceptance Criteria Guide, the minimum supportable throughput per Worker node (irrespective of whether VM or Bare-metal) is 200MB/s. This implies that at least 4 gbps of East-West network bandwidth is available to the node, for proper performance.

Minimum Per-VM throughput (MB/s)	Minimum per-VM network throughput (gbps) (EW)	Recommended per-VM throughput (MB/s)	Recommended per-VM network throughput (gbps) (EW)
200	4	800	16

[1] By this we mean that the storage being used for HDFS can be leveraged for Kudu as well. The same drives that back HDFS can be used for Kudu as well.

The minimum throughput per Master node is, 120MB/s. The table below shows the minimum and recommended.

Minimum Per-VM throughput (MB/s)	Minimum per-VM network throughput (gbps) (EW)	Recommended per-VM throughput (MB/s)	Recommended per-VM network throughput (gbps) (EW)
120	2	240	4

These parameters than can be utilized to build the infrastructure that supports VMs that provide these minimum characteristics. For more details, refer to the guide referenced earlier.

Cluster Sizing methodology

Sizing the infrastructure can be approached in two different ways.

The first approach is to build the underlying infrastructure based on HDFS capacity required, while at the same time ensuring that the clusters will be well balanced between performance and capacity.

Once the physical infrastructure has been sized, then it can be converted into a private cloud and VMs built to fit as required. We call this the **Build for Capacity** approach.

The other approach is to consider a minimum throughput per core and the aggregate throughput needed for the cluster, which would then define the details about number of cores, network bandwidth, storage bandwidth (and therefore number of spindles/LUNs). We call this the **Build for Need** approach.

These two approaches help address two different scenarios. In cases where only the required HDFS capacity is known, the "Build for Capacity" approach will help size the private cloud so that it delivers a reasonable medium of both performance as well as capacity, in terms of storage, network, and compute resources.

The "Build for Need" approach will be more applicable where the required throughput of a workload is known, has better defined SLAs, etc. The virtual infrastructure can then be sized and the private cloud built.

The 'Build for Capacity' Approach

To summarize, in terms of formulae that can be reused, let us articulate the key determinants first.

A. UC (HDFS Capacity required/Usable Capacity)
B. RC (HDFS Raw Capacity)
C. CT (Cluster throughput)
D. PND (Number of Drives per node)
E. PDC (Per drive capacity)

F. PDT (Per Drive throughput)
G. NBW (Network Bandwidth)
H. NC (Number of Cores)
I. CPS (Cores per Socket)
J. NS (Number of Sockets)
K. SPN (Sockets per Node)
L. NN (Number of Nodes)
M. STPN (Storage throughput per Node)
N. SCPN (Storage Capacity Per Node)
O. PCT (Per-core Throughput)
P. RT (Required Throughput)

Storage-based Sizing formulae

RC = UC x 4[2]

NN = (RC / (PND x PDC)) + 3[3]

NBW = PND x PDT x 8 x 2[4]

CT = PND x NN x PDT

SPN = 2 (assumption) [5]

NS = NN x SPN

CPS = 10 (assumption)

NC = NS x CPS

Working through an example --

Following requirements are available --

- Usable HDFS capacity required is 400TB
- Standard hardware model in datacenter calls for 2-socket boxes with 10 Cores each.
- Standard 4TB SATA drives are used, with 12 drives per node.
- Minimum 128GB RAM per node.

---

[2] This assumes the standard HDFS replication factor of 3. Add to that the 25% raw storage for intermediate storage, gives us the number 4.
[3] Here the number 3 represents the minimum number of Master nodes
[4] Here 8 is the factor that converts B/s to b/s (Bytes to bits) and 2 is factoring in 2x network bandwidth recommended for best performance
[5] Here assuming standard 2-socket servers are being considered. This value will change if a more compute heavy node is selected.

UC = 400TB
CPS = 10 cores
PDC = 4 TB
PND = 12 drives
PDT =100MB/s (good estimate for SATA drives)

RC = UC x 4 = 1600TB
NN = (RC/(PND x PDC)) + 3 = (1600TB/(12x4TB)) + 3 = 37 nodes (rounded)
NBW = PND x PDT x 8 x 2 = 12 x 100MB/s x 8 x 2 = 19200Mbps (~ 20 Gbps)
NS = NN x NS = 37 nodes x 2 sockets = 74 sockets
NC = NS x CPS = 74 x 10 cores = 740 cores
CT = PND x NN x PDT = 12 drives x 37 nodes x 100MB/s = 44,400MB/s

This gives us the following blanket size to start a configuration with --

- 37 nodes - 34 Workers + 3 Masters
- Each node with 2 x 10-Core sockets and 128GB RAM
- Each node with 20Gbps NICs (pair of 10GbE NICs bonded)
- This cluster would theoretically provide about 1.6PB of raw capacity, or ~ 400TB of usable HDFS capacity. It would also generate ~ 44GB/s of sequential IO throughput.
- This cluster would have ~ 740 Physical CPU cores or 1480 Hyper-threaded CPU cores.

The 'Build for Need' Approach

Throughput-based sizing Formulae

NC = RT/PCT

CPS = 10 (assumption)

NS = NC/CPS

SPN = 2 (assumption)

NN = NS/SPN

NBW = (RT x 8 x 2) / NN

STPN = RT / NN

Working through an example with this --

PCT = 50MB/s (assumption)

RT = 20GB/s (based on requirements)

NC = RT/PCT = 20GBps/50MBps = 400

NS = NC/CPS = 400/10 = 40

NN = NS/SPN = 40/2 = 20

NBW = (RT x 8 x 2) / NN = (20GBps x 8 x 2) / 20 = 16Gbps

STPN = RT / NN = 20GBps / 20 = 1 GBps

In this exercise, we start with the requirement that the cluster needs to provide 20GB/s of throughput, and expected per core throughput is 50MB/s.

Certain other assumptions are made, such as each node would have two sockets each with ten cores. We arrive at a basic envelope size --

- Require 400 cores or 40 sockets, which results in 20 physical nodes.
- The network bandwidth required to achieve the 50MB/s per core throughput is 16Gbps, or two 10GbE NICs bonded together (or a single 25Gbps NIC).
- This doesn't do a good job of HDFS capacity sizing, but if we follow the assumption that we would require enough storage spindles to achieve 20GBps of IO throughput, we would require at least 1 GBps of throughput per node.
    - This can help derive the number of spindles per node (either Direct attached or Remote) provided we know the throughput per spindle.
        - For example, if we use Direct attached SATA drives and each drive can provide 100MB/s of throughput[6], we would need 10 such spindles to provide 1GBps throughput.
        - If we use remote block storage, with each spindle providing say 40MB/s of throughput, we would need 25 such spindles (LUNs). That could call for more VMs to be deployed if a per-VM throughput limitation is in place.
        - This would also allow the derivation of infrastructure details such as, the type of SAN HBA (Host Bus Adapter) that would be needed. 1GBps is 8 gbps, which implies 2 x 8 gbps FC Adapters could be used for fault tolerance, better load balancing across multiple paths and so on (two SAN fabrics are the norm in most shops).
        - The capacity of the storage would have to be calculated based on HDFS capacity required for the cluster.

NOTE:

- Here effective network throughput/bandwidth being calculated is assumed to be 2x of effective throughput to adequately accommodate E-W traffic patterns.
- An assumption has to be made on throughput achievable at the compute layer, on a per-CPU Core basis. Our tests show we can expect at least 50MB/s of throughput per physical core in a properly designed cluster. So the value of PCT should at least be 50.

[6] This is typically on the lower end for SATA spindles. It is not unusual to get up to 120MB/s of sequential IO.

- In terms of per drive throughput, certain assumptions are made, considering locally attached SATA drives which can produce about 100-120MB/s sequential IO throughput
- While considering remote block storage, the overall throughput capabilities of the storage array(s) being considered should be kept in mind. For instance, even if a per-LUN sequential throughput of 40MB/s can be guaranteed, the storage array itself will have practical limitations depending on the number of spindles[7] that back the storage pool/RAID group that feeds the LUNs.
- For Remote block storage acceptance criteria, please refer to The Cloudera Enterprise Storage Device Acceptance Criteria Guide. This guide is the cloudera artifact that will articulate supportable/acceptable performance guidelines for storage device support.

Network Topology Considerations

The preferred network topology is spine-leaf with no more than 4:1 oversubscription[8] between leaf and spine switches, ideally aiming for no oversubscription[9].

Let us step through an example scenario to better understand this. Assuming that this is a greenfield setup:

- The cluster comprises of 40 worker nodes (VMs), each with 5 SATA drives.
    - This implies that the ideal IO throughput (for planning) of the cluster would be ~ 20GB/s, and a per-VM throughput of 500MB/s.
    - This is the equivalent of 8 Gbps of network throughput per VM (2 x 500 MB/s x 8)
    - This infrastructure could be built with 20 2-socket nodes each with 20 Gbps of network bandwidth and 10 SATA drives (for data).

The best network topology for Hadoop clusters is spine-leaf. Each rack of hardware has its own leaf switch and each leaf is connected to each spine switch. Ideally we would not like to have any oversubscription between spine and leaf. That would result in having full line rate between any two nodes in the cluster. However, beyond a certain size in clusters (more than 500 nodes), having upto 4:1 oversubscription is acceptable, since the cost of maintaining 1:1 oversubscription gets higher as a cluster scales to greater than 500 nodes.

[7] This point was considering mainly the backend spindle count. In shared storage arrays, the L1 and L2 caches can easily be overrun by hadoop workloads. Typically in SAN topologies, network bandwidth is not the concern. Due to the need for multiple (typically 2 for fault tolerance) fabrics, there is sufficient capacity built in. In case of Fiber Channel, 2 x 8 gbps is standard, 2 x 16 gbps more prevalent these days and if using iSCSI, 2 x 10 gbps is the norm.
[8] Going over 4:1 oversubscription results in bottlenecks in E-W traffic patterns, which in turn end up impacting performance of the cluster overall.
[9] While it is ideal to have no oversubscription, there are practical limitations that determine its feasibility, such as budget, existing physical space, and so on..

The choice of switches, bandwidth, and other networking equipment would be predicated on the calculations in the previous section.

If we are to build the desired network topology based on the sizing exercise from above, we would need the following.

Network Per-port Bandwidth	Number of Ports required	Notes
25	37	Per sizing exercise, we needed 20 Gbps per node. However, to simplify the design, let us pick single 25 gbps ports instead of bonded pair of 10gbps per node.

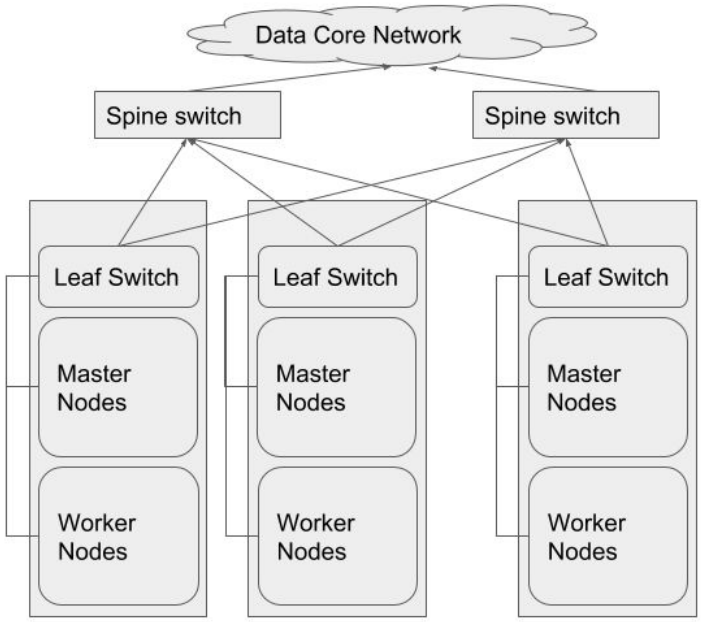
Assuming all the nodes are 2 RU in form factor, we would require 3 racks to house this entire set up, and leave enough room for growth.
If we place the nodes from each layer as evenly distributed across the 3 Racks as we can, we would end up with following configuration.

Rack1	Rack2	Rack3
Spine (18 x 100 gbps uplink + 2 x 100 gbps to Core)		Spine (18 x 100 gbps uplink + 2 x 100 gbps to Core)
ToR (12 x 25 gbps + 6 x 100 gbps uplink)	ToR (12 x 25 gbps + 6 x 100 gbps uplink)	ToR (13 x 25 gbps + 6 x 100 gbps uplink)
1 x Master	1 x Master	1 x Master
11 x Worker	11 x Worker	12 x Worker

So that implies that we would have to choose ToR or Leaf (Top of Rack) switches that have at least 13 x 25 gbps ports and 6 x 100 gbps uplink ports. Also the Spine switches would need at least 20 x 100 gbps ports.

Using six 100 gbps uplinks from the leaf switches would result in almost 1:1 oversubscription ratio between leaves (upto 13 x 25 Gbps ports) and the spine (3 x 100 gbps per spine switch).

Mixing the Workload and Storage nodes in the way shown below, will help localize some traffic to each leaf and thereby reduce the pressure of N-S traffic (between Workload and Storage clusters) patterns.



NOTE: For sake of clarity, the spine switches have been shown outside the racks.

Defining the IaaS architecture

Virtualization Design details

Hypervisor definition

This section defines two types of hypervisors and what is involved in their selection.

Node Type	CPU	Memory	Storage	Network
Worker	As sized[10]	As sized[3]	Storage - Single virtual disk per physical disk, locally attached or remote block storage with appropriate SAN	Non-SDN[11]

[10] "As Sized" here is predicated on a sizing exercise undertaken, or at least based on the recommendations provided in Cloudera's Hardware Requirements Guide.
[11] The SDN and Non-SDN demarcation has to do with leveraging the physical networking with minimal virtualization overhead. With SDN features enabled, such as using VXLAN, there is a encapsulation-decapsulation penalty to be paid at the network layer, and that may negatively impact performance.

			adapters (bandwidth, etc)	
Master	As sized	As sized	WL - RAID5 of multiple locally attached disks or remote block storage	non-SDN

Instance type definition

Instance Name/Type	vCPUs	Memory	Root Disk	Additional Storage
cdh-worker	>=8 or as sized[12]	>= 32GB or As sized	400GB	N x Storage Volumes
cdh-master	16	>= 64GB	400GB	N x Master Volumes

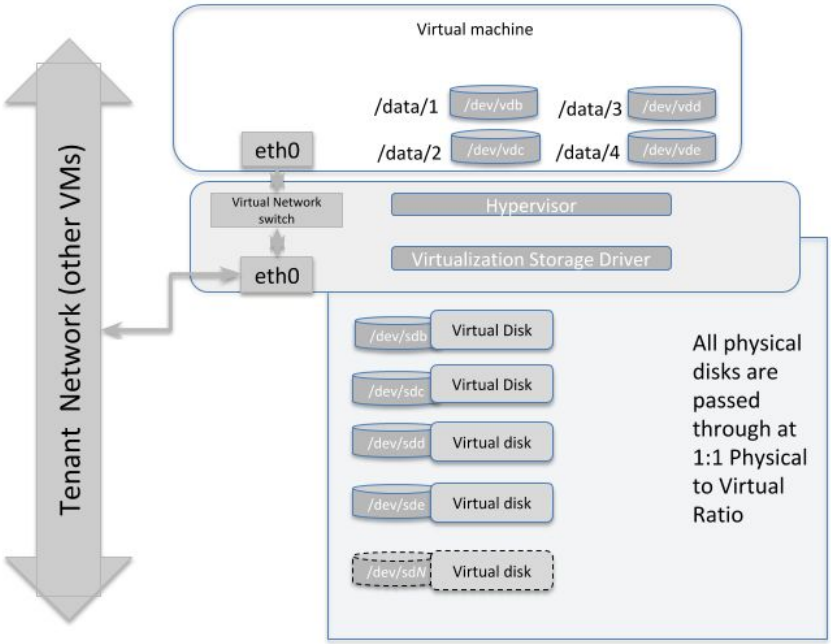
NOTE:

- Different sizes for CPU/Memory are possible for workload clusters and more than one instance type of cdh-worker might be defined depending on the workloads that will be run on them.
- Cloudera's Hardware Requirements Guide provides a good starting point for ascertaining minimum dimensions of these instances.

[12] "As Sized" here is predicated on a sizing exercise undertaken, or at least based on the recommendations provided in Cloudera's Hardware Requirements Guide.

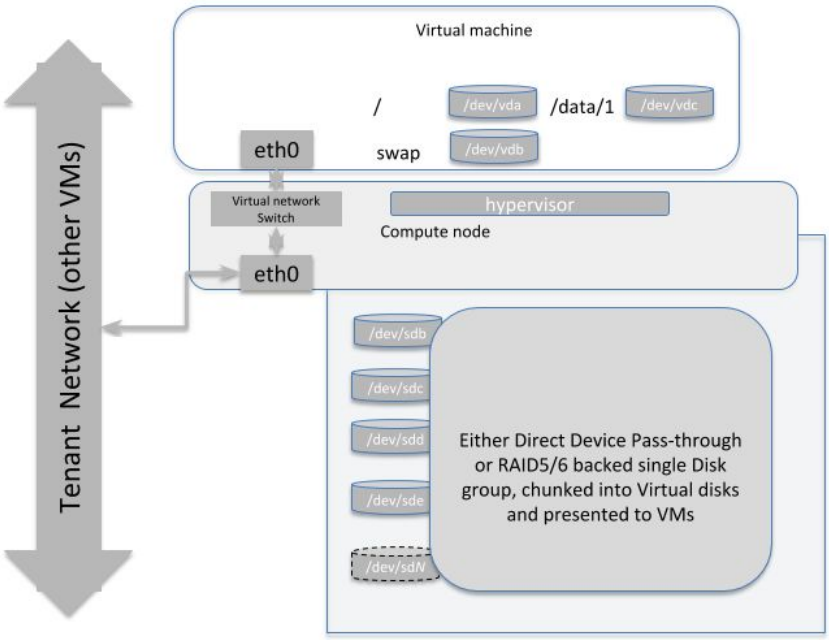
Storage Architecture

DAS Nodes



The diagram above illustrates how the storage subsystem of a worker-type node should be designed. The best practice is to obtain hardware with directly attached storage, and pass those devices through to the virtual machines running in this environment with a physical to virtual ratio of 1:1. So each physical spindle at the hypervisor level, maps to a virtual disk spindle at the VM level.

Master Nodes



The diagram above illustrates how a master-type hypervisor system should be designed. The virtual disks provisioned on these VMs can either be Direct Device Pass-throughs as in the storage nodes, or can be a Virtual disk carved out of a RAID5/6[13] backed disk group (at hypervisor layer).

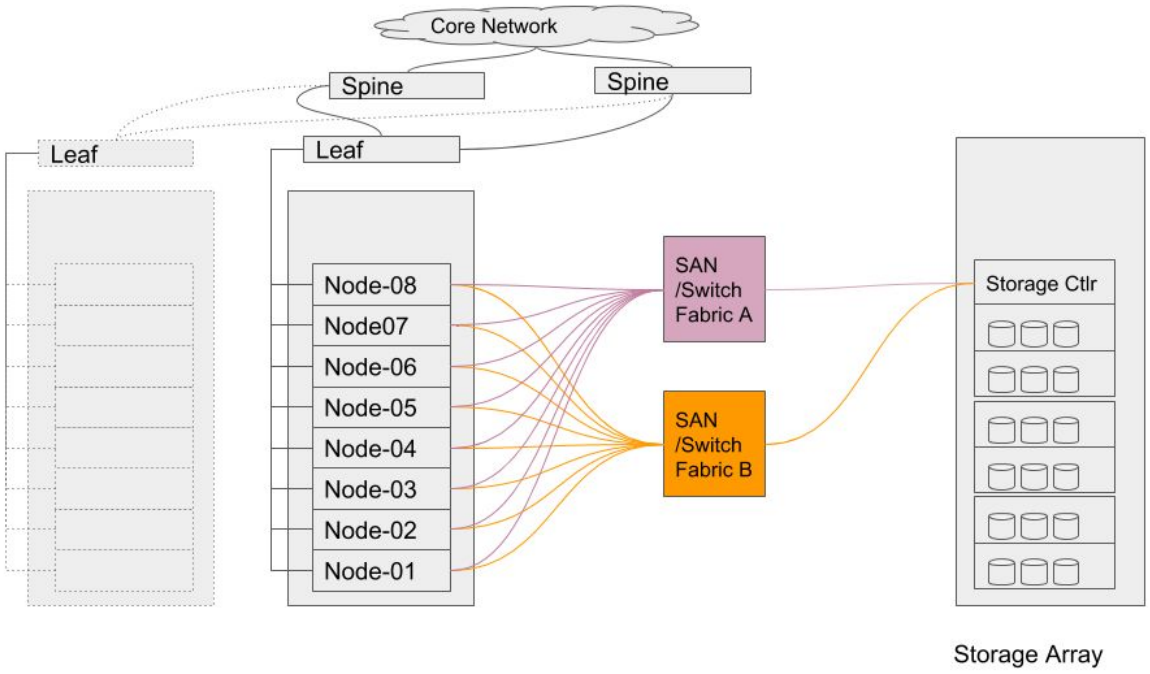
Remote Block Storage based VMs

In SAN-based deployments there are typically two SAN fabrics, which are fault tolerant and most modern SAN HBAs can be effectively multipathed to provide load-balancing and higher aggregate bandwidth, depending on the SAN Storage Array controller (whether it is Active/Active or Active/Passive). The hypervisor layer needs to be aware of this topology and be configured to leverage best performance from the storage backend.

It is also possible to leverage more modern, distributed storage platforms, where-in drivers specific to the Software Defined Storage (SDS) system might need to be installed at the hypervisor layer. The storage

[13] The RAID level used here should be well thought through, depending on the number of spindles available at the hypervisor layer. If 6-8 disks are being used, then RAID5 or RAID6 would make sense, as they will provide both data protection as well as leverage the spindle count to provide higher throughput.

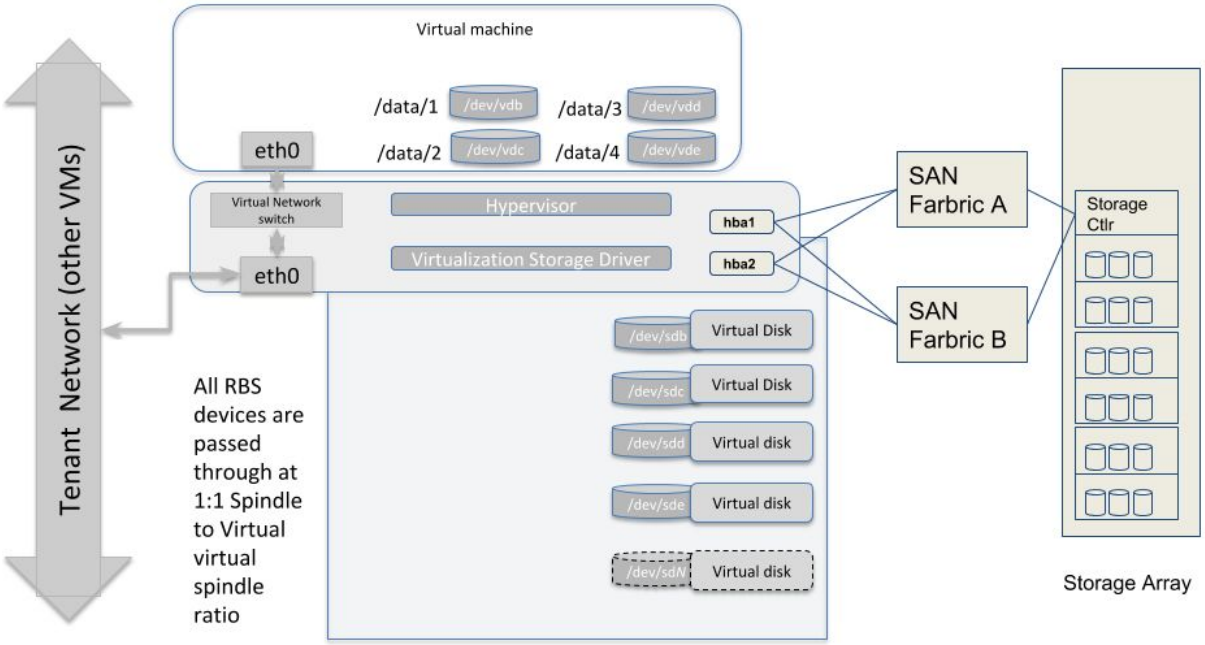
then is presented to the VM layer over Ethernet/iSCSI (or similar protocols).



A typical SAN-based topology would resemble the diagram shown above.

With remote block storage devices (RBS) , there may not be a need for as many spindles at the VM level as we would require with locally attached storage. However, the RBS device should be able to provide adequate IO throughput to meet the minimum requirements already articulated for each type of node (masters and workers). And also, it would be operationally prudent[14] to maintain a 1:1 mapping of RBS device to virtual storage device.

[14] This makes troubleshooting storage hotspotting etc easier to track. Maintaining a 1:1 relationship between LUN and Virtual device ensures that device mappings don't span Guest VM boundaries.


Virtual machine
/data/1
/dev/vdb
/data/3
/dev/vdd
/data/2
/dev/vdc
/data/4
/dev/vde
eth0
Virtual Network switch
Hypervisor
Virtualization Storage Driver
eth0
hba1
hba2
Tenant Network (other VMs)
All RBS devices are passed through at 1:1 Spindle to Virtual virtual spindle ratio
/dev/sdb
Virtual Disk
/dev/sdc
Virtual Disk
/dev/sdd
Virtual disk
/dev/sde
Virtual disk
/dev/sdN
Virtual disk
SAN Farbric A
SAN Farbric B
Storage Ctlr
Storage Array

Moreover, attention must be paid to running stress tests on the infrastructure as articulated in the Storage Acceptance Guide, such that a baseline be established showing that the minimum performance characteristics are being achieved when running workloads in a distributed manner. Such tests should be run at regular intervals to ensure that acceptable SLAs are being maintained by the underlying infrastructure as clusters evolve in their compute needs or as data volumes and data velocities grow..

Cloudera Software stack

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

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 Worker Nodes (VMs)

The Worker Node count depends on the required size of HDFS storage to deploy. The following document identifies service roles for different node types -- Recommended Cluster Hosts and Role Distribution.

Follow the guidelines in the Virtualization design details section to provision instance types.

- Ensure that CPU and Memory resources are not overcommitted while provisioning these node instances on the virtualized infrastructure.
- Automated movement of VMs must be disabled. There should be no Migration/Live Migration of VMs allowed in this deployment model.
- Master Nodes should be provisioned on disparate physical hardware; if possible configure them in separate physical racks or configure anti-affinity between the Master node VMs so they cannot be colocated on the same VM server..

Enabling Hadoop Virtualization Extensions (HVE)

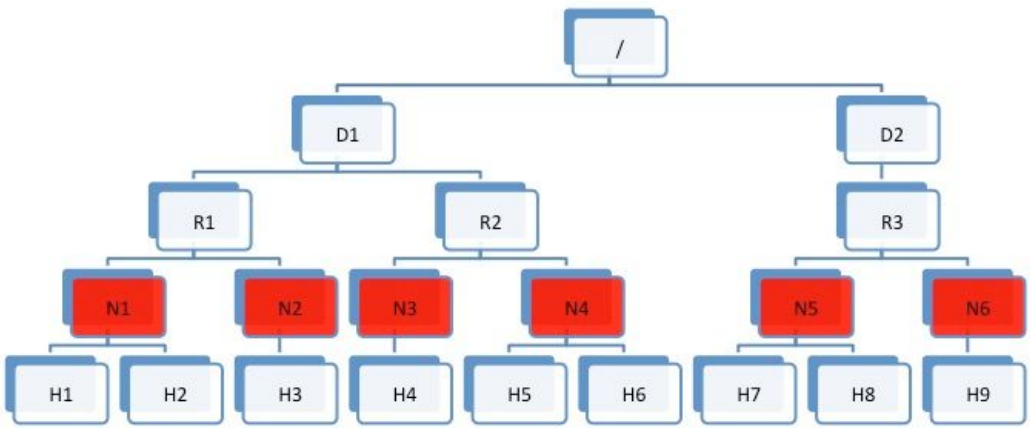
NOTE: While this document refers to hypervisors and virtual machines, this methodology is applicable to all any scenario where a "shared" something is involved. This is a strategy to help mitigate single points of failure, be it a shared power supply, a shared chassis, a shared storage tray, and so on.

Referring to the HDFS-side HVE JIRA (HADOOP-8468), the 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 5

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):

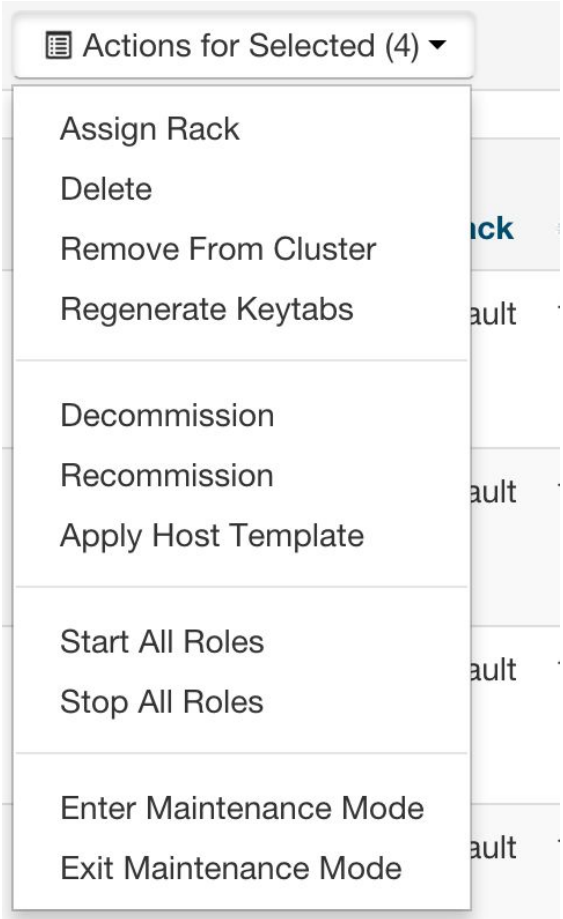
```
<property>
     <name>net.topology.impl</name>
       <value>org.apache.hadoop.net.NetworkTopologyWithNodeGroup<
       /value>
</property>
<property>
     <name>net.topology.nodegroup.aware</name>
     <value>true</value>
</property>
<property>
     <name>dfs.block.replicator.classname</name>
 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

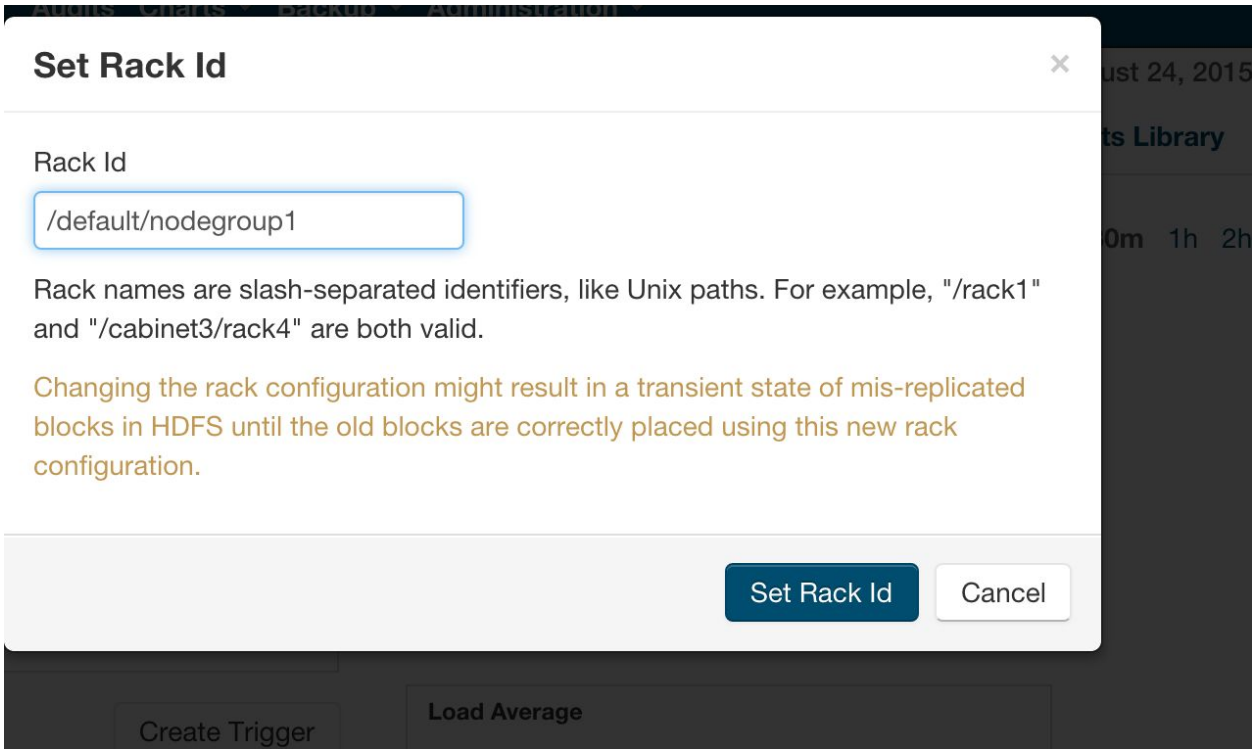
```
<property>
        <name>mapred.jobtracker.nodegroup.aware</name>
        <value>true</value>
</property>
<property>
        <name>mapred.task.cache.levels </name>
        <value>3</value>
</property>
```

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

- Cloudera Installation Guide
- Cloudera Hardware Requirements Guide
- Cloudera Enterprise Storage Device Acceptance Criteria Guide
- Cloudera Bare-Metal Reference Architecture
- Cloudera Security Guide

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.
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.
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 Availability	Configuration that addresses availability issues in a cluster. In a standard 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 and above (YARN-18). The HDFS component is supported in CDH 5.4 and above.
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.
NodeManager	The process that starts application processes and manages resources on the DataNodes.

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.
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.

Apache License

Version 2.0, January 2004

http://www.apache.org/licenses/

TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION

1. Definitions.

"License" shall mean the terms and conditions for use, reproduction, and distribution as defined by Sections 1 through 9 of this document.

"Licensor" shall mean the copyright owner or entity authorized by the copyright owner that is granting the License.

"Legal Entity" shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common control with that entity. For the purposes of this definition, "control" means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity.

"You" (or "Your") shall mean an individual or Legal Entity exercising permissions granted by this License.

"Source" form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files.

"Object" form shall mean any form resulting from mechanical transformation or translation of a Source form, including but not limited to compiled object code, generated documentation, and conversions to other media types.

"Work" shall mean the work of authorship, whether in Source or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work (an example is provided in the Appendix below).

"Derivative Works" shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an original work of authorship. For the purposes of this License, Derivative Works shall not include works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof.

"Contribution" shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, "submitted" means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to communication on electronic mailing lists, source code

control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as "Not a Contribution."

"Contributor" shall mean Licensor and any individual or Legal Entity on behalf of whom a Contribution has been received by Licensor and subsequently incorporated within the Work.

2. Grant of Copyright License.

Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare Derivative Works of, publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form.

3. Grant of Patent License.

Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those patent claims licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s) with the Work to which such Contribution(s) was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses granted to You under this License for that Work shall terminate as of the date such litigation is filed.

4. Redistribution.

You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions:

1. You must give any other recipients of the Work or Derivative Works a copy of this License; and
2. You must cause any modified files to carry prominent notices stating that You changed the files; and
3. You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and
4. If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License.

You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as

a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License.

5. Submission of Contributions.

Unless You explicitly state otherwise, any Contribution intentionally submitted for inclusion in the Work by You to the Licensor shall be under the terms and conditions of this License, without any additional terms or conditions. Notwithstanding the above, nothing herein shall supersede or modify the terms of any separate license agreement you may have executed with Licensor regarding such Contributions.

6. Trademarks.

This License does not grant permission to use the trade names, trademarks, service marks, or product names of the Licensor, except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of the NOTICE file.

7. Disclaimer of Warranty.

Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License.

8. Limitation of Liability.

In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall any Contributor be liable to You for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising as a result of this License or out of the use or inability to use the Work (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if such Contributor has been advised of the possibility of such damages.

9. Accepting Warranty or Additional Liability.

While redistributing the Work or Derivative Works thereof, You may choose to offer, and charge a fee for, acceptance of support, warranty, indemnity, or other liability obligations and/or rights consistent with this License. However, in accepting such obligations, You may act only on Your own behalf and on Your sole responsibility, not on behalf of any other Contributor, and only if You agree to indemnify, defend, and hold each Contributor harmless for any liability incurred by, or claims asserted against, such Contributor by reason of your accepting any such warranty or additional liability.

END OF TERMS AND CONDITIONS

APPENDIX: How to apply the Apache License to your work

To apply the Apache License to your work, attach the following boilerplate notice, with the fields enclosed by brackets "[]" replaced with your own identifying information. (Don't include the brackets!) The text should be enclosed in the appropriate comment syntax for the file format. We also recommend that a file or class name and description of purpose be included on the same "printed page" as the copyright notice for easier identification within third-party archives.

```
  Copyright [yyyy] [name of copyright owner]

   Licensed under the Apache License, Version 2.0 (the "License");
   you may not use this file except in compliance with the License.
   You may obtain a copy of the License at

     http://www.apache.org/licenses/LICENSE-2.0

   Unless required by applicable law or agreed to in writing,
software
   distributed under the License is distributed on an "AS IS" BASIS,
   WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or
implied.
   See the License for the specific language governing permissions
and
   limitations under the License.
```