Red Hat Reference Architecture Series

Deploying Cloudera 5

on Red Hat Enterprise Linux 6

Jacob Liberman, Red Hat

Woody Christy, Cloudera

Version 1.0

March 2015

redhat.

100 East Davie Street
Raleigh NC 27601 USA
Phone: +1 919 754 3700
Phone: 888 733 4281
Fax: +1 919 754 3701
PO Box 13588
Research Triangle Park NC 27709 USA

The GPG fingerprint of the security@redhat.com key is:
CA 20 86 86 2B D6 9D FC 65 F6 EC C4 21 91 80 CD DB 42 A6 0E

Send feedback to refarch-feedback@redhat.com

Comments and Feedback

In the spirit of open source, we invite anyone to provide feedback and comments on any reference architectures. Although we review our papers internally, sometimes issues or typographical errors are encountered. Feedback allows us to not only improve the quality of the papers we produce, but allows the reader to provide their thoughts on potential improvements and topic expansion to the papers.

Feedback on the papers can be provided by emailing refarch-feedback@redhat.com. Please refer to the title within the email.

Staying In Touch

Join us on some of the popular social media sites where we keep our audience informed on new reference architectures as well as offer related information on things we find interesting.

Like us on Facebook:

https://www.facebook.com/rhrefarch

Follow us on Twitter:

https://twitter.com/RedHatRefArch

Plus us on Google+:

https://plus.google.com/u/0/b/114152126783830728030/

Table of Contents

1 Executive Summary

Cloudera, the leader in enterprise analytic data management powered by Apache Hadoop®, and Red Hat, Inc., the world's leading provider of open source solutions have partnered to deliver a jointly engineered reference architecture of Cloudera Enterprise on Red Hat Enterprise Linux (RHEL) 6.5. This solution delivers a reliable, high-performance and comprehensive enterprise data management platform for both physical and virtual environments.

While this reference architecture reflects an integrated solution, each component provides its own distinct advantages, namely:

- Red Hat Enterprise Linux – A mission-critical open platform for the enterprise data center with extensive global ecosystem of support and unparalleled stability and flexibility.
- Cloudera Enterprise – A mission critical Hadoop-based platform that provides advanced system and data management tools with the enterprise capabilities necessary to succeed with Apache Hadoop

Aside from the individual value of the components, both Cloudera and Red Hat provide dedicated support and community advocacy from our world-class teams of developers and experts. The recommendations in this guide are derived from both Cloudera and Red Hat's respective Engineering, Field Technical Services, and Proactive Support teams that provide round-the-clock support throughout the private and public sectors.

The first section of this paper introduces the principal technologies used in this reference architecture: Cloudera CDH 5 and Red Hat Enterprise Linux Server 6. Next the paper describes hardware selection and sizing guidelines while planning a Cloudera installation. The third section of the paper shares best practices for configuring RHEL servers prior to Cloudera installation. The paper concludes with instructions for validating the installation and optimizing it for production use.

2 Solution Overview

This document explores integration points between Red Hat and Cloudera products. It describes how to deploy CDH 5.1.3 on Red Hat Enterprise Linux 6.5 to create an Enterprise Data Hub. The recommendations in this guide are derived from shared best practices between Cloudera and Red Hat's respective engineering, field services, and support teams. By combining Cloudera with RHEL, customers can deploy a state of the art analytics platform on top of rock solid core data center infrastructure software.

2.1 Cloudera Product Overview

Over the years, organizations have built solutions aimed at specific business problems: relational databases for transaction processing, data warehouse systems for analysis and exploration, document management systems for storing and searching business documents, and application solutions such as ERP and CRM to run major functional areas in the company. At the same time, in volume, in variety and in velocity, there is more data streaming in than ever before. As data grows it becomes increasingly burdensome to move it around for each new business question or form of analysis.

Many organizations are attacking these challenges head on by building out a new capability in their data centers. They deploy a new platform, an Enterprise Data Hub (EDH), that puts data at the center of the enterprise to give them the power and flexibility to be information-driven at superior price relative to traditional data management offerings. A CDH is one place to store all data, for as long as desired or required, in its original fidelity; integrated with existing infrastructure and tools; with the flexibility to run a variety of enterprise workloads—including batch processing, interactive SQL, enterprise search, and advanced analytics—together with the robust security, governance, data protection, and management that enterprises require.

This section of the paper describes the Cloudera products used to create an EDH in this reference architecture.

2.1.1 Cloudera Enterprise

At the core of Cloudera Enterprise is CDH, which combines Apache Hadoop with a number of other Apache Licensed open source projects to create a single, massively scalable system that unites storage with an array of powerful processing and analytic frameworks.

2.1.2 Automated Cluster Management – Cloudera Manager

Cloudera Enterprise includes Cloudera Manager to easily deploy, manage, monitor, and diagnose issues with the cluster. Cloudera Manager is critical for operating clusters at scale. Cloudera Manager is used in this reference architecture to deploy CDH.

2.2 Red Hat Product Overview

This section describes some of the technologies that can be used in conjunction with Cloudera CDH.

2.2.1 Red Hat Enterprise Linux

Red Hat Enterprise Linux (RHEL) is a commercially supported Linux operating system tailored to meet the requirements of enterprise customers. Business-critical applications need a platform that is proven to be stable. RHEL Server fulfills core operating system functions and includes additional capabilities that provide a firm foundation for application infrastructure.

RHEL Server 6 includes many enhancements and new functionality related to efficiency, scalability, and reliability. These include efficient scheduling using the Completely Fair Scheduler (CFS), active state power management, scalability to large core counts and memory, high availability, and enterprise storage, networking, and file system support. All of these features are built on a thoroughly tested Linux code base and backed by Red Hat's technical support and engineering.

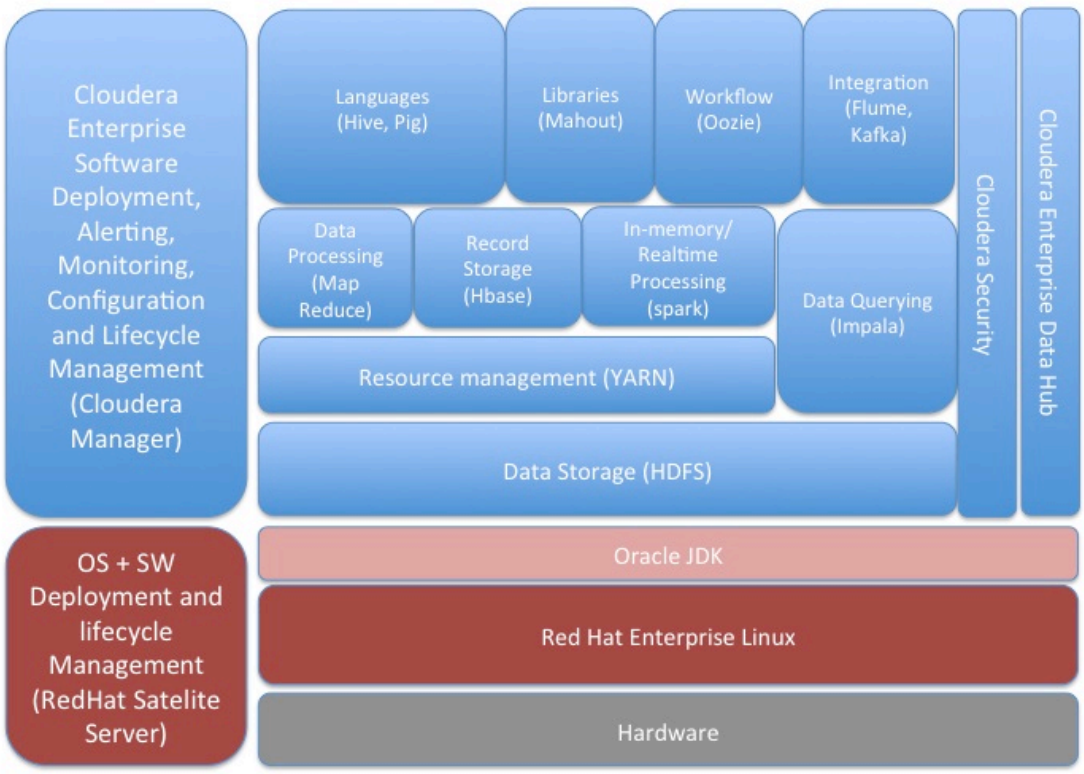
2.2.2 Red Hat Satellite Server

Red Hat Satellite server is an easy-to-use, advanced life-cycle management platform for Red Hat Enterprise Linux infrastructure. Providing the tools to efficiently deploy, update, monitor, and manage systems, Red Hat Satellite helps reduce repetitive and time-consuming tasks.

Red Hat Satellite server makes RHEL deployable, scalable, and manageable. It is ideally suited to deploy and manage large scale Hadoop cluster installations. Additionally, Red hat Satellite can be configured to cache frequently used packages within the datacenter. This alleviates network bandwidth bottlenecks often associated with large scale systems deployment and management.

2.3 Cloudera on Red Hat Enterprise Linux

In this diagram Cloudera provides the components in blue and Red Hat provides all of the red components. The JDK is provided by Cloudera as part of the installation of Cloudera Manager and its agents.



Graphic 2.1: Cloudera + RHEL Stack

3 Architectural Considerations

This section of the paper discusses high-level considerations for planning a Cloudera Enterprise on RHEL cluster installation including system design, cluster sizing, and server hardware selection.

3.1 System Architecture

This section describes, at a high level, Cloudera's recommendations for Hadoop cluster system architecture. Most items are described in more detail in later sections.

3.1.1 Utility Server

The Utility Server runs core Hadoop services. The utility server is a good place to install client configurations for access to the cluster, such as for YARN, Hive, HBase, Spark and Sqoop. The utility server is also the right location to install management and monitoring services, including Cloudera Manager. Cloudera recommends using a dedicated server for the Utility server role.

3.1.2 Right-size Server Configurations

Cloudera recommends deploying three machine types into production:

Master Node

The Master Node runs the master daemons:

- NameNode
- SecondaryNameNode (Standby NameNode in HA)
- ResourceManager (and a Standby Resource Manager in HA).
- HBase Master daemon

Master nodes are also the location where Zookeeper and JournalNodes are installed for use by HBase, and HA for ResourceManager and NameNode. The daemons can run on a single server, or split and run in separate servers, depending on the cluster size. These nodes also run the Impala processes of Catalog Server and State Store.

Worker Node

The Worker Nodes run the Hadoop and HBase worker daemons:

- DataNode
- Nodemanager
- RegionServer

Utility Node

The Utility Node provides core services such as DNS and NFS if needed. It can also host a MySQL (or another supported) database instance, which is leveraged by Hive and other Hadoop-related projects. The utility node may also be used for Cloudera Manager and Hue.

3.1.3 Cluster Architecture

Refer to the appendix for cluster architecture diagrams. We have multiple cluster architecture recommendations depending on the number of worker nodes. The following diagram shows a standard layout of software installation. Cloudera Manager may be installed on one of the masters or a dedicated utility node. Utility nodes are added as a way to scale out client access

to the cluster. If more users are running ingest jobs from utility nodes they should be scaled as demand requires.

	Master Node 1	Master Node 2	Master Node 3	Worker nodes 1..n
ZooKeeper	ZooKeeper	ZooKeeper	ZooKeeper	
HDFS	Name Node, Quorum Journal Node	Name Node, Quorum Journal Node	Quorum Journal Node	Data Node
YARN	ResourceManager	ResourceManager	History Server	NodeManager
Hive			MetaStore, WebHCat, HiveServer2	
Management	Cloudera Agent	Cloudera Agent	Cloudera Agent, Oozie, Cloudera Manager, Management Services	Cloudera Agent
Navigator			Navigator, Key Management Services	
HUE			HUE	
HBASE	HMaster	HMaster	HMaster	RegionServer

For most workloads 3 masters work for at least 50 worker nodes. As demand scales master nodes should be added and ZooKeeper nodes scaled out in odd numbers. Typically 5 nodes can handle several hundred servers for most workloads.

3.1.4 Dedicated Network Hardware

Hadoop may consume all available network bandwidth. For this reason, Cloudera recommends that Hadoop be placed in a separate physical network with its own core switch.

3.1.5 Switch Per Rack

Hadoop supports the concept of rack locality and takes advantage of the network topology to minimize network congestion. Ideally, nodes in one rack should connect to one or two physical switches. Each rack switch (i.e., top-of-rack or TOR switch) links up to a core switch with a significantly bigger backplane. Cloudera recommends 10GbE for the TOR switches with 1 or more 40GbE uplinks to the core switch.

3.1.6 Redundant Core Switches

Having redundant core switches in a full mesh configuration allows the cluster to continue operating in the event of a core switch failure.

3.1.7 NIC Bonding

Most standard rack servers have 2x 10 Gbe network interfaces built-in to the motherboard. These can be bonded, which effectively doubles the bandwidth to the TOR switch. There are quite a few caveats to make multi-homing work on a Hadoop system and Cloudera does not recommend that configuration.

3.1.8 Use Cloudera 5

Cloudera 5 is the latest version of the Cloudera's Distribution including Apache Hadoop (CDH) as part of a Cloudera Enterprise subscription, which includes the most up-to-date stable versions of Hadoop and HBase, Cloudera Impala, as well as the other Hadoop ecosystem components.

CDH is open source and freely available. CDH is a prerequisite for a Cloudera Enterprise support contract.

3.2 Cluster Sizing

The following are the key Hadoop cluster sizing considerations when determining total available storage:

- The total raw storage (in GB/TB) for each worker node's dedicated HDFS drives is used to calculate the total available storage for the cluster.

- These calculations assume some percentage of disk space is allocated for job temporary storage. Cloudera recommends allocating between 15% and 25% of the raw disk space as a general guideline.
- The calculations assume some factor of HDFS replication. Cloudera strongly recommends that the default replication factor (3x) be utilized.
- Settings for HDFS Storage and YARN, Spark and Impala Intermediate Storage (temporary) can be configured to meet the specific needs of customers, through Cloudera Manager. This is recommended after analyzing cluster performance during operational workloads.

The following formula would be used to calculate total usable storage for HDFS:

$$Total\ Capacity = \frac{(n \cdot h \cdot c \cdot a)}{r}$$

Where:

n=number of worker nodes

h=number of hard drives per machine

c=capacity of hard drives

a=portion of drive allocated to HDFS

r=HDFS replication factor

The following table provides a couple examples of this calculation.

Number of Worker Nodes	Number of Hard Drives Per Machine	Size of Hard Drive	Default Replication Factor	Reserved for intermediate data	Total Usable
50	12	2 TB	3	20%	320 TB
25	24	1 TB	3	20%	160 TB

3.3 Cluster Hardware Selection

This section gives a high-level overview of how different hardware component selections impact the performance of a Hadoop cluster.

3.3.1 Number of Spindles

Hadoop is a giant I/O platform. Unlike the number of cores in a CPU and the density of RAM, the speed at which data can be read from a hard drive spindle has not changed much in the last 10 years. In order to counter the slowness of hard drive read/write speed, Hadoop reads and writes from many drives in parallel. Every additional spindle added per node increases the overall read/write speed of the cluster.

Additional spindles also come with the likelihood of more network traffic in the cluster. For the majority of cases, network traffic between nodes is limited by how fast data can be written to or read from a node. Therefore, the rule normally follows that network speed requirements increase with more spindles. The Teragen benchmark is a good performance test for network capacity. Teragen uses about *three to nine times* more network capacity than that of normal cluster operations.

Generally speaking, the more spindles a node has, the lower the cost per TB. However, care should be taken with this information, as the more data stored on one node yields longer recovery time when a node goes down. Hadoop clusters are designed to have many nodes. It is generally better to have more average nodes than fewer super nodes. More nodes increase data protection and parallelism for jobs that run inside the cluster.

Cloudera recommends, for Impala and other I/O intensive frameworks, 16-24 drives which provides a good trade off on density versus I/O per node.

3.3.2 Data Density Per Drive

Hard drives come in many sizes. Popular drive sizes are one, two, and three terabytes. Consider the following when selecting a hard drive model for the cluster:

- **Lower Cost Per TB.** A rule of thumb is: the larger the drive, the cheaper the price per TB, which makes for better ROI.
- **Replication Storms.** Larger drives mean drive failures produce larger re-replication storms, which can take longer and saturate the network.
- **Cluster Performance.** Drive size has little impact on cluster performance. The exception is when drives have different read/write speeds and a use case that leverages this gain. Hadoop is designed for long sequential reads and writes, so latency timings are generally not as important. HBase can potentially benefit from faster drives, but that is dependent on a variety of factors, such as HBase access patterns and schema design. Spending extra for faster drives typically does not lead to good ROI.

3.3.3 Number of Cores and Multithreading

Other than cost, there is no negative for buying more and better CPUs. However, the ROI on additional CPU power must be evaluated carefully. Here are some points to consider:

- **Cluster Bottleneck.** In general, CPU resources (and lack thereof) do not bottleneck YARN and HBase. The performance bottleneck is usually drive and/or network performance. There are certainly exceptions to this, such as inefficient Hive queries.
- **Additional Cores/Threads.** Within a given job, a single task typically uses one thread at a time. As outlined earlier, the number of slots allocated per node is often a function of the number of drives in the node. As long as there is not a huge disparity in the number of cores (threads) and the number of drives, it does not make sense to pay for additional cores. In addition, a YARN task for typical jobs is going to be mostly performing I/O operations, thus a given thread used by the task has a large amount of idle time while waiting for I/O response. Spark jobs that have data cached in a Resilient Distributed Dataset tend to be CPU and memory bound
- **Clock Speed.** Additional CPU clock speed is not a good investment unless the types of YARN jobs are CPU-intensive, which is rare. However, price points on CPUs may warrant an upgrade choice and allow for more flexibility to future workload types. The faster CPU cores tend to have lower density so trading off more cores for slower clock spend might be advisable. A good rule of thumb is roughly two hyper threaded cores per spindle to have a balanced CPU to IO profile.

3.3.4 RAM

Other than cost, more memory is always good, so it is recommended to purchase as much as the budget allows. Cloudera recommends on Impala clusters to have at least 128 GB of RAM. The below breakdown table lists the amount of memory that is typically needed/allocated for each worker node in the cluster:

Item	RAM Allocated	RAM Recommended
Operating System	2 GB	2 GB
DataNode	1 GB	4 GB
NodeManager	1 GB	4 GB
NodeManager Child Tasks	1 - 2 GB / task	1 - 2 GB / task
RegionServer	4 GB	16 GB
Impala	48 GB	Workload dependent

NOTE: More memory aids YARN during sort. It is also leveraged for uses cases such as tree building, grid traversal, and windowing.

3.3.5 Network Bandwidth

Networking is a critical component for a Hadoop cluster. Typically, 1GbE is not enough bandwidth for production Hadoop cluster nodes. Most production cluster nodes have 10Gbe. Speeds beyond 2x10Gbe bonded are generally not necessary for worker nodes.

As mentioned earlier, Teragen tests the cluster's network capacity. Compression optimizes network performance by effectively increases the amount of data that can be sent across the wire. Typical compression codecs are Snappy and GZip.

3.3.6 Power Supplies

Hadoop software is designed around the expectation that nodes fail. An extra power supply is meant to reduce failure but at a cost: additional power supplies, power cables, and electrical sockets. Some hardware designs also mean that additional power supplies increase energy costs. While it is generally not important to utilize redundant power supplies for worker nodes, Cloudera recommends redundant power supplies for master nodes. This is even more critical for non-HA configurations.

3.4 Hardware Specification

This section contains a series of tables that provide a detailed breakout of the physical hardware for the Hadoop cluster including server and networking hardware.

3.4.1 Network Specification

Cloudera recommends a ratio of total access port bandwidth to uplink bandwidth as close to 1:1 as economically possible. This is especially important for heavy ETL workloads and MapReduce jobs that have a lot of data sent to reducers in the shuffle phase. Heavy joins in Spark and Impala also generate a lot of network traffic. Ratios up to 3:1 are fine if network monitoring ensures uplink bandwidth is not the bottleneck for Hadoop.

The following table provides some examples as a point of reference:

Access Port Bandwidth (In Use)	Uplink Port Bandwidth (Bonded)	Ratio
48 x 1GigE = 48 Gbit/s	4 x 10GigE = 40 Gbit/s	1.2:1
24 x 10GigE = 240 Gbit/s	2 x 40Gig CFP = 80 Gbit/s	3:1
48 x 10GigE = 480 Gbit/s	4 x 40Gig CFP = 160 Gbit/s	3:1

4 Implementation

4.1 Hadoop Deployment Strategy

This section is a reference for deployment configurations.

4.1.1 Master Nodes

The Master Nodes should have dedicated volumes for each of ZooKeeper, Quorum Journal Node, HDFS Name Node metadata, and for any database services. Besides the boot volume, 4 other dedicated volumes should be provided.

4.1.2 Utility Nodes

The utility nodes serve as the configuration management point in the cluster, a client to the Hadoop cluster, as well as a staging area for new data to be ingested. The following management and user-facing processes should be installed:

- Cloudera Manager Server
- Oozie
- Hue/Beeswax
- HttpFS
- Flume

The utility nodes do not run core Hadoop daemons. They are configured as gateway nodes within Cloudera Manager and configured for client access to the cluster. Additionally, the following CDH components are installed and configured on the utility nodes by Cloudera Manager:

- Beeline
- Pig
- Sqoop
- Impala Shell

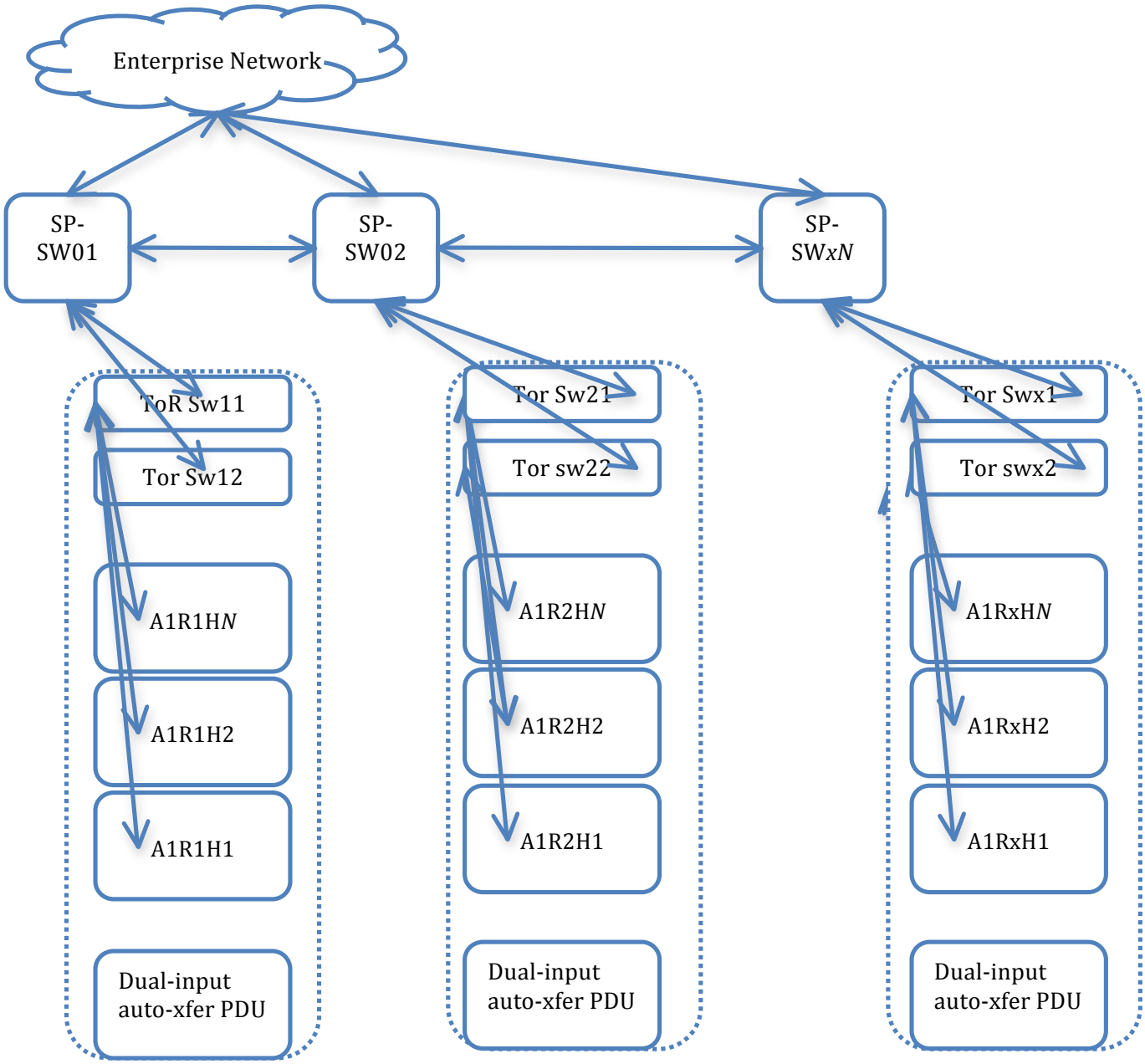
4.1.3 Worker Nodes

Each worker node in the cluster runs three Hadoop daemons: HDFS DataNode, YARN NodeManager, HBase Region Server and Impala Impalad.

> **NOTE:** If HBase is chosen for installation, it is required that YARN virtual cores and memory be lowered in order to minimize impact on performance. The system should be monitored via Cloudera Manager to ensure that YARN jobs are not starving HBase for resources.

4.1.4 Server and Network Topology

This section displays a diagram/graphic that shows the basic server and network topology for the cluster.



Graphic 4.1.4: Server and Network Topology

4.2 Install Red Hat Enterprise Linux 6.5

Install Red Hat Enterprise Linux Server on all servers that participate in the cluster.

> **NOTE:** The process for installing RHEL Server is documented at: https://access.redhat.com/documentation/en-US/Red_Hat_Enterprise_Linux/6/pdf/Installation_Guide/Red_Hat_Enterprise_Linux-6-Installation_Guide-en-US.pdf.

4.2.1 Operating System Version

Cloudera Manager and CDH components require a supported operating system. Red Hat Enterprise Linux 6.5 is supported for Cloudera CDH 5.1.3.

Verify the operating system version:

```
[root@cdh0 ~]# cat /etc/redhat-release
Red Hat Enterprise Linux Server release 6.5 (Santiago)
```

A 64-bit version of the operating system is necessary to utilize system memory over 4GB:

```
[root@cdh0 ~]# uname -r
2.6.32-431.el6.x86_64
```

> **NOTE:** Cloudera's support site lists supported operating systems: http://www.cloudera.com/content/cloudera/en/documentation/cdh5/latest/CDH5-Requirements-and-Supported-Versions/cdhrsv_os.html.

4.2.2 Use a Local Red Hat Satellite Server

The network bandwidth required to install a large number of servers in parallel can be a significant deployment bottleneck. Installing from a local Red Hat Satellite server can reduce deployment time.

> **NOTE:** Instructions for installing Red Hat Satellite Server: https://access.redhat.com/documentation/en-US/Red_Hat_Satellite/6.0/html/Installation_Guide/index.html.

4.3 Configure RHEL for Cloudera

This section describes how to prepare Red Hat Enterprise Linux hosts for a Cloudera installation. It includes steps required to complete an installation via Cloudera Manager as well as steps intended to optimize performance post-installation.

4.3.1 Hostname Naming Convention

Cloudera recommends using a hostname convention that allows for easy recognition of roles and/or physical connectivity. This is especially important for easily configuring rack awareness within Cloudera Manager. Using a project name identifier, followed by the rack ID, the machine class, and a machine ID is an easy way to encode useful information about the cluster. For example:

```
acme-test-r01m01
```

This hostname would represent the Acme customer's test project, rack #1, master node #1.

The hostnames used in this reference architecture are of the form *<company>-<project>-<rack><role>*.

Set the hostname using the **hostname** command:

```
[root@rh-ra-r01m01 ~]# hostname rh-ra-r01m01

[root@rh-ra-r01m01 ~]# hostname
rh-ra-r01m01
```

Set the hostname in */etc/sysconfig/network* to ensure that it persists after reboot:

```
[root@rh-ra-r01m01 ~]# cat /etc/sysconfig/network
NETWORKING=yes
HOSTNAME=rh-ra-r01m01
```

4.3.2 Name Resolution

Cloudera recommends using DNS for hostname resolution. All hosts in the cluster must have forward and reverse lookups be the inverse of each other for Hadoop to function properly. An easy test to perform on the hosts to ensure proper DNS resolution is to execute **dig** against both the host name and IP address:

```
[root@rh-ra-r01m01 ~]# dig $(hostname -s)
; <<>> DiG 9.8.2rc1-RedHat-9.8.2-0.17.rc1.el6_4.6 <<>> rh-ra-r01m01
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NXDOMAIN, id: 33851
;; flags: qr rd ra; QUERY: 1, ANSWER: 0, AUTHORITY: 1, ADDITIONAL: 0

;; QUESTION SECTION:
;rh-ra-r01m01.                  IN    A

;; AUTHORITY SECTION:
.               10800 IN    SOA   a.root-servers.net. nstld.verisign-grs.com.
2014101401 1800 900 604800 86400

;; Query time: 39 msec
;; SERVER: 10.19.143.247#53(10.19.143.247)
;; WHEN: Tue Oct 14 13:33:47 2014
;; MSG SIZE rcvd: 105

[root@rh-ra-r01m01 ~]# dig -x 10.19.137.100
; <<>> DiG 9.8.2rc1-RedHat-9.8.2-0.17.rc1.el6_4.6 <<>> -x 10.19.137.100
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 17309
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 2, ADDITIONAL: 2

;; QUESTION SECTION:
;100.137.19.10.in-addr.arpa.     IN    PTR

;; ANSWER SECTION:
100.137.19.10.in-addr.arpa. 86400 IN   PTR   rh-ra-
r01m01.cloud.lab.eng.bos.redhat.com.

;; AUTHORITY SECTION:
137.19.10.in-addr.arpa.     86400 IN    NS
     refarch.cloud.lab.eng.bos.redhat.com.
```

```
137.19.10.in-addr.arpa.    86400 IN    NS    ra-
ns1.cloud.lab.eng.bos.redhat.com.

;; ADDITIONAL SECTION:
ra-ns1.cloud.lab.eng.bos.redhat.com. 86400 IN A   10.19.143.247
refarch.cloud.lab.eng.bos.redhat.com. 86400 IN A 10.19.143.248

;; Query time: 1 msec
;; SERVER: 10.19.143.247#53(10.19.143.247)
;; WHEN: Tue Oct 14 13:33:47 2014
;; MSG SIZE rcvd: 174
```

The usage of */etc/hosts* becomes cumbersome quickly, and it routinely is the source of hard-to-diagnose problems. */etc/hosts* should only contain an entry for 127.0.0.1, and `localhost` should be the only name that resolves to it. If an */etc/hosts* file is used, it should be in the following format:

```
127.0.0.1    localhost localhost.localdomain
10.19.137.100  rh-ra-r01m01.cloud.lab.eng.bos.redhat.com rh-ra-r01m01
10.19.137.101  rh-ra-r01m02.cloud.lab.eng.bos.redhat.com rh-ra-r01m02
10.19.137.102  rh-ra-r01m03.cloud.lab.eng.bos.redhat.com rh-ra-r01m03
10.19.137.103  rh-ra-r01u01.cloud.lab.eng.bos.redhat.com rh-ra-r01u01
10.19.137.107  rh-ra-r01d01.cloud.lab.eng.bos.redhat.com rh-ra-r01d01
10.19.137.108  rh-ra-r01d02.cloud.lab.eng.bos.redhat.com rh-ra-r01d02
10.19.137.109  rh-ra-r01d03.cloud.lab.eng.bos.redhat.com rh-ra-r01d03
...
```

> **NOTE:** The IP address in the left-most column should be immediately followed by the server Fully Qualified Domain Name (FQDN) and 127.0.0.1 should only resolve to *localhost*. This is a common, but costly area where mistakes are made. The machine name must not resolve to the 127.0.0.1 address.

4.3.3 Time

All machines in the cluster need to have the same time and date settings, including time zones. Use of the Network Time Protocol (NTP) is highly recommended. HBase is very sensitive to time, and, as such, all HBase roles need to be synchronized across the cluster. The following commands enable the NTP daemon:

```
[root@rh-ra-r01m01 ~]# chkconfig ntpdate on

[root@rh-ra-r01m01 ~]# chkconfig ntpd on

[root@rh-ra-r01m01 ~]# service ntpdate start
ntpdate: Synchronizing with time server:          [ OK ]

[root@rh-ra-r01m01 ~]# service ntpd start
Starting ntpd:                            [ OK ]
```

4.3.4 Name Server Caching

It is recommended that name server caching be enabled for clusters of 50 nodes or more and for clusters that use non-local Hadoop functional accounts, such as the `hdfs` and `mapred` users. This becomes critical in the case where the latter is combined with using Kerberos. Many difficult-to-diagnose problems can arise when name server lookups timeout or fail during heavy cluster utilization.

Enable the Name Server Caching Daemon:

```
[root@rh-ra-r01m01 ~]# yum install -y -q nscd

[root@rh-ra-r01m01 ~]# service nscd start
Starting nscd:                             [ OK ]

[root@rh-ra-r01m01 ~]# chkconfig nscd on
```

4.3.5 SELinux

At this time, Cloudera requires SELinux to be disabled on all machines in the Hadoop cluster. Disable SELinux on RHEL by editing `/etc/selinux/config` and setting `SELINUX=disabled`. This change must be done as root (or with proper **sudo** access) and requires a reboot.

```
[root@rh-ra-r01m01 ~]# sed -i -e 's/=permissive/=disabled/g' \
/etc/selinux/config

[root@rh-ra-r01m01 ~]# shutdown -r now
```

Verify SELinux is disabled:

```
[root@rh-ra-r01m01 ~]# sestatus
SELinux status:          disabled

[root@rh-ra-r01m01 ~]# cat /etc/selinux/config
# This file controls the state of SELinux on the system.
# SELINUX= can take one of these three values:
#    enforcing - SELinux security policy is enforced.
#    permissive - SELinux prints warnings instead of enforcing.
#    disabled - SELinux is fully disabled.
SELINUX=disabled
# SELINUXTYPE= type of policy in use. Possible values are:
#    targeted - Only targeted network daemons are protected.
#    strict - Full SELinux protection.
SELINUXTYPE=targeted
```

4.3.6 IPv6

Hadoop does not support IPv6. IPv6 configurations should be removed, and IPv6-related services should be stopped.

4.3.7 IPTables

Cloudera recommends disabling **iptables** on the cluster. Many difficult problems to diagnose result from incorrect/conflicting table entries that interfere with normal cluster communication.

Flush and save **iptables** rules:

```
[root@rh-ra-r01m01 ~]# iptables -F

[root@rh-ra-r01m01 ~]# service iptables save
iptables: Saving firewall rules to /etc/sysconfig/iptables:[ OK ]
```

Stop and disable **iptables**:

```
[root@rh-ra-r01m01 ~]# service iptables stop
iptables: Setting chains to policy ACCEPT: filter      [ OK ]
iptables: Flushing firewall rules:                     [ OK ]
```

```
iptables: Unloading modules:                [ OK ]

[root@rh-ra-r01m01 ~]# chkconfig iptables off

[root@rh-ra-r01m01 ~]# service ip6tables stop

[root@rh-ra-r01m01 ~]# chkconfig ip6tables off
```

4.3.8 Startup Services

As with any production server, unused services should be removed and disabled. Some example services that are on by default in a standard RHEL that are not needed by CDH include:

- bluetooth
- cups
- iptables
- ip6tables
- postfix

This list is not exhaustive. Examine enabled services with chkconfig and disable those that are not needed.

Disable unused services:

```
[root@rh-ra-r01m01 ~]# for i in autofs bluetooth cups nfslock portreserve
postfix rpcbind rpcgssd iptables ip6tables
do
    chkconfig $i off && service $i stop
done

Stopping NFS locking:                       [ OK ]
Stopping NFS statd:                         [ OK ]
Shutting down postfix:                      [ OK ]
Stopping rpcbind:                           [ OK ]
Stopping RPC gssd:                          [ OK ]
```

4.3.9 Process Memory

The memory on each node is allocated out to the various Hadoop processes. This predictability reduces the chance of Hadoop processes inadvertently running out of memory and so paging to disk, which, in turn, leads to severe degradation in performance.

It is critical to performance that the total memory allocated to *all* Hadoop-related processes (including processes such as HBase) is less than the total memory on the node, taking into account the operating system and non-Hadoop processes. It can also be harmful to performance to unnecessarily over-allocate memory to a Hadoop process as this can lead to long Java garbage collection pauses.

Reserve a minimum of 20 percent of memory on all nodes for operating system and other non-Hadoop use. This amount should be raised if additional non-Hadoop applications are running on the cluster nodes, such as third-party active monitoring/alerting tools.

Verify free memory with the **free** command:

```
[root@rh-ra-r01m01 ~]# free -g
             total       used       free     shared    buffers     cached
Mem:            29          7         22          0          0          6
-/+ buffers/cache:          0         28
Swap:           0          0          0
```

Memory requirements and allocation for Hadoop components are discussed in detail in later sections.

4.3.10 Reduce Swappiness

Hadoop depends on heavy usage of RAM resources on machines in the cluster. If pages in RAM are swapped to disk, Hadoop performance degrades heavily. This is especially the case with HBase to the extreme that RegionServers can be marked down if delays are present due to excessive paging. As such, the parent OS needs to be tuned to reduce the likelihood that swapping to disk occurs.

The kernel parameter `vm.swappiness` controls this setting. The possible value ranges are between 0 and 100, where 0 means the OS is less likely to swap to disk, and 100 means the OS is more likely to swap to disk. By default, Linux distributions ship with `vm.swappiness` set to as high as 60. Cloudera recommends this be set at 0.

To change this setting on a running machine, execute the command as root:

```
[root@rh-ra-r01m01 ~]# sysctl -w vm.swappiness=0
vm.swappiness = 0

[root@rh-ra-r01m01 ~]# sysctl -a | grep vm.swappiness
vm.swappiness = 0
```

Modify */etc/sysctl.conf* file by adding *vm.swappiness = 0* to make this change persist across reboots:

```
[root@rh-ra-r01m01 ~]# echo "vm.swappiness=0" >> /etc/sysctl.conf
```

4.3.11 Apply tuned Profile

Red Hat Enterprise Linux 6 contains a number of pre-defined performance profiles that can be enabled with the tuned-adm command. For typical Hadoop workloads, Red Hat performance engineering recommends the *throughput-performance* profile. This profile makes the following changes:

- Disables tuned and ktune power save mechanisms
- cpuspeed mode changed to performance
- deadline I/O elevator is used for each device
- cpu_dma_latency parameter is registered a value of 0
- kernel.sched_min_granularity_ns is set to 10ms
- kernel.sched_wakeup_granularity_ns *is set to 15ms*
- vm.dirty_ratio is set to 40%
- Transparent Huge Pages are enabled

  NOTE: Although this profile enables THP, this feature should be disabled. Following the steps outlined in the "Disable Transparent huge Pages" section below.

To apply the *throughput-performance* profile:

```
[root@rh-ra-r01m01 ~]# yum -y -q install tuned

[root@rh-ra-r01m01 ~]# tuned-adm profile throughput-performance
Switching to profile 'throughput-performance'
Applying deadline elevator: dm-0 sda sdb sdc sdd           [  OK  ]
Applying ktune sysctl settings:
/etc/ktune.d/tunedadm.conf:                                [  OK  ]
Calling '/etc/ktune.d/tunedadm.sh start':                  [  OK  ]
```

```
Applying sysctl settings from /etc/sysctl.conf
Starting tuned:                                            [  OK  ]
```

Verify the profile is applied:

```
[root@rh-ra-r01m01 ~]# tuned-adm active
Current active profile: throughput-performance
Service tuned: enabled, running
Service ktune: enabled, running
```

> **NOTE:** Not all Hadoop workloads benefit from the throughput-performance profile. Workloads should be tested on an individual basis. Enable this profile only where appropriate. More information on tuned-adm can be found at https://access.redhat.com/documentation/en-US/Red_Hat_Enterprise_Linux/6/html/Performance_Tuning_Guide/ch07s02.html.

4.3.12 Disable Transparent Huge Pages

Transparent Huge Pages (THP) is a feature designed to boost performance for applications with a large memory footprint by allocating larger memory pages at boot. Cloudera recommends disabling Transparent Huge Pages as it has a negative impact on almost all Hadoop workloads.

Disable THP through */boot/grub/grub.conf*:

```
[root@rh-ra-r01m01 ~]# sed -i.orig 's/rhgb quiet/rhgb quiet
transparent_hugepage=never/' /boot/grub/grub.conf
```

The following lines should also be added to */etc/rc.local* to ensure that THP are disabled after services start:

```
if test -f /sys/kernel/mm/transparent_hugepage/enabled
then
     echo never > /sys/kernel/mm/transparent_hugepage/enabled
fi

if test -f /sys/kernel/mm/transparent_hugepage/defrag
then
     echo never > /sys/kernel/mm/transparent_hugepage/defrag
fi
```

Verify THP are disabled:

```
[root@rh-ra-r01m01 ~]# sysctl -a | grep hugepage
vm.nr_hugepages = 0
vm.nr_hugepages_mempolicy = 0
vm.hugepages_treat_as_movable = 0
vm.nr_overcommit_hugepages = 0

[root@rh-ra-r01m01 ~]# cat
/sys/kernel/mm/redhat_transparent_hugepage/enabled
always madvise [never]

[root@rh-ra-r01m01 ~]# grep -i AnonHugePages /proc/meminfo
AnonHugePages:         0 kB
```

> **NOTE:** More information on THP can be found at: https://access.redhat.com/documentation/en-US/Red_Hat_Enterprise_Linux/6/html/Performance_Tuning_Guide/s-memory-transhuge.html

4.3.13 Hard Disks

In Linux there are several choices for formatting and organizing drives. That being said, only a few choices are optimal for Hadoop.

Here are guidelines for optimizing hard disks for Hadoop under RHEL 6:

1. The Logical Volume Manager (LVM) should never be used. It is not optimal for Hadoop and can lead to combining multiple drives into one logical disk, which is in complete contrast to how Hadoop manages fault tolerance across HDFS.
2. Cloudera recommends using an extent-based file system. This includes `ext3`, `ext4`, and `xfs`. For RHEL 6 Cloudera recommends using ext4.
3. HDFS is a fault tolerant file system. As such, all drives used by DataNode machines for data should be mounted without the use of RAID.
4. Drives should be mounted in `/etc/fstab` using the `noatime` option (which also implies `nodiratime`). Mounting disks with this option reduces disk I/O because disk reads no longer read the access time information.

Additionally, for ease of administration, it is recommended to mount all of the disks on the DataNode machines with a naming pattern. For example, in this reference architecture, the following naming scheme was used on the data nodes:

```
[root@rh-ra-r01d01 ~]# grep data /etc/fstab
/dev/sdb1 /data01 ext4 defaults,noatime 0 0
/dev/sdc1 /data02 ext4 defaults,noatime 0 0
/dev/sdd1 /data03 ext4 defaults,noatime 0 0
/dev/sde1 /data04 ext4 defaults,noatime 0 0
/dev/sdf1 /data05 ext4 defaults,noatime 0 0
/dev/sdg1 /data06 ext4 defaults,noatime 0 0
/dev/sdh1 /data07 ext4 defaults,noatime 0 0
/dev/sdi1 /data08 ext4 defaults,noatime 0 0
/dev/sdj1 /data09 ext4 defaults,noatime 0 0
/dev/sdk1 /data10 ext4 defaults,noatime 0 0
/dev/sdl1 /data11 ext4 defaults,noatime 0 0
/dev/sdm1 /data12 ext4 defaults,noatime 0 0

[root@rh-ra-r01d01 ~]# mount -v | grep data
/dev/sdb1 on /data01 type ext4 (rw,noatime)
/dev/sdc1 on /data02 type ext4 (rw,noatime)
/dev/sdd1 on /data03 type ext4 (rw,noatime)
/dev/sde1 on /data04 type ext4 (rw,noatime)
/dev/sdf1 on /data05 type ext4 (rw,noatime)
/dev/sdg1 on /data06 type ext4 (rw,noatime)
/dev/sdh1 on /data07 type ext4 (rw,noatime)
/dev/sdi1 on /data08 type ext4 (rw,noatime)
/dev/sdj1 on /data09 type ext4 (rw,noatime)
/dev/sdk1 on /data10 type ext4 (rw,noatime)
/dev/sdl1 on /data11 type ext4 (rw,noatime)
/dev/sdm1 on /data12 type ext4 (rw,noatime)
```

4.4 Install CDH

Cloudera recommends installing Cloudera 5 using Cloudera Manager. During the Cloudera 5 installation via CM, there is the choice to install using parcels or native packages. A parcel is a new, binary distribution format supported in Cloudera Manager version 4.5 or greater. Parcels offer a number of benefits, including consistency, flexible installation location, installation without sudo, reduced upgrade downtime, rolling upgrades, and easy downgrades. Cloudera recommends using parcels, although using packages is still supported.

The following installation guide contains instructions for installing CDH via Cloudera Manager:
http://www.cloudera.com/content/cloudera/en/documentation/core/latest/topics/installation_installation.html

In this reference archiecture the base server roles for a *Core Hadoop* installation were accepted. This option installs core Hadoop services, YARN, Oozie, Hive, Hue, Sqoop, Zookeeper, and HDFS. **Graphic 4.4.2.1: Server Roles by Host** shows the server roles assigned by Cloudera Manager. In this example there are only three Data Nodes – *rh-ra-r01-d0[1-3]* – but this could be extended up to 50 for a small deployment. *rh-ra-r01m01* is the Name Node and *rh-ra-r01m02* is the Secondary Name Node.

View By Host

This table is grouped by hosts having the same roles assigned to them.

Hosts	Count	Existing Roles	Added Roles
rh-ra-r01d01.cloud.lab.eng.bos.redhat.com	1		DN G HMS HS2 NM S
rh-ra-r01d[02-03].cloud.lab.eng.bos.redhat.com	2		DN G NM
rh-ra-r01m01.cloud.lab.eng.bos.redhat.com	1		NN B G HMS HS2 HS SM HM RM ES AP OS S2S RM JHS S
rh-ra-r01m02.cloud.lab.eng.bos.redhat.com	1		SNN G
rh-ra-r01m03.cloud.lab.eng.bos.redhat.com; rh-ra-r01u01.cloud.lab.eng.bos.redhat.com	2		G

Graphic 4.4.2.1: Server Roles by Host

The Cloudera Manager installation guide includes complete descriptions of server roles and recommdations for role placement: http://www.cloudera.com/content/cloudera/en/documentation/core/latest/topics/installation_installation.html

4.5 Optimize CDH

This section contains information about how the cluster is configured, including Cloudera recommendations specific to the customer hardware being used. This section is not an exhaustive description of every configuration, but rather a focus on important and/or configurations that have been changed from the default setting.

4.5.1 Cloudera Manager Configuration

Cloudera Manager supports the usage of several different mechanisms for user authentication and authorization. By default, an admin user is created during Cloudera Manager installation. Additional users can be added either as administrators or as regular users. Cloudera Manager also supports integration with Active Directory and OpenLDAP directory services. Furthermore, Cloudera Manager (as of 4.1.2) supports a general external authentication mechanism to use a custom script developed by the customer.

Cloudera Manager and CDH make use of dedicated functional accounts for the associated daemon processes. By default these accounts are created as local accounts on every machine in the cluster that needs them if they do not already exist (locally or from a directory service, such as LDAP). The following account names are used:

- hdfs
- mapred
- hbase
- hive

- hue
- httpfs
- oozie
- sqoop
- flume
- zookeeper
- yarn
- cloudera-scm

If LDAP or some other directory service is used for these accounts, see the section on name server caching.

4.5.2 Database

By default, Cloudera Manager uses an embedded PostgreSQL database for storage. Note: Cloudera recommends, for customers deploying a production system, not to use the embedded databases, as migration paths for converting from these (e.g., PostgreSQL) to another type (e.g., MySQL) can be problematic. In these cases, it is preferable that customers utilize an external database for deployment otherwise known as Installation path B in the install guides.

It is recommended to use the same type of database across all services that require one. Below is a summary that shows all of the databases used on the cluster and their defaults.

Service Name	Default Database Type
Cloudera Manager (if applicable)	Embedded PostgreSQL
Activity Monitor	Embedded PostgreSQL
Host Monitor	Embedded PostgreSQL
Report Manager	Embedded PostgreSQL
Service Monitor	Embedded PostgreSQL
Hue	Embedded SQLite
Hive Metastore (Beeswax)	Embedded Derby
Oozie	Embedded Derby
Navigator	Embedded PostgreSQL

4.5.3 HDFS

4.5.3.1 Java Heap Sizes

NameNode memory should be increased over time as more blocks and files are stored in HDFS. Cloudera Manager can monitor and alert on memory usage. A rough estimate is that the NameNode needs 1 GB of memory for every 1 million files. Setting the heap size too large when it is not needed leads to inefficient Java garbage collection, which can lead to erratic behavior that is hard to diagnose. NameNode and Secondary/Standby NameNode heap sizes must always be the same, and thus must be adjusted together.

The below table shows the recommended HDFS daemon heap sizes to be used on the cluster, based upon the hardware used:

Daemon	Cloudera Manager Default
NameNode	1 GB – 10 GB (calculated)
DataNode	1 GB

4.5.3.2 NameNode Metadata Locations

Cloudera recommends that NameNode daemons write metadata to multiple locations. A quorum-based HA NameNode configuration should be used and the JournalNodes handle the storage of metadata writes. The NameNode daemons require a local location to store metadata as well. Cloudera recommends that only a single directory be used if the underlying disks are configured as RAID, or two directories on different disks if the disks are mounted as JBOD.

4.5.3.3 Block Size

HDFS stores files in blocks that are distributed over the cluster. A block is typically stored contiguously on disk to provide high read throughput. The choice of block size influences how long these high throughput reads run, and over how many nodes a file is distributed. When reading the many blocks of a single file, a too low block size spends more overall time in slow disk seek, and a too high block size has reduced parallelism. Data processing that is I/O-heavy benefits from larger block sizes, and data processing that is CPU-heavy benefits from smaller block sizes.

Cloudera recommends the default block size of 128MB. The default provided by Cloudera Manager is 128MB. The block size is a client-overrideable configuration and can be specified on the command line during job submissions or within the code itself

4.5.3.4 Replication

Bottlenecks can occur on a small number of nodes when only small subsets of files on HDFS are being heavily accessed. Increasing the replication factor of the files so that their blocks are replicated over more nodes can alleviate this. This is done at the expense of storage capacity on the cluster. This can be set on individual files, or recursively on directories with the -R parameter, by using the Hadoop shell command `hadoop fs -setrep`. By default, the replication factor is 3.

4.5.3.5 Rack Awareness

Hadoop can optimize performance and redundancy when rack awareness is configured for clusters that span across multiple racks, and Cloudera recommends doing so. Rack assignments for nodes can be easily configured within Cloudera Manager. As mentioned in the hostname naming convention section, this configuration can be made easier if it is obvious to discern the rack by looking at the hostname.

4.5.3.6 DataNode Failed Volumes Tolerated

By default, a DataNode is considered down if one disk fails (a setting of zero). A single node going down causes a nontrivial amount of network traffic to accommodate the replicating of blocks that are on the failed node.

The failed volume threshold should be increased to half of the data drives in a DataNode. In other words, if each DataNode has eight drives dedicated to data storage, this threshold should be set to four, meaning that Hadoop marks the DataNode dead on the fifth drive failure. This number may need to be adjusted up or down depending on internal policies regarding hard drive replacements or because of evaluating what behavior is actually seen on the cluster under normal operating conditions. Setting the value too high has a negative impact on the Hadoop cluster. Specifically for YARN, the number of total container slots available on the node with many drive failures is the same as nodes without drive failures, meaning data locality is less likely on the former, leading to more network traffic and slower performance.

It is important to note that multiple drive failures in a short amount of time might be indicative of a larger problem with the machine, such as a failed disk controller.

4.5.3.7 DataNode Xciever Count

Xcievers are handlers in the DataNode that take care of sending and receiving block data. HBase tends to use a lot of Xcievers. If the cluster uses HBase, Cloudera recommends increasing the default dfs.datanode.max.xcievers from 256 to 2048. This can be safely increased further up to 8092 for larger clusters or heavier workloads.

Note that Cloudera Manager typically configures a high-enough Xciever count when an HBase service is deployed, but it should be verified.

4.5.3.8 Balancing

Hadoop tries to spread data evenly across the cluster in order to optimize read access, MapReduce, and node utilization. However, over time, the cluster can become out of balance due to a variety of reasons. Hadoop helps mitigate this by rebalancing data across the cluster using the Balancer tool. By default, Cloudera Manager configures the Balancer to rebalance when a given DataNode is utilized 10% more or less from the average utilization across the cluster. This is a good default, but is worth mentioning explicitly in this document. Running the Balancer is a manual process that can be executed from within Cloudera Manager as well as from the command line. This should be done regularly and during off-peak hours to minimize the impact it may have on normal cluster operation. Individual DataNode utilization can be viewed from within Cloudera Manager (if installed/utilized).

Going hand-in-hand with the above information is the maximum amount of bandwidth a DataNode uses for rebalancing purposes. By default, this is set to 10 MB/second (80 Mbit/second). The table below should be used to capture the network and balancing configuration information:

Setting	Value
Cluster Network Capacity	(e.g., 10 MB/second)
Balancer Bandwidth Setting	(e.g., 8 MB/second)
Percentage of Maximum Bandwidth	(e.g., 80%)

This ensures that the rebalancing process can finish faster, but still not dominate the maximum bandwidth on the network that is needed for normal cluster use (e.g., YARN or ingesting data to HDFS). This change, if made within Cloudera Manager, requires a restart of the HDFS service. However, Hadoop allows this change to be made instantly across all nodes without a configuration change by using the command:

```
hdfs dfsadmin -setBalancerBandwidth <bytes_per_second>
```

All dfsadmin commands must be run as the `hdfs` superuser. This is a convenient way to change the setting without restarting the cluster, but, since it is a dynamic change, it does not persist if the cluster is restarted.

Important:
Cloudera generally does not recommend running the Balancer on an HBase cluster as it affects data locality for the RegionServers, which can reduce performance. Unfortunately, when HBase and YARN services are collocated, and heavy usage is expected on both, there is not a good way to ensure the cluster is optimally balanced.

4.5.4 YARN

YARN is the next generation Resource management framework for Hadoop Clusters and allows for applications other than Map Reduce to run within the cluster. With finer grained resource control and scheduling mechanisms, YARN is used heavily within the Cloudera Stack to enable various Cloudera Hadoop sub-components to inter-operate with each other seamlessly.

4.5.4.1 Java Heap Sizes

The Java Heap size varies by server role. The defaults provided by Cloudera Manager are a good starting point and

Daemon	Cloudera Manager Default
ResourceManager	1 GB – 10 GB (calculated)
NodeManager	1 GB
NodeManager Child Tasks	1 GB

4.5.4.2 YARN Configuration

On a dedicated YARN cluster with a balanced workload, Cloudera recommends allocating [1.5 x # of physical CPU cores] virtual cores per NodeManager. If the CPU is using HyperThreading, Cloudera recommends allocating [.8 x # of total CPU cores].

For example, if all of machines have dual eight-core processors, for a total of 16 physical cores. The CPUs support

HyperThreading, which means the YARN virtual cores should be configured to be [.8 x 32 = ~26] virtual cores.

If the customer is also using HBase, and chooses to collocate HBase and Yarn services on the same physical hardware, this needs to be adjusted. Due to the batch-oriented nature of YARN, hardware resources on a given worker node are easily maximized under heavy load. This can cause severe performance degradation of, if not outright failures of, HBase and its components. As such, Cloudera recommends the following three options:

1. Provide a dedicated cluster for HBase
2. Separate the roles of NodeManagers and RegionServers such that they are not collocated
3. Leave the roles collocated and reduce the number of YARN virtual cores and memory configured

Option one may not be feasible due to budget constraints. Option two is more intrusive to both services because YARN loses a large portion of data locality, and HBase RegionServers are required to handle the same workload with fewer machines (RegionServers), which leads to less performance. Option three effectively throttles back YARN , but does not degrade its data locality. Additionally, HBase keeps the maximum number of RegionServers to it, allowing for the most optimal performance. Because of this, Cloudera recommends option three if option one is not possible.

It is important to note that the above calculations for YARN virtual cores and memory per NodeManager were based on a dedicated YARN cluster. If HBase RegionServers are to be collocated with NodeManagers, Cloudera recommends scaling back to between 0.5 and 0.75 times the number of physical cores. This can be adjusted upon observing the performance of HBase and YARN under typical workloads.

4.5.4.3 Schedulers

YARN has the ability to use three different job schedulers: FIFOScheduler, FairScheduler, and CapacityScheduler.

The FIFOScheduler, as the name implies, is a first-in first-out scheduler. It is the most basic scheduler available to YARN. It benefits from an easy configuration, but it has major drawbacks during high cluster activity. Long-running jobs can easily dominate resources over short-running jobs that are submitted afterwards.

The FairScheduler is the answer to the FIFOScheduler drawbacks. It allows for the creation of pools to assign jobs to. These pools can be configured in great detail to provide a granular SLA for various types of jobs, users, and groups.

The CapacityScheduler is similar to the FairScheduler, except that its primary purpose is to provide a guarantee on cluster resources. This scheduler is mostly used in large clusters where multiple lines of business and/or applications are sharing the cluster. This scheduler is the most effective for addressing these types of SLAs.

Cloudera recommends using the Fair Scheduler. This is default scheduler chosen by Cloudera Manager and is the only one that currently takes into consideration both memory and CPU when scheduling containers.

4.6 Validate the Installation

Use **`TeraGen`** and **`TeraSort`** to validate the installation. **`TeraSort`** is a standard **`MapReduce`** sorting benchmark. It takes data generated by **`TeraGen`** as input.

> **NOTE:** A complete description of these benchmarks can be found at:
> https://hadoop.apache.org/docs/current/api/org/apache/hadoop/examples/terasort/package-summary.html

4.6.1 Run TeraGen

`TeraGen` stresses the network subsystem. Run **`TeraGen`** as the *hdfs* user.

```
[root@rh-ra-r01m01 ~]# su - hdfs

[hdfs@rh-ra-r01m01 ~]$ which java
```

```
/usr/java/default/bin/java

[hdfs@rh-ra-r01m01 cdh]$ /usr/bin/hadoop jar /opt/cloudera/parcels/CDH-
5.1.3-1.cdh5.1.3.p0.12/lib/hadoop-0.20-mapreduce/hadoop-examples.jar teragen
-Dmapreduce.job.reduces=24 107374182 /user/hdfs/tsortin
14/10/23 09:52:09 INFO client.RMProxy: Connecting to ResourceManager at rh-
ra-r01m01.cloud.lab.eng.bos.redhat.com/10.19.137.100:8032
14/10/23 09:52:10 INFO terasort.TeraSort: Generating 107374182 using 2
14/10/23 09:52:10 INFO mapreduce.JobSubmitter: number of splits:2
14/10/23 09:52:10 INFO mapreduce.JobSubmitter: Submitting tokens for job:
job_1414001930025_0011
14/10/23 09:52:10 INFO impl.YarnClientImpl: Submitted application
application_1414001930025_0011
14/10/23 09:52:10 INFO mapreduce.Job: The url to track the job: http://rh-
ra-
r01m01.cloud.lab.eng.bos.redhat.com:8088/proxy/application_1414001930025_001
1/
14/10/23 09:52:10 INFO mapreduce.Job: Running job: job_1414001930025_0011
14/10/23 09:52:22 INFO mapreduce.Job: Job job_1414001930025_0011 running in
uber mode : false
14/10/23 09:52:22 INFO mapreduce.Job:  map 0% reduce 0%
14/10/23 09:52:38 INFO mapreduce.Job:  map 6% reduce 0%
14/10/23 09:52:41 INFO mapreduce.Job:  map 8% reduce 0%
14/10/23 09:52:42 INFO mapreduce.Job:  map 9% reduce 0%
14/10/23 09:52:44 INFO mapreduce.Job:  map 11% reduce 0%
14/10/23 09:52:45 INFO mapreduce.Job:  map 12% reduce 0%
14/10/23 09:52:47 INFO mapreduce.Job:  map 14% reduce 0%
14/10/23 09:52:48 INFO mapreduce.Job:  map 16% reduce 0%
14/10/23 09:52:50 INFO mapreduce.Job:  map 17% reduce 0%
14/10/23 09:52:51 INFO mapreduce.Job:  map 19% reduce 0%
14/10/23 09:52:53 INFO mapreduce.Job:  map 21% reduce 0%
14/10/23 09:52:54 INFO mapreduce.Job:  map 22% reduce 0%
14/10/23 09:52:56 INFO mapreduce.Job:  map 24% reduce 0%
14/10/23 09:52:57 INFO mapreduce.Job:  map 26% reduce 0%
14/10/23 09:53:00 INFO mapreduce.Job:  map 29% reduce 0%
14/10/23 09:53:03 INFO mapreduce.Job:  map 32% reduce 0%
14/10/23 09:53:06 INFO mapreduce.Job:  map 36% reduce 0%
14/10/23 09:53:09 INFO mapreduce.Job:  map 39% reduce 0%
14/10/23 09:53:12 INFO mapreduce.Job:  map 40% reduce 0%
14/10/23 09:53:13 INFO mapreduce.Job:  map 42% reduce 0%
14/10/23 09:53:15 INFO mapreduce.Job:  map 44% reduce 0%
14/10/23 09:53:16 INFO mapreduce.Job:  map 45% reduce 0%
14/10/23 09:53:18 INFO mapreduce.Job:  map 47% reduce 0%
14/10/23 09:53:19 INFO mapreduce.Job:  map 49% reduce 0%
14/10/23 09:53:21 INFO mapreduce.Job:  map 50% reduce 0%
14/10/23 09:53:22 INFO mapreduce.Job:  map 52% reduce 0%
14/10/23 09:53:24 INFO mapreduce.Job:  map 54% reduce 0%
14/10/23 09:53:25 INFO mapreduce.Job:  map 55% reduce 0%
14/10/23 09:53:27 INFO mapreduce.Job:  map 57% reduce 0%
14/10/23 09:53:28 INFO mapreduce.Job:  map 58% reduce 0%
14/10/23 09:53:30 INFO mapreduce.Job:  map 60% reduce 0%
14/10/23 09:53:31 INFO mapreduce.Job:  map 62% reduce 0%
14/10/23 09:53:33 INFO mapreduce.Job:  map 63% reduce 0%
14/10/23 09:53:34 INFO mapreduce.Job:  map 65% reduce 0%
14/10/23 09:53:36 INFO mapreduce.Job:  map 67% reduce 0%
```

```
14/10/23 09:53:37 INFO mapreduce.Job:  map 68% reduce 0%
14/10/23 09:53:39 INFO mapreduce.Job:  map 70% reduce 0%
14/10/23 09:53:40 INFO mapreduce.Job:  map 71% reduce 0%
14/10/23 09:53:42 INFO mapreduce.Job:  map 73% reduce 0%
14/10/23 09:53:43 INFO mapreduce.Job:  map 75% reduce 0%
14/10/23 09:53:45 INFO mapreduce.Job:  map 76% reduce 0%
14/10/23 09:53:46 INFO mapreduce.Job:  map 78% reduce 0%
14/10/23 09:53:48 INFO mapreduce.Job:  map 80% reduce 0%
14/10/23 09:53:49 INFO mapreduce.Job:  map 81% reduce 0%
14/10/23 09:53:51 INFO mapreduce.Job:  map 83% reduce 0%
14/10/23 09:53:52 INFO mapreduce.Job:  map 84% reduce 0%
14/10/23 09:53:54 INFO mapreduce.Job:  map 86% reduce 0%
14/10/23 09:53:55 INFO mapreduce.Job:  map 88% reduce 0%
14/10/23 09:53:57 INFO mapreduce.Job:  map 89% reduce 0%
14/10/23 09:53:58 INFO mapreduce.Job:  map 91% reduce 0%
14/10/23 09:54:00 INFO mapreduce.Job:  map 92% reduce 0%
14/10/23 09:54:01 INFO mapreduce.Job:  map 94% reduce 0%
14/10/23 09:54:05 INFO mapreduce.Job:  map 97% reduce 0%
14/10/23 09:54:08 INFO mapreduce.Job:  map 100% reduce 0%
14/10/23 09:54:08 INFO mapreduce.Job: Job job_1414001930025_0011 completed
successfully
14/10/23 09:54:08 INFO mapreduce.Job: Counters: 31
    File System Counters
          FILE: Number of bytes read=0
          FILE: Number of bytes written=214020
          FILE: Number of read operations=0
          FILE: Number of large read operations=0
          FILE: Number of write operations=0
          HDFS: Number of bytes read=170
          HDFS: Number of bytes written=10737418200
          HDFS: Number of read operations=8
          HDFS: Number of large read operations=0
          HDFS: Number of write operations=4
    Job Counters
          Launched map tasks=2
          Other local map tasks=2
          Total time spent by all maps in occupied slots (ms)=193675
          Total time spent by all reduces in occupied slots (ms)=0
          Total time spent by all map tasks (ms)=193675
          Total vcore-seconds taken by all map tasks=193675
          Total megabyte-seconds taken by all map tasks=198323200
    Map-Reduce Framework
          Map input records=107374182
          Map output records=107374182
          Input split bytes=170
          Spilled Records=0
          Failed Shuffles=0
          Merged Map outputs=0
          GC time elapsed (ms)=1503
          CPU time spent (ms)=214090
          Physical memory (bytes) snapshot=385581056
          Virtual memory (bytes) snapshot=3395674112
          Total committed heap usage (bytes)=1244659712
    org.apache.hadoop.examples.terasort.TeraGen$Counters
          CHECKSUM=230593859918397906
```

```
    File Input Format Counters
          Bytes Read=0
    File Output Format Counters
          Bytes Written=10737418200
```

4.6.2 Run TeraSort

TeraSort stresses the storage subsystem. Run **TeraSort** against the data created by **TeraGen** in the previous step.

```
[hdfs@rh-ra-r01m01 cdh]$ /usr/bin/hadoop jar /opt/cloudera/parcels/CDH-
5.1.3-1.cdh5.1.3.p0.12/lib/hadoop-0.20-mapreduce/hadoop-examples.jar
terasort -Dmapreduce.job.reduces=24 -
Dmapreduce.job.reduce.slowstart.completedmaps=1.0 -
Dmapreduce.input.fileinputformat.split.minsize=536870912
/user/hdfs/tsortin/part-* /user/hdfs/tsortout
14/10/23 09:55:55 INFO terasort.TeraSort: starting
14/10/23 09:55:56 INFO input.FileInputFormat: Total input paths to process :
2
Spent 152ms computing base-splits.
Spent 2ms computing TeraScheduler splits.
Computing input splits took 155ms
Sampling 10 splits of 20
Making 24 from 100000 sampled records
Computing parititions took 860ms
Spent 1017ms computing partitions.
14/10/23 09:55:57 INFO client.RMProxy: Connecting to ResourceManager at rh-
ra-r01m01.cloud.lab.eng.bos.redhat.com/10.19.137.100:8032
14/10/23 09:55:58 INFO mapreduce.JobSubmitter: number of splits:20
14/10/23 09:55:58 INFO mapreduce.JobSubmitter: Submitting tokens for job:
job_1414001930025_0012
14/10/23 09:55:58 INFO impl.YarnClientImpl: Submitted application
application_1414001930025_0012
14/10/23 09:55:58 INFO mapreduce.Job: The url to track the job: http://rh-
ra-
r01m01.cloud.lab.eng.bos.redhat.com:8088/proxy/application_1414001930025_001
2/
14/10/23 09:55:58 INFO mapreduce.Job: Running job: job_1414001930025_0012
14/10/23 09:56:10 INFO mapreduce.Job: Job job_1414001930025_0012 running in
uber mode : false
14/10/23 09:56:10 INFO mapreduce.Job:  map 0% reduce 0%
14/10/23 09:56:21 INFO mapreduce.Job:  map 13% reduce 0%
14/10/23 09:56:22 INFO mapreduce.Job:  map 28% reduce 0%
14/10/23 09:56:24 INFO mapreduce.Job:  map 35% reduce 0%
14/10/23 09:56:25 INFO mapreduce.Job:  map 47% reduce 0%
14/10/23 09:56:27 INFO mapreduce.Job:  map 48% reduce 0%
14/10/23 09:56:28 INFO mapreduce.Job:  map 51% reduce 0%
14/10/23 09:56:30 INFO mapreduce.Job:  map 54% reduce 0%
14/10/23 09:56:31 INFO mapreduce.Job:  map 66% reduce 0%
14/10/23 09:56:34 INFO mapreduce.Job:  map 67% reduce 0%
14/10/23 09:56:36 INFO mapreduce.Job:  map 69% reduce 0%
14/10/23 09:56:37 INFO mapreduce.Job:  map 77% reduce 0%
14/10/23 09:56:39 INFO mapreduce.Job:  map 79% reduce 0%
14/10/23 09:56:40 INFO mapreduce.Job:  map 93% reduce 0%
14/10/23 09:56:41 INFO mapreduce.Job:  map 97% reduce 0%
14/10/23 09:56:42 INFO mapreduce.Job:  map 100% reduce 0%
```

```
14/10/23 09:56:58 INFO mapreduce.Job:  map 100% reduce 1%
14/10/23 09:56:59 INFO mapreduce.Job:  map 100% reduce 9%
14/10/23 09:57:00 INFO mapreduce.Job:  map 100% reduce 20%
14/10/23 09:57:01 INFO mapreduce.Job:  map 100% reduce 25%
14/10/23 09:57:02 INFO mapreduce.Job:  map 100% reduce 29%
14/10/23 09:57:03 INFO mapreduce.Job:  map 100% reduce 33%
14/10/23 09:57:04 INFO mapreduce.Job:  map 100% reduce 35%
14/10/23 09:57:05 INFO mapreduce.Job:  map 100% reduce 46%
14/10/23 09:57:06 INFO mapreduce.Job:  map 100% reduce 60%
14/10/23 09:57:07 INFO mapreduce.Job:  map 100% reduce 67%
14/10/23 09:57:08 INFO mapreduce.Job:  map 100% reduce 69%
14/10/23 09:57:09 INFO mapreduce.Job:  map 100% reduce 74%
14/10/23 09:57:10 INFO mapreduce.Job:  map 100% reduce 79%
14/10/23 09:57:11 INFO mapreduce.Job:  map 100% reduce 87%
14/10/23 09:57:12 INFO mapreduce.Job:  map 100% reduce 90%
14/10/23 09:57:13 INFO mapreduce.Job:  map 100% reduce 92%
14/10/23 09:57:14 INFO mapreduce.Job:  map 100% reduce 96%
14/10/23 09:57:18 INFO mapreduce.Job:  map 100% reduce 99%
14/10/23 09:57:23 INFO mapreduce.Job:  map 100% reduce 100%
14/10/23 09:57:23 INFO mapreduce.Job: Job job_1414001930025_0012 completed
successfully
14/10/23 09:57:23 INFO mapreduce.Job: Counters: 49
    File System Counters
          FILE: Number of bytes read=9589594344
          FILE: Number of bytes written=14315542384
          FILE: Number of read operations=0
          FILE: Number of large read operations=0
          FILE: Number of write operations=0
          HDFS: Number of bytes read=10737421180
          HDFS: Number of bytes written=10737418200
          HDFS: Number of read operations=132
          HDFS: Number of large read operations=0
          HDFS: Number of write operations=48
    Job Counters
          Launched map tasks=20
          Launched reduce tasks=24
          Data-local map tasks=20
          Total time spent by all maps in occupied slots (ms)=587757
          Total time spent by all reduces in occupied slots (ms)=512832
          Total time spent by all map tasks (ms)=587757
          Total time spent by all reduce tasks (ms)=512832
          Total vcore-seconds taken by all map tasks=587757
          Total vcore-seconds taken by all reduce tasks=512832
          Total megabyte-seconds taken by all map tasks=601863168
          Total megabyte-seconds taken by all reduce tasks=525139968
    Map-Reduce Framework
          Map input records=107374182
          Map output records=107374182
          Map output bytes=10952166564
          Map output materialized bytes=4770198845
          Input split bytes=2980
          Combine input records=0
          Combine output records=0
          Reduce input groups=107374182
          Reduce shuffle bytes=4770198845
```

```
            Reduce input records=107374182
            Reduce output records=107374182
            Spilled Records=322122546
            Shuffled Maps =480
            Failed Shuffles=0
            Merged Map outputs=480
            GC time elapsed (ms)=16216
            CPU time spent (ms)=1198480
            Physical memory (bytes) snapshot=33026736128
            Virtual memory (bytes) snapshot=75533418496
            Total committed heap usage (bytes)=33015463936
    Shuffle Errors
            BAD_ID=0
            CONNECTION=0
            IO_ERROR=0
            WRONG_LENGTH=0
            WRONG_MAP=0
            WRONG_REDUCE=0
    File Input Format Counters
            Bytes Read=10737418200
    File Output Format Counters
            Bytes Written=10737418200
14/10/23 09:57:23 INFO terasort.TeraSort: done
```

5 Conclusion

The combination of Red Hat enterprise Linux and Cloudera Enterprise creates a stable and tested infrastructure that enables enterprises to build and grow their Big Data solutions and provides application developers and data scientists with a viable, flexible and tightly integrated platform.

Red Hat and Cloudera worked together on this document in the hopes of providing a valuable and informative resource to their mutual customers. If the advice provided in this document is followed, many problems that Hadoop users experience such as incorrectly configured Operating System or making suboptimal cluster configuration choices during configuration can be avoided.