Cisco UCS Integrated Infrastructure for Big Data and Analytics with Cloudera for Data Science at Scale

Building a 28-Node Cluster

Last Updated: June 7, 2018

About the Cisco Validated Design Program
The Cisco Validated Design (CVD) program consists of systems and solutions designed, tested, and documented to facilitate faster, more reliable, and more predictable customer deployments. For more information, visit:


ALL DESIGNS, SPECIFICATIONS, STATEMENTS, INFORMATION, AND RECOMMENDATIONS (COLLECTIVELY, "DESIGNS") IN THIS MANUAL ARE PRESENTED "AS IS," WITH ALL FAULTS. CISCO AND ITS SUPPLIERS DISCLAIM ALL WARRANTIES, INCLUDING, WITHOUT LIMITATION, THE WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT OR ARISING FROM A COURSE OF DEALING, USAGE, OR TRADE PRACTICE. IN NO EVENT SHALL CISCO OR ITS SUPPLIERS BE LIABLE FOR ANY INDIRECT, SPECIAL, CONSEQUENTIAL, OR INCIDENTAL DAMAGES, INCLUDING, WITHOUT LIMITATION, LOST PROFITS OR LOSS OR DAMAGE TO DATA ARISING OUT OF THE USE OR INABILITY TO USE THE DESIGNS, EVEN IF CISCO OR ITS SUPPLIERS HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

THE DESIGNS ARE SUBJECT TO CHANGE WITHOUT NOTICE. USERS ARE SOLELY RESPONSIBLE FOR THEIR APPLICATION OF THE DESIGNS. THE DESIGNS DO NOT CONSTITUTE THE TECHNICAL OR OTHER PROFESSIONAL ADVICE OF CISCO, ITS SUPPLIERS OR PARTNERS. USERS SHOULD CONSULT THEIR OWN TECHNICAL ADVISORS BEFORE IMPLEMENTING THE DESIGNS. RESULTS MAY VARY DEPENDING ON FACTORS NOT TESTED BY CISCO.

CCDE, CCENT, Cisco Eos, Cisco Lumin, Cisco Nexus, Cisco StadiumVision, Cisco TelePresence, Cisco WebEx, the Cisco logo, DCE, and Welcome to the Human Network are trademarks; Changing the Way We Work, Live, Play, and Learn and Cisco Store are service marks; and Access Registrar, Aironet, AsyncOS, Bringing the Meeting To You, Catalyst, CCDA, CCDP, CCIE, CCIP, CCNA, CCNP, CCSP, CCVP, Cisco, the Cisco Certified Internetwork Expert logo, Cisco IOS, Cisco Press, Cisco Systems, Cisco Systems Capital, the Cisco Systems logo, Cisco Unified Computing System (Cisco UCS), Cisco UCS B-Series Blade Servers, Cisco UCS C-Series Rack Servers, Cisco UCS S-Series Storage Servers, Cisco UCS Manager, Cisco UCS Management Software, Cisco Unified Fabric, Cisco Application Centric Infrastructure, Cisco Nexus 9000 Series, Cisco Nexus 7000 Series, Cisco Prime Data Center Network Manager, Cisco NX-OS Software, Cisco MDS Series, Cisco Unity, Collaboration Without Limitation, EtherFast, EtherSwitch, Event Center, Fast Step, Follow Me Browsing, FormShare, GigaDrive, HomeLink, Internet Quotient, IOS, iPhone, iQuick Study, LightStream, Linksys, MediaTone, MeetingPlace, MeetingPlace Chime Sound, MGX, Networkers, Networking Academy, Network Registrar, PCNow, PIX, PowerPanels, ProConnect, ScriptShare, SenderBase, SMARTnet, Spectrum Expert, StackWise, The Fastest Way to Increase Your Internet Quotient, TransPath, WebEx, and the WebEx logo are registered trademarks of Cisco Systems, Inc. and/or its affiliates in the United States and certain other countries.

All other trademarks mentioned in this document or website are the property of their respective owners. The use of the word partner does not imply a partnership relationship between Cisco and any other company. (0809R)

© 2018 Cisco Systems, Inc. All rights reserved.
# Table of Contents

Executive Summary .......................................................................................................................... 7

Solution Overview .......................................................................................................................... 8
  Introduction .................................................................................................................................... 8
  Audience ........................................................................................................................................ 8
  Purpose of this Document .............................................................................................................. 8
  Solution Summary ......................................................................................................................... 9
    Scaling the Solution ..................................................................................................................... 12
  Cloudera Data Science Workbench ............................................................................................... 13
  NVIDIA GPU ................................................................................................................................... 15

Technology Overview ..................................................................................................................... 16
  Cisco UCS Integrated Infrastructure for Big Data and Analytics .................................................... 16
    Cisco UCS 6300 Series Fabric Interconnects ............................................................................... 16
    Cisco UCS C-Series Rack-Mount Servers ..................................................................................... 16
    Cisco UCS Virtual Interface Cards (VICs) .................................................................................... 18
    Cisco UCS Manager .................................................................................................................... 19
  NVIDIA CUDA ............................................................................................................................... 19
  Cloudera (CDH 5.13.0) ............................................................................................................... 19
    Cloudera Data Science Workbench ............................................................................................ 21
    Docker Containers ...................................................................................................................... 21
    Kubernetes ................................................................................................................................. 22

Solution Design .............................................................................................................................. 23
  Requirements ................................................................................................................................. 23
  Rack and PDU Configuration ......................................................................................................... 23
    Port Configuration on Fabric Interconnects ................................................................................ 25
    Server Configuration and Cabling for Cisco UCS C240 M5 .................................................... 25
  Software Distributions and Versions ............................................................................................ 25
    Cloudera (CDH 5.13.0) .............................................................................................................. 25
    Red Hat Enterprise Linux (RHEL) ............................................................................................... 25
    Software Versions ......................................................................................................................... 25

Fabric Configuration ...................................................................................................................... 26

Performing Initial Setup of Cisco UCS 6332 Fabric Interconnects .............................................. 27
  Configure Fabric Interconnect A .................................................................................................... 27
  Configure Fabric Interconnect B ................................................................................................. 28
  Logging Into Cisco UCS Manager ............................................................................................... 28
  Upgrading Cisco UCS Manager Software to Version 3.2(2b) .................................................... 28
  Adding a Block of IP Addresses for KVM Access ....................................................................... 28
  Enabling Uplink Ports .................................................................................................................. 30
Executive Summary

The technology that implements big data systems has matured to the point where it is in wide use across all industries addressing a wide variety of complex business problems; yet even as the technology has matured, the rate of data growth has increased. In addition, machine learning is gaining in prominence. Machine learning is a set of techniques for sophisticated pattern matching that came out of research into Artificial Intelligence. Machine learning needs all the data from the big data systems plus very high-performance computing.

Businesses are now faced with a new set of challenges, namely, making this data available to the diverse set of people who need it, publishing their results so the organization can make use of it, enabling the automated production of those results, while also managing the data for compliance and governance, and doing all of this in an efficient way that scales as the data continues to grow.

New, better solutions to old problems, and new applications, with new revenue streams, are now within grasp, but require approaches to hardware and software where agility is a primary design driver. IT organizations with traditional infrastructure and software solutions struggle to respond to changing business conditions and to manage this infrastructure. These environments require administrators to spend excessive time configuring new server, storage, and network resources in order to keep up with the scale demanded by growing computing and storage needs.

In addition, the introduction of machine learning into the mix adds new requirements. Many machine learning tasks, especially deep learning tasks, require the use of GPUs, a specialized, very high-performance processor that is massively parallel in nature. GPUs are installed on the servers and it is critically important that these high-performance processors also scale with the data growth.

Cisco UCS Integrated Infrastructure for Big Data and Analytics is an optimal choice where world class performance and reliability are base requirements. It is the strong foundation upon which solutions are built. The architecture has the designed-in ability to scale from a small starting solution to thousands of servers and hundreds of petabytes of storage, all managed from a single pane.

Cloudera Enterprise combines distributed data processing with machine learning and analytics into a single scalable platform that enables businesses to tackle their most complex problems. Cloudera is a leading provider of Apache Hadoop distributions and its integrated ecosystem of projects. These tools are used by companies around the world to tackle problems as diverse as predictive maintenance on fleets of hundreds of thousands of vehicles to real-time processing of petabyte-scale data for market surveillance and compliance on stock exchanges.

Together, Cisco and Cloudera combine to create a dependable deployment system to address today’s most challenging problems.
Solution Overview

Introduction

Both big data and machine learning technology have progressed to the point where they are being implemented in production systems running 24x7. There exists a very clear need for a proven, dependable, high-performance platform for the ingestion, processing, storage and analysis of the data, as well as the seamless dissemination of the output, results and insights of the analysis.

This solution implements the Cisco UCS Integrated Infrastructure for Big Data and Analytics, a world-class platform specifically designed for demanding workloads that is both easy to scale and easy to manage, even as the requirements grow to thousands of servers and petabytes of storage; and the Cloudera Data Science Workbench, an integrated set of tools designed to enable flexible, fast access to the entire data store.

Many companies, recognizing the immense potential of big data and machine learning technology, are gearing up to leverage these new capabilities, building out departments and increasing hiring. However, these efforts face a new set of challenges:

- making the data available to the diverse set of people who need it,
- enabling access to high-performance computing resources, GPUs, that also scale with the data growth
- allowing people to work with the data using the environments they are familiar with,
- publishing their results so the organization can make use of it,
- enabling the automated production of those results,
- managing the data for compliance and governance,
- scaling the system as the data grows
- managing and administering the system in an efficient, cost-effective way

This solution is based on the Cisco UCS Integrated Infrastructure for Big Data and Analytics and includes computing, storage, connectivity, and unified management capabilities to help companies manage the immense amount of data being collected. It is built on the Cisco Unified Computing System (Cisco UCS) infrastructure, using Cisco UCS 6332 Series Fabric Interconnects, and Cisco UCS C-Series Rack Servers. This architecture is specifically designed for performance and linear scalability for big data and machine learning workloads.

Audience

The intended audience of this document includes sales engineers, field consultants, professional services, IT managers, partner engineering and customers who want to deploy the Cloudera Distribution with Apache Hadoop (CDH 5.13.0) and Cloudera Data Science Workbench (CDSW 1.3.0) on the Cisco UCS Integrated Infrastructure for Big Data and Analytics (Cisco UCS M5 Rack mount servers).

Purpose of this Document

This document describes the architecture and deployment procedures for Cloudera 5.13.0 with Cloudera Data Science Workbench 1.3.0 on a 28-node Cisco UCS C240 M5 cluster based on Cisco UCS Integrated Infrastructure for Big Data and Analytics.
Solution Summary

This CVD describes in detail the process of installing Cloudera 5.13.0 with Cloudera Data Science Workbench (CDSW 1.3.0) and the configuration details of the cluster. The current version of Cisco UCS Integrated Infrastructure for Big Data and Analytics offers the following configurations depending on the compute and storage requirements as shown in Table 1.

**Table 1  Cisco UCS Integrated Infrastructure for Big Data and Analytics Configuration Options**

<table>
<thead>
<tr>
<th></th>
<th>Performance (UCS-SP-C240M5-A2)</th>
<th>Capacity (UCS-SPC240M5L-S1)</th>
<th>High Capacity (UCS-SP-53260-BV)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Servers</td>
<td>16 x Cisco UCS C240 M5 Rack Servers with SFF drives</td>
<td>16 x Cisco UCS C240 M5 Rack Servers with LFF drives</td>
<td>8 x Cisco UCS S3260 Storage Servers</td>
</tr>
<tr>
<td>CPU</td>
<td>2 x Intel Xeon Processor Scalable Family 6132 (2 x 14 cores, 2.6 GHz)</td>
<td>2 x Intel Xeon Processor Scalable Family 4110 (2 x 8 cores, 2.1 GHz)</td>
<td>2 x Intel Xeon Processor Scalable Family 632 (2 x 14 cores, 2.6 GHz)</td>
</tr>
<tr>
<td>Memory</td>
<td>6 x 32 GB 2666 MHz (192 GB)</td>
<td>6 x 32 GB 2666 MHz (192 GB)</td>
<td>6 x 32 GB 2666 MHz (192 GB)</td>
</tr>
<tr>
<td>Boot</td>
<td>M.2 with 2 x 240-GB SSDs</td>
<td>M.2 with 2 x 240-GB SSDs</td>
<td>M.2 with 2 x 240-GB SSDs</td>
</tr>
<tr>
<td>Storage</td>
<td>26 x 1.8 TB 10k rpm SFF SAS HDDs or 12 x 1.6 TB Enterprise Value SATA SSDs</td>
<td>12 x 8 TB 7.2K rpm LFF SAS HDDs + 2 SFF rear hot-swappable 1.6 TB Enterprise Value SATA SSDs</td>
<td>24 x 6 TB 7.2K rpm LFF SAS SSDs</td>
</tr>
<tr>
<td>VIC</td>
<td>40 Gigabit Ethernet (Cisco UCS VIC 1387)</td>
<td>40 Gigabit Ethernet (Cisco UCS VIC 1387)</td>
<td>40 Gigabit Ethernet (Cisco UCS VIC 1387)</td>
</tr>
<tr>
<td>Storage Controller</td>
<td>Cisco 12-Gbps SAS Modular RAID Controller with 4-GB flash-based write cache (FBWC) or Cisco 12-Gbps Modular SAS Host Bus Adapter (HBA)</td>
<td>Cisco 12-Gbps SAS Modular RAID Controller with 2-GB flash-based write cache (FBWC) or Cisco 12-Gbps Modular SAS Host Bus Adapter (HBA)</td>
<td>Cisco 12-Gbps SAS Modular RAID Controller with 4-GB flash-based write cache (FBWC)</td>
</tr>
<tr>
<td>Network Connectivity</td>
<td>Cisco UCS 6332 Fabric Interconnect</td>
<td>Cisco UCS 6332 Fabric Interconnect</td>
<td>Cisco UCS 6332 Fabric Interconnect</td>
</tr>
</tbody>
</table>

Table 2 lists the configuration details for Cloudera Data Science Workbench. These servers provide the high-performance GPU compute capacity.

**Table 2  Cisco UCS Integrated Infrastructure for Big Data and Analytics for CDSW**

<table>
<thead>
<tr>
<th></th>
<th>Starter</th>
<th>High Performance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Servers</td>
<td>4 x Cisco UCS C240 M5 Rack Servers</td>
<td>4 x Cisco UCS C480 M5 Rack Servers</td>
</tr>
<tr>
<td>CPU</td>
<td>2 x Intel Xeon Processor Scalable Family 6132 (2 x 14 cores, 2.6 GHz)</td>
<td>2 x Intel Xeon Processor Scalable Family 632 (2 x 14 cores, 2.6 GHz)</td>
</tr>
<tr>
<td>Memory</td>
<td>12 x 32 GB DDR4 (384 GB)</td>
<td>24 x 32 GB DDR4 (768 GB)</td>
</tr>
<tr>
<td>Boot</td>
<td>M.2 with 2 x 240-GB SSDs</td>
<td>M.2 with 2 x 240-GB SSDs</td>
</tr>
<tr>
<td>Storage</td>
<td>4 x 1.6 TB Enterprise Value SATA SSDs</td>
<td>8 x 1.6 TB Enterprise Value SATA SSDs</td>
</tr>
<tr>
<td>VIC</td>
<td>40 Gigabit Ethernet (Cisco UCS VIC 1387)</td>
<td>40 Gigabit Ethernet (Cisco UCS VIC 1387)</td>
</tr>
<tr>
<td></td>
<td>Starter</td>
<td>High Performance</td>
</tr>
<tr>
<td>------------------------</td>
<td>-------------------------------------------------------------------------</td>
<td>--------------------------------------------------------</td>
</tr>
<tr>
<td><strong>Storage Controller</strong></td>
<td>Cisco 12-Gbps SAS Modular RAID Controller with 4-GB flash-based write cache (FBWC) or Cisco 12-Gbps Modular SAS Host Bus Adapter (HBA)</td>
<td>Cisco 12-Gbps SAS Modular RAID Controller with 4-GB flash-based write cache (FBWC) or Cisco 12-Gbps Modular SAS Host Bus Adapter (HBA)</td>
</tr>
<tr>
<td><strong>Network Connectivity</strong></td>
<td>Cisco UCS 6332 Fabric Interconnect</td>
<td>Cisco UCS 6332 Fabric Interconnect</td>
</tr>
<tr>
<td><strong>GPU</strong></td>
<td>2 x NVIDIA TESLA V100</td>
<td>6 x NVIDIA TESLA V100</td>
</tr>
</tbody>
</table>

Figure 1 depicts a 28-node starter cluster. Rack #1 has 16 Cisco UCS C240 M5 servers. Each link in the figure represents a 40 Gigabit Ethernet link from each of the 16 servers directly connected to a Fabric Interconnect. Rack #2 has 12 Cisco UCS C240 M5 servers. Every server is connected to both Fabric Interconnects.
Figure 1 28 Node Starter Cluster Configuration for CDSW

2 x Cisco UCS 6332 Fabric Interconnect

16 x C240 M5

8 x C240 M5

4 x C240 M5 with 2 x GPU each

Note: Power requirements per rack must be calculated as the exact values will change based on the power needs of the GPUs.

Figure 2 shows an alternate configuration for cases where more GPU capacity is needed. Four of the Cisco UCS C240 M5 servers from the previous figure are replaced with Cisco UCS C480 M5 servers. These servers support up to six GPUs each.
Solution Overview

Figure 2 28 Node High Performance Cluster Configuration with additional GPU capacity

2 x Cisco UCS 6332 Fabric Interconnect

16 x C240 M5

8 x C240 M5

4 x Cisco UCS C480 M5 with 6 x GPU each

Note: Power requirements per rack must be calculated as the exact values will change based on the power needs of the GPUs.

Scaling the Solution

Figure 3 shows how to scale the solution. Each pair of Cisco UCS 6332 Fabric Interconnects has 28 Cisco UCS C240 M5 servers connected to it. This allows for four uplinks from each Fabric Interconnect to the Cisco Nexus 9332 switch. Six pairs of 6332 FI’s can connect to a single switch with four uplink ports each. With 28 servers per FI, a total of 168 servers can be supported. Additionally, the can scale to thousands of nodes with the Nexus 9500 series family of switches.
Cloudera Data Science Workbench (CDSW) is a web application that allows data scientists to use a variety of open source languages and libraries to directly and securely access the data in the Hadoop cluster. Direct access to the big data cluster means no more working with small subsets of the data on desktop systems; no sampling is required as the entire data set is available for use directly by the user. Further, users are not restricted to a single environment. Many popular open source libraries and languages are supported, including R, Python and Scala, which means users become productive faster with no need for retraining and no time lost learning a new programming language.
CDSW is addressing the key challenge that every team or user may require a different language, library or framework in order to be productive while the organization requires reproducibility and collaboration. By making the entire set of data in the cluster available to the user, CSDW eliminates the problem that what works on small samples or extracts of the data on a user’s desktop computer may not scale across a large cluster. Cloudera Data Science Workbench gives data scientists the flexibility and simplicity they need to be productive and innovative at scale.

In addition, CDSW enables seamless access to high-performance processors in the form of GPUs. CSDW makes use of lightweight container architecture to rapidly and securely provide the environment and resources to the users.

Cloudera Data Science Workbench is directly aimed at helping data scientists build and test new analyses and analytics projects as quickly as possible in secure manner even in large scale environments. This flexibility improves the efficiency of the exploration process, a key requirement to meet in order to move rapidly from idea to answer. Most analytics problems, especially those with transformative power, are not standard analyses and require advanced models and iterative methods. Experimentation and innovation are the heart and soul of data science, but security is needed for compliance and governance.

Data has become one of the most strategic assets in the organization. Leveraging the data to drive the business forward is the primary motivation for building an enterprise data hub to support advanced analytics. Typically, when forced to make a choice between the security of the data and the flexibility to access it, security wins locking away the data from the people who most need it. CDSW address this issue by providing full authentication and access controls against data in the cluster, including complete Kerberos integration. It offers data science teams per-project isolation and reproducibility with no effort.

Cloudera Data Science Workbench allows you to automate analytics workloads with a built-in job and pipeline scheduling system that supports real-time monitoring, job history, and email alerts. Jobs are created and can be configured to run on a recurring schedule, as well as providing alerts for successful and failed runs. Multiple jobs can be scheduled together to create an automated pipeline; e.g., the first job performs data acquisition, the next data cleansing, then analytics, and so on.

Collaboration and sharing of results are implemented via project sharing (either globally or to specific users, and project forking. To share results, CSDW enables publishing output for viewing via a browser, and even makes the console log itself
available for viewing both during and after the run. Cloudera Data Science Workbench is a web application. It has no
desktop footprint making it very easy to administer and maintain.

**NVIDIA GPU**

Graphics Processing Units, or GPUs, are specialized processors designed to render images, animation and video for
computer displays. They perform this task by running many operations simultaneously. While the number and kinds of
operations they can do are limited, they make up for it by being able run many thousands in parallel. As the graphics
capabilities of GPUs increased, it soon became apparent that the massive parallelism of GPUs could be put to other uses
beside rendering graphics.

NVIDIA® GPU used in this document, NVIDIA Tesla® V100 is advanced data center GPU built to accelerate AI, HPC, and
graphics. It’s powered by NVIDIA Volta architecture, comes in 16 and 32 GB configurations.

NVIDIA GPUs bring two key advantages to the table. First, they make possible solutions that were simply not
computationally possible before. Second, by providing the same processing power as scores of traditional CPUs they
reduce the requirements for rack space, power, networking and cooling in the data center.
Technology Overview

Cisco UCS Integrated Infrastructure for Big Data and Analytics

The Cisco UCS Integrated Infrastructure for Big Data and Analytics solution for Cloudera is based on Cisco UCS Integrated Infrastructure for Big Data and Analytics, a highly scalable architecture designed to meet a variety of scale-out application demands with seamless data integration and management integration capabilities built using the components described in this section.

Cisco UCS 6300 Series Fabric Interconnects

Cisco UCS 6300 Series Fabric Interconnects provide high-bandwidth, low-latency connectivity for servers, with integrated, unified management provided for all connected devices by Cisco UCS Manager. Deployed in redundant pairs, Cisco fabric interconnects offer the full active-active redundancy, performance, and exceptional scalability needed to support the large number of nodes that are typical in clusters serving big data applications. Cisco UCS Manager enables rapid and consistent server configuration using service profiles, automating ongoing system maintenance activities such as firmware updates across the entire cluster as a single operation. Cisco UCS Manager also offers advanced monitoring with options to raise alarms and send notifications about the health of the entire cluster.

The Cisco UCS 6300 series Fabric interconnects are a core part of Cisco UCS, providing low-latency, lossless 10 and 40 Gigabit Ethernet, Fiber Channel over Ethernet (FCoE), and Fiber Channel functions with management capabilities for the entire system. All servers attached to Fabric interconnects become part of a single, highly available management domain.

Figure 4 Cisco UCS 6332 UP 32 - Port Fabric Interconnect

Cisco UCS C-Series Rack-Mount Servers

The Cisco UCS C240 M5 Rack-Mount Server (Figure 5) is a 2-socket, 2-Rack-Unit (2RU) rack server offering industry-leading performance and expandability. It supports a wide range of storage and I/O-intensive infrastructure workloads, from big data and analytics to collaboration. Cisco UCS C-Series Rack Servers can be deployed as standalone servers or as part of a Cisco Unified Computing System (Cisco UCS) managed environment to take advantage of Cisco's standards-based unified computing innovations that help reduce customers' Total Cost of Ownership (TCO) and increase their business agility.

In response to ever-increasing computing and data-intensive real-time workloads, the enterprise-class Cisco UCS C240 M5 server extends the capabilities of the Cisco UCS portfolio in a 2RU form factor. It incorporates the Intel® Xeon® Scalable processors, supporting up to 20 percent more cores per socket, twice the memory capacity, and five times more Non-Volatile Memory Express (NVMe) PCI Express (PCIe) Solid-State Disks (SSDs) compared to the previous generation of servers. These improvements deliver significant performance and efficiency gains that will improve your application performance. The Cisco UCS C240 M5 delivers outstanding levels of storage expandability with exceptional performance, along with the following:

- Latest Intel Xeon Scalable CPUs with up to 28 cores per socket
- Up to 24 DDR4 DIMMs for improved performance
- Up to 26 hot-swappable Small-Form-Factor (SFF) 2.5-inch drives, including 2 rear hot-swappable SFF drives (up to 10 support NVMe PCIe SSDs on the NVMe-optimized chassis version), or 12 Large-Form-Factor (LFF) 3.5-inch drives plus 2 rear hot-swappable SFF drives
- Support for 12-Gbps SAS modular RAID controller in a dedicated slot, leaving the remaining PCIe Generation 3.0 slots available for other expansion cards
- Modular LAN-On-Motherboard (mLOM) slot that can be used to install a Cisco UCS Virtual Interface Card (VIC) without consuming a PCIe slot, supporting dual 10- or 40-Gbps network connectivity
- Dual embedded Intel x550 10GBASE-T LAN-On-Motherboard (LOM) ports
- Modular M.2 or Secure Digital (SD) cards that can be used for boot

**Figure 5 Cisco UCS C240 M5 Rack-Mount Server**

The Cisco UCS C480 M5 Rack-Mount Server is a storage- and I/O-optimized enterprise-class rack server that delivers industry-leading performance for in-memory databases, big data analytics, virtualization, Virtual Desktop Infrastructure (VDI), and bare-metal applications. The Cisco UCS C480 M5 (Figure 6) delivers outstanding levels of expandability and performance for standalone or Cisco Unified Computing System™ (Cisco UCS) managed environments in a 4RU form-factor. And because of its modular design, you pay for only what you need. It offers these capabilities:

- Latest Intel® Xeon® Scalable processors with up to 28 cores per socket and support for two- or four-processor configurations
- 2666-MHz DDR4 memory and 48 DIMM slots for up to 6 TeraBytes (TB) of total memory
- 12 PCI Express (PCIe) 3.0 slots
  - Six x8 full-height, full length slots
  - Six x16 full-height, full length slots
- Flexible storage options with support up to 32 Small-Form-Factor (SFF) 2.5-inch, SAS, SATA, and PCIe NVMe disk drives
- Cisco® 12-Gbps SAS Modular RAID Controller in a dedicated slot
- Internal Secure Digital (SD) and M.2 boot options
- Dual embedded 10 Gigabit Ethernet LAN-On-Motherboard (LOM) ports

**Figure 6 Cisco UCS C480 M5 Rack-Mount Server**
Cisco UCS Virtual Interface Cards (VICs)

Cisco UCS Virtual Interface Cards (VIC) are unique to Cisco. Cisco UCS Virtual Interface Cards incorporate next-generation converged network adapter (CNA) technology from Cisco, and offer dual 10- and 40-Gbps ports designed for use with Cisco UCS servers. Optimized for virtualized networking, these cards deliver high performance and bandwidth utilization, and support up to 256 virtual devices.

The Cisco UCS Virtual Interface Card 1387 offers dual-port Enhanced Quad Small Form-Factor Pluggable (QSFP+) 40 Gigabit Ethernet and Fiber Channel over Ethernet (FCoE) in a modular-LAN-on-motherboard (mLOM) form factor. The mLOM slot can be used to install a Cisco VIC without consuming a PCIe slot providing greater I/O expandability.
Cisco UCS Manager

Cisco UCS Manager resides within the Cisco UCS 6300 Series Fabric Interconnect. It makes the system self-aware and self-integrating, managing all of the system components as a single logical entity. Cisco UCS Manager can be accessed through an intuitive graphical user interface (GUI), a command-line interface (CLI), or an XML application-programming interface (API). Cisco UCS Manager uses service profiles to define the personality, configuration, and connectivity of all resources within Cisco UCS, radically simplifying provisioning of resources so that the process takes minutes instead of days. This simplification allows IT departments to shift their focus from constant maintenance to strategic business initiatives.

NVIDIA CUDA

GPUs are very good at running the same operation on different data simultaneously. This is often referred to as single instruction, multiple data, or SIMD. This is exactly what’s needed to render graphics but many other computing problems can benefit from this approach. As a result, NVIDIA created CUDA. CUDA is a parallel computing platform and programming model that makes it possible to use a GPU for many general purpose computing tasks via commonly used programming languages like C and C++.

In addition to the general-purpose computing capabilities that CUDA enables there is also a special CUDA library for deep learning called the CUDA Deep Neural Network library, or cuDNN. cuDNN makes it easier to implement deep machine learning architectures that take full advantage of the GPU’s capabilities.

Cloudera (CDH 5.13.0)

Built on the transformative Apache Hadoop open source software project, Cloudera Enterprise is a hardened distribution of Apache Hadoop and related projects designed for the demanding requirements of enterprise customers. Cloudera is the leading contributor to the Hadoop ecosystem, and has created a rich suite of complementary open source projects that are included in Cloudera Enterprise.

All the integration and the entire solution is thoroughly tested and fully documented. By taking the guesswork out of building out a Hadoop deployment, CDH gives a streamlined path to success in solving real business problems.

Cloudera Enterprise with Apache Hadoop is:
- **Unified** – one integrated system, bringing diverse users and application workloads to one pool of data on common infrastructure; no data movement required
- **Secure** – perimeter security, authentication, granular authorization, and data protection
- **Governed** – enterprise-grade data auditing, data lineage, and data discovery
- **Managed** – native high-availability, fault-tolerance and self-healing storage, automated backup and disaster recovery, and advanced system and data management
- **Open** – Apache-licensed open source to ensure both data and applications remain copy righted, and an open platform to connect with all of the existing investments in technology and skills.

**Figure 8 Cloudera Data Hub**

Cloudera provides a scalable, flexible, integrated platform that makes it easy to manage rapidly increasing volumes and varieties of data in any enterprise. Industry-leading Cloudera products and solutions enable to deploy and manage Apache Hadoop and related projects, manipulate and analyze data, and keep that data secure and protected.

Cloudera provides the following products and tools:
- **CDH**—The Cloudera distribution of Apache Hadoop and other related open-source projects, including Spark. CDH also provides security and integration with numerous hardware and software solutions.
- **Apache Spark**—An integrated part of CDH and supported with Cloudera Enterprise, Spark is an open standard for flexible in-memory data processing for batch, real time and advanced analytics. Via the one platform Cloudera is committed to adopting Spark as the default data execution engine for analytic workloads.
- **Cloudera Manager**—A sophisticated application used to deploy, manage, monitor, and diagnose issues with CDH deployments. Cloudera Manager provides the Admin Console, a web-based user interface that makes administration of any enterprise data simple and straightforward. It also includes the Cloudera Manager API, which can be used to obtain cluster health information and metrics, as well as configure Cloudera Manager.

- **Cloudera Navigator**—An end-to-end data management tool for the CDH platform. Cloudera Navigator enables administrators, data managers, and analysts to explore the large amounts of data in Hadoop. The robust auditing, data management, lineage management, and life cycle management in Cloudera Navigator allow enterprises to adhere to stringent compliance and regulatory requirements.

**Cloudera Data Science Workbench**

Cloudera Data Science Workbench (CDSW) is a web application that allows data scientists to use a variety of open source languages and libraries to directly and securely access the data in the Hadoop cluster. Direct access to the big data cluster means no more working with small subsets of the data on desktop systems; no sampling is required as the entire data set is available for use directly by the user. Further, users are not restricted to a single environment. Many popular open source libraries and languages are supported, including R, Python and Scala, as well as all of the ML/DL frameworks such as TensorFlow, Theano, Pytorch, etc. In addition, CDSW enables access to available GPU resources for deep learning workloads.

**Docker Containers**

Cloudera Data Science Workbench makes use of container technology. Containers are conceptually similar to virtual machines, but instead of virtualizing the hardware, a container virtualizes the operating system. With a VM there is an entire operating system sitting on top of the hypervisor. Containers dispense with this time-consuming and resource hungry requirement by sharing the host system’s kernel. As a result, a container is far smaller, and its lightweight nature means they can be instantiated quickly. In fact, they can be instantiated so quickly that new application architectures are possible.

Docker is an open-source project based on Linux containers. It uses Linux kernel features like namespaces and control groups to create containers. These features are not new, but Docker has taken these concepts and improved them in the following ways:

- **Ease of use:** Docker makes easier for anyone—developers, systems admins, architects and others—to take advantage of containers in order to quickly build and test portable applications. It allows anyone to package an application on their development system, which can then run **unmodified** on any cloud or bare metal server. The basic idea is to create a “build once, run anywhere” system.

- **Speed:** Docker containers are very fast with a small footprint. Ultimately, containers are just sandboxed environments running on the kernel so they take up few resources. You can create and run a Docker container in seconds. Compare this to a VM which takes much longer because it has to boot up a full virtual operating system every time.

- **Modularity:** Docker makes it easy to take an application and breaks its functionality into separate individual containers. These containers can then be spun up and run as needed. This is particularly useful for cases where an application needs to hold and lock a particular resource, like a GPU, and then release it once it’s done using it. Modularity also enables each component, i.e., container to be updated independently.

- **Scalability:** modularity enables scalability. With different parts of the system running in different containers it becomes possible, and with Docker, it becomes easy to connect these containers together to create an application, which can then be scaled out as needed.
Kubernetes

Applications built using container technology provide a great deal of flexibility in terms of their architecture, deployment and scaling. Since containers provide VM-like separation of concerns but with far less overhead they allow system developers to package different services of the same application into separate containers. These containers can then be deployed in a very flexible manner including across clusters of physical and virtual machines. This builds the ability to scale directly into the application architecture. This ability in turn requires a tool to aid in deploying, managing and scaling container-based applications.

Kubernetes is an open source project specifically designed for deploying and managing multi-container applications at scale. Kubernetes automates and simplifies the following tasks:

- Deploying multi-container applications. With the application split into separate containers for different services, Kubernetes manages the deployment of the containers both at initial startup and in real-time as needed by the application.

- Scaling containers. Applications need to spin up and down containers to suit demand, to balance incoming load, and make better use of physical resources. Kubernetes provides the mechanisms for doing these things in a completely automated way.

- Updating applications. One advantage of container-based application development is the ability to independently change, improve and fix individual containers. Kubernetes has mechanisms for allowing graceful updates to new versions of container images, including rollbacks if something does not go as planned.

Kubernetes manages application status and any replication and load balancing needs. It also handles hardware resource allocation including GPUs. Kubernetes also has facilities for maximizing the use of hardware resources including memory, storage I/O, and network bandwidth. Applications can have soft and hard limits set on their resource usage. For example, many small applications that use minimal resources can be run together on the same hardware while resource hungry applications can be placed on different hardware and scale out as needed.
Solution Design

Requirements

This CVD describes architecture and deployment procedures for Cloudera (CDH 5.13.0) and Cloudera Data Science Workbench (CDSW 1.3.0) on a 28-node cluster based on Cisco UCS Integrated Infrastructure for Big Data and Analytics. The solution goes into detail configuring CDH 5.13.0 on the infrastructure, as well as the complete installation and configuration of CDSW 1.3.0 and all of its dependencies.

The cluster configuration consists of the following:

- Two Cisco UCS 6332UP Fabric Interconnects
- 28 UCS C240 M5 Rack-Mount servers
- 8 NVIDIA GPU
- Two Cisco R42610 standard racks
- Four Vertical Power distribution units (PDUs) (Country Specific)

Rack and PDU Configuration

Each rack consists of two vertical PDUs. The first rack consists of two Cisco UCS 6332UP Fabric Interconnects, 16 Cisco UCS C240 M5 Rack Servers connected to each of the vertical PDUs for redundancy; thereby, ensuring availability during power source failure. The second rack consists of 12 Cisco UCS C240 M5 Servers connected to each of the vertical PDUs for redundancy; thereby, ensuring availability during power source failure, similar to the first rack.

Note: Please contact your Cisco representative for country specific information.

<table>
<thead>
<tr>
<th>Table 3</th>
<th>Rack Configurations</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Cisco</strong></td>
<td><strong>First Rack</strong></td>
</tr>
<tr>
<td>42URack</td>
<td>42URack</td>
</tr>
<tr>
<td>42</td>
<td>Cisco UCS FI 6332UP</td>
</tr>
<tr>
<td>41</td>
<td>Cisco UCS FI 6332UP</td>
</tr>
<tr>
<td>40</td>
<td>Unused</td>
</tr>
<tr>
<td>39</td>
<td>Unused</td>
</tr>
<tr>
<td>38</td>
<td>Unused</td>
</tr>
<tr>
<td>37</td>
<td>Unused</td>
</tr>
<tr>
<td>36</td>
<td>Unused</td>
</tr>
<tr>
<td>35</td>
<td></td>
</tr>
<tr>
<td>Cisco</td>
<td>First Rack</td>
</tr>
<tr>
<td>-------</td>
<td>-----------------</td>
</tr>
<tr>
<td>34</td>
<td>Unused</td>
</tr>
<tr>
<td>33</td>
<td></td>
</tr>
<tr>
<td>32</td>
<td>Cisco UCS C240 M5</td>
</tr>
<tr>
<td>31</td>
<td></td>
</tr>
<tr>
<td>30</td>
<td>Cisco UCS C240 M5</td>
</tr>
<tr>
<td>29</td>
<td></td>
</tr>
<tr>
<td>28</td>
<td>Cisco UCS C240 M5</td>
</tr>
<tr>
<td>27</td>
<td></td>
</tr>
<tr>
<td>26</td>
<td>Cisco UCS C240 M5</td>
</tr>
<tr>
<td>25</td>
<td></td>
</tr>
<tr>
<td>24</td>
<td>Cisco UCS C240 M5</td>
</tr>
<tr>
<td>23</td>
<td></td>
</tr>
<tr>
<td>22</td>
<td>Cisco UCS C240 M5</td>
</tr>
<tr>
<td>21</td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>Cisco UCS C240 M5</td>
</tr>
<tr>
<td>19</td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>Cisco UCS C240 M5</td>
</tr>
<tr>
<td>17</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>Cisco UCS C240 M5</td>
</tr>
<tr>
<td>15</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>Cisco UCS C240 M5</td>
</tr>
<tr>
<td>13</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>Cisco UCS C240 M5</td>
</tr>
<tr>
<td>11</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>Cisco UCS C240 M5</td>
</tr>
<tr>
<td>9</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>Cisco UCS C240 M5</td>
</tr>
</tbody>
</table>
Solution Design

<table>
<thead>
<tr>
<th>Cisco</th>
<th>First Rack</th>
<th>Cisco</th>
<th>Second Rack</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td></td>
<td>7</td>
<td>with 2x NVIDIA GPUs</td>
</tr>
<tr>
<td>6</td>
<td>Cisco UCS C240 M5</td>
<td>6</td>
<td>Cisco UCS C240 M5</td>
</tr>
<tr>
<td>5</td>
<td></td>
<td>5</td>
<td>with 2x NVIDIA GPUs</td>
</tr>
<tr>
<td>4</td>
<td>Cisco UCS C240 M5</td>
<td>4</td>
<td>Cisco UCS C240 M5</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>3</td>
<td>with 2x NVIDIA GPUs</td>
</tr>
<tr>
<td>2</td>
<td>Cisco UCS C240 M5</td>
<td>2</td>
<td>Cisco UCS C240 M5</td>
</tr>
<tr>
<td>1</td>
<td></td>
<td>1</td>
<td>with 2x NVIDIA GPUs</td>
</tr>
</tbody>
</table>

Port Configuration on Fabric Interconnects

<table>
<thead>
<tr>
<th>Port Type</th>
<th>Port Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>Network</td>
<td>29-32</td>
</tr>
<tr>
<td>Server</td>
<td>1-28</td>
</tr>
</tbody>
</table>

Server Configuration and Cabling for Cisco UCS C240 M5

The Cisco UCS C240 M5 rack server is equipped with 2 x Intel Xeon Processor Scalable Family 6132 (2 x 14 cores, 2.6 GHz), 192 GB of memory, Cisco UCS Virtual Interface Card 1337, Cisco 12-Gbps SAS Modular Raid Controller with 4-GB FBWC, 26 x 1.8 TB 10K rpm SFF SAS HDDs or 12 x 1.6 TB Enterprise Value SATA SSDs, M.2 with 2 x 240-GB SSDs for Boot.

For information on physical connectivity and single-wire management see:

For more information on physical connectivity illustrations and cluster setup, see:

Software Distributions and Versions

The software distributions required versions are listed below.

Cloudera (CDH 5.13.0)

The Cloudera Distribution for Apache Hadoop version used is 5.13.0. For more information visit www.cloudera.com.

Red Hat Enterprise Linux (RHEL)

The operating system supported is Red Hat Enterprise Linux 7.4. For more information visit http://www.redhat.com.

Software Versions

The software versions tested and validated in this document are shown in Table 4.
<table>
<thead>
<tr>
<th>Layer</th>
<th>Component</th>
<th>Version or Release</th>
</tr>
</thead>
<tbody>
<tr>
<td>Compute</td>
<td>Cisco UCS C240-M5</td>
<td>C240M5.3.1.2a.0.09</td>
</tr>
<tr>
<td>Network</td>
<td>Cisco UCS 6332</td>
<td>UCS 3.2(2b) A</td>
</tr>
<tr>
<td></td>
<td>Cisco UCS VIC1387 Firmware</td>
<td>4.2.2(a)</td>
</tr>
<tr>
<td></td>
<td>Cisco UCS VIC1387 Driver</td>
<td>2.3.0.44</td>
</tr>
<tr>
<td>Storage</td>
<td>SAS Expander</td>
<td>65.02.13.00</td>
</tr>
<tr>
<td></td>
<td>Cisco 12G Modular Raid controller</td>
<td>50.1.0-07.26</td>
</tr>
<tr>
<td></td>
<td>LSI MegaRAID SAS Driver</td>
<td>07.703.06.00</td>
</tr>
<tr>
<td>Software</td>
<td>Red Hat Enterprise Linux Server</td>
<td>7.4 (x86_64)</td>
</tr>
<tr>
<td></td>
<td>Cisco UCS Manager</td>
<td>3.2(2b)</td>
</tr>
<tr>
<td></td>
<td>CDH</td>
<td>5.13.0</td>
</tr>
<tr>
<td></td>
<td>CDSW</td>
<td>1.3.0</td>
</tr>
<tr>
<td>GPU</td>
<td>CUDA</td>
<td>8.1</td>
</tr>
<tr>
<td></td>
<td>GPU Driver</td>
<td>390</td>
</tr>
</tbody>
</table>

Note: The latest drivers can be downloaded from the link below:
https://software.cisco.com/download/home/283862063/type/283853158/release/3.1%25283%2529.

Note: The Latest Supported RAID controller Driver is already included with the RHEL 7.4 operating system.

Note: Cisco UCS C240 M5 Rack Servers with Intel Scalable Processor Family CPUs are supported from Cisco UCS firmware 3.2 onwards.

Fabric Configuration
This section provides details for configuring a fully redundant, highly available Cisco UCS 6332 fabric configuration.
- Initial setup of the Fabric Interconnect A and B
• Connect to Cisco UCS Manager using virtual IP address of using the web browser
• Launch Cisco UCS Manager.
• Enable server, uplink and appliance ports.
• Start discovery process.
• Create pools and polices for service profile template
• Create Service Profile template and 28 Service profiles
• Associate Service Profiles to servers

Performing Initial Setup of Cisco UCS 6332 Fabric Interconnects

This section describes the initial setup of the Cisco UCS 6332 Fabric Interconnects A and B.

Configure Fabric Interconnect A

1. Connect to the console port on the first Cisco UCS 6332 Fabric Interconnect.
2. At the prompt to enter the configuration method, enter console to continue.
3. If asked to either perform a new setup or restore from backup, enter setup to continue.
4. Enter y to continue to set up a new Fabric Interconnect.
5. Enter y to enforce strong passwords.
6. Enter the password for the admin user.
7. Enter the same password again to confirm the password for the admin user.
8. When asked if this fabric interconnect is part of a cluster, answer y to continue.
10. Enter the cluster name for the system name.
11. Enter the Mgmt0 IPv4 address.
12. Enter the Mgmt0 IPv4 netmask.
13. Enter the IPv4 address of the default gateway.
14. Enter the cluster IPv4 address.
15. To configure DNS, answer y.
16. Enter the DNS IPv4 address.
17. Answer y to set up the default domain name.
18. Enter the default domain name.
19. Review the settings that were printed to the console, and if they are correct, answer yes to save the configuration.

20. Wait for the login prompt to make sure the configuration has been saved.

Configure Fabric Interconnect B

1. Connect to the console port on the second Cisco UCS 6332 Fabric Interconnect.

2. When prompted to enter the configuration method, enter console to continue.

3. The installer detects the presence of the partner Fabric Interconnect and adds this fabric interconnect to the cluster. Enter y to continue the installation.

4. Enter the admin password that was configured for the first Fabric Interconnect.

5. Enter the Mgmt0 IPv4 address.

6. Answer yes to save the configuration.

7. Wait for the login prompt to confirm that the configuration has been saved.

For more information on configuring Cisco UCS 6332 Series Fabric Interconnect, refer to:

Logging Into Cisco UCS Manager

To log into Cisco UCS Manager, complete the following steps:

1. Open a Web browser and navigate to the Cisco UCS 6332 Fabric Interconnect cluster address.

2. Click the Launch link to download the Cisco UCS Manager software.

3. If prompted to accept security certificates, accept as necessary.

4. When prompted, enter admin for the username and enter the administrative password.

5. Click Login to log in to the Cisco UCS Manager.

Upgrading Cisco UCS Manager Software to Version 3.2(2b)

This document assumes the use of UCS 3.2(2b). Refer to Cisco UCS 3.2 Release (upgrade the Cisco UCS Manager software and UCS 6332 Fabric Interconnect software to version 3.2(2b). Also, make sure the UCS C-Series version 3.2(2b) software bundles are installed on the Fabric Interconnects.

Adding a Block of IP Addresses for KVM Access

To create a block of KVM IP addresses for server access in the Cisco UCS environment, complete the following steps:

1. Select the LAN tab at the top of the left window.


3. Right-click IP Pool ext-mgmt.
4. **Select** Create Block of IPv4 Addresses.
5. Enter the starting IP address of the block and number of IPs needed, as well as the subnet and gateway information.

6. Click OK to create the IP block.

7. Click OK in the message box.

Enabling Uplink Ports

To enable uplinks ports, complete the following steps:

1. Select the Equipment tab on the top left of the window.


3. Expand the Unconfigured Ethernet Ports section.

4. Select port 29-32 that is connected to the uplink switch, right-click, then select Reconfigure > Configure as Uplink Port.
5. Select Show Interface and select 40GB for Uplink Connection.

6. A pop-up window appears to confirm your selection. Click Yes then OK to continue.


8. Expand the Unconfigured Ethernet Ports section.

9. Select port number 29-32, which is connected to the uplink switch, right-click, then select Reconfigure > Configure as Uplink Port.

10. Select Show Interface and select 40GB for Uplink Connection.

11. A pop-up window appears to confirm your selection. Click Yes then OK to continue.

Figure 11 Enabling Uplink Ports Part 1
Configuring VLANs

VLANs are configured as in shown in Table 5.

<table>
<thead>
<tr>
<th>VLAN</th>
<th>NIC Port</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>VLAN13</td>
<td>etho</td>
<td>Data</td>
</tr>
</tbody>
</table>

The NIC will carry the data traffic from VLAN13. A single vNIC is used in this configuration and the Fabric Failover feature in Fabric Interconnects will take care of any physical port down issues. It will be a seamless transition from an application perspective.

To configure VLANs in the Cisco UCS Manager GUI, complete the following steps:

1. Select the LAN tab in the left pane in the UCSM GUI.
2. Select LAN > LAN Cloud > VLANs.
3. Right-click the VLANs under the root organization.
4. Select Create VLANs to create the VLAN.
5. Enter `vlan13` for the VLAN Name.

6. Keep multicast policy as `<not set>`.


8. Enter `13` in the VLAN IDs field for the Create VLAN IDs.

9. Click OK and then, click Finish.

10. Click OK in the success message box.
11. Click OK and then, click Finish.

**Enabling Server Ports**

To enable server ports, complete the following steps:

1. Select the Equipment tab on the top left of the window.


3. Expand the Unconfigured Ethernet Ports section.

4. Select all the ports that are connected to the Servers right-click them, and select Reconfigure > Configure as a Server Port.

5. A pop-up window appears to confirm your selection. Click Yes then OK to continue.


7. Expand the Unconfigured Ethernet Ports section.

8. Select all the ports that are connected to the Servers right-click them, and select Reconfigure > Configure as a Server Port.

9. A pop-up window appears to confirm your selection. Click Yes, then OK to continue.
After the Server Discovery, Port 29-32 will be a Network Port and 1-28 will be Server Ports.

Figure 22  Ports status after the Server Discover

Creating Pools for Service Profile Templates

Creating an Organization

Organizations are used as a means to arrange and restrict access to various groups within the IT organization, thereby enabling multi-tenancy of the compute resources. This document does not assume the use of Organizations; however, the necessary steps are provided for future reference.

To configure an organization within the Cisco UCS Manager GUI, complete the following steps:

1. Click Quick Action icon on the top right corner in the right pane in the Cisco UCS Manager GUI.

2. Select Create Organization from the options
3. Enter a name for the organization.

4. (Optional) Enter a description for the organization.

5. Click OK.

6. Click OK in the success message box.

Creating MAC Address Pools

To create MAC address pools, complete the following steps:

1. Select the LAN tab on the left of the window.

2. Select Pools > root > MAC Pools

3. Right-click MAC Pools under the root organization.

4. Select Create MAC Pool to create the MAC address pool. Enter ucs for the name of the MAC pool.

5. (Optional) Enter a description of the MAC pool.

6. Select Assignment Order Sequential.

7. Click Next.

8. Click Add.

9. Specify a starting MAC address.
10. Specify a size of the MAC address pool, which is sufficient to support the available server resources.

11. Click OK.
12. Click Finish.

13. When the message box displays, click OK.

Creating a Server Pool

A server pool contains a set of servers. These servers typically share the same characteristics. Those characteristics can be their location in the chassis, or an attribute such as server type, amount of memory, local storage, type of CPU, or local drive configuration. You can manually assign a server to a server pool, or use server pool policies and server pool policy qualifications to automate the assignment.

To configure the server pool within the Cisco UCS Manager GUI, complete the following steps:
1. Select the **Servers** tab in the left pane in the UCS Manager GUI.

2. Select **Pools > root**.

3. Right-click the **Server Pools**.

4. Select **Create Server Pool**.

5. Enter your required **name** (*ucs*) for the Server Pool in the name text box.

6. (Optional) enter a description for the organization.

7. Click **Next >** to add the servers.

8. Select all the Cisco UCS C240M5 servers to be added to the server pool that was previously created (*ucs*), then **Click >>** to add them to the pool.
Creating Policies for Service Profile Templates

Creating Host Firmware Package Policy

Firmware management policies allow the administrator to select the corresponding packages for a given server configuration. These include adapters, BIOS, board controllers, FC adapters, HBA options, and storage controller properties as applicable.

To create a firmware management policy for a given server configuration using the Cisco UCS Manager GUI, complete the following steps:

1. Select the Servers tab in the left pane in the UCS Manager GUI.
2. Select Policies > root.
3. Right-click Host Firmware Packages.
4. Select Create Host Firmware Package.
5. Enter the required Host Firmware package name (ucs).
6. Select Simple radio button to configure the Host Firmware package.
7. Select the appropriate Rack package that has been installed.
8. Click OK to complete creating the management firmware package.
9. Click OK.
Creating QoS Policies

To create the QoS policy for a given server configuration using the Cisco UCS Manager GUI, complete the following steps:

Platinum Policy

1. Select the LAN tab in the left pane in the Cisco UCS Manager GUI.
2. Select Policies > root.
4. Select Create QoS Policy.
5. Enter Platinum as the name of the policy.

6. Select Platinum from the drop-down list.

7. Keep the Burst (Bytes) field set to default (10240).

8. Keep the Rate (Kbps) field set to default (line-rate).

9. Keep Host Control radio button set to default (none).

10. When the pop-up window appears, click OK to complete the creation of the Policy.

Setting Jumbo Frames

To set Jumbo frames and enable QoS, complete the following steps:

1. Select the LAN tab in the left pane in the Cisco UCS Manager GUI.
2. Select **LAN Cloud > QoS System Class**.

3. In the right pane, select the **General tab**

4. In the **Platinum row**, enter 9216 for MTU.

5. Check the **Enabled Check box next to Platinum**.

6. In the **Best Effort row**, select **none** for weight.

7. In the **Fiber Channel row**, select **none** for weight.

8. Click **Save Changes**.

9. Click **OK**.

---

**Creating the Local Disk Configuration Policy**

To create local disk configuration in the Cisco UCS Manager GUI, complete the following steps:

1. Select the **Servers tab** on the left pane in the Cisco UCS Manager GUI.

2. Go to **Policies > root**.

3. Right-click **Local Disk Config Policies**.

4. Select **Create Local Disk Configuration Policy**.

5. Enter **ucs** as the local disk configuration policy name.

6. Change the **Mode** to **Any Configuration**. Check the **Protect Configuration box**.

7. Keep the **FlexFlash State field** as default (Disable).

8. Keep the **FlexFlash RAID Reporting State field** as default (Disable).

9. Click **OK** to complete the creation of the Local Disk Configuration Policy.
10. Click OK.

Creating Server BIOS Policy

The BIOS policy feature in Cisco UCS automates the BIOS configuration process. The traditional method of setting the BIOS is manually, and is often error-prone. By creating a BIOS policy and assigning the policy to a server or group of servers, can enable transparency within the BIOS settings configuration.

Note: BIOS settings can have a significant performance impact, depending on the workload and the applications. The BIOS settings listed in this section is for configurations optimized for best performance which can be adjusted based on the application, performance, and energy efficiency requirements.

To create a server BIOS policy using the Cisco UCS Manager GUI, complete the following steps:

1. Select the Servers tab in the left pane in the UCS Manager GUI.

2. Select Policies > root.


4. Select Create BIOS Policy.

5. Enter your preferred BIOS policy name (ucs).

6. Change the BIOS settings as shown in the following figures.

7. Only changes that need to be made are in the Processor and RAS Memory settings.
Creating the Boot Policy

To create boot policies within the Cisco UCS Manager GUI, complete the following steps:

1. Select the Servers tab in the left pane in the UCS Manager GUI.
2. Select Policies > root.
3. Right-click the Boot Policies.
4. Select Create Boot Policy.
5. Enter ucs as the boot policy name.
6. (Optional) enter a description for the boot policy.
7. Keep the Reboot on Boot Order Change check box unchecked.
8. Keep Enforce vNIC/vHBA/iSCSI Name check box checked.


10. Expand Local Devices > Add CD/DVD and select Add Local CD/DVD.

11. Expand Local Devices and select Add Local Disk.

12. Expand vNICs and select Add LAN Boot and enter eth0.

13. Click OK to add the Boot Policy.

14. Click OK.
Creating Power Control Policy

To create Power Control policies within the Cisco UCS Manager GUI, complete the following steps:

1. Select the Servers tab in the left pane in the Cisco UCS Manager GUI.

2. Select Policies > root.

3. Right-click the Power Control Policies.

4. Select Create Power Control Policy.

5. Enter ucs as the Power Control policy name.

6. (Optional) enter a description for the boot policy.

7. Select Performance for Fan Speed Policy.

8. Select No cap for Power Capping selection.

9. Click OK to create the Power Control Policy.

10. Click OK.
Creating a Service Profile Template

To create a Service Profile Template, complete the following steps:

1. Select the Servers tab in the left pane in the Cisco UCS Manager GUI.

2. Right-click Service Profile Templates.

3. Select Create Service Profile Template.

The Create Service Profile Template window appears.

To identify the service profile template, complete the following steps:
1. Name the service profile template as **ucs**. Select the **Updating Template** radio button.

2. In the **UUID** section, select **Hardware Default** as the **UUID pool**.

3. Click **Next** to continue to the next section.

![Create Service Profile Template](image)

**Configuring the Storage Provisioning for the Template**

To configure storage policies, complete the following steps:

1. Go to the **Local Disk Configuration Policy** tab, and select **ucs** for the **Local Storage**.

2. Click **Next** to continue to the next section.
3. Click Next once the Networking window appears to go to the next section.

**Configuring Network Settings for the Template**

1. Keep the Dynamic vNIC Connection Policy field at the default.
2. Select Expert radio button for the option how would you like to configure LAN connectivity?
3. Click Add to add a vNIC to the template.
4. The Create vNIC window displays. Name the vNIC as eth0.

5. Select UCS in the MAC Address Assignment pool.

6. Select the Fabric A radio button and check the Enable failover check box for the Fabric ID.

7. Check the VLAN13 check box for VLANs and select the Native VLAN radio button.

8. Select MTU size as 9000.

9. Select adapter policy as Linux.

10. Select QoS Policy as Platinum.

11. Keep the Network Control Policy as Default.

12. Click OK.
Note: Optionally Network Bonding can be setup on the vNICs for each host for redundancy as well as for increased throughput.

13. Click Next to continue with SAN Connectivity.

14. Select no vHBAs for How would you like to configure SAN Connectivity?
15. Click Next to continue with Zoning.

16. Click Next to continue with vNIC/vHBA placement.
17. Click **Next** to configure vMedia Policy.

**Configuring the vMedia Policy for the Template**

1. Click **Next** once the vMedia Policy window appears to go to the next section.

**Configuring Server Boot Order for the Template**

To set the boot order for the servers, complete the following steps:
1. Select ucs in the Boot Policy name field.

2. Review to make sure that all of the boot devices were created and identified.

3. Verify that the boot devices are in the correct boot sequence.

4. Click OK.

5. Click Next to continue to the next section.

6. In the Maintenance Policy window, apply the maintenance policy.

7. Keep the Maintenance policy at no policy used by default. Click Next to continue to the next section.
Configuring Server Assignment for the Template

In the Server Assignment window, to assign the servers to the pool, complete the following steps:

1. Select ucs for the Pool Assignment field.
2. Select the power state to be Up.
4. Check the Restrict Migration check box.
5. Select ucs in Host Firmware Package.
Configuring Operational Policies for the Template

In the Operational Policies Window, complete the following steps:

1. Select `ucs` in the BIOS Policy field.

2. Select `ucs` in the Power Control Policy field.
3. Click **Finish** to create the Service Profile template.

4. Click **OK** in the pop-up window to proceed.

5. Select the **Servers** tab in the left pane of the Cisco UCS Manager GUI.

6. Go to Service Profile Templates > root.

7. Right-click **Service Profile Templates ucs**.

8. Select **Create Service Profiles From Template**.
The Create Service Profiles from Template window appears.

Association of the Service Profiles will take place automatically.

**Installing Red Hat Enterprise Linux 7.4**

The following section provides detailed procedures for installing Red Hat Enterprise Linux 7.4 using Software RAID (OS based Mirroring) on Cisco UCS C240 M5 servers. There are multiple ways to install the Red Hat Linux operating system. The installation procedure described in this deployment guide uses KVM console and virtual media from Cisco UCS Manager.

*Note: This requires RHEL 7.4 DVD/ISO for the installation*

To install the Red Hat Linux 7.4 operating system, complete the following steps:

1. Log in to the Cisco UCS 6332 Fabric Interconnect and launch the Cisco UCS Manager application.

2. Select the **Equipment** tab.
3. In the navigation pane expand Rack-Mounts and then Servers.

4. In the right pane, click the KVM Console >>.

5. Click O.K on KVM Console – Select IP address pop-up window.

6. Click the link to launch the KVM console.

7. Point the cursor over the top right corner, select the Virtual Media tab.
8. Click the **Activate Virtual Devices** found in Virtual Media tab.

9. Click the Virtual Media tab again to select **CD/DVD**.

10. Select **Map Drive** in the Virtual Disk Management windows.

**Note:** The Red Hat Enterprise Linux 7.4 DVD is assumed to be on the client machine.

12. Click **Open** to add the image to the list of virtual media.

13. In the KVM window, select the **KVM** tab to monitor during boot.

14. In the KVM window, select the **Macros > Static Macros > Ctrl-Alt-Del button** in the upper left corner.

15. Click **OK**.
16. Click OK to reboot the system.

17. Press F6 key on the keyboard to select install media.

Note: Press F6 on your keyboard as soon as possible when the screen above appears to avoid the server reboot again.

18. On reboot, the machine detects the presence of the Red Hat Enterprise Linux Server 7.4 install media.

20. Skip the Media test and start the installation. Select language of installation and click Continue.

21. Select Date and time, which pops up another window as shown below:
22. Select the location on the map, set the time, and click Done.
23. Click Installation Destination.
24. This opens a new window with the boot disks. Make the selection, and choose I will configure partitioning. Click Done.

25. This opens the new window for creating the partitions. Click on the + sign to add a new partition as shown below, boot partition of size 2048 MB.

26. Click Add MountPoint to add the partition.
27. Change the Device type to RAID and make sure the RAID Level is RAID1 (Redundancy) and click on Update Settings to save the changes.

28. Click the + sign to create the swap partition of size 2048 MB as shown below.

29. Change the Device type to RAID and RAID level to RAID1 (Redundancy) and click Update Settings.
30. Click + to add the / partition. The size can be left empty so it uses the remaining capacity and click Add Mountpoint.

31. Change the Device type to RAID and RAID level to RAID1 (Redundancy). Click Update Settings.
32. Click Done to go back to the main screen and continue the Installation.

33. Click Software Selection.
34. Select Infrastructure Server and select the Add-Ons as noted below. Click Done.

35. Click Network and Hostname and configure Hostname and Networking for the Host.
36. Type in the hostname as shown below.

37. Click Configure to open the Network Connectivity window. Click IPV4Settings.
38. Change the Method to **Manual** and click **Add** to enter the IP Address, Netmask, and Gateway details.
Solution Design
Click Save and update the hostname and turn Ethernet ON. Click Done to return to the main menu.
40. Click Begin Installation in the main menu.
41. Select Root Password in the User Settings.

42. Enter the Root Password and click Done.
43. When the installation is complete reboot the system.

44. Repeat steps 1 to 43 to install Red Hat Enterprise Linux 7.4 on the remaining servers.

Note: The OS installation and configuration of the nodes that is mentioned above can be automated through PXE boot or third party tools.
The hostnames and their corresponding IP addresses are shown in Table 6.

<table>
<thead>
<tr>
<th>Hostname</th>
<th>eth0</th>
</tr>
</thead>
<tbody>
<tr>
<td>rhel1</td>
<td>10.13.1.50</td>
</tr>
<tr>
<td>rhel2</td>
<td>10.13.1.51</td>
</tr>
<tr>
<td>rhel3</td>
<td>10.13.1.52</td>
</tr>
<tr>
<td>rhel4</td>
<td>10.13.1.53</td>
</tr>
<tr>
<td>rhel5</td>
<td>10.13.1.54</td>
</tr>
<tr>
<td>rhel6</td>
<td>10.13.1.55</td>
</tr>
<tr>
<td>rhel7</td>
<td>10.13.1.56</td>
</tr>
<tr>
<td>rhel8</td>
<td>10.13.1.57</td>
</tr>
<tr>
<td>rhel9</td>
<td>10.13.1.58</td>
</tr>
<tr>
<td>rhel10</td>
<td>10.13.1.59</td>
</tr>
<tr>
<td>rhel11</td>
<td>10.13.1.60</td>
</tr>
<tr>
<td>rhel12</td>
<td>10.13.1.61</td>
</tr>
<tr>
<td>rhel13</td>
<td>10.13.1.62</td>
</tr>
<tr>
<td>rhel14</td>
<td>10.13.1.63</td>
</tr>
<tr>
<td>rhel15</td>
<td>10.13.1.64</td>
</tr>
<tr>
<td>rhel16</td>
<td>10.13.1.65</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>rhel24</td>
<td>10.13.1.73</td>
</tr>
<tr>
<td>cdsw1.cisco.com</td>
<td>10.13.1.250</td>
</tr>
<tr>
<td>Cdsw2.cisco.com</td>
<td>10.13.1.251</td>
</tr>
<tr>
<td>Cdsw3.cisco.com</td>
<td>10.13.1.252</td>
</tr>
<tr>
<td>Cdsw4.cisco.com</td>
<td>10.13.1.253</td>
</tr>
</tbody>
</table>
Note: Cloudera does not recommend multi-homing configurations, so please assign only one network to each node.

Post OS Install Configuration

Choose one of the nodes of the cluster or a separate node as the Admin Node for management such as CDH installation, cluster parallel shell, creating a local Red Hat repo and others. In this document, we use rhel1 for this purpose.

Setting Up Password-less Login

To manage all of the clusters nodes from the admin node password-less login needs to be setup. It assists in automating common tasks with clustershell (clush, a cluster wide parallel shell), and shell-scripts without having to use passwords.

When Red Hat Linux is installed across all the nodes in the cluster, to enable password-less login across all the nodes, complete the following steps:

1. Login to the Admin Node (rhel1).
   
   ```bash
   #ssh 10.13.1.50
   ```

2. Run the ssh-keygen command to create both public and private keys on the admin node.

   ```bash
   [root@rhel1 ~]$ ssh-keygen
   Generating public/private rsa key pair.
   Enter file in which to save the key (/root/.ssh/id_rsa):
   /root/.ssh/id_rsa already exists.
   Overwrite (y/n)? y
   Enter passphrase (empty for no passphrase):
   Enter same passphrase again:
   Your identification has been saved in /root/.ssh/id_rsa.         ⬜
   Your public key has been saved in /root/.ssh/id_rsa.pub.
   The key fingerprint is:
   SHA256:LAJF8TpvkirkJQ07YjZxQ1mLFXk3Mu1Fc+O0cQ1go root@rhel1
   The key's randomart image is:
   +-----[RSA 2048]-----+
   | ooo oo |
   | ...oo. = oo |
   | + ooo  |
   | .. oE + . |
   | +0.080 = 0 |
   | =+.. + .. |
   | Bo+o + . |
   | . oo=00 . |
   | oo oo |
   +-----[SHA256]-----+
   [root@rhel1 ~]#
   ```

   Then run the following command from the admin node to copy the public key id_rsa.pub to all the nodes of the cluster.
   
   ```bash
   ssh-copy-id appends the keys to the remote-host's .ssh/authorized_keys.
   ```

   ```bash
   #for IP in {50..73}; do echo -n "$IP -> "; ssh-copy-id -i ~/.ssh/id_rsa.pub 10.13.1.IP; done
   ```
#for IP in {250..253}; do echo -n "$IP -> "; ssh-copy-id -i ~/.ssh/id_rsa.pub 10.13.1.$IP; done

4. Enter yes for Are you sure you want to continue connecting (yes/no)?

5. Enter the password of the remote host.

Configuring /etc/hosts

Setup /etc/hosts on the Admin node; this is a pre-configuration to setup DNS as shown in the next section.

To create the host file on the admin node, complete the following steps:

1. Populate the host file with IP addresses and corresponding hostnames on the Admin node (rhel1) and other nodes as follows:

2. On Admin Node (rhel1):

   #vi /etc/hosts

   127.0.0.1 localhost localhost.localdomain localhost4 \ localhost4.localdomain4
   ::1 localhost localhost.localdomain localhost6 \ localhost6.localdomain6

   10.13.1.50 rhel1
   10.13.1.51 rhel2
   10.13.1.52 rhel3
   10.13.1.53 rhel4
   10.13.1.54 rhel5
   10.13.1.55 rhel6
   10.13.1.56 rhel7

   ... 

   10.13.1.73 rhel24
   10.13.1.250 cdsw1.cisco.com
   10.13.1.251 cdsw2.cisco.com
   10.13.1.252 cdsw3.cisco.com
   10.13.1.253 cdsw4.cisco.com

Creating a Red Hat Enterprise Linux (RHEL) 7.4 Local Repo

To create a repository using RHEL DVD or ISO on the admin node (in this deployment rhel1 is used for this purpose), create a directory with all the required RPMs, run the createrepo command and then publish the resulting repository.

1. Log on to rhel1. Create a directory that would contain the repository.

   #mkdir -p /var/www/html/rhelrepo

2. Copy the contents of the Red Hat DVD to /var/www/html/rhelrepo

3. Alternatively, if you have access to a Red Hat ISO Image, Copy the ISO file to rhel1.

4. Log back into rhel1 and create the mount directory.

   #scp rhel-server-7.4-x86_64-dvd.iso rhel1:/root/
   #mkdir -p /mnt/rheliso

   #mount -t iso9660 -o loop /root/rhel-server-7.4-x86_64-dvd.iso /mnt/rheliso/

   #cp -r /mnt/rheliso/* /var/www/html/rhelrepo

6. On rhel1 create a .repo file to enable the use of the yum command.

   #vi /var/www/html/rhelrepo/rheliso.repo
   [rhel7.4]
   name=Red Hat Enterprise Linux 7.4
   baseurl=http://10.13.1.50/rhelrepo
   gpgcheck=0
   enabled=1


   #cp /var/www/html/rhelrepo/rheliso.repo /etc/yum.repos.d/

Note: Based on this repo file yum requires httpd to be running on rhel1 for other nodes to access the repository.

8. To make use of repository files on rhel1 without httpd, edit the baseurl of repo file
    /etc/yum.repos.d/rheliso.repo to point repository location in the file system.

   #vi /etc/yum.repos.d/rheliso.repo
   [rhel7.4]
   name=Red Hat Enterprise Linux 7.4
   baseurl=file:///var/www/html/rhelrepo
   gpgcheck=0
   enabled=1

Creating the Red Hat Repository Database

1. Install the createrepo package on admin node (rhel1). Use it to regenerate the repository database(s) for the local copy of the RHEL DVD contents.

   #yum -y install createrepo
2. Run createrepo on the RHEL repository to create the repo database on admin node

   # cd /var/www/html/rhelrepo
   # createrepo .

Setting up ClusterShell

ClusterShell (or clush) is the cluster-wide shell that runs commands on several hosts in parallel. To set up the ClusterShell, complete the following steps:

1. From the system connected to the Internet download Cluster shell (clush) and install it on rhel1. Cluster shell is available from EPEL (Extra Packages for Enterprise Linux) repository.

#scp clustershell-1.7-1.el7.noarch.rpm rhel1:/root/
Login to rhel1 and install cluster shell.
#yum -y install clustershell-1.71.el7.noarch.rpm

2. Edit /etc/clustershell/groups.d/local.cfg file to include hostnames for all the nodes of the cluster. This set of hosts is taken when running clush with the `-a` option.

3. For 28 node cluster as in our CVD, set groups file as follows:

   ```bash
   ```

Note: For more information and documentation on ClusterShell, visit https://github.com/ceahpc/clustershell/wiki/UserAndProgrammingGuide.

Note: clustershell will not work if not ssh to the machine earlier (as it requires to be in known_hosts file), for instance, as in the case below for rhel<host>.
Installing httpd

Setting up RHEL repo on the admin node requires httpd. To set up RHEL repository on the admin node, complete the following steps:

1. Install httpd on the admin node to host repositories.
   
   The Red Hat repository is hosted using HTTP on the admin node, this machine is accessible by all the hosts in the cluster.
   
   #yum -y install httpd

2. Add ServerName and make the necessary changes to the server configuration file.
   
   #vi /etc/httpd/conf/httpd.conf
   ServerName 10.13.1.50:80

3. Start httpd
   
   #service httpd start
   #chkconfig httpd on

Set Up All Nodes to use the RHEL Repository

Note: Based on this repo file yum requires httpd to be running on rhel1 for other nodes to access the repository.

1. Copy the rheliso.repo to all the nodes of the cluster.
   
   #clush -w rhel[1-24],cdsw[1-4].cisco.com -b -c
   /var/www/html/rhelrepo/rheliso.repo --dest=/etc/yum.repos.d/

2. Also copy the /etc/hosts file to all nodes.
   
   #clush -a -b -c /etc/hosts --dest=/etc/hosts

3. Purge the yum caches after this
   
   #clush -a -B yum clean all
   #clush -a -B yum repolist

Note: While suggested configuration is to disable SELinux as shown below, if for any reason SELinux needs to be enabled on the cluster, then ensure to run the following to make sure that the httpd is able to read the Yum reposfiles.

   #chcon -R -t httpd_sys_content_t /var/www/html/

Configuring DNS

This section details setting up DNS using dnsmasq as an example based on the /etc/hosts configuration setup in the earlier section.

To create the host file across all the nodes in the cluster, complete the following steps:

1. Disable Network manager on all nodes.
Solution Design

#clush -a -b service NetworkManager stop
#clush -a -b chkconfig NetworkManager off

2. Update /etc/resolv.conf file to point to Admin Node.

   #vi /etc/resolv.conf

   nameserver 10.13.1.50

Note: Cloudera CDSW requires Wildcard entry dns configurations as detailed in section Cloudera Data Science Workbench.

Note: Alternatively #systemctl start NetworkManager.service can be used to start the service. #systemctl stop NetworkManager.service can be used to stop the service. Use #systemctl disable NetworkManager.service to stop a service from being automatically started at boot time.

3. Install and Start dnsmasq on Admin node

   #service dnsmasq start

   #chkconfig dnsmasq on

4. Deploy /etc/resolv.conf from the admin node (rhel1) to all the nodes via the following clush command:

   #clush -a -B -c /etc/resolv.conf

Note: A clush copy without –dest copies to the same directory location as the source-file directory.

5. Make sure DNS is working fine by running the following command on Admin node and any data-node

   [root@rhel2 ~]# nslookup rhel1
   Server:  10.13.1.50
   Address: 10.13.1.50#53
   Name:    rhel1
   Address: 10.13.1.50

Note: yum install -y bind-utils will need to be run for nslookup utility to run.

Upgrading the Cisco Network Driver for VIC1387

The latest Cisco Network driver is required for performance and updates. The latest drivers can be downloaded from the link below:

https://software.cisco.com/download/home/286318800/type/283853158/release/3.1%25283%2529

In the ISO image, the required driver kmod/enic-2.3.0.44-rhel7u4.el7.x86_64.rpm can be located at Linux\Network\Cisco\VIC\RHEL\RHEL7.4.

1. From a node connected to the Internet, download, extract and transfer kmod-enic-2.3.0.44-rhel7u4.el7.x86_64.rpm to rhel1 (admin node).
2. Install the rpm on all nodes of the cluster using the following clush commands. For this example, the rpm is assumed to be in present working directory of rhel1.

3. [root@rhel1 ~]# clush -a -b -c kmod-enic-2.3.0.44-rhel7u4.el7.x86_64.rpm

4. [root@rhel1 ~]# clush -a -b "rpm -ivh kmod-enic-2.3.0.44-rhel7u4.el7.x86_64.rpm "

5. Make sure that the above installed version of kmod-enic driver is being used on all nodes by running the command "modinfo enic" on all nodes

   [root@rhel1 ~]# clush -a -b "modinfo enic | head -5"

6. Also it is recommended to download the kmod-megaraid driver for higher performance, the RPM can be found in the same package at \Linux\Storage\LSI\Cisco_Storage_12G_SAS_RAID_controller\RHEL\RHEL7.4

Installing xfsprogs

From the admin node rhel1 run the command shown below to Install xfsprogs on all the nodes for xfs filesystem.

#clush -a -B yum -y install xfsprogs

NTP Configuration

The Network Time Protocol (NTP) is used to synchronize the time of all the nodes within the cluster. The Network Time Protocol daemon (ntpd) sets and maintains the system time of day in synchronism with the timeserver located in the admin node (rhel1). Configuring NTP is critical for any Hadoop Cluster. If server clocks in the cluster drift out of sync, serious problems will occur with HBase and other services.

#clush -a -b "yum -y install ntp"

Note: Installing an internal NTP server keeps your cluster synchronized even when an outside NTP server is inaccessible.

1. Configure /etc/ntp.conf on the admin node only with the following contents:

   #vi /etc/ntp.conf
driftfile /var/lib/ntp/drift
restrict 127.0.0.1
restrict -6 ::1
server 127.127.1.0
fudge 127.127.1.0 stratum 10
includefile /etc/ntp/crypto/pw
keys /etc/ntp/keys

2. Create /root/ntp.conf on the admin node and copy it to all nodes:

   #vi /root/ntp.conf
server 10.13.1.50
driftfile /var/lib/ntp/drift
restrict 127.0.0.1
restrict -6 ::1
includefile /etc/ntp/crypto/pw
keys /etc/ntp/keys
3. Copy ntp.conf file from the admin node to /etc of all the nodes by executing the following commands in the admin node (rhel1)

```bash
# for SERVER in {50..73}; do scp /root/ntp.conf 10.13.1.$SERVER:/etc/ntp.conf; done
# for SERVER in {250..253}; do scp /root/ntp.conf 10.13.1.$SERVER:/etc/ntp.conf; done
```

4. Run the following to syncronize the time and restart NTP daemon on all nodes.

```bash
# clush -a -b "service ntpd stop"
# clush -a -b "ntpdate rhel1"
# clush -a -b "service ntpd start"
```

5. Make sure to restart of NTP daemon across reboots:

```bash
# clush -a -b "systemctl enable ntpd"
```

Alternatively, the new Chrony service can be installed, that is quicker to synchronize clocks in mobile and virtual systems.

6. Install the Chrony service:

```bash
# yum install -y chrony
```

7. Activate the Chrony service at boot:

```bash
# systemctl enable chronyd
```

8. Start the Chrony service:

```bash
# systemctl start chronyd
```

The Chrony configuration is in the `/etc/chrony.conf` file, configured similar to `/etc/ntp.conf`.

### Enabling Syslog

Syslog must be enabled on each node to preserve logs regarding killed processes or failed jobs. Modern versions such as syslog-ng and rsyslog are possible, making it more difficult to be sure that a syslog daemon is present. One of the following commands should suffice to confirm that the service is properly configured:

```bash
#clush -B -a rsyslogd -v
#clush -B -a service rsyslog status
```
**Setting ulimit**

On each node, `ulimit -n` specifies the number of inodes that can be opened simultaneously. With the default value of 1024, the system appears to be out of disk space and shows no inodes available. This value should be set to 64,000 on every node.

Higher values are unlikely to result in an appreciable performance gain.

1. For setting the ulimit on Redhat, edit `/etc/security/limits.conf` on admin node rhel1 and add the following lines:

   ```
   root soft nofile 64000
   root hard nofile 64000
   ```

   ```bash
   [root@rhel1 ~]# cat /etc/security/limits.conf | grep 64000
   root soft nofile 64000
   root hard nofile 64000
   ```

2. Copy the `/etc/security/limits.conf` file from admin node (rhel1) to all the nodes using the following command.

   ```bash
   #clush -a -b -c /etc/security/limits.conf --dest=/etc/security/
   ```

3. Make sure that the `/etc/pam.d/su` file contains the following settings:

   ```
   #PAM-1.0
   auth        sufficient    pam_rootOK.so
   # Uncomment the following line to implicitly trust users in the "wheel" group.
   #auth        sufficient    pam_wheel.so trust use_uid
   # Uncomment the following line to require a user to be in the "wheel" group.
   #auth        required      pam_wheel.so use_uid
   auth        include        system-auth
   account      sufficient    pam_succeed_if.so uid = 0 use_uid quiet
   account      include        system-auth
   password     include        system-auth
   session      include        system-auth
   session      optional      pam_xauth.so
   ```

   Note: The ulimit values are applied on a new shell, running the command on a node on an earlier instance of a shell will show old values.

**Disabling SELinux**

SELinux must be disabled during the install procedure and cluster setup. SELinux can be enabled after installation and while the cluster is running.

SELinux can be disabled by editing `/etc/selinux/config` and changing the `SELINUX` line to `SELINUX=disabled`. The following command will disable SELINUX on all nodes.
Solution Design

#clush -a -b "sed -i 's/SELINUX=enforcing/SELINUX=disabled/g' /etc/selinux/config"

#clush -a -b "setenforce 0"

Note: The above command may fail if SELinux is already disabled.

Reboot the machine, if needed for SELinux to be disabled in case it does not take effect. It can checked using
#clush -a -b sestatus

Set TCP Retries

Adjusting the tcp_retries parameter for the system network enables faster detection of failed nodes. Given the advanced networking features of UCS, this is a safe and recommended change (failures observed at the operating system layer are most likely serious rather than transitory). On each node, set the number of TCP retries to 5 can help detect unreachable nodes with less latency.

1. Edit the file /etc/sysctl.conf and on admin node rhel1 and add the following lines:

   net.ipv4.tcp_retries2=5

2. Copy the /etc/sysctl.conf file from admin node (rhel1) to all the nodes using the following command:

   #clush -a -b -c /etc/sysctl.conf --dest=/etc/

3. Load the settings from default sysctl file /etc/sysctl.conf by running.

   #clush -B -a sysctl -p

Disabling the Linux Firewall

The default Linux firewall settings are far too restrictive for any Hadoop deployment. Since the UCS Big Data deployment will be in its own isolated network there is no need for that additional firewall.

#clush -a -b " firewall-cmd --zone=public --add-port=80/tcp --permanent"
#clush -a -b "firewall-cmd --reload"
#clush -a -b “systemctl disable firewalld”

Disable Swapping

1. In order to reduce Swapping, run the following on all nodes. Variable `vm.swappiness` defines how often swap should be used, 60 is default.

   #clush -a -b " echo 'vm.swappiness=1' >> /etc/sysctl.conf"

2. Load the settings from default sysctl file /etc/sysctl.conf.

   #clush -a -b "sysctl -p"

Disable Transparent Huge Pages

Disabling Transparent Huge Pages (THP) reduces elevated CPU usage caused by THP.

#clush -a -b "echo never > /sys/kernel/mm/transparent_hugepage/defrag"
#clush -a -b "echo never > /sys/kernel/mm/transparent_hugepage/_enabled"
1. The above commands must be run for every reboot, so copy this command to /etc/rc.local so they are executed automatically for every reboot.

2. On the Admin node, run the following commands:

   # rm -f /root/thp_disable
   # echo "echo never > /sys/kernel/mm/transparent_hugepage/enabled" >> /root/thp_disable
   # echo "echo never > /sys/kernel/mm/transparent_hugepage/defrag " >> /root/thp_disable

3. Copy file to each node:

   # clush -a -b -c /root/thp_disable

4. Append the content of file thp_disable to /etc/rc.local:

   # clush -a -b "cat /root/thp_disable >> /etc/rc.local"

### Disable IPv6 Defaults

1. Disable IPv6 as the addresses used are IPv4.

   # clush -a -b "echo 'net.ipv6.conf.all.disable_ipv6 = 1' >> /etc/sysctl.conf"
   # clush -a -b "echo 'net.ipv6.conf.default.disable_ipv6 = 1' >> /etc/sysctl.conf"
   # clush -a -b "echo 'net.ipv6.conf.lo.disable_ipv6 = 1' >> /etc/sysctl.conf"

2. Load the settings from default sysctl file /etc/sysctl.conf.

   # clush -a -b "sysctl -p"

### Configuring Data Drives on Name Node And Other Management Nodes

This section describes steps to configure non-OS disk drives as RAID1 using Storcli command as described below. All the drives are going to be part of a single RAID1 volume. This volume can be used for Staging any client data to be loaded to HDFS. This volume will not be used for HDFS data.


2. Extract the zip file and copy storcli-007.0504.0000.0000-1.noarch.rpm from the linux directory.

3. Download storcli and its dependencies and transfer to Admin node.

   # scp storcli-007.0504.0000.0000-1.noarch.rpm rhel1:/root/

4. Copy storcli rpm to all the nodes using the following commands:

   # clush -a -b -c /root/storcli-007.0504.0000.0000-1.noarch.rpm --dest=/root/
5. Run the below command to install storcli on all the nodes

   #clush -a -b “rpm -ivh storcli-007.0504.0000.0000-1.noarch.rpm”

6. Run the below command to copy storcli64 to root directory.

   #cd /opt/MegaRAID/storcli/
   #cp storcli64 /root/

   [root@rhel ~]# cd /opt/MegaRAID/storcli/
   [root@rhel storcli]# ls
   install.log  libstorelib-2.so  libstorelibic-2.so.14.07-0  storcli64
   [root@rhel storcli]# cp storcli64 /root/

7. Copy storcli64 to all the nodes using the following commands:

   #clush -a -b -c /root/storcli64 --dest=/root/

8. Run the following script as root user on rhel1 to rhel3 to create the virtual drives for the management nodes.

   #vi /root/raid1.sh
   .;/storcli64 -cfgldadd
   The above script requires enclosure ID as a parameter.

9. Run the following command to get enclosure id.

   #./storcli64 pdlist -a0 | grep Enc | grep -v 252 | awk '{print $4}' | sort | uniq -c | awk '{print $2}'
   #chmod 755 raid1.sh

10. Run MegaCli script as follows.

    #./raid1.sh <EnclosureID> obtained by running the command above
    WB: Write back
    RA: Read Ahead
    NoCachedBadBBU: Do not write cache when the BBU is bad.
    Strpsz1024: Strip Size of 1024K

---

Note: The command above will not override any existing configuration. To clear and reconfigure existing configurations refer to Embedded MegaRAID Software Users Guide available at www.broadcom.com.

Cloudera recommends the following disk configuration for the master nodes:

- At least 10 physical disks in following configuration
- 2 x RAID1 OS (Root disk)
- 4 x RAID 10 (DB filesystems)
- 2 x RAID 1 HDFS NameNode metadata
- 1 x JBOD - ZooKeeper
- 1 x JBOD - Quorum JournalNode
Configuring Data Drives on Data Nodes

To configure non-OS disk drives as individual RAID0 volumes using StorCli command, complete the following steps. These volumes are going to be used for HDFS Data.

1. Issue the following command from the admin node to create the virtual drives with individual RAID0 configurations on all the data nodes.

   ```
   #clush -w rhel[4-24] -B ./storcli64 -cfgeachdskraid0 WB RA direct
   NoCachedBadBBU strpsz1024 -a0
   WB: Write back
   RA: Read Ahead
   NoCachedBadBBU: Do not write cache when the BBU is bad.
   Strpsz1024: Strip Size of 1024K
   ```

   **Note:** Create Raid 1 for the Cloudera Data Science workbench nodes as shown as management nodes.

   **Note:** The command above will not override existing configurations. To clear and reconfigure existing configurations refer to Embedded MegaRAID Software Users Guide available at www.broadcom.com.

Configuring the Filesystem for NameNodes and Datanodes

The following script will format and mount the available volumes on each node whether it is Namenode or Data node. OS boot partition is going to be skipped. All drives are mounted based on their UUID as /data/disk1, /data/disk2, etc. To configure the filesystem for NameNodes and DataNodes, complete the following steps:

1. On the Admin node, create a file containing the following script.

```bash
#!/bin/bash
[[ "-x" == "$(1)" ]] && set -x && set -v && shift 1
count=1
for X in /sys/class/scsi_host/host?/scan
do
echo '- - -' > ${X}
done
for X in /dev/sd?
do
list+=$(echo $X " ")
done
for X in /dev/sd??
do
list+=$(echo $X " ")
done
for X in $list
```

   **Note:** The script assumes there are no partitions already existing on the data volumes. If there are partitions, delete them before running the script. This process is documented in the "Note" section at the end of the section.
```bash
do
echo "========"
echo $X
echo "========"
if [[ -b ${X} && `/sbin/parted -s ${X} print quit|/bin/grep -c boot` -ne 0 ]]
then
  echo "$X bootable - skipping."
  continue
else
  Y=${X##*/}
echo "Formatting and Mounting Drive => ${X}"
  /sbin/mkfs.xfs -f ${X}
  (( $? )) && continue
  #Identify UUID
  UUID=`blkid ${X} | cut -d " " -f2 | cut -d ":" -f2 | sed 's//\"//g'`
  /bin/mkdir -p /data/disk${count}
  (( $? )) && continue
  echo "UUID of ${X} = ${UUID}, mounting ${X} using UUID on /data/disk${count}"
  /bin/mount -t xfs -o inode64,noatime,nobarrier -U ${UUID} /data/disk${count}
  (( $? )) && continue
  echo "UUID=${UUID} /data/disk${count} xfs inode64,noatime,nobarrier 0 0" >> /etc/fstab
  ((count++))
fi
done
```

Note: Do not run this script on the Cloudera Data Science Workbench nodes

3. Run the following command to copy driveconf.sh to all the nodes:

   ```bash
   #chmod 755 /root/driveconf.sh
   #clush -a -B -c /root/driveconf.sh
   ```

4. Run the following command from the admin node to run the script across all data nodes

   ```bash
   #clush -a -B /root/driveconf.sh
   ```

5. Run the following from the admin node to list the partitions and mount points

   ```bash
   #clush -a -B df -h
   #clush -a -B mount
   #clush -a -B cat /etc/fstab
   ```

Note: If there is a need to delete any partitions, it can be done so using the following.

6. Run the mount command (`mount`) to identify which drive is mounted to which device /dev/sd<*>

7. umount the drive for which partition is to be deleted and run fdisk to delete as shown below.

Note: Care should be taken not to delete the OS partition as this will wipe out the OS.

96
Cluster Verification

This section describes the steps to create the script `cluster_verification.sh` that helps to verify the CPU, memory, NIC, and storage adapter settings across the cluster on all nodes. This script also checks additional prerequisites such as NTP status, SELinux status, ulimit settings, JAVA_HOME settings and JDK version, IP address and hostname resolution, Linux version and firewall settings.

1. Create the script `cluster_verification.sh` as shown, on the Admin node (rhel1).

```bash
#!/bin/bash
shopt -s expand_aliases,

# Setting Color codes
green='\e[0;32m'
red='\e[0;31m'
NC='\e[0m' # No Color

echo "${green} === Cisco UCS Integrated Infrastructure for Big Data and Analytics \ Cluster Verification === ${NC}"

echo "${green} ==== System Information  ==== ${NC}"

crush -a-B " `which dmidecode` |grep -A2 '^System Information'"

echo "${green}BIOS ${NC}"

crush -a-B " `which dmidecode` | grep -A3 '^BIOS I'"

echo "${green}Memory ${NC}"

crush -a -B "cat /proc/meminfo | grep -i ^memt | uniq"

echo "${green}Number of Dimms ${NC}"

crush -a -B "echo -n 'DIMM slots: '; `which dmidecode` |grep -c "^[[:space:]]*Locator:'"

crush -a -B "echo -n 'DIMM count is: '; `which dmidecode` | grep "^"Size"| grep -c "MB"

crush -a -B " `which dmidecode` | awk '/Memory Device$/,'/"$"/ {print}' | grep -e '^Mem' -e Size: -e Speed: -e Part | sort -u | grep -v -e 'NO \ DIMM' -e 'No Module Installed' -e Unknown"

echo "${green}CPU ${NC}"

crush -a -B "grep '^[model name]' /proc/cpuinfo | sort -u"

echo "${green}CPU ${NC}"

crush -a -B "grep '^[model name]' /proc/cpuinfo | sort -u"

# probe for cpu info #

echo "${green}CPU ${NC}"

crush -a -B "grep '^[model name]' /proc/cpuinfo | sort -u"

echo "${green}CPU ${NC}"

crush -a -B "grep '^[model name]' /proc/cpuinfo | sort -u"

```

# probe for nic info#
echo -e "${green}NIC ${NC}"
clush -a -B "which ifconfig` | grep -e `(e|^p)` | awk '{print "$1"}' | xargs -1 `which ethtool` | grep -e ^Settings -e Speed"
echo ""
clush -a -B `which lspci` | grep -i ether"
echo ""
echo ""

# probe for disk info#
echo -e "${green}Storage ${NC}"
clush -a -B "echo 'Storage Controller: '; `which lspci` | grep -i -e raid -e storage -e lsi"
echo ""
clush -a -B "dmesg | grep -i raid | grep -i scsi"
echo ""
clush -a -B "lsblk -id | awk '{print "$1","$4"}'}|sort | nl"
echo ""
echo ""
echo -e "$ {green} ================ Software ================ ${NC}"  
.echo ""
.echo ""
echo -e "${green}Linux Release ${NC}" 
clush -a -B "cat /etc/*release | uniq"
echo ""
echo ""
echo -e "${green}Linux Version ${NC}" 
clush -a -B "uname -srvm | fmt"
echo ""
echo ""
echo -e "${green}Date ${NC}" 
clush -a -B date 
echo ""
echo ""
echo -e "${green}NTP Status ${NC}" 
clush -a -B "ntpstat 2>&1 | head -1"
echo ""
echo ""
echo -e "${green}SELINUX ${NC}" 
clush -a -B "echo -n 'SElinux status: '; grep ^SELINUX= /etc/selinux/config 2>&1"
echo ""
echo ""
clush -a -B "echo -n 'CPUspeed Service: '; `which service` cpuspeed \ status 2>&1"
clush -a -B "echo -n 'CPUspeed Service: '; `which chkconfig` --list \ cpuspeed 2>&1"
clush -a -B "echo -n 'Java Version${NC}" 
clush -a -B "java -version 2>&1; echo JAVA_HOME is ${JAVA_HOME:-Not Defined!)"
clush -a -B " ip addr show"
clush -a -B "ip addr show"
clush -a -B "ip addr show"
change permissions to executable.

chmod 755 cluster_verification.sh

3. Run the Cluster Verification tool from the admin node. This can be run before starting Hadoop to identify any discrepancies in Post OS Configuration between the servers or during troubleshooting of any cluster / Hadoop issues.

#./cluster_verification.sh

Installing Cloudera

Cloudera’s Distribution including Apache Hadoop (CDH) is an enterprise grade, hardened Hadoop distribution. CDH offers Apache Hadoop and several related projects into a single tested and certified product. It offers the latest innovations from the open source community with the testing and quality expected from enterprise quality software.

Prerequisites for CDH Installation

This section details the prerequisites for CDH installation such as setting up CDH Repo.

Cloudera Manager Repository

1. From a host connected to the Internet, download the Cloudera’s repositories as shown below and transfer it to the admin node.

   #mkdir -p /tmp/clouderarepo/

2. Download Cloudera Manager Repository.

   #cd /tmp/clouderarepo/
   #wget http://archive.cloudera.com/cm5/redhat/7/x86_64/cm/cloudera-manager.repo
   #reposync --config=./cloudera-manager.repo --repoid=cloudera-manager

   This downloads the Cloudera Manager RPMs needed for the Cloudera repository.

3. Run the following command to move the RPMs

4. Copy the repository directory to the admin node (rhel1)

   #scp -r /tmp/clouderarepo/ rhel1:/var/www/html/

5. On admin node (rhel1) run create repo command.

   #cd /var/www/html/clouderarepo/
   #createrepo --baseurl http://10.13.1.50/clouderarepo/cloudera-manager/
   /var/www/html/clouderarepo/cloudera-manager
6. Create the Cloudera Manager repo file with following contents:

```bash
# vi /var/www/html/clouderarepo/cloudera-manager/cloudera-manager.repo
[cloudera-manager]
name=Cloudera Manager
baseurl=http://10.13.1.50/clouderarepo/cloudera-manager/
gpgcheck=0
enabled=1
```

7. Copy the file cloudera-manager.repo into /etc/yum.repos.d/ on the admin node to enable it to find the packages that are locally hosted.

```bash
# cp /var/www/html/clouderarepo/cloudera-manager/cloudera-manager.repo /etc/yum.repos.d/
```

8. From the admin node copy the repo files to /etc/yum.repos.d/ of all the nodes of the cluster.

```bash
# clush --a -B -c /etc/yum.repos.d/cloudera-manager.repo
```

**Setting Up the Local Parcels for CDH 5.13.0**

From a host connected the internet, download the appropriate CDH 5.13.0 parcels that are meant for RHEL7.4 from the URL: [http://archive.cloudera.com/cdh5/parcels/](http://archive.cloudera.com/cdh5/parcels/) and place them in the directory "/var/www/html/CDH5.13.0.parcel" of the Admin node.

The following are the relevant files for RHEL7.4:

- CDH-5.13.0-1.cdh5.13.0.p0.29-el7.parcel
- CDH-5.13.0-1.cdh5.13.0.p0.29-el7.parcel.sha1 and manifest.json

**Downloading Parcels**

From a host connected to the Internet, download the Cloudera’s parcels as shown below and transfer it to the admin node.

```bash
# mkdir -p /tmp/clouderarepo/CDH5.13.0.parcel
```

1. Download parcels:
Now edit the /tmp/clouderarepo/CDH5.13.0parcels/manifest.json file and remove the scripts that are not meant for RHEL7.4. Below is that script which can be copy and pasted.

```json
{
    "lastUpdated": 15075981980000,
    "parcels": [
        {
            "parcelName": "CDH-5.13.0-1.cdh5.13.0.p0.29-el7.parcel",
            "components": [
                {
                    "pkg_version": "0.7.0+cdh5.13.0+0",
                    "pkg_release": "1.cdh5.13.0.p0.34",
                    "name": "bigtop-tomcat",
                    "version": "6.0.53-cdh5.13.0"
                },
                {
                    "pkg_version": "0.11.0+cdh5.13.0+101",
                    "pkg_release": "1.cdh5.13.0.p0.34",
                    "name": "crunch",
                    "version": "0.11.0-cdh5.13.0"
                },
                {
                    "pkg_version": "1.6.0+cdh5.13.0+169",
                    "pkg_release": "1.cdh5.13.0.p0.34",
                    "name": "flume-ng",
                    "version": "1.6.0-cdh5.13.0"
                },
                {
                    "pkg_version": "2.6.0+cdh5.13.0+2639",
                    "pkg_release": "1.cdh5.13.0.p0.34",
                    "name": "hadoop-0.20-mapreduce",
                    "version": "2.6.0-cdh5.13.0"
                },
                {
                    "pkg_version": "2.6.0+cdh5.13.0+2639",
                    "pkg_release": "1.cdh5.13.0.p0.34",
                    "name": "hadoop",
                    "version": "2.6.0-cdh5.13.0"
                },
                {
                    "pkg_version": "2.6.0+cdh5.13.0+2639",
                    "pkg_release": "1.cdh5.13.0.p0.34",
                    "name": "hadoop-hdfs",
                    "version": "2.6.0-cdh5.13.0"
                }
            ]
        }
    ]
}
```

Note: Please make sure the script starts and end with initial additional braces.
"pkg_version": "2.6.0+cdh5.13.0+2639",
"pkg_release": "1.cdh5.13.0.p0.34",
"name": "hadoop-httpfs",
"version": "2.6.0-cdh5.13.0"
},

{"pkg_version": "2.6.0+cdh5.13.0+2639",
"pkg_release": "1.cdh5.13.0.p0.34",
"name": "hadoop-kms",
"version": "2.6.0-cdh5.13.0"
},

{"pkg_version": "2.6.0+cdh5.13.0+2639",
"pkg_release": "1.cdh5.13.0.p0.34",
"name": "hadoop-mapreduce",
"version": "2.6.0-cdh5.13.0"
},

{"pkg_version": "2.6.0+cdh5.13.0+2639",
"pkg_release": "1.cdh5.13.0.p0.34",
"name": "hadoop-yarn",
"version": "2.6.0-cdh5.13.0"
},

{"pkg_version": "1.2.0+cdh5.13.0+411",
"pkg_release": "1.cdh5.13.0.p0.34",
"name": "hbase",
"version": "1.2.0-cdh5.13.0"
},

{"pkg_version": "1.5+cdh5.13.0+71",
"pkg_release": "1.cdh5.13.0.p0.34",
"name": "hbase-solr",
"version": "1.5-cdh5.13.0"
},

{"pkg_version": "1.1.0+cdh5.13.0+1269",
"pkg_release": "1.cdh5.13.0.p0.34",
"name": "hive",
"version": "1.1.0-cdh5.13.0"
},

{"pkg_version": "1.1.0+cdh5.13.0+1269",
"pkg_release": "1.cdh5.13.0.p0.34",
"name": "hive-hcatalog",
"version": "1.1.0-cdh5.13.0"
},

{"pkg_version": "3.9.0+cdh5.13.0+7079",
"pkg_release": "1.cdh5.13.0.p0.34",
"name": "hue",
"version": "3.9.0-cdh5.13.0"
},

{"pkg_version": "2.10.0+cdh5.13.0+0",
"pkg_release": "1.cdh5.13.0.p0.34",
"name": "impala",
"version": "2.10.0-cdh5.13.0"}

{
  "pkg_version": "1.0.0+cdh5.13.0+145",
  "pkg_release": "1.cdh5.13.0.p0.34",
  "name": "kite",
  "version": "1.0.0-cdh5.13.0"
},
{
  "pkg_version": "1.5.0+cdh5.13.0+0",
  "pkg_release": "1.cdh5.13.0.p0.34",
  "name": "kudu",
  "version": "1.5.0-cdh5.13.0"
},
{
  "pkg_version": "1.0.0+cdh5.13.0+0",
  "pkg_release": "1.cdh5.13.0.p0.34",
  "name": "llama",
  "version": "1.0.0-cdh5.13.0"
},
{
  "pkg_version": "0.9+cdh5.13.0+34",
  "pkg_release": "1.cdh5.13.0.p0.34",
  "name": "mahout",
  "version": "0.9-cdh5.13.0"
},
{
  "pkg_version": "4.1.0+cdh5.13.0+458",
  "pkg_release": "1.cdh5.13.0.p0.34",
  "name": "oozie",
  "version": "4.1.0-cdh5.13.0"
},
{
  "pkg_version": "1.5.0+cdh5.13.0+191",
  "pkg_release": "1.cdh5.13.0.p0.34",
  "name": "parquet",
  "version": "1.5.0-cdh5.13.0"
},
{
  "pkg_version": "0.12.0+cdh5.13.0+110",
  "pkg_release": "1.cdh5.13.0.p0.34",
  "name": "pig",
  "version": "0.12.0-cdh5.13.0"
},
{
  "pkg_version": "1.5.1+cdh5.13.0+410",
  "pkg_release": "1.cdh5.13.0.p0.34",
  "name": "sentry",
  "version": "1.5.1-cdh5.13.0"
},
{
  "pkg_version": "4.10.3+cdh5.13.0+519",
  "pkg_release": "1.cdh5.13.0.p0.34",
  "name": "solr",
  "version": "4.10.3-cdh5.13.0"
},
{
  "pkg_version": "1.6.0+cdh5.13.0+530",
  "pkg_release": "1.cdh5.13.0.p0.34",
  "name": "stunnel",
  "version": "1.6.0-cdh5.13.0"
}
"name": "spark",
"version": "1.6.0-cdh5.13.0"
},
{
"pkg_version": "1.99.5+cdh5.13.0+46",
"pkg_release": "1.cdh5.13.0.p0.34",
"name": "sqoop2",
"version": "1.99.5-cdh5.13.0"
},
{
"pkg_version": "1.4.6+cdh5.13.0+116",
"pkg_release": "1.cdh5.13.0.p0.34",
"name": "sqoop",
"version": "1.4.6-cdh5.13.0"
},
{
"pkg_version": "0.9.0+cdh5.13.0+23",
"pkg_release": "1.cdh5.13.0.p0.34",
"name": "whirr",
"version": "0.9.0-cdh5.13.0"
},
{
"pkg_version": "3.4.5+cdh5.13.0+118",
"pkg_release": "1.cdh5.13.0.p0.34",
"name": "zookeeper",
"version": "3.4.5-cdh5.13.0"
}
"conflicts": "SPARK2 (<< 2.0.0.cloudera2)",
"replaces": "IMPALA, SOLR, SPARK, KUDU",
"hash": "bef6f3f074e0a88cd79d6d37abc6698471e3d279"
}]

3. Copy /tmp/clouderarepo/CDH5.13.0parcels to the admin node (rhel1)

   #scp -r /tmp/clouderarepo/CDH5.13.0parcels/ rhel1:/var/www/html/

4. Verify that these files are accessible by visiting the URL http://10.13.1.50/CDH5.13.0parcels/ in admin node.

Setting Up the MariaDB Database for Cloudera Manager

- Install the MariaDB Server
- Configure and Start the MariaDB Server
- Install the MariaDB/MySQL JDBC Driver
- Create Databases for Activity Monitor, Reports Manager, Hive Metastore Server, Sentry Server, Cloudera Navigator Audit Server, and Cloudera Navigator Metadata Server

Installing the MariaDB Server

To use a MariaDB database, complete the following steps:
1. In the admin node where Cloudera Manager will be installed, use the following command to install the mariadb/mysql server.

   #yum -y install mariadb-server

2. To configure and start the MySQL Server, stop the MariaDB server if it is running.

   #service mariadb stop

3. Move the old InnoDB log if exists.

4. Move files /var/lib/mysql/ib_logfile0 and /var/lib/mysql/ib_logfile1 out of /var/lib/mysql/ to a backup location.

   #mv /var/lib/mysql/ib_logfile0 /root/ib_logfile0.bkp
   #mv /var/lib/mysql/ib_logfile1 /root/ib_logfile1.bkp

5. Determine the location of the option file, my.cnf and edit/add following lines:

   #vi /etc/my.cnf

   [mysqld]
   transaction-isolation = READ-COMMITTED
   # InnoDB settings
   innodb_flush_method = O_DIRECT
   max_connections = 550

   [root@zhell -]# vi /etc/my.cnf
   [root@zhell -]# cat /etc/my.cnf

   [mysqld]
   datadir=/var/lib/mysql
   socket=/var/lib/mysql/mysql.sock
   user=mysql
   transaction-isolation = READ-COMMITTED
   # Disabling symbolic-links is recommended to prevent assorted security risks
   symbolic-links=0

   [mysqld_safe]
   log-error=/var/log/mysqld.log
   pid-file=/var/run/mysqld/mysqld.pid

   # InnoDB settings
   innodb_flush_method = O_DIRECT
   max_connections = 550

   Note: The max_connections need to be increased based on number of nodes and applications. Please follow the recommendations as mentioned in the Cloudera document

   http://www.cloudera.com/documentation/enterprise/latest/topics/install_cm_mariadb.html - install_cm_mariadb_config

6. Make sure MySQL Server starts at boot:

   #systemctl enable mariadb.service
7. Start the MySQL Server:

```
# service mariadb start
```

8. Set the MySQL root password on admin node (rhel1)

```
# cd /usr/bin/
# mysql_secure_installation
```

Enter current password for root (enter for none):
Ok, successfully used password, moving on...

Setting the root password ensures that nobody can log into the MariaDB root user without the proper authorisation.

Set root password? [Y/n] y
New password:
Re-enter new password:
Password updated successfully!
Reloading privilege tables..
.. Success!

By default, a MariaDB installation has an anonymous user, allowing anyone to log into MariaDB without having to have a user account created for them. This is intended only for testing, and to make the installation go a bit smoother. You should remove them before moving into a production environment.

Remove anonymous users? [Y/n] y
.. Success!

Normally, root should only be allowed to connect from 'localhost'. This ensures that someone cannot guess at the root password from the network.

Disallow root login remotely? [Y/n] n
.. skipping.

By default, MariaDB comes with a database named 'test' that anyone can access. This is also intended only for testing, and should be removed before moving into a production environment.

```
- Removing test database...
... Success!
- Removing privileges on test database...
... Success!
```

Reloading the privilege tables will ensure that all changes made so far will take effect immediately.

```
Reload privilege tables now? [Y/n] y
... Success!
```

Cleaning up...

All done! If you've completed all of the above steps, your MariaDB installation should now be secure.
Installing the MySQL JDBC Driver

Install the JDBC driver on the Cloudera Manager Server host, as well as hosts which run the Activity Monitor, Reports Manager, Hive Metastore Server, Sentry Server, Cloudera Navigator Audit Server, and Cloudera Navigator Metadata Server roles.

1. From a host connected to the Internet, download the MySQL JDBC Driver and transfer it to the admin node. Download the MySQL JDBC driver from the URL [http://www.mysql.com/downloads/connector/j/5.1.html](http://www.mysql.com/downloads/connector/j/5.1.html)

2. Copy mysql-connector-java-5.1.37.tar.gz to admin node(rhel1)

   ```
   #scp mysql-connector-java-5.1.37.tar.gz rhel1:/root/
   ```

3. Log in to the admin node and extract the file:

   ```
   #tar xzvf mysql-connector-java-5.1.37.tar.gz
   ```

4. Create the /usr/share/java/ directory on the admin node (rhel1)

   ```
   #mkdir -p /usr/share/java/
   ```

5. Go to the mysql-connector-java-5.1.37 directory on the admin node (rhel1) and copy mysql-connector-java-5.1.37-bin.jar to /usr/share/java/

   ```
   #cd mysql-connector-java-5.1.37
   #cp mysql-connector-java-5.1.37-bin.jar /usr/share/java/mysql-connector-java.jar
   ```

Creating Databases for Activity Monitor, Reports Manager, Hive Metastore Server, Navigator Audit Server, and Navigator Metadata Server

1. In the admin node Log into MySQL as the root user:

   ```
   #mysql -u root -p
   ```

2. Enter password that was previously supplied.

3. Create databases for the Activity Monitor, Reports Manager and Hive Metastore Server using the command below:

   ```
   mysql> create database amon DEFAULT CHARACTER SET utf8;
   mysql> create database rman DEFAULT CHARACTER SET utf8;
   mysql> create database metastore DEFAULT CHARACTER SET utf8;
   mysql> create database nav DEFAULT CHARACTER SET utf8;
   mysql> create database navms DEFAULT CHARACTER SET utf8;
   mysql> create database sentry DEFAULT CHARACTER SET utf8;
   mysql> create database oozie DEFAULT CHARACTER SET utf8;
   ```

   ```
   mysql> grant all on amon.* TO 'root'@'%' IDENTIFIED BY 'password';
   mysql> grant all on rman.* TO 'rman'@'%' IDENTIFIED BY 'password';
   mysql> grant all on metastore.* TO 'hive'@'%' IDENTIFIED BY 'password';
   ```

   ```
   mysql> grant all on amon.* TO 'root'@'%' IDENTIFIED BY 'password';
   mysql> grant all on nav.* TO 'root'@'%' IDENTIFIED BY 'password';
   mysql> grant all on navms.* TO 'root'@'%' IDENTIFIED BY 'password';
   mysql> grant all on sentry.* TO 'root'@'%' IDENTIFIED BY 'password';
   mysql> grant all privileges on oozie.* to root@'%' IDENTIFIED BY 'password';
   mysql> grant all on rman.* TO 'rman'@'%' IDENTIFIED BY 'password';
   mysql> grant all on metastore.* TO 'hive'@'%' IDENTIFIED BY 'password';
   ```
Cloudera Manager Installation

The following section describes installation of Cloudera Manager first and then using Cloudera Manager to install CDH 5.13.0.

Setting Up the Cloudera Manager Server Database

The Cloudera Manager Server Database stores information about service and host configurations.
Installing Cloudera Manager

Cloudera Manager, an end to end management application, is used to install and configure CDH. During CDH Installation, Cloudera Manager's Wizard will help to install Hadoop services on all nodes using the following procedure:

- Discovery of the cluster nodes
- Configure the Cloudera parcel or package repositories
- Install Hadoop, Cloudera Manager Agent (CMA) and Impala on all the cluster nodes.
- Install the Oracle JDK if it is not already installed across all the cluster nodes.
- Assign various services to nodes.
- Start the Hadoop services.

To install Cloudera Manager, complete the following steps:

1. Update the repo files to point to local repository.
   
   ```
   #rm -f /var/www/html/clouderarepo/*.repo
   #cp /etc/yum.repos.d/c*.repo /var/www/html/clouderarepo/
   ```

2. Install the Oracle Java Development Kit on the Cloudera Manager Server host.
   
   ```
   #yum install oracle-j2sdk1.7
   ```

3. Install the Cloudera Manager Server packages either on the host where the database is installed, or on a host that has access to the database.
   
   ```
   #yum install cloudera-manager-daemons cloudera-manager-server
   ```

Preparing a Cloudera Manager Server External Database

1. Run the `scm_prepare_database.sh` script on the host where the Cloudera Manager Server package is installed (rhel1) admin node.

   ```
   #cd /usr/share/cmf/schema
   #./scm_prepare_database.sh mysql amon root <password>
   #./scm_prepare_database.sh mysql rman root <password>
   #./scm_prepare_database.sh mysql metastore root <password>
   #./scm_prepare_database.sh mysql nav root <password>
   #./scm_prepare_database.sh mysql navms root <password>
   #./scm_prepare_database.sh mysql sentry root <password>
   #./scm_prepare_database.sh mysql oozie root <password>
   ```

2. Verify the database connectivity using the following command.

   ```
   [root@rhel1 ~]# mysql -u root -p
   mysql> connect amon
   mysql> connect rman
   mysql> connect metastore
   mysql> connect nav
   mysql> connect navms
   mysql> connect sentry
   ```
Starting The Cloudera Manager Server

1. Start the Cloudera Manager Server:

   #service cloudera-scm-server start

2. Access the Cloudera Manager using the URL, [http://10.13.1.50:7180](http://10.13.1.50:7180) to verify that the server is up.

3. Once the installation of Cloudera Manager is complete, install CDH5 using the Cloudera Manager Web interface.

Installing Cloudera Enterprise Data Hub (CDH5)

   To install the Cloudera Enterprise Data Hub, complete the following steps:

1. Login to the Cloudera Manager. Enter "admin" for both the Username and Password fields.
Welcome to Cloudera Manager

End User License Terms and Conditions

Cloudera Standard License
Version 2019-05-26

END USER LICENSE TERMS AND CONDITIONS

THESE TERMS AND CONDITIONS (THESE "TERMS") APPLY TO YOUR USE OF THE PRODUCTS (AS DEFINED BELOW) PROVIDED BY CLOUDEA, INC. ("CLOUDEA").

PLEASE READ THESE TERMS CAREFULLY.

IF YOU ("YOU" OR "CUSTOMER") PLAN TO USE ANY OF THE PRODUCTS ON BEHALF OF A COMPANY OR OTHER ENTITY, YOU REPRESENT THAT YOU ARE THE EMPLOYEE OR AGENT OF SUCH COMPANY (OR OTHER ENTITY) AND YOU HAVE THE AUTHORITY TO ACCEPT ALL OF THE TERMS AND CONDITIONS SET FORTH IN AN ACCEPTED REQUEST (AS DEFINED BELOW) AND THESE TERMS (COLLECTIVELY, THE "AGREEMENT") ON BEHALF OF SUCH COMPANY (OR OTHER ENTITY).

BY USING ANY OF THE PRODUCTS, YOU ACKNOWLEDGE AND AGREE THAT:
(A) YOU HAVE READ ALL OF THE TERMS AND CONDITIONS OF THIS AGREEMENT;
(B) YOU UNDERSTAND ALL OF THE TERMS AND CONDITIONS OF THIS AGREEMENT;
(C) YOU AGREE TO BE LEGALLY BOUND BY ALL OF THE TERMS AND CONDITIONS SET FORTH IN THIS AGREEMENT

IF YOU DO NOT AGREE WITH ANY OF THE TERMS OR CONDITIONS OF THESE TERMS, YOU MAY NOT USE ANY PORTION OF THE PRODUCTS.

Yes, I accept the End User License Terms and Conditions.

If your download and use of Cloudera Manager are on behalf of a company that has an existing agreement with Cloudera for the use of the software, your action does not modify that existing agreement.
2. If you do not have a Cloudera license, select Cloudera Enterprise Data Hub Trial Edition. If you do have a Cloudera license, click “Upload License” and select your license.

3. Based on requirement, choose appropriate Cloudera Editions for the Installation.
4. Click Continue on the confirmation page.
Edit the Cloudera Enterprise Parcel Settings to Use the CDH 5.13.0 Parcels

1. Open another tab in the same browser window and visit the URL: http://10.13.1.50:7180/cmf/parcel/status for modifying the parcel settings.

2. Click Configuration on this page.

3. Click to remove the entire remote repository URLs, and add the URL to the location where we kept the CDH 5.13.0 parcels i.e. http://10.13.1.50/CDH5.13.oparcels/.
4. Click Save Changes to finish the configuration.


6. Click Continue on the confirmation page.
7. Specify the hosts that are part of the cluster using their IP addresses or hostname. The figure below shows use of a pattern to specify the IP addresses range.

\[ 10.13.1.[50-73] \text{ or } \text{rhel}[1-24] \]

8. After the IP addresses or hostnames are entered, click Search.

**Figure 25** Searching for Cluster Nodes

9. Cloudera Manager will "discover" the nodes in the cluster. Verify that all desired nodes have been found and selected for installation.
10. Click Continue.

11. For the method of installation, select the Use Parcels (Recommended) radio button.

12. For the CDH version, select the CDH5.13.0-1.cdh5.13.0.p0.29 radio button.

13. For the specific release of Cloudera Manager, select the Custom Repository radio button.

Cluster Installation

Select Repository

Cloudera recommends the use of parcels for installation over packages, because parcels enable Cloudera Manager to easily manage the software on your cluster, automating the deployment and upgrade of service binaries. Electing not to use parcels will require you to manually upgrade packages on all hosts in your cluster when software updates are available, and will prevent you from using Cloudera Manager’s rolling upgrade capabilities.

Choose Method
- Use Packages
- Use Parcels (Recommended)

Select the version of CDH
- CDH-5.13.0-1.cdh5.13.0.p0.29

Versions of CDH that are too new for this version of Cloudera Manager (5.13.0) will not be shown.

Select the specific release of the Cloudera Manager Agent you want to install on your hosts.
- Matched release for this Cloudera Manager Server
- Custom Repository


Example for SLES, Redhat or other RPM based distributions:
https://archive.cloudera.com/cm/redhat/6/x86_64/cm/5/

Example for Ubuntu or other Debian based distributions:

deb https://archive.cloudera.com/cm/ubuntu/lucid/5.6/cm/deb Contrib

Enter a custom URL for the location of the GPG signing key (applies to all custom repositories and without internet access).

Custom GPG Key URL: 

Continue

Cluster Installation

JDK Installation Options

Oracle Binary Code License Agreement for the Java SE Platform Products and JavaFX

ORACLE AMERICA, INC. ("ORACLE"), FOR AND ON BEHALF OF ITSELF AND ITS SUBSIDIARIES AND AFFILIATES UNDER COMMON CONTROL, IS WILLING TO LICENSE THE SOFTWARE TO YOU ONLY UPON THE CONDITION THAT YOU ACCEPT ALL OF THE TERMS CONTAINED IN THIS BINARY CODE LICENSE AGREEMENT AND SUPPLEMENTAL LICENSE TERMS (COLLECTIVELY "AGREEMENT"). PLEASE READ THE AGREEMENT CAREFULLY. BY SELECTING THE "ACCEPT LICENSE AGREEMENT" (OR THE EQUIVALENT) BUTTON AND/OR BY USING THE SOFTWARE YOU ACKNOWLEDGE THAT YOU HAVE READ THE TERMS AND AGREE TO THEM. IF YOU ARE AGREEING TO THESE TERMS ON BEHALF OF A COMPANY OR OTHER LEGAL ENTITY, YOU REPRESENT THAT YOU HAVE THE LEGAL AUTHORITY TO BIND THE LEGAL ENTITY TO THESE TERMS. IF YOU DO NOT HAVE SUCH AUTHORITY, OR IF YOU DO NOT WISH TO BE BOUND BY THE TERMS, THEN SELECT THE "DECLINE LICENSE AGREEMENT" (OR THE EQUIVALENT) BUTTON AND YOU MUST NOT USE THE SOFTWARE ON THIS SITE OR ANY OTHER MEDIA ON WHICH THE SOFTWARE IS CONTAINED.

1. DEFINITIONS. "Software" means the software identified above in binary form that you selected for download, install or use (in the version You selected for download, install or use from Oracle or its authorized licensees, any other machine readable materials (including, but not limited to, libraries, source files, header files, and data files), any updates or error corrections provided by Oracle, and any user manuals, programming guides and other documentation provided to you by Oracle under this Agreement. "General Purpose Desktop Computers and Servers" means computers, including desktop and laptop computers, or servers, used for general computing functions and not user control (such as but not specifically limited to email, general purpose Internet browsing, and office suite productivity tools). The use of Software in systems and solutions that provide dedicated functionality (other than as mentioned above) or designed for use in embedded or function-specific software applications, for example but not limited to: Software embedded in or bundled with industrial control systems, wireless mobile telephones, wireless handheld devices, kiosks, TV/STB, Blu-ray Disc devices, telematics and network control switching equipment, printers and storage management systems, and other related systems are excluded from this definition and not licensed under this Agreement. "Programs" means (a) Java technology applets and applications intended to run on the Java Platform, Standard Edition platform on Java-enabled General Purpose Desktop Computers and Servers; and (b) JavaFX technology applications intended to run on the JavaFX Runtime on JavaFX-enabled General Purpose Desktop Computers and Servers. "Commercial Features" means those features identified in

- Install Oracle Java SE Development Kit (JDK)
- Check this box to accept the Oracle Binary Code License Agreement and install the JDK. Leave it unchecked to use a currently installed JDK.

- Install Java Unlimited Strength Encryption Policy Files
- Check this box if local laws permit you to deploy unlimited strength encryption and you are running a secure cluster.

118
15. Provide SSH login credentials for the cluster and click Continue.

**Figure 26  Login Credentials to Start CDH Installation**

16. Installation using parcels begins.
17. Once the installation is completed successfully, click Continue to select the required services.

18. Wait for Cloudera Manager to inspect the hosts on which it has just performed the installation.

19. Review and verify the summary. Click Continue.
20. Select services that need to be started on the cluster.
21. This is a critical step in the installation. Inspect and customize the role assignments of all the nodes based on your requirements and click Continue.

22. Reconfigure the service assignment to match Table 7 below.

**Table 7  Service Assignments**

<table>
<thead>
<tr>
<th>Service Name</th>
<th>Host</th>
</tr>
</thead>
<tbody>
<tr>
<td>NameNode</td>
<td>rhel1, rhel2 (HA)</td>
</tr>
<tr>
<td>HistoryServer</td>
<td>rhel1</td>
</tr>
<tr>
<td>JournalNodes</td>
<td>rhel1, rhel2, rhel3</td>
</tr>
<tr>
<td>ResourceManager</td>
<td>rhel2, rhel3 (HA)</td>
</tr>
<tr>
<td>Hue Server</td>
<td>rhel2</td>
</tr>
<tr>
<td>HiveMetastore Server</td>
<td>rhel1</td>
</tr>
<tr>
<td>HiveServer2</td>
<td>rhel2</td>
</tr>
<tr>
<td>HBase Master</td>
<td>rhel2</td>
</tr>
<tr>
<td>Oozie Server</td>
<td>rhel1</td>
</tr>
<tr>
<td>Service Name</td>
<td>Host</td>
</tr>
<tr>
<td>--------------------------------------</td>
<td>----------------------------</td>
</tr>
<tr>
<td>ZooKeeper</td>
<td>rhel3, rhel2, rhel1</td>
</tr>
<tr>
<td>DataNode</td>
<td>rhel4 to rhel24</td>
</tr>
<tr>
<td>NodeManager</td>
<td>rhel4 to rhel24</td>
</tr>
<tr>
<td>RegionServer</td>
<td>rhel4 to rhel24</td>
</tr>
<tr>
<td>Sqoop Server</td>
<td>rhel1</td>
</tr>
<tr>
<td>Impala Catalog Server Daemon</td>
<td>rhel1</td>
</tr>
<tr>
<td>Impala State Store</td>
<td>rhel2</td>
</tr>
<tr>
<td>Impala Daemon</td>
<td>rhel4 to rhel24</td>
</tr>
<tr>
<td>Solr Server</td>
<td>rhel4 (can be installed on all hosts if needed, if there is a search use case)</td>
</tr>
<tr>
<td>Spark History Server</td>
<td>rhel1</td>
</tr>
<tr>
<td>Spark Executors</td>
<td>rhel4 to rhel24</td>
</tr>
</tbody>
</table>

**Cluster Setup**

Customize Role Assignments

You can customize the role assignments for your new cluster here, but if assignments are made incorrectly, such as assigning too many roles to a single host, this can impact the performance of your services. Cloudera does not recommend altering assignments unless you have specific requirements, such as having pre-selected a specific host for a specific role.

You can also view the role assignments by host: View By Host

### HBase

<table>
<thead>
<tr>
<th>Role</th>
<th>Quantity</th>
<th>Host</th>
</tr>
</thead>
<tbody>
<tr>
<td>Master</td>
<td>1 New</td>
<td>rhel2</td>
</tr>
<tr>
<td>HBase REST Server</td>
<td>Select hosts</td>
<td></td>
</tr>
<tr>
<td>HBase Thrift Server</td>
<td>Select hosts</td>
<td></td>
</tr>
<tr>
<td>RegionServer</td>
<td>3 New</td>
<td>Same As DataNode</td>
</tr>
</tbody>
</table>

### HDFS

<table>
<thead>
<tr>
<th>Role</th>
<th>Quantity</th>
<th>Host</th>
</tr>
</thead>
<tbody>
<tr>
<td>NameNode</td>
<td>1 New</td>
<td>rhel1</td>
</tr>
<tr>
<td>SecondaryNameNode</td>
<td>1 New</td>
<td>rhel2</td>
</tr>
<tr>
<td>Balancer</td>
<td>1 New</td>
<td>rhel1</td>
</tr>
<tr>
<td>HttpFS</td>
<td>Select hosts</td>
<td></td>
</tr>
</tbody>
</table>

### Hive

<table>
<thead>
<tr>
<th>Role</th>
<th>Quantity</th>
<th>Host</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gateway</td>
<td>4 New</td>
<td>rhel [1-24]</td>
</tr>
<tr>
<td>Hive Metastore Server</td>
<td>1 New</td>
<td>rhel1</td>
</tr>
<tr>
<td>WebHCat Server</td>
<td>1 New</td>
<td>rhel1</td>
</tr>
<tr>
<td>HiveServer2</td>
<td>1 New</td>
<td>rhel2</td>
</tr>
</tbody>
</table>
Setting up the Database

The role assignment recommendation above is for clusters of up to 64 servers. For clusters larger than 64 nodes, use the HA recommendation defined in Table 7 above.

1. In the Database Host Name sections use port 3306 for TCP/IP because connection to the remote server always uses TCP/IP.
2. Enter the Database Name, username and password that were used during the database creation stage earlier in this document.

3. Click Test Connection to verify the connection and click Continue.

**Figure 29  Database Setup**

4. Review and customize the configuration changes based on your requirements.
5. Click Continue to start running the cluster services.
### Cluster Setup

**First Run Command**

<table>
<thead>
<tr>
<th>Step</th>
<th>Description</th>
<th>Status</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Ensuring that the expected software releases are installed on hosts.</td>
<td>Completed</td>
<td>37 ms</td>
</tr>
<tr>
<td>2</td>
<td>Deploying Client Configuration</td>
<td>Completed</td>
<td>16.23 s</td>
</tr>
<tr>
<td>3</td>
<td>Start Cloudera Management Service, ZooKeeper</td>
<td>Completed</td>
<td>24.68 s</td>
</tr>
<tr>
<td>4</td>
<td>Start HDFS</td>
<td>Completed</td>
<td>40.21 s</td>
</tr>
<tr>
<td>5</td>
<td>Start HBase, Solr</td>
<td>Completed</td>
<td>32.35 s</td>
</tr>
<tr>
<td>6</td>
<td>Start Key-Value Store index, YARN (MR2 included)</td>
<td>Completed</td>
<td>26.86 s</td>
</tr>
<tr>
<td>7</td>
<td>Start Spark</td>
<td>Completed</td>
<td>31.27 s</td>
</tr>
<tr>
<td>8</td>
<td>Start Hive</td>
<td>Completed</td>
<td>30.35 s</td>
</tr>
<tr>
<td>9</td>
<td>Start Impala, Oozie</td>
<td>Completed</td>
<td>30.35 s</td>
</tr>
<tr>
<td>10</td>
<td>Start Hue</td>
<td>Completed</td>
<td>30.35 s</td>
</tr>
</tbody>
</table>

Finished First Run of the following services successfully: ZooKeeper, HDFS, HBase, Solr, YARN (MR2 Included), Key-Value Store Indexer, Spark, Hive, Impala, Oozie, Hue, Cloudera Management Service.
Starting the Cluster Services

1. Hadoop services are installed, configured and now running on all the nodes of the cluster. Click Finish to complete the installation.

Figure 31  Installation Completion

Cloudera Manager now displays the status of all Hadoop services running on the cluster.
Scaling the Cluster

The role assignment recommendation above is for clusters with at least 64 servers and in High Availability (HA). For smaller clusters running without HA, the recommendation is to dedicate one server for NameNode and a second server for secondary name node and YARN Resource Manager. For larger clusters larger than 28 nodes, the recommendation is to dedicate one server each for NameNode, YARN Resource Manager, and one more for running both NameNode (HA) and Resource Manager (HA) as in the table (no Secondary NameNode when in HA).

For production clusters, it is recommended to set up NameNode and Resource Manager in HA mode.

This implies that there will be at least 3 master nodes, running the NameNode, YARN Resource Manager, the failover counter-part being designated to run on another node and a third node that would have similar capacity as the other two nodes.

All the three nodes will also need to run Zookeeper and quorum journal node services. It is also recommended to have a minimum of 5 DataNodes in a cluster. Please refer to the next section for details on how to enable HA.

Enabling High Availability

Note: Setting up HA is done after the Cloudera Installation is completed.

HDFS High Availability

The HDFS HA feature provides the option of running two NameNodes in the same cluster, in an Active/Passive configuration. These are referred to as the Active NameNode and the Standby NameNode. Unlike the Secondary NameNode, the Standby NameNode is a hot standby, allowing a fast failover to a new NameNode in the case that a machine crashes, or a graceful administrator-initiated failover for the purpose of planned maintenance. There cannot be more than two NameNodes.
For more information go to:

http://www.cloudera.com/documentation/enterprise/latest/topics/cdh_hag_hdfs_ha_intro.html - topic_2_1

Setting Up HDFS HA

The Enable High Availability workflow leads through adding a second (standby) NameNode and configuring JournalNodes. During the workflow, Cloudera Manager creates a federated namespace.

1. Log in to the admin node (rhel1) and create the Edit directory for the JournalNode

   #clush -w rhel[1-3] mkdir -p /data/disk1/namenode-edits
   #clush -w rhel[1-3] chmod 777 /data/disk1/namenode-edits

   [root@rhel1 ~]# clush -w rhel[1-3] mkdir -p /data/disk1/namenode-edits
   [root@rhel1 ~]# clush -w rhel[1-3] chmod 777 /data/disk1/namenode-edits

2. Log in to the Cloudera manager and go to the HDFS service.

3. In the top right corner Select Actions> Enable High Availability. A screen showing the hosts that are eligible to run a standby NameNode and the JournalNodes displays.

4. Specify a name for the nameservice or accept the default name nameservice1 and click Continue.

5. In the NameNode Hosts field, click Select a host. The host selection dialog displays.

6. Check the checkbox next to the hosts (rhel2) where the standby NameNode is to be set up and click OK.
Note: The standby NameNode cannot be on the same host as the active NameNode, and the host that is chosen should have the same hardware configuration (RAM, disk space, number of cores, and so on) as the active NameNode.

7. In the **JournalNode Hosts** field, click **Select hosts**. The host selection dialog displays.

8. Check the checkboxes next to an odd number of hosts (a minimum of three) to act as JournalNodes and click **OK**. Here we are using the same nodes as Zookeeper nodes.

**Enable High Availability for HDFS**

**Assign Roles**

- **NameNode Hosts**:
  - rhe1 (Current)
  - rhe2

- **JournalNode Hosts**:
  - rhe{1-3}

We recommend that JournalNodes be hosted on machines of similar hardware specifications as the NameNodes. The hosts of NameNodes and the ResourceManager are generally good options. You must have a minimum of three and an odd number of JournalNodes.

9. Click **Continue**.

10. In the **JournalNode Edits Directory** property, enter a directory location created earlier in step 1 for the JournalNode ed-
  its directory into the fields for each JournalNode host.
Enable High Availability for HDFS

Review Changes

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Group</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NameNode Data Directories</td>
<td>rhel1</td>
<td>/data/disk1/dfs/nn</td>
<td>Determines where on the local file system the NameNode should store the name table (fsimage). For redundancy, enter a comma-delimited list of directories to replicate the name table in all of the directories. Typical values are /data/Ndfs/nn where N=1..3.</td>
</tr>
<tr>
<td></td>
<td>rhel2</td>
<td>/data/disk1/dfs/nn</td>
<td>Inherited from: NameNode Default Group</td>
</tr>
<tr>
<td></td>
<td>rhel1</td>
<td>/data/disk1/namenode-edits</td>
<td>Directory on the local file system where NameNode edits are written.</td>
</tr>
<tr>
<td></td>
<td>rhel2</td>
<td>/data/disk1/namenode-edits</td>
<td>Inherited from: NameNode Default Group</td>
</tr>
<tr>
<td></td>
<td>rhel3</td>
<td>/data/disk1/namenode-edits</td>
<td>Reset to empty default value</td>
</tr>
</tbody>
</table>

Note: The directories specified should be empty, and must have the appropriate permissions.

Extra Options: Decide whether Cloudera Manager should clear existing data in ZooKeeper, Standby NameNode, and JournalNodes. If the directories are not empty (for example, re-enabling a previous HA configuration), Cloudera Manager will not automatically delete the contents—select to delete the contents by keeping the default checkbox selection. The recommended default is to clear the directories.

Note: If you choose to not configure any of the extra options, the data should be in sync across the edits directories of the JournalNodes and should have the same version data as the NameNodes.

11. Click Continue.

Cloudera Manager executes a set of commands that will stop the dependent services, delete, create, and configure roles and directories as appropriate, create a nameservice and failover controller, and restart the dependent services and deploy the new client configuration.
Enable High Availability for HDFS

✓ Enable High Availability Command

Status: Finished  Context: HDFS  Start Time: May 1, 9:41:17 PM  Duration: 10.9m

Successfully enabled High Availability and Automatic Failover

Details (Completed 20 of 20 steps):

Step                  Context                  Start Time              Duration   Actions

✓ Check that name directories for the new Standby NameNode either do not exist or are writable and empty. Can optionally clear directories. Process host-validate-writable-empty-dirs (/d=01) on host rhel7 (/d=4) exited with 0 and expected 0

May 1, 9:41:17 PM    15.14s

✓ Check that /nameservice either do not exist or are writable and empty. Can optionally clear directories.

May 1, 9:41:32 PM    15.25s

Note: Formatting the name directory is expected to fail, if the directories are not empty.

Enable High Availability for HDFS

Congratulations!

Successfully enabled High Availability.

The following manual steps must be performed after completing this wizard:
• Configure the HDFS Web Interface Role of Hue service(s) Hue to be an HTTPS role instead of a NameNode. Documentation
• For each of the Hive service(s) Hive, stop the Hive service, back up the Hive Metastore Database to a persistent store, run the service command "Update Hive Metastore NameNodes", then restart the Hive services.
12. In the next screen additional steps are suggested by the Cloudera Manager to update the Hue and Hive metastore. Click Finish for the screen shown above.

Note: The following subsections cover configuring Hue and Hive for HA as needed.

13. In the Cloudera Manager, Click Home > HDFS > Instances to see Namenode in High Availability.

Configuring Hive Metastore to Use HDFS HA

To configure the Hive Megastore to use HDFS high availability, complete the following steps:

1. Go the Hive service.
2. Select Actions > Stop.
3. Click Stop to confirm the command.
4. Back up the Hive metastore database (if any existing data is present).
5. Select Actions> Update Hive Metastore NameNodes and confirm the command.
6. Select Actions> Start.
7. Restart the Hue and Impala services if stopped prior to updating the Metastore.

Configuring Hue to Work with HDFS HA

To configure Hue to work with HDFS HA, complete the following steps:

1. Go to the HDFS service.
2. Click the Instances tab.
3. Click Add Role Instances.
4. Select the text box below the HttpFS field. The Select Hosts dialog displays.
5. Select the host on which to run the role and click OK.
6. Click Continue.
7. Check the checkbox next to the HttpFS role and select Actions for Selected> Start.
8. After the command has completed, go to the Hue service.

9. Click the Configuration tab.

10. Locate the HDFS Web Interface Role property or search for it by typing its name in the Search box.

11. Select the HttpFS role that was just created instead of the NameNode role, and save your changes.

12. Restart the Hue service.
YARN High Availability

The YARN Resource Manager (RM) is responsible for tracking the resources in a cluster and scheduling applications (for example, MapReduce jobs). Before CDH 5, the RM was a single point of failure in a YARN cluster. The RM high availability (HA) feature adds redundancy in the form of an Active/Standby RM pair to remove this single point of failure. Furthermore, upon failover from the Standby RM to the Active, the applications can resume from their last check-pointed state; for example, completed map tasks in a MapReduce job are not re-run on a subsequent attempt. This allows events such the following to be handled without any significant performance effect on running applications.

- Unplanned events such as machine crashes.
- Planned maintenance events such as software or hardware upgrades on the machine running the ResourceManager

For more information, go to:
http://www.cloudera.com/documentation/enterprise/latest/topics/cdh_hag_rm_ha_config.html

Setting up YARN HA

To set up YARN HA, complete the following steps:

1. Log in to the Cloudera manager and go to the YARN service.
2. Select Actions> Enable High Availability.
   A screen showing the hosts that are eligible to run a standby ResourceManager displays.
   The host where the current ResourceManager is running is not available as a choice.
3. Select the host (rhel3) where the standby ResourceManager is to be installed, and click Continue.
Cloudera Manager proceeds to execute a set of commands that stop the YARN service, add a standby ResourceManager, initialize the ResourceManager high availability state in ZooKeeper, restart YARN, and redeploy the relevant client configurations.
4. Click Finish once the installation is completed successfully.

Configuring Yarn (MR2 Included) and HDFS Services

The parameters in Table 8 are used for Cisco UCS Integrated Infrastructure for Big Data and Analytics Performance Optimized cluster configuration described in this document. These parameters are to be changed based on the cluster configuration, number of nodes and specific workload.

<table>
<thead>
<tr>
<th>Service</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>mapreduce.map.memory.mb</td>
<td>3GiB</td>
</tr>
<tr>
<td>mapreduce.reduce.memory.mb</td>
<td>3GiB</td>
</tr>
<tr>
<td>mapreduce.map.java.opts.max.heap</td>
<td>2560 MiB</td>
</tr>
<tr>
<td>yarn.nodemanager.resource.memoryMb</td>
<td>180 GiB</td>
</tr>
<tr>
<td>yarn.nodemanager.resource.cpu-vcores</td>
<td>32</td>
</tr>
<tr>
<td>yarn.scheduler.minimum-allocation-mb</td>
<td>4 GiB</td>
</tr>
<tr>
<td>yarn.scheduler.maximum-allocation-mb</td>
<td>180 GiB</td>
</tr>
<tr>
<td>yarn.scheduler.maximum-allocation-vcores</td>
<td>48</td>
</tr>
<tr>
<td>mapreduce.task.io.sort.mb</td>
<td>256 MiB</td>
</tr>
</tbody>
</table>
HDFS

<table>
<thead>
<tr>
<th>Service</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>dfs.datanode.failed.volumes.tolerated</td>
<td>6</td>
</tr>
<tr>
<td>dfs.datanode.du.reserved</td>
<td>50 GiB</td>
</tr>
<tr>
<td>dfs.datanode.data.dir.perm</td>
<td>755</td>
</tr>
<tr>
<td>Java Heap Size of Namenode in Bytes</td>
<td>2628 MiB</td>
</tr>
<tr>
<td>dfs.namenode.handler.count</td>
<td>54</td>
</tr>
<tr>
<td>dfs.namenode.service.handler.count</td>
<td>54</td>
</tr>
<tr>
<td>Java Heap Size of Secondary namenode in Bytes</td>
<td>2628 MiB</td>
</tr>
</tbody>
</table>

Apache Kafka Installation and Configuration

Cloudera Manager 5.4 or higher includes the Kafka service. To install, download Kafka using Cloudera Manager, distribute Kafka to the cluster, activate the new parcel, and add the service to the cluster.

1. Download the Kafka Parcels as shown below.

2. On a server that is accessible to the internet.

3. Create a directory for the Kafka parcels.

```
#mkdir /tmp/Kafka2.0.1Parcels
#cd /tmp/Kafka2.0.1Parcels
#wget http://archive.cloudera.com/kafka/parcels/2.0.1/KAFKA-2.0.1-1.2.0.1.p0.5-el7.parcel
#wget http://archive.cloudera.com/kafka/parcels/2.0.1/KAFKA-2.0.1-1.2.0.1.p0.5-el7.parcel.sha1
#wget http://archive.cloudera.com/kafka/parcels/2.0.1/manifest.json
```
4. Change the contents of manifest.json to match the following:

```json
{
    "lastUpdated": 14598141460000,
    "parcels": [
        {
            "parcelName": "KAFKA-2.0.1-1.2.0.1.p0.5-el7.parcel",
            "components": [
                {
                    "pkg_version": "0.9.0+kafka2.0.1",
                    "pkg_release": "1.2.0.1.p0.5",
                    "name": "kafka",
                    "version": "0.9.0-kafka2.0.1"
                }
            ],
            "depends": "CDH (>= 5.2), CDH (<= 6.0)",
            "replaces": "CLABS_KAFKA",
            "hash": "180d8322f2026f2b3609741216d2c25dd2dfb294"
        }
    ]
}
```
5. Copy the Kafka parcels over to rhel1 under /var/www/html.

   \#scp -r Kafka2.0.1Parcels rhel1:/var/www/html/


7. Click Configure. Add in a new parcel by clicking on the + button and giving the path to the new Kafka parcels that were downloaded in the previous step.

8. Click Save Changes.

9. Click Download and then Distribute to download and distribute the parcels.

10. Click Activate to add the service to the cluster.
Solution Design

Configuring Spark

The two main resources that Spark (and YARN) are dependent on are CPU and memory. Disk and network I/O, of course, play a part in Spark performance as well, but neither Spark nor YARN currently can actively manage them. Every Spark executor in any application has the same fixed number of cores and same fixed heap size. The number of cores can be specified with the executor-cores flag when invoking spark-submit, spark-shell, and pyspark from the command line, or by setting the spark.executor.cores property in the spark-defaults.conf file or in the SparkConf object.

And the heap size can be controlled with the executor-memory flag or the spark.executor.memory property. The cores property controls the number of concurrent tasks an executor can run, executor-cores = 5 mean that each executor can run a maximum of five tasks at the same time. The memory property impacts the amount of data Spark can cache, as well as the maximum sizes of the shuffle data structures used for grouping, aggregations, and joins.

The num-executors command-line flag or spark.executor.instances configuration property control the number of executors requested. Dynamic Allocation can be enabled from CDH5.4 instead setting the spark.dynamicAllocation.enabled to true. Dynamic allocation enables a Spark application to request executors when there is a backlog of pending tasks and free up executors when idle.

Asking for five executor cores will result in a request to YARN for five virtual cores. The memory requested from YARN is a little more complex for a couple reasons:
- **executor-memory/spark.executor.memory** controls the executor heap size, but JVMs can also use some memory off heap, for example for VM overhead, interned Strings and direct byte buffers. The value of the `spark.yarn.executor.memoryOverhead` property is added to the executor memory to determine the full memory request to YARN for each executor. It defaults to max \((384, 0.10 \times \text{spark.executor.memory})\).

- YARN may round the requested memory up a little. YARN's `yarn.scheduler.minimum-allocation-mb` and `yarn.scheduler.increment-allocation-mb` properties control the minimum and increment request values respectively.

- The application master is a non-executor container with the special capability of requesting containers from YARN, takes up resources of its own that must be budgeted in. In **yarn-client** mode, it defaults to a 1024MB and one vcore. In **yarn-cluster** mode, the application master runs the driver, so it’s often useful to add its resources with the --driver-memory and --driver-cores properties.

- Running executors with too much memory often results in excessive garbage collection delays. 64GB is a rough guess at a good upper limit for a single executor.

- A good estimate is that at most five tasks per executor can achieve full write throughput, so it’s good to keep the number of cores per executor around that number.

- Running tiny executors (with a single core and just enough memory needed to run a single task, for example) throws away the benefits that come from running multiple tasks in a single JVM. For example, broadcast variables need to be replicated once on each executor, so many small executors will result in many more copies of the data.

### Tuning Resource Allocation for Spark

Below is an example of configuring a Spark application to use as much of the cluster as possible, we are using an example cluster with 16 nodes running NodeManagers, each equipped with 56 cores and 256GB of memory. `yarn.nodemanager.resource.memory-mb` and `yarn.nodemanager.resource.cpu-vcores` should be set to 180 * 1024 = 184320 (megabytes) and 48 respectively.

```plaintext
spark.executor.instances/num-executors = 63
spark.executor.cores/--executor-cores = 5
spark.executor.memory/--executor-memory = 41G
```

This configuration results in four executors on all nodes except for the one with the AM, which will have three executors.

```plaintext
executor-memory is derived as \((180/4 \text{ executors per node}) = 45\); 45 * 0.10 = 4.5 45 - 4.5 ~ 40.
```

For taking care of long running processes use 2G for the spark driver

```plaintext
spark.driver.memory = 2G
```

### Submitting a Job

```plaintext
--driver --memory 2G --executor --memory 40G --num-executors 63 --executor-cores 5 --properties-file /opt/cloudera/parcels/CDH/etc/spark/conf.dist/spark-defaults.conf
```

In **yarn-cluster** mode, the local directories used by the Spark executors and the Spark driver will be the local directories configured for YARN (Hadoop YARN config `yarn.nodemanager.local-dirs`). If the user specifies `spark.local.dir`, it will be ignored.

In **yarn-client** mode, the Spark executors will use the local directories configured for YARN while the Spark driver will use those defined in `spark.local.dir`. The Spark driver does not run on the YARN cluster in **yarn-client** mode, only the Spark executors do.
spark.local.dir /tmp  (Directory to use for “scratch” space in Spark, including map output files and RDDs that get stored on disk. This should be on a fast, local disk in your system).

Every Spark stage has a number of tasks, each of which processes data sequentially. In tuning Spark jobs, this parallelism number is the most important parameter in determining performance. The number of tasks in a stage is the same as the number of partitions in the last RDD in the stage. The number of partitions in an RDD is the same as the number of partitions in the RDD on which it depends, with a couple exceptions: the coalesce transformation allows creating an RDD with fewer partitions than its parent RDD, the union transformation creates an RDD with the sum of its parents’ number of partitions, and Cartesian creates an RDD with their product.

RDDs produced by a file have their partitions determined by the underlying MapReduce InputFormat that’s used. Typically there will be a partition for each HDFS block being read. Partitions for RDDs produced by parallelize come from the parameter given by the user, or spark.default.parallelism if none is given.

The primary concern is that the number of tasks will be too small. If there are fewer tasks than slots available to run them in, the stage won’t be taking advantage of all the CPU available.

If the stage in question is reading from Hadoop, your options are:

- Use the repartition transformation, which will trigger a shuffle.
- Configure your InputFormat to create more splits.
- Write the input data out to HDFS with a smaller block size.

If the stage is getting its input from another stage, the transformation that triggered the stage boundary will accept a numPartitions argument.

The most straightforward way to tune the number of partitions is experimentation: Look at the number of partitions in the parent RDD and then keep multiplying that by 1.5 until performance stops improving.

In contrast with MapReduce for Spark when in doubt, it is almost always better to be on the side of a larger number of tasks (and thus partitions).

**Shuffle Performance Improvement**

spark.shuffle.compress true  (compress map output files)

spark.broadcast.compress true (compress broadcast variables before sending them)

spark.io.compression.codec org.apache.spark.io.SnappyCompressionCodec (codec used to compress internal data such as RDD partitions, broadcast variables and shuffle outputs)

spark.shuffle.spill.compress true (Whether to compress data spilled during shuffles.)

spark.shuffle.io.numConnectionsPerPeer 4 (Connections between hosts are reused in order to reduce connection buildup for large clusters. For clusters with many hard disks and few hosts, this may result in insufficient concurrency to saturate all disks, and so users may consider increasing this value.)

spark.shuffle.file.buffer 64K (Size of the in-memory buffer for each shuffle file output stream. These buffers reduce the number of disk seeks and system calls made in creating intermediate shuffle file)

**Improving Serialization Performance**

Serialization plays an important role in the performance of any distributed application. Often, this will be the first thing that should be tuned to optimize a Spark application.
Solution Design

spark.serializer org.apache.spark.serializer.KryoSerializer (when speed is necessary)
spark.kryo.referenceTracking false
spark.kryoserilizer.buffer 2000 (if the objects are large, may need to increase the size further to fit the size of the object being deserialized).

SparkSQL is ideally suited for mixed procedure jobs where SQL code is combined with Scala, Java, or Python programs. In general the SparkSQL command line interface is used for single user operations and ad hoc queries.

For multi-user SparkSQL environments, it is recommended to use a Thrift server connected via JDBC.

**Spark SQL Tuning**

Below are some guidelines for Spark SQL tuning:

1. To compile each query to Java bytecode on the fly, turn on sql.codegen. This can improve performance for large queries, but can slow down very short queries.
   
   spark.sql.codegen true
   spark.sql.unsafe.enabled true

2. Configuration of in-memory caching can be done using the setConf method on SQLContext or by running SET key=value commands using SQL.

3. spark.sql.inMemoryColumnarStorage.compressed true (will automatically select a compression codec for each column based on statistics of the data)

4. spark.sql.inMemoryColumnarStorage.batchSize 5000 (Controls the size of batches for columnar caching. Larger batch sizes can improve memory utilization and compression, but risk OOMs when caching data)

5. The columnar nature of the ORC format helps avoid reading unnecessary columns, but it is still possible to read unnecessary rows. ORC avoids this type of overhead by using predicate push-down with three levels of built-in indexes within each file: file level, stripe level, and row level. This combination of indexed data and columnar storage reduces disk I/O significantly, especially for larger datasets where I/O bandwidth becomes the main bottleneck for performance.

6. By default, ORC predicate push-down is disabled in Spark SQL. To obtain performance benefits from predicate push-down, enable it explicitly, as follows:

7. spark.sql.orc.filterPushdown=true

8. In SparkSQL to automatically determine the number of reducers for joins and groupbys, use the parameter,

9. spark.sql.shuffle.partitions 200, (default value is 200)

This property can be put into hive-site.xml to override the default value.

10. Set log to WARN in log4j.properties to reduce log level.

---

Note: Running Thrift server and connecting to spark-sql through beeline is the recommended option for multi-session testing.
### Compression for Hive

Set the following Hive parameters to compress the Hive output files using Snappy compression:

```
hive.exec.compress.output=true
hive.exec.orc.default.compress=SNAPPY
```

### Changing the Log Directory for All Applications

To change the default log from the `/var` prefix to `/data/disk1`, complete the following steps:

1. Log into the cloudera home page and click `My Clusters`.
2. From the configuration drop-down list select “All Log Directories.”
3. Click Save changes.
Cloudera Data Science Workbench (CDSW)

CDSW has pre-requisites, one of which is CUDA. CUDA itself, also has prerequisites. The order of installation is:

- CUDA pre-requisites
- CUDA
- CDSW pre-requisites
- CDSW

Installing Prerequisites for CUDA

Note: Details of CUDA installation can be found here: http://docs.nvidia.com/cuda/cuda-installation-guide-linux/index.html.

Note: These commands are run as root or sudo.

List GPUs and CPUs installed:

   lspci | grep -i nvidia

   # lspci | grep -i nvidia

   5e:00.0 3D controller: NVIDIA Corporation GP100GL [Tesla P100 PCIe 16GB] (rev a1)
   af:00.0 3D controller: NVIDIA Corporation GP100GL [Tesla P100 PCIe 16GB] (rev a1)
   #lscpu
GCC Installation

Make sure gcc is installed in the system:

```
# gcc --version

gcc (GCC) 4.8.5 20150623 (Red Hat 4.8.5-16)

Copyright (C) 2015 Free Software Foundation, Inc.

This is free software; see the source for copying conditions.
```

Install Kernel Headers and Installation Packages

```
# yum install kernel-devel-$\{uname -r\} kernel-headers-$\{uname -r\}

# yum install kernel-devel-$\{uname -r\} kernel-headers-$\{uname -r\}

Package kernel-devel-3.10.0-693.el7.x86_64 already installed and latest version

Package kernel-headers-3.10.0-693.el7.x86_64 already installed and latest version

Nothing to do
```

Install dkms

The NVIDIA driver RPM packages depend on other external packages, such as DKMS. Those packages are only available on third-party repositories, such as EPEL.

http://rpmfind.net/linux/rpm2html/search.php?query=dkms for RHEL 7.x
RHEL 7.x [http://rpmfind.net/linux/epel/7/x86_64/Packages/d/dkms-2.4.0-1.20170926git959bd74.el7.noarch.rpm](http://rpmfind.net/linux/epel/7/x86_64/Packages/d/dkms-2.4.0-1.20170926git959bd74.el7.noarch.rpm)

```
#wget http://rpmfind.net/linux/epel/7/x86_64/Packages/d/dkms-2.4.0-1.20170926git959bd74.el7.noarch.rpm
```

Copy dkms rpm to all the GPU servers

Install dkms with yum install

```
yum install dkms-2.4.0-1.20170926git959bd74.el7.noarch.rpm
```

Install NVIDIA GPU Drivers

Download this NVIDIA GPU driver from [http://docs.nvidia.com/deeplearning/sdk/cudnn-install/index.html#installdriver](http://docs.nvidia.com/deeplearning/sdk/cudnn-install/index.html#installdriver).

```
# wget http://us.download.nvidia.com/tesla/390.12/nvidia-diag-driver-local-repo-rhel7-390.12-1.0-1.x86_64.rpm
```

```
# rpm -ivh nvidia-diag-driver-local-repo-rhel7-390.12-1.0-1.x86_64.rpm
```

```
warning: nvidia-diag-driver-local-repo-rhel7-390.12-1.0-1.x86_64.rpm: Header V3 RSA/SHA512 Signature, key ID 7fa2af80: NOKEY
Preparing... #........................................................ [100%]
Updating / installing...
1:nvidia-diag-driver-local-repo-rhel7-390.12-1.0-1.x86_64 RPM:[100%]
```

**Installing CUDA**

Download CUDA 8 (Tensorflow 1.4 needed CUDA 8, later version of tensorflow uses CUDA 9).

Note: Tensorflow needs CUDA version 8 or 9, make sure which version of CUDA is supported by tensorflow before installing CUDA. Earlier versions of CUDA are at [https://developer.nvidia.com/cuda-toolkit-archive](https://developer.nvidia.com/cuda-toolkit-archive).

Latest version of CUDA is available at the following link:


This CVD is showing sample steps to install CUDA 8 to use Tensorflow 1.4. Please use the appropriate CUDA as needed.

```
#wget https://developer.nvidia.com/compute/cuda/8.0/Prod2/local_installers/cuda-repo-rhel7-8-0-local-ga2-8.0.61-1.x86_64-rpm
```

```
#rpm -i cuda-repo-rhel7-8-0-local-ga2-8.0.61-1.x86_64.rpm
```

```
warning: cuda-repo-rhel7-8-0-local-ga2-8.0.61-1.x86_64.rpm: Header V3 RSA/SHA512 Signature, key ID 7fa2af80: NOKEY
# yum clean all
```

Cleaning repos: cuda-8-0-local-ga2 nvidia-diag-driver-local-390.12 rhel7.4

Cleaning up everything
Install CUDA on the GPU nodes as follows:

```
[root@cdsw1 ~]# yum install cuda -nogpgcheck
# yum install cuda --nogpgcheck
```

Total size: 1.5 G
Total download size: 1.5 G
Installed size: 2.5 G
Is this ok [y/d/N]: y

Download and Setup NVIDIA CUDA Deep Neural Network library (cuDNN)

Download cuDNN 6.5


Copy it into all the GPU servers.


```
tar -xzvf cudnn-6.5-linux-x64-v2.tgz
```

```
cp cudnn.h /usr/local/cuda-8.0/include
```

```
cp libcudnn* /usr/local/cuda-8.0/lib64
```

```
chmod a+r /usr/local/cuda-8.0/include/cudnn.h /usr/local/cuda-8.0/lib64/libcudnn*
```

```
[root@UCS ~]# cp cuda/include/cudnn.h /usr/local/cuda-9.1/include
[root@UCS ~]# cp cuda/lib64/libcudnn* /usr/local/cuda-9.1/lib64
```
Solution Design

Post Installation Steps

Add CUDA in PATH and LD_LIBRARY_PATH variable

The PATH and LD_LIBRARY_PATH variable needs to include /usr/local/<cuda 8 or 9 path>/bin

$ export PATH=/usr/local/<cuda location>/bin${PATH:+:${PATH}}
$ export LD_LIBRARY_PATH=/usr/local/<cuda location>/lib\n   ${LD_LIBRARY_PATH:+:${LD_LIBRARY_PATH}}

Note: For more information, refer to: https://docs.nvidia.com/cuda/cuda-installation-guide-linux/index.html#post-installation-actions.

Verify Drivers

Restart the servers to verify the drivers have been installed:

# nvidia-smi

![nvidia-smi output]

[root@cdsw1 bin]# pwd
/usr/local/cuda-8.0/bin

[root@cdsw1 bin]# ./nvcc --version
	nvcc: NVIDIA (R) Cuda compiler driver
Copyright (c) 2005-2017 NVIDIA Corporation
Built on Fri_Nov_3_21:07:56_CDT_2017
Cuda compilation tools, release 8.0, V8.0

Based on the Driver version, add the following in ~/.bash_profile of root

export NVIDIA_DRIVER_VERSION=390.12
Installation Prerequisites for CDSW

Complete the following steps on all CDSW nodes:


Set Up a Wildcard DNS Subdomain

Cloudera Data Science Workbench uses subdomains to provide isolation for user-generated HTML and JavaScript, and routing requests between services. To set up subdomains for Cloudera Data Science Workbench, configure your DNS server with an A record for a wildcard DNS name such as *.cdsw.<your_domain>.com for the master host, and a second A record for the root entry of cdsw.<your_domain>.com.

```
[root@cdsw ~]# cat /etc/hosts
127.0.0.1   localhost localhost.localdomain localhost4 localhost4.localdomain4 ::1         localhost localhost.localdomain localhost6 localhost6.localdomain6
10.13.1.50 rhel1
10.13.1.51 rhel2
10.13.1.52 rhel3
10.13.1.53 rhel4
10.13.1.54 rhel5
10.13.1.55 rhel6
10.13.1.56 rhel7
10.13.1.57 rhel8
...
10.13.1.73 rhel24
10.13.1.250 cdsw1.cisco.com
10.13.1.251 cdsw2.cisco.com
10.13.1.252 cdsw3.cisco.com
10.13.1.253 cdsw4.cisco.com
```

You can also use a wildcard CNAME record if it is supported by your DNS provider.

---

This Wildcard DNS subdomain need to be used by the jump host/edge node or the bastion server as well.

With dnsmasq, to add a wildcard DNS subdomain, do the following for just the master host.

**Update /etc/dnsmasq.conf to enable Wildcard entry**

```
address=/cdsw/10.13.1.250
```

```
#service dnsmasq restart
```

**Test the working of wildcard DNS with dig or nslookup**

```
#nslookup *.cdsw.<your_domain>.com
#dig cdsw.<your_domain>.com
#dig *.cdsw.<your_domain>.com
```
Supported JDK Version

The entire CDH cluster, including Cloudera Data Science Workbench gateway nodes, must use Oracle JDK. OpenJDK is not supported by CDH, Cloudera Manager, or Cloudera Data Science Workbench.

Spark 2.2 which is needed by CDSW to run Spark jobs (on Hadoop nodes) requires JDK 1.8. On CSD-based deployments, Cloudera Manager automatically detects the path and version of Java installed on Cloudera Data Science Workbench gateway hosts.

To upgrade your entire CDH cluster to JDK 1.8, see Upgrading to Oracle JDK 1.8.

IP Tables and Security on CDSW Nodes

Disable all pre-existing iptables rules. While Kubernetes makes extensive use of iptables, it is difficult to predict how pre-existing iptables rules will interact with the rules inserted by Kubernetes. Therefore, Cloudera recommends you use the following commands to disable all pre-existing rules before you proceed with the installation.

```
yum -y install iptables-service
yum install initscripts
systemctl stop firewalld
systemctl mask firewalld
systemctl disable firewalld
systemctl enable iptables
systemctl start iptables
iptables -P INPUT ACCEPT
iptables -P FORWARD ACCEPT
iptables -P OUTPUT ACCEPT
iptables -t nat -F
iptables -t mangle -F
iptables -F
iptables -X
service iptables save
```
Disable SELinux

```
  sed -i 's/SELINUX=enforcing/SELINUX=disabled/g' /etc/selinux/config
  setenforce 0
  sestatus
```

Configure Block Devices

**Docker Block Device**

The Cloudera Data Science Workbench installer will format and mount Docker on each gateway host. Make sure there is no important data stored on these devices. Do not mount these block devices prior to installation.

**Application Block Device or Mount Point**

The master host on Cloudera Data Science Workbench requires at least 500 GB for database and project storage. This recommended capacity is contingent on the expected number of users and projects on the cluster. While large data files should be stored on HDFS, it is not uncommon to find gigabytes of data or libraries in individual projects. Running out of storage will cause the application to fail. Make sure you continue to carefully monitor disk space usage and I/O using Cloudera Manager.

⚠️ Note: To enable data resilience, enable this drive as RAID1 of SSDs (using commands as shown in configuring namenode).

Cloudera Data Science Workbench will store all application data at `/var/lib/cdsw`. In a CSD-based deployment, this location is not configurable. Cloudera Data Science Workbench will assume the system administrator has formatted and mounted one or more block devices to `/var/lib/cdsw`.

Regardless of the application data storage configuration you choose, `/var/lib/cdsw` must be stored on a separate block device (the RAID1 of SSDs created for this).

Download and Install CDSW with Cloudera Manager

To download and install CDSW, complete the following steps:

1. Download the Cloudera Data Science Workbench Custom Service Descriptor (CSD) `CLOUDERA_DATA_SCIENCE_WORKBENCH-1.3.0.jar`

   ⚠️ Note: For more information, refer to: [https://www.cloudera.com/documentation/data-science-workbench/latest/topics/cdsw_install_parcel.html#csd](https://www.cloudera.com/documentation/data-science-workbench/latest/topics/cdsw_install_parcel.html#csd).

2. Log on to the Cloudera Manager Server host, and place the CSD file under `/opt/cloudera/csd`, which is the default location for CSD files.

3. Once cloudera Manager is installed under `/opt/cloudera/csd` set the file ownership to `cloudera-scm:cloudera-scm` with permission `644`

4. `chown cloudera-scm:cloudera-scm CLOUDERA_DATA_SCIENCE_WORKBENCH-1.3.0.jar`

5. `chmod 644 CLOUDERA_DATA_SCIENCE_WORKBENCH-1.3.0.jar`

6. Restart cloudera-management service
7. `service cloudera-scm-server restart`

8. Log into the Cloudera Manager Admin Console and restart the Cloudera Management Service.
   a. Select Clusters > Cloudera Management Service.
   b. Select Actions > Restart.
   c. Download Cloudera CDSW parcel from [http://archive.cloudera.com/cdsw/1/parcels/1.3.0/](http://archive.cloudera.com/cdsw/1/parcels/1.3.0/) into the server hosting Cloudera parcels copy all these files to CDSW folder and move it to /var/www/html/
   d. Log into the Cloudera Manager Admin Console.
   e. Click Hosts > Parcels in the main navigation bar.
   f. Download the Cloudera Data Science Workbench parcel, distribute the parcel to the hosts in your cluster, and then activate the parcel.

---

For more information, refer to: [https://www.cloudera.com/documentation/data-science-workbench/latest/topics/cdsw_install_parcel.html](https://www.cloudera.com/documentation/data-science-workbench/latest/topics/cdsw_install_parcel.html)

---

**Installing Apache Spark 2 on YARN**

To download and install Apache Spark 2 on YARN, complete the following steps:

To install CDS Powered by Apache Spark, refer to: [https://www.cloudera.com/documentation/spark2/latest/topics/spark2_installing.html](https://www.cloudera.com/documentation/spark2/latest/topics/spark2_installing.html).

---

1. Download these files from this link
   [https://www.cloudera.com/documentation/spark2/latest/topics/spark2_packaging.html#versions](https://www.cloudera.com/documentation/spark2/latest/topics/spark2_packaging.html#versions) and place them under /opt/cloudera/csd:

   - `manifest.json`
   - `SPARK2-2.2.0.cloudera2-1.cdh5.12.0.p0.232957-e17.parcel.sha1`
   - `SPARK2-2.2.0.cloudera2-1.cdh5.12.0.p0.232957-e17.parcel`
2. **Set the file ownership to** `cloudera-scm:cloudera-scm` **with permission 644**

   
   ```bash
   chown cloudera-scm:cloudera-scm SPARK2_ON_YARN-2.2.0.cloudera2.jar
   chmod 644 SPARK2_ON_YARN-2.2.0.cloudera2.jar
   ```

3. **Copy all files to spark2 folder and move them to /var/www/html/**:

   ```bash
   Service clouderar-scm-server restart
   ```

4. **Click the Parcel tab > host add configuration link then download > add service and download and distribute.**

5. **Restart the Cloudera Management service.**

   ```bash
   Select Clusters > Cloudera Management Service.
   Select Actions > Restart.
   ```

6. **Add new hosts for CDSW and add the repository for CDH, CDSW and Spark2:**
   
   a. **Hosts > Now add new host to the cluster and add cdsw**
   b. **Hosts > Now add new host to the cluster and add Spark2**

   ![Parcel Settings](image)

   The repository is being distributed and activated.
7. Add the Spark 2 service to the cluster and deploy Spark 2 Gateway roles to the CDSW hosts, which are going to deploy Spark Job through CDSW.

Note: CDSW need Spark 2 to deploy Spark Jobs.
Sentry service stores authorization policy metadata and provides clients concurrent and secure access to this metadata.

Solr is a distributed service for indexing and searching data stored in HDFS.

Apache Spark is an open source cluster computing system. This service runs Spark as an application on YARN.

Apache Spark is an open source cluster computing system. This is the standalone version of the service which does not use YARN for resource management. Cloudera recommends using Spark on YARN instead of this standalone version.

Apache Spark is an open source cluster computing system. This service runs Spark 2 as an application on YARN. Before adding this service, ensure that you have installed the Spark2 binaries, which are not included in CDH.

Configuration and connector management for Sqoop 1.

Sqoop is a tool designed for efficiently transferring bulk data between Apache Hadoop and structured datastores such as relational databases. The version supported by Cloudera is Sqoop 2.
Add the Cloudera Data Science Workbench Service

To install CDSW, complete the following steps:

- Log into the Cloudera Manager Admin Console.
- On the Home > Status tab, right-click the cluster name and select Add a Service to launch the wizard. A list of services will be displayed.

For more information, refer to: https://www.cloudera.com/documentation/data-science-workbench/latest/topics/cdsw_install_parcel.html.
3. Select the Cloudera Data Science Workbench service and click Continue.

4. Assign the Master and Worker roles to the gateway hosts. You must assign the Cloudera Data Science Workbench Master role to one gateway host, and optionally, assign the Worker role to one or more gateway hosts.

Note: Do not assign Masters and Workers to the same host; even on single node deployments, the Master can perform the functions of a Worker as needed.
5. Configure the following parameters and click Continue.

Note: For more information about the parameters, refer to: https://www.cloudera.com/documentation/data-science-workbench/latest/topics/cdsw_install_parcel.html.
6. The wizard will now begin a First Run of the Cloudera Data Science Workbench service. This includes deploying client configuration for HDFS, YARN and Spark 2, installing the package dependencies on all hosts, and formatting the Docker block device. The wizard will also assign the Application role to the host running Master, and the Docker Daemon role to all the Cloudera Data Science Workbench gateway hosts.

7. Once the First Run command has completed successfully, click Finish to go back to the Cloudera Manager home page.

- Note: CDSW will take 10-15 minutes to come online for the first time; they can follow the Docker/Master/Application process logs to track progress.
Create the Administrator Account

After your installation is complete, set up the initial administrator account. Go to the Cloudera Data Science Workbench web application at http://cdsw.<your_domain>.com.

You must access Cloudera Data Science Workbench from the Cloudera Data Science Workbench Domain configured when setting up the service, and not the hostname of the master node. Visiting the hostname of the master node will result in a 404 error.

The first account that you create becomes the site administrator. You may now use this account to create a new project and start using the workbench to run data science workloads.
Non-Kerberized Clusters

Note: In this CVD, we did not enable Kerberos.

Note: To enable Kerberos, refer to: https://www.cloudera.com/documentation/data-science-workbench/latest/topics/cdsw_kerberos.html.

1. To disable Kerberos, delete the file /etc/krb5.conf on the CDSW nodes.

2. For a non-kerberized cluster, by default, your Hadoop username will be set to your Cloudera Data Science Workbench login username. Override this default and set an alternative HADOOP_USER_NAME ("admin" in this case which was the admin user created).

3. Create User admin in hdfs.
   
   hdfs dfs -mkdir /user/admin
   hdfs dfs -chown admin:admin /user/admin

4. Go to the Hadoop Username Override setting at Account settings > Hadoop Authentication.

User Settings

Profile  SSH Keys  Hadoop Authentication  API Keys

Hadoop Username Override

Set the HADOOP_USER_NAME environmental variable to override your Hadoop username for all sessions and jobs.

admin

Authenticate

5. Restart CDSW from Cloudera Manager.
Using GPUs for Cloudera Data Science Workbench Workloads

A GPU is a specialized processor that can be used to accelerate highly parallelized computationally-intensive workloads. Because of their computational power, GPUs have been found to be particularly well-suited to deep learning workloads. Ideally, CPUs and GPUs should be used in tandem for data engineering and data science workloads. A typical machine learning workflow involves data preparation, model training, model scoring, and model fitting. You can use existing general-purpose CPUs for each stage of the workflow, and optionally accelerate the math-intensive steps with the selective application of special-purpose GPUs. For example, GPUs allow you to accelerate model fitting using frameworks such as Tensorflow, PyTorch, Keras, MXNet, and Microsoft Cognitive Toolkit (CNTK).

By enabling GPU support, data scientists can share GPU resources available on Cloudera Data Science Workbench nodes. Users can request a specific number of GPU instances, up to the total number available on a node, which are then allocated to the running session or job for the duration of the run. Projects can use isolated versions of libraries, and even different CUDA and cuDNN versions via Cloudera Data Science Workbench’s extensible engine feature.

Note: For more information, refer to: https://www.cloudera.com/documentation/data-science-workbench/latest/topics/cdsw_gpu.html.

Enabling GPU with CDSW

- Cloudera Data Science Workbench only supports CUDA-enabled NVIDIA GPU cards.
- Cloudera Data Science Workbench does not support heterogeneous GPU hardware in a single deployment.
- Cloudera Data Science Workbench does not include an engine image that supports NVIDIA libraries.

This section provides instructions about creating your own custom CUDA-capable engine image.

To enable Docker containers to use the GPUs, the previously installed NVIDIA driver libraries must be consolidated in a single directory named after the <driver_version>, and mounted into the containers. This is done using the nvidia-docker package, which is a thin wrapper around the Docker CLI and a Docker plugin.

The following sample steps demonstrate how to use nvidia-docker to set up the directory structure for the drivers so that they can be easily consumed by the Docker containers that will leverage the GPU. Perform these steps on all nodes with GPU hardware installed.

1. Download and install nvidia-docker.

   wget https://github.com/NVIDIA/nvidia-docker/releases/download/v1.0.1/nvidia-docker-1.0.1-1.x86_64.rpm

   yum install nvidia-docker-1.0.1-1.x86_64.rpm

   Note: This example uses nvidia-docker 1.0.

2. Start the necessary services and plugins:

   systemctl start nvidia-docker
   systemctl enable nvidia-docker

3. Run a small container to create the Docker volume structure

   nvidia-docker run --rm nvidia/cuda nvidia-smi
4. Verify that the `/var/lib/nvidia-docker/volumes/nvidia_driver/$NVIDIA_DRIVER_VERSION/` directory was created.

5. Use the following Docker command to verify that Cloudera Data Science Workbench can access the GPU:

   ```bash
   docker run --net host \
   --device=/dev/nvidiactl \
   --device=/dev/nvidia-uvm \
   --device=/dev/nvidia0 \
   -v /var/lib/nvidia-docker/volumes/nvidia_driver/$NVIDIA_DRIVER_VERSION:/usr/local/nvidia/ \
   -it nvidia/cuda \
   /usr/local/nvidia/bin/nvidia-smi
   ```
Note: On a multi-GPU machine the output of this command will show exactly one GPU. This is because we have run this sample Docker container with only one device (/dev/nvidia0).

6. Restart CDSW.

**Enable GPU Support in Cloudera Data Science Workbench**

To enable Cloudera Data Science Workbench to identify the GPUs installed, complete the following steps:

1. In Cloudera Manager – Cloudera Data Science Workbench Configuration, set the following properties:
   - Enable GPU Support
   - Complete path to the NVIDIA driver libraries 
     
     /var/lib/nvidia-docker/volumes/nvidia_driver/

2. Restart CDSW from Cloudera Manager for GPU to display in Cloudera Data Science Workbench:
Create a Custom CUDA-Capable Engine Image

The base engine image (docker.repository.cloudera.com/cdsw/engine:4) that ships with Cloudera Data Science Workbench will need to be extended with CUDA libraries to make it possible to use GPUs in jobs and sessions.

For more information, refer to https://www.cloudera.com/documentation/data-science-workbench/latest/topics/cdsw_gpu.html.

For this we first create a local registry

Run a Local Registry

To start the registry container, run the following command:

```
$ docker run -d -p 5000:5000 --restart=always --name registry registry:2
```

For more information about creating a local docker registry, refer to: https://docs.docker.com/registry/deploying/#copy-an-image-from-docker-hub-to-your-registry

The following sample Dockerfile illustrates an engine on top of which machine learning frameworks such as Tensorflow and PyTorch can be used. This Dockerfile uses a deep learning library from NVIDIA called NVIDIA CUDA Deep Neural Network (cuDNN). Make sure you check with the machine learning framework that you intend to use in order to know which version of cuDNN is needed. As an example, Tensorflow 1.4 uses CUDA 8.0 and requires cuDNN 6.0.
To create the cuda.Dockerfile, run the following command:

```
FROM  docker.repository.cloudera.com/cdsw/engine:4

RUN NVIDIA_GPGKEY_SUM=d1be581509378368edeec8c1eb2958702feedf3bc3d17011adbf24efacce4ab5 &
    NVIDIA_GPGKEY_FPR=ae09fe4b223a84b2ccf3e3f60f4b3d7fa2af80 &
    apt-key adv --fetch-keys
    http://developer.download.nvidia.com/compute/cuda/repos/ubuntu1604/x86_64/7fa2af80.pub &
    apt-key adv --export --no-emit-version -a $NVIDIA_GPGKEY_FPR | tail -n +5 >
    cudasign.pub &
    echo "$NVIDIA_GPGKEY_SUM  cudasign.pub" | sha256sum -c --strict - &
    rm cudasign.pub &
    echo "deb http://developer.download.nvidia.com/compute/cuda/repos/ubuntu1604/x86_64 /" >
    /etc/apt/sources.list.d/cuda.list

ENV CUDA_VERSION 8.0.61
LABEL com.nvidia.cuda.version="${CUDA_VERSION}"

ENV CUDA_PKG_VERSION 8-0=${CUDA_VERSION}-1
RUN apt-get update &
    apt-get install --no-install-recommends \
        cuda-nvrtc=${CUDA_PKG_VERSION} \
        cuda-nvgraph=${CUDA_PKG_VERSION} \
        cuda-cusolver=${CUDA_PKG_VERSION} \
        cuda-cublas-8-0=8.0.61.2-1 \
        cuda-cufft=${CUDA_PKG_VERSION} \
        cuda-curand=${CUDA_PKG_VERSION} \
        cuda-cusparse=${CUDA_PKG_VERSION} \
        cuda-npp=${CUDA_PKG_VERSION} \
        cuda-cudart=${CUDA_PKG_VERSION} &
    ln -s cuda-8.0 /usr/local/cuda &
    rm -rf /var/lib/apt/lists/*

RUN echo "/usr/local/cuda/lib64" >> /etc/ld.so.conf.d/cuda.conf &
    ldconfig

RUN echo "/usr/local/nvidia/lib" >> /etc/ld.so.conf.d/nvidia.conf &
    echo "/usr/local/nvidia/lib64" >> /etc/ld.so.conf.d/nvidia.conf

ENV PATH /usr/local/nvidia/bin:/usr/local/cuda/bin:${PATH}
ENV LD_LIBRARY_PATH /usr/local/nvidia/lib:/usr/local/nvidia/lib64

RUN echo "deb http://developer.download.nvidia.com/compute/machine-learning/repos/ubuntu1604/x86_64 /" >
    /etc/apt/sources.list.d/nvidia-ml.list

ENV CUDNN_VERSION 6.0.21
LABEL com.nvidia.cudnn.version="${CUDNN_VERSION}"

RUN apt-get update &
    apt-get install --no-install-recommends \
        libcudnn6=${CUDNN_VERSION}-1+cuda8.0 &
    rm -rf /var/lib/apt/lists/*
```

Build the custom engine image out of cuda.Dockerfile using the following command:
# docker build --network host -t localhost/cisco-gpu-demo:2 . -f cuda.Dockerfile

```
[root@cdsw1 CUDA]# docker build --network host -t localhost/cisco-gpu-demo:2 . -f cuda.Dockerfile
```

Solution Design

```
Step 1/4 : FROM docker.repository.cloudera.com/cdsdm/engine:4
---> 62a5f1dfb8ad

Step 2/4 : RUN NVIDIA_GPGKEY_SIM_SLM=be8159093783679ee6d81ec8f485e02f5fedf3bc3d710113afdb24f4acce4e3a && NVIDIA_GPGKEY
download.nvidia.com/compute/cuda/repos/ubuntu1804/x6_04_7f0a3f080.pub && apt-key adv --export --no-md5 SIM
cud-sign.pub || sha256sum -c --strict -r cudesign.pub & echo "deb http://developer.download.nvidia.com/compute/cuda/repos/ubuntu1804/x86_64_7f0a3f080.pub key
[cuda.r/W] 19350113a0b72c46bb555fa4335202d4f

Sending build context to Docker daemon layer 2.625 GB

Step 3/4 : ENV CUDALUMIARY 8.0.61
---> 9b8b68c409ac

Step 4/4 : LABEL com.nvidia.cuda.version "$CUDALUMIARY" 
---> Running in 6883946d3f9d

Removing intermediate container 64be3bc17f43

Removing intermediate container 9eb8f8586d06

Removing intermediate container 0066f1c1f33f

Sending build context to Docker daemon layer 1.294 MB

Step 1/4 : FROM ubuntu:18.04
---> 62a5f1dfb8ad

Step 2/4 : RUN apt-get update && apt-get install -y 
```

# docker images

```
[root@cdsw1 CUDA]# docker images
```

<table>
<thead>
<tr>
<th>REPOSITORY</th>
<th>TAG</th>
<th>IMAGE ID</th>
<th>CREATED</th>
<th>SIZE</th>
</tr>
</thead>
<tbody>
<tr>
<td>localhost/cisco-gpu-demos</td>
<td>2</td>
<td>3716c3f9f8b8</td>
<td>13 seconds ago</td>
<td>7.05 GB</td>
</tr>
<tr>
<td>nvidia/cuda</td>
<td>latest</td>
<td>49dfb7f70c76</td>
<td>11 days ago</td>
<td>2.25 GB</td>
</tr>
<tr>
<td>docker.repository.cloudera.com/cdsdm/engine:4</td>
<td>62a5f1dfb8ad</td>
<td>5866628f0</td>
<td>5866628f0</td>
<td>5866628f0</td>
</tr>
</tbody>
</table>

[Root@cdsw1 CUDA]# docker tag localhost/cisco-gpu-demo:2 localhost:5000/cisco-gpu-demo:2

```
[Root@cdsw1 CUDA]# docker push localhost:5000/cisco-gpu-demo:2
```

Tag the image as localhost:5000/<image name>. This creates an additional tag for the existing image. When the first part of
the tag is a hostname and port, Docker interprets this as the location of a registry, when pushing.

```
[root@cdsw1 CUDA]# docker tag localhost/cisco-gpu-demo:2 localhost:5000/cisco-gpu-demo:2

Push the image to the local registry running at localhost:5000:
```

```
$ docker push localhost:5000/cisco-gpu-demo:2
```

```
[Root@cdsw1 CUDA]# docker images
```

<table>
<thead>
<tr>
<th>REPOSITORY</th>
<th>TAG</th>
<th>IMAGE ID</th>
<th>CREATED</th>
<th>SIZE</th>
</tr>
</thead>
<tbody>
<tr>
<td>localhost/cisco-gpu-demos</td>
<td>2</td>
<td>3716c3f9f8b8</td>
<td>13 seconds ago</td>
<td>7.05 GB</td>
</tr>
<tr>
<td>nvidia/cuda</td>
<td>latest</td>
<td>49dfb7f70c76</td>
<td>11 days ago</td>
<td>2.25 GB</td>
</tr>
<tr>
<td>docker.repository.cloudera.com/cdsdm/engine:4</td>
<td>62a5f1dfb8ad</td>
<td>5866628f0</td>
<td>5866628f0</td>
<td>5866628f0</td>
</tr>
</tbody>
</table>
Allocate GPUs for Sessions and Jobs

Once Cloudera Data Science Workbench has been enabled to use GPUs, a site administrator must whitelist the CUDA-capable engine image created in the previous step. Site administrators can also set a limit on the maximum number of GPUs that can be allocated per session or job.

1. Sign in to Cloudera Data Science Workbench as a site administrator.

2. Click Admin.

3. Go to the Engines tab.

4. From the **Maximum GPUs per Session/Job drop-down** list, select the maximum number of GPUs that can be used by an engine.

5. Under **Engine Images**, add the custom CUDA-capable engine image created in the previous step. This whitelists the image and allows project administrators to use the engine in their jobs and sessions.

6. Click Update.
Project administrators can now whitelist the CUDA engine image to make it available for sessions and jobs within a particular project by completing the following steps:

1. Go to Projects Overview
2. Click on any projects and choose Settings
3. Go to the Engines tab in the above page
4. Under Engine Image, add the CUDA-capable engine image.
Bill of Materials

This section provides the BOM for the 24 Nodes Hadoop Base Rack and 8 Nodes Hadoop Expansion Rack. See Table 9 for BOM for the Hadoop Base rack, Table 10 for BOM for Hadoop Expansion Rack, 0 for BOM for Hadoop GPU Rack, Table 12, and Table 13. Table 15 for software components. Table 14 lists Cloudera SKUs available from Cisco.

Note: If UCS-SP-CPA4-P2 is added to the BOM all the required components for 16 servers only are automatically added. If not customers can pick each of the individual components that are specified after this and build the BOM manually.

<table>
<thead>
<tr>
<th>Table 9</th>
<th>Bill of Materials for C240M5SX Hadoop Nodes Base Rack</th>
</tr>
</thead>
<tbody>
<tr>
<td>Part Number</td>
<td>Description</td>
</tr>
<tr>
<td>UCS-SP-C240M5-A2</td>
<td>SP C240 M5SX w/2x6132,6x32GB mem, VIC1387</td>
</tr>
<tr>
<td>CON-OSP-C240M5A2</td>
<td>SNTC 24X7X4OS UCS C240 M5 A2</td>
</tr>
<tr>
<td>UCS-CPU-6132</td>
<td>2.6 GHz 6132/140W 14C/19.25MB Cache/DDR4 2666MHz</td>
</tr>
<tr>
<td>UCS-MR-X32G2RS-H</td>
<td>32GB DDR4-2666-MHz RDIMM/PC4-21300/dual rank/x4/1.2v</td>
</tr>
<tr>
<td>UCSC-PCI-1-C240M5</td>
<td>Riser 1 including 3 PCIe slots (x8, x16, x8); slot 3 required CPU2</td>
</tr>
<tr>
<td>UCSC-MLOM-C40Q-03</td>
<td>Cisco VIC 1387 Dual Port 40Gb QSFP CNA MLOM</td>
</tr>
<tr>
<td>UCSC-PSU1-1600W</td>
<td>Cisco UCS 1600W AC Power Supply for Rack Server</td>
</tr>
<tr>
<td>CAB-9K12A-NA</td>
<td>Power Cord, 125VAC 13A NEMA 5-15 Plug, North America</td>
</tr>
<tr>
<td>UCSC-RAILB-M4</td>
<td>Ball Bearing Rail Kit for C220 &amp; C240 M4 &amp; M5 rack servers</td>
</tr>
<tr>
<td>CIMC-LATEST</td>
<td>IMC SW (Recommended) latest release for C-Series Servers.</td>
</tr>
<tr>
<td>UCSC-HS-C240M5</td>
<td>Heat sink for UCS C240 M5 rack servers 150W CPUs &amp; below</td>
</tr>
<tr>
<td>UCSC-BBLKD-S2</td>
<td>UCS C-Series M5 SFF drive blanking panel</td>
</tr>
<tr>
<td>UCSC-PCIF-240M5</td>
<td>C240 M5 PCIe Riser Blanking Panel</td>
</tr>
<tr>
<td>CBL-SC-MR12GM5P</td>
<td>Super Cap cable for UCSC-RAID-M5HD</td>
</tr>
<tr>
<td>UCSC-SCAP-M5</td>
<td>Super Cap for UCSC-RAID-M5, UCSC-MRAID1GB-KIT</td>
</tr>
<tr>
<td>UCSC-RAID-M5HD</td>
<td>Cisco 12G Modular RAID controller with 4GB cache</td>
</tr>
<tr>
<td>UCS-SP-FI6332-2X</td>
<td>UCS SP Select 2 x 6332 Fl</td>
</tr>
<tr>
<td>UCS-SP-FI6332</td>
<td>(Not sold standalone) UCS 6332 1RU Fl/12 QSFP+</td>
</tr>
<tr>
<td>CON-OSP-SPFI6332</td>
<td>ONSITE 24X7X4 (Not sold standalone) UCS 6332 1RU Fl/No PSU/3</td>
</tr>
<tr>
<td>UCS-PSU-6332-AC</td>
<td>UCS 6332 Power Supply/100-240VAC</td>
</tr>
<tr>
<td>CAB-9K12A-NA</td>
<td>Power Cord, 125VAC 13A NEMA 5-15 Plug, North America</td>
</tr>
<tr>
<td>QSFP-H40G-CU3M</td>
<td>40GBASE-CR4 Passive Copper Cable, 3m</td>
</tr>
<tr>
<td>QSFP-40G-SR-BD</td>
<td>QSFP40G BiDi Short-reach Transceiver</td>
</tr>
<tr>
<td>N10-MGT015</td>
<td>UCS Manager v3.2(1)</td>
</tr>
<tr>
<td>UCS-ACC-6332</td>
<td>UCS 6332 Chassis Accessory Kit</td>
</tr>
<tr>
<td>UCS-FAN-6332</td>
<td>UCS 6332 Fan Module</td>
</tr>
<tr>
<td>QSFP-H40G-CU3M</td>
<td>40GBASE-CR4 Passive Copper Cable, 3m</td>
</tr>
<tr>
<td>Part Number</td>
<td>Description</td>
</tr>
<tr>
<td>-------------</td>
<td>---------------------------------------------------------</td>
</tr>
<tr>
<td>UCS-SP-H1P8TB-4X</td>
<td>UCS SP 1.8 TB 12G SAS 10K RPM SFF HDD (4K) 4Pk</td>
</tr>
<tr>
<td>UCS-SP-H1P8TB</td>
<td>1.8 TB 12G SAS 10K RPM SFF HDD (4K)</td>
</tr>
<tr>
<td>UCS-SP-HD-1P8T-2</td>
<td>1.8TB 12G SAS 10K RPM SFF HDD (4K) 2 Pack</td>
</tr>
<tr>
<td>UCS-SP-HD-1P8T</td>
<td>SP 1.8TB 12G SAS 10K RPM SFF HDD (4K)</td>
</tr>
</tbody>
</table>

Table 10  Bill of Materials for Hadoop Nodes Expansion Rack

<table>
<thead>
<tr>
<th>Part Number</th>
<th>Description</th>
<th>Qty</th>
</tr>
</thead>
<tbody>
<tr>
<td>UCS-SP-C240M5-A2</td>
<td>SP C240 M5SX w/2x6132,6x32GB mem,VIC1387</td>
<td>8</td>
</tr>
<tr>
<td>CON-OSP-C240M5A2</td>
<td>SNTC 24x7X4QOS UCS C240 M5 A2</td>
<td>8</td>
</tr>
<tr>
<td>UCS-CPU-6132</td>
<td>2.6 GHz 6132/140W 14C/19.25MB Cache/DDR4 2666MHz</td>
<td>16</td>
</tr>
<tr>
<td>UCS-MR-X32G2RS-H</td>
<td>32GB DDR4-2666-MHz RDIMM/PC4-21300/dual rank/x4/1.2v</td>
<td>48</td>
</tr>
<tr>
<td>UCSC-PCI-1-C240M5</td>
<td>Riser 1 including 3 PCIe slots (x8, x16, x8); slot 3 required CPU2</td>
<td>8</td>
</tr>
<tr>
<td>UCSC-MLOM-C40Q-03</td>
<td>Cisco VIC 1387 Dual Port 40Gb QSFP CNA MLOM</td>
<td>8</td>
</tr>
<tr>
<td>UCSC-PSU1-1600W</td>
<td>Cisco UCS 1600W AC Power Supply for Rack Server</td>
<td>16</td>
</tr>
<tr>
<td>CAB-9K12A-NA</td>
<td>Power Cord, 125VAC 13A NEMA 5-15 Plug, North America</td>
<td>16</td>
</tr>
<tr>
<td>UCSC-RAILB-M4</td>
<td>Ball Bearing Rail Kit for C220 &amp; C240 M4 &amp; M5 rack servers</td>
<td>8</td>
</tr>
<tr>
<td>CIMC-LATEST</td>
<td>IMC SW (Recommended) latest release for C-Series Servers.</td>
<td>8</td>
</tr>
<tr>
<td>UCSC-HS-C240M5</td>
<td>Heat sink for UCS C240 M5 rack servers 150W CPUs and below</td>
<td>16</td>
</tr>
<tr>
<td>UCSC-BBLKD-S2</td>
<td>UCS C-Series M5 SFF drive blanking panel</td>
<td>208</td>
</tr>
<tr>
<td>UCSC-PCIF-240M5</td>
<td>C240 M5 PCIe Riser Blanking Panel</td>
<td>8</td>
</tr>
<tr>
<td>CBL-SC-MR12GM5P</td>
<td>Super Cap cable for UCSC-RAID-M5JD</td>
<td>8</td>
</tr>
<tr>
<td>UCSC-SCAP-M5</td>
<td>Super Cap for UCSC-RAID-M5, UCSC-MRAID2GB-KIT</td>
<td>8</td>
</tr>
<tr>
<td>UCSC-RAID-M5HD</td>
<td>Cisco 12G Modular RAID controller with 4GB cache</td>
<td>8</td>
</tr>
<tr>
<td>UCS-SP-H1P8TB-4X</td>
<td>UCS SP 1.8 TB 12G SAS 10K RPM SFF HDD (4K) 4Pk</td>
<td>48</td>
</tr>
<tr>
<td>UCS-SP-H1P8TB</td>
<td>1.8 TB 12G SAS 10K RPM SFF HDD (4K)</td>
<td>192</td>
</tr>
<tr>
<td>UCS-SP-HD-1P8T-2</td>
<td>1.8TB 12G SAS 10K RPM SFF HDD (4K) 2 Pack</td>
<td>8</td>
</tr>
<tr>
<td>UCS-SP-HD-1P8T</td>
<td>SP 1.8TB 12G SAS 10K RPM SFF HDD (4K)</td>
<td>16</td>
</tr>
</tbody>
</table>

Table 11  Bill of Materials for CDSW Nodes Rack

<table>
<thead>
<tr>
<th>Part Number</th>
<th>Description</th>
<th>Qty</th>
</tr>
</thead>
<tbody>
<tr>
<td>UCS-SP-C240M5-A2</td>
<td>SP C240 M5SX w/2x6132,6x32GB mem,VIC1387</td>
<td>4</td>
</tr>
<tr>
<td>CON-OSP-C240M5A2</td>
<td>SNTC 24x7X4QOS UCS C240 M5 A2</td>
<td>4</td>
</tr>
<tr>
<td>UCS-CPU-6132</td>
<td>2.6 GHz 6132/140W 14C/19.25MB Cache/DDR4 2666MHz</td>
<td>8</td>
</tr>
<tr>
<td>UCS-MR-X32G2RS-H</td>
<td>32GB DDR4-2666-MHz RDIMM/PC4-21300/dual rank/x4/1.2v</td>
<td>24</td>
</tr>
<tr>
<td>UCSC-PCI-1-C240M5</td>
<td>Riser 1 including 3 PCIe slots (x8, x16, x8); slot 3 required CPU2</td>
<td>4</td>
</tr>
<tr>
<td>UCSC-MLOM-C40Q-03</td>
<td>Cisco VIC 1387 Dual Port 40Gb QSFP CNA MLOM</td>
<td>4</td>
</tr>
<tr>
<td>UCSC-PSU1-1600W</td>
<td>Cisco UCS 1600W AC Power Supply for Rack Server</td>
<td>8</td>
</tr>
<tr>
<td>CAB-9K12A-NA</td>
<td>Power Cord, 125VAC 13A NEMA 5-15 Plug, North America</td>
<td>8</td>
</tr>
<tr>
<td>UCSC-RAILB-M4</td>
<td>Ball Bearing Rail Kit for C220, C240 M4 and M5 rack servers</td>
<td>4</td>
</tr>
<tr>
<td>CIMC-LATEST</td>
<td>IMC SW (Recommended) latest release for C-Series Servers.</td>
<td>4</td>
</tr>
<tr>
<td>-------------</td>
<td>--------------------------------------------------------</td>
<td>---</td>
</tr>
<tr>
<td>UCSC-HS-C240M5</td>
<td>Heat sink for UCS C240 M5 rack servers 150W CPUs &amp; below</td>
<td>8</td>
</tr>
<tr>
<td>UCSC-BBLKDS2</td>
<td>UCS C-Series M5 SFF drive blanking panel</td>
<td>104</td>
</tr>
<tr>
<td>CBL-SC-MR22GM5P</td>
<td>Super Cap cable for UCSC-RAID-M5HD</td>
<td>4</td>
</tr>
<tr>
<td>UCSC-SCAP-M5</td>
<td>Super Cap for UCSC-RAID-M5, UCSC-MRAID1GB-KIT</td>
<td>4</td>
</tr>
<tr>
<td>UCSC-RAID-M5HD</td>
<td>Cisco 12G Modular RAID controller with 4GB cache</td>
<td>4</td>
</tr>
<tr>
<td>UCSC-PCI-2A-240M5</td>
<td>Riser 2A including 3 PCIe slots (x8, x16, x16) supports GPU</td>
<td>4</td>
</tr>
<tr>
<td>UCSC-SP-SD-1P6TB-4X</td>
<td>UCS SP 1.6TB 2.5 inch Ent. Perf 12G SAS SSD(10X endurance)4Pk</td>
<td>4</td>
</tr>
<tr>
<td>UCSC-SP-SD-1P6TB</td>
<td>1.6TB 2.5 inch Ent. Performance 12G SAS SSD(10X endurance)</td>
<td>16</td>
</tr>
<tr>
<td>UCSC-GPU-P100-16G=</td>
<td>NVIDIA P100 16GB</td>
<td>8</td>
</tr>
<tr>
<td>CON-OSP-UCSCLM16G</td>
<td>SNTC-24X7X4OS NVIDIA P100 16GB</td>
<td>8</td>
</tr>
<tr>
<td>UCS-P100CBL-240M5=</td>
<td>C240M5 NVIDIA P100 / V100 Cable</td>
<td>8</td>
</tr>
</tbody>
</table>

**Table 12 Red Hat Enterprise Linux License**

<table>
<thead>
<tr>
<th>Red Hat Enterprise Linux</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>RHEL-2S2V-3A</td>
<td>Red Hat Enterprise Linux</td>
</tr>
<tr>
<td>CON-IVS1-EL2S2V3A</td>
<td>3 year Support for Red Hat Enterprise Linux</td>
</tr>
</tbody>
</table>

**Table 13 Cloudera Data Science Work Bench Software**

<table>
<thead>
<tr>
<th>UCS-BD-CDSWB=</th>
<th>UCS-BD-CDSWB-1Y</th>
<th>Cloudera Data Science Work Bench, 10-user pack - 1 Year</th>
</tr>
</thead>
<tbody>
<tr>
<td>UCS-BD-CDSWB=</td>
<td>UCS-BD-CDSWB-2Y</td>
<td>Cloudera Data Science Work Bench, 10-user pack - 2 Year</td>
</tr>
<tr>
<td>UCS-BD-CDSWB=</td>
<td>UCS-BD-CDSWB-3Y</td>
<td>Cloudera Data Science Work Bench, 10-user pack - 3 Year</td>
</tr>
</tbody>
</table>

**Table 14 Cloudera SKU’s Available at Cisco**

<table>
<thead>
<tr>
<th>Cisco TOP SKU</th>
<th>Cisco PID with Duration</th>
<th>Product Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>UCS-BD-CEBN-BZ=</td>
<td>UCS-BD-CEBN-BZ-3Y</td>
<td>Cloudera Enterprise Basic Edition, Node License, Bronze Support - 3 Year</td>
</tr>
<tr>
<td>UCS-BD-CEBN-BZI=</td>
<td>UCS-BD-CEBN-BZI-3Y</td>
<td>Cloudera Enterprise Basic Edition + Indemnification, Node License, Bronze Support - 3 Year</td>
</tr>
<tr>
<td>UCS-BD-CEBN-GDI=</td>
<td>UCS-BD-CEBN-GDI-3Y</td>
<td>Cloudera Enterprise Basic Edition + Indemnification, Node License, Gold Support - 3 Year</td>
</tr>
<tr>
<td>Cisco TOP SKU</td>
<td>Cisco PID with Duration</td>
<td>Product Name</td>
</tr>
<tr>
<td>---------------------</td>
<td>-------------------------</td>
<td>------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
About the Authors


Karthik Kulkarni is an Architect with the Computing Systems Product Group. His focus includes Big Data and analytics system, next gen data center architecture and performance.

Manan Trivedi, Big Data Solutions Architect, Cisco Systems, Inc.

Manan Trivedi is a Big Data Solutions Architect at Computing Systems Product Group. He is part of the solution engineering team focusing on big data infrastructure, solutions, and performance.

Acknowledgements

The authors would like to thank the following for their support and contribution to the design, creating, and validation of this Cisco Validated Design:

- Rajesh Shroff, Big Data Solutions Architect, Data Center Solutions Group, Cisco Systems, Inc.
- Shane Handy, Big Data Solutions Engineer, Data Center Solutions Group, Cisco Systems, Inc.