Cisco Data Intelligence Platform with Hortonworks Data Platform 3.1 and Cloudera Data Science Workbench 1.5

Deployment Guide for the Cisco Data Intelligence Platform with Hortonworks Data Platform 3.1.0

Last Updated: October 6, 2019
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, go to:


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, 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)

© 2019 Cisco Systems, Inc. All rights reserved.
# Table of Contents

- Executive Summary ................................................................................................................. 8
- Solution Overview .................................................................................................................... 9
  - Cisco Data Intelligence Platform .......................................................................................... 9
  - Audience ............................................................................................................................... 11
  - Purpose of this Document .................................................................................................... 11
  - What’s New in this Release? ................................................................................................. 11
  - What’s Next? ........................................................................................................................ 11
  - Reference Architecture ........................................................................................................ 12
  - Data Lake Reference Architecture ...................................................................................... 12
  - AI Computing Farm Reference Architecture ........................................................................ 13
  - Scaling the Solution .............................................................................................................. 16
  - Scaled Architecture with 3:1 Oversubscription with Cisco Fabric Interconnects and Cisco ACI ................................................................................................................ 17
  - Scaled Architecture with 2:1 Oversubscription with Cisco ACI .......................................... 18
- Technology Overview ............................................................................................................... 20
  - Cisco UCS Integrated Infrastructure for Big Data and Analytics ........................................... 20
    - Cisco Unified Computing System ..................................................................................... 20
    - Cisco UCS 6300 Series Fabric Interconnects ..................................................................... 20
    - Cisco UCS C-Series Rack-Mount Servers ........................................................................ 20
    - Cisco UCS Virtual Interface Cards .................................................................................. 25
    - Cisco UCS Manager ......................................................................................................... 25
  - NVIDIA GPU ....................................................................................................................... 26
    - NVIDIA CUDA ................................................................................................................ 26
    - Cloudera Enterprise Data Hub and Hortonworks Data Platform ..................................... 26
    - Cloudera (CDH 6.2.0) .................................................................................................... 26
    - Cloudera Data Science Workbench .................................................................................... 27
  - Hortonworks Data Platform ................................................................................................. 29
    - Apache Ambari .................................................................................................................. 29
    - HDP for Data Access ......................................................................................................... 30
    - Submarine .......................................................................................................................... 31
    - Docker Containerization .................................................................................................... 32
    - YARN Support For Docker .............................................................................................. 32
    - NVIDIA Docker .............................................................................................................. 33
    - GPU Pooling and Isolation ............................................................................................... 33
    - Red Hat Ansible Automation ........................................................................................... 34
  - Solution Design ....................................................................................................................... 35
Choose Services ........................................................................................................................................... 125
Assign Masters ........................................................................................................................................... 126
Assign Slaves and Clients .......................................................................................................................... 127
Customize Services .................................................................................................................................. 128
HDFS .......................................................................................................................................................... 132
MapReduce2 .............................................................................................................................................. 134
YARN ......................................................................................................................................................... 134
Configure the HDFS NameNode High Availability .................................................................................... 135
HBase ........................................................................................................................................................ 136
Zookeeper .................................................................................................................................................. 136
Storm .......................................................................................................................................................... 136
Ambari Metrics ......................................................................................................................................... 136
Accumulo .................................................................................................................................................. 137
Atlas ........................................................................................................................................................... 137
Kafka .......................................................................................................................................................... 138
Knox ........................................................................................................................................................... 138
SmartSense ............................................................................................................................................... 138
Spark .......................................................................................................................................................... 139
Review ......................................................................................................................................................... 139
Deploy ......................................................................................................................................................... 140
Summary of the Installation Process ......................................................................................................... 141
High Availability for HDFS NameNode and YARN ResourceManager .................................................... 142
Configure the HDFS NameNode High Availability .................................................................................... 142
Configure the YARN ResourceManager HA ............................................................................................. 150
Bill of Materials ....................................................................................................................................... 155
Appendix – A ........................................................................................................................................... 160
Configure Data Drives on Name Node and Other Management Nodes .................................................... 160
Configure Data Drives on Data Nodes ........................................................................................................ 161
Cloudera Data Science Workbench (CDSW) .............................................................................................. 162
Install the Prerequisites for CUDA ........................................................................................................... 163
Install GCC ............................................................................................................................................... 164
Install Kernel Headers and Installation Packages ....................................................................................... 165
Install DKMS ........................................................................................................................................... 166
Install NVIDIA GPU Drivers ................................................................................................................... 167
Install CUDA .......................................................................................................................................... 168
Download and Setup NVIDIA CUDA Deep Neural Network Library (cuDNN) ........................................... 170
Installation Prerequisites for CDSW ........................................................................................................ 171
Executive Summary

Data scientists are constantly searching for newer techniques and methodologies that can unlock the value of big data and distill this data further to identify additional insights which could transform productivity and provide business differentiation.

One such area is Artificial Intelligence/Machine Learning (AI/ML), which has seen tremendous development with bringing in new frameworks and new forms of compute (CPU, GPU and FPGA) to work on data to provide key insights. While data lakes have historically been data intensive workloads, these advancements in technologies have led to a new growing demand of compute intensive workloads to operate on the same data.

While data scientists want to be able to use the latest and greatest advancements in AI/ML software and hardware technologies on their datasets, the IT team is also constantly looking at enabling these data scientists to be able to provide such a platform to a data lake. This has led to architecturally siloed implementations. When data, which is ingested, worked, and processed in a data lake, needs to be further operated by AI/ML frameworks, it often leaves the platform and has to be on-boarded to a different platform to be processed. This would be fine if this demand is seen only on a small percentage of workloads. However, AI/ML workloads working closely on the data in a data lake are seeing an increase in adoption. For instance, data lakes in customer environment are seeing deluge of data from new use cases such as IoT, autonomous driving, smart cities, genomics and financials, who are all seeing more and more demand of AI/ML processing of this data.

IT is demanding newer solutions to enable data scientists to operate on both a data lake and an AI/ML platform (or a compute farm) without worrying about the underlying infrastructure. IT also needs this to seamlessly grow to cloud scale while reducing the TCO of this infrastructure and without affecting utilization. Thus, driving a need to plan a data lake along with an AI/ML platform in a systemic fashion.

Seeing this increasing demand by IT, and also envisioning this as a natural extension of a data lake, we announced Cisco Data Intelligence Platform. Cisco Data Intelligence Platform is discussed in detail here.

This CVD implements Cisco Data Intelligence Platform on Cisco Unified Computing System (Cisco UCS) using Hortonworks Data Platform 3.1.
Cisco Data Intelligence Platform

Cisco Data Intelligence Platform (CDIP) is a cloud scale architecture which brings together big data, AI/compute farm, and storage tiers to work together as a single entity while also being able to scale independently to address the IT issues in the modern data center. This architecture allows for:

- Extremely fast data ingest, and data engineering done at the data lake
- AI compute farm allowing for different types of AI frameworks and compute types (GPU, CPU, FPGA) to work on this data for further analytics
- A storage tier, allowing to gradually retire data which has been worked on to a storage dense system with a lower $/TB providing a better TCO
- Seamlessly scale the architecture to thousands of nodes with a single pane of glass management using Cisco Application Centric Infrastructure (ACI)

Cisco Data Intelligence Platform caters to the evolving architecture bringing together a fully scalable infrastructure with centralized management and fully supported software stack (in partnership with industry leaders in the space) to each of these three independently scalable components of the architecture including data lake, AI/ML and Object stores.

Cisco has developed numerous industry leading Cisco Validated Designs (CVDs) in the area of Big Data (CVDs with Cloudera, Hortonworks, and MapR), compute farm with Kubernetes (CVD with RedHat OpenShift) and Object store (Scality, SwiftStack, Cloudian, and others).

This Cisco Data Intelligence Platform can be deployed in two variants:

- CDIP with Cloudera with Data Science Workbench (powered by Kubernetes) and Tiered Storage with Hadoop
- CDIP with Hortonworks with Apache Hadoop 3.1 and Data Science Workbench (powered by Kubernetes) and Tiered Storage with Hadoop
This architecture can start from a single rack and scale to thousands of nodes with a single pane of glass management with Cisco Application Centric Infrastructure (ACI).
Solution Overview

Audience
The intended audience of this document includes, but not limited to, sales engineers, field consultants, professional services, IT managers, partner engineering and customers who want to deploy Cisco Data Intelligence Platform using Hortonworks Data Platform (HDP 3.1.0) on Cisco UCS. You are assumed to have intermediate level of knowledge for Apache Hadoop and Cisco UCS based scale-out infrastructure.

Purpose of this Document
This document describes the architecture and step by step guidelines of deployment procedures for Cisco Data Intelligence Platform using Hortonworks Data Platform (HDP) 3.1.0 on Cisco UCS C240 M5.

This document walks through the process of deploying the three independently scalable components of the architecture including data lake, AI/ML and Object stores:

- Data Lake with Hortonworks Data Platform 3.1 or Cloudera Enterprise Data Hub 6.2
- Kubernetes / AI farm with Cloudera Data Science Workbench
- Hadoop Tiered storage with S3260
- Distributed AI/ML with Apache Submarine

What’s New in this Release?
This CVD describes the deployment procedure for the following:

- Data Lake with Cloudera Enterprise Data Hub 6.2.0 or Hortonworks Data Platform 3.1.0
- Kubernetes / AI farm with Cloudera Data Science Workbench (CDSW) 1.5
  - Enable CUDA for the GPUs
  - Enable GPU as a resource to the Docker Containers through CDSW
  - Enable GPU isolation and scheduling (with Docker Containers) through YARN 2.0
  - Downloading a TensorFlow image from NVIDIA Cloud (NGC)
  - Adding trusted registries for Docker for YARN 2.0
  - Execute a sample TensorFlow job accessing data from Hadoop and running on a Docker container with GPU as a resource scheduled by YARN 2.0
- Distributed Deep Learning with Submarine

What’s Next?
This CVD showcases Cisco UCS Manager (UCSM). This solution can also be deployed using Cisco Intersight. This along with other additional Cisco UCS features will be added to the appendix section in the following months. Some of these include,

- Cisco Intersight
- Cloudera Data Science Workbench
- Tiered Storage with HDFS on Cisco UCS S3260
Solution Overview

- Cisco Boot optimized M.2 Raid controller for hardware RAID
- 4th Generation Fabric Interconnect
- Hadoop data offload to S3 compliant storage

Reference Architecture

Table 1, Table 2, and Table 3 summarize the reference architecture configuration details for the data lake, AI/ML components of the data lake, and tiered storage.

Data Lake Reference Architecture

Table 1 lists the data lake reference architecture configuration details for Cisco UCS Integrated Infrastructure for Big Data and Analytics.

<table>
<thead>
<tr>
<th>Table 1</th>
<th>Cisco UCS Integrated Infrastructure for Big Data and Analytics Configuration Options</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Servers</strong></td>
<td><strong>Performance (UCS-SP-C240M5-A2)</strong></td>
</tr>
<tr>
<td><strong>CPU</strong></td>
<td>16 x Cisco UCS C240 M5 Rack Servers with SFF drives</td>
</tr>
<tr>
<td><strong>Memory</strong></td>
<td>2 x 2nd Gen Intel Xeon Processor Scalable Family 6230 (2 x 20 cores, 2.1 GHz)</td>
</tr>
<tr>
<td><strong>Storage</strong></td>
<td>12 x 32 GB 2933 MHz (384 GB)</td>
</tr>
<tr>
<td><strong>Boot</strong></td>
<td>M.2 with 2 x 240-GB SSDs</td>
</tr>
<tr>
<td><strong>Storage Controller</strong></td>
<td>26 x 2.4 TB 10K rpm SFF SAS HDDs or 12 x 1.6 TB Enterprise Value SATA SSDs</td>
</tr>
<tr>
<td><strong>Network Connectivity</strong></td>
<td>40 Gigabit Ethernet (Cisco UCS VIC 1387) or 25 Gigabit Ethernet (Cisco UCS VIC 1455)</td>
</tr>
<tr>
<td><strong>GPU (Optional)</strong></td>
<td>2 x NVIDIA TESLA V100 with 32G memory each</td>
</tr>
</tbody>
</table>
## AI Computing Farm Reference Architecture

Table 2 lists the AI computing farm reference architecture configuration details for high-density CPU cores and GPU nodes.

<table>
<thead>
<tr>
<th></th>
<th>Select stack</th>
<th>Elite stack</th>
<th>Premier stack</th>
</tr>
</thead>
<tbody>
<tr>
<td>Servers</td>
<td>8 x Cisco UCS C240 M5 Rack Servers</td>
<td>8 x Cisco UCS C240 M5 Rack Servers</td>
<td>8 x Cisco UCS C4200 Rack Servers Each with: 4 x Cisco UCS C425 M5 Rack Servers</td>
</tr>
<tr>
<td></td>
<td>4 x Cisco UCS C480 M5 Rack Servers</td>
<td>4 x Cisco UCS C480 M5 Rack Servers</td>
<td></td>
</tr>
<tr>
<td>CPU</td>
<td>2 x 2nd Gen Intel Xeon Scalable 6230 processors (2 x 20 cores, at 2.1 GHz)</td>
<td>2 x Intel Xeon Scalable 6230 processors (2 x 16 cores, at 2.6 GHz)</td>
<td>2 x AMD EPYC 7401 processors (2 x 24 cores, at 2.0 or 2.8 GHz)</td>
</tr>
<tr>
<td>Memory</td>
<td>24 x 32GB 2933 MHz DDR4 (384 GB)</td>
<td>12 x 32GB 2666MHz DDR4 (384 GB)</td>
<td>16 x 32GB 2666 MHz DDR4 (512 GB)</td>
</tr>
<tr>
<td>Boot</td>
<td>M.2 with 2 x 960-GB SSDs</td>
<td>M.2 with 2 x 960-GB SSDs</td>
<td>M.2 with 2 x 240-GB SATA 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>24 x 1.8-TB 10K rpm SFF SAS HDDs or 12 x 1.6-TB Enterprise Value SATA SSDs</td>
<td>6 x 3.8-TB Enterprise Value SATA SSDs</td>
</tr>
<tr>
<td>VIC</td>
<td>40 Gigabit Ethernet (Cisco UCS VIC 1387) or 25 Gigabit Ethernet (Cisco UCS VIC 1455)</td>
<td>40 Gigabit Ethernet (Cisco UCS VIC 1387) or 25 Gigabit Ethernet (Cisco UCS VIC 1455)</td>
<td>25 Gigabit Ethernet (Cisco UCS VIC 1455)</td>
</tr>
<tr>
<td>Storage controller</td>
<td>Cisco 12-Gbps SAS modular RAID controller with 4-GB FBWC or Cisco 12-Gbps modular SAS HBA</td>
<td>Cisco 12-Gbps SAS modular RAID controller with 4-GB FBWC or Cisco 12-Gbps modular SAS HBA</td>
<td>Cisco 12-Gbps SAS 9460-8i RAID controller with 2-GB FBWC</td>
</tr>
<tr>
<td>Network connectivity</td>
<td>Cisco UCS 6332 Fabric Interconnect or Cisco UCS 6454 Fabric Interconnect</td>
<td>Cisco UCS 6332 Fabric Interconnect or Cisco UCS 6454 Fabric Interconnect</td>
<td>Cisco UCS 6454 Fabric Interconnect</td>
</tr>
<tr>
<td>GPU</td>
<td>For C240 M5: 2 x NVIDIA TESLA V100 with 32-GB memory each or 2 x NVIDIA T4</td>
<td>For C240 M5: 2 x NVIDIA TESLA V100 with 32-GB memory each or 2 x NVIDIA T4</td>
<td>For C240 M5: 2 x NVIDIA TESLA V100 with 32-GB memory each or 2 x NVIDIA T4</td>
</tr>
<tr>
<td></td>
<td>For C480 M5: 4 x NVIDIA TESLA V100 with 32-GB memory each or 4 x NVIDIA T4</td>
<td>For C480 M5 ML: 8 x NVIDIA TESLA V100 with 32-GB memory each and with NVLink</td>
<td>For C480 M5 ML: 8 x NVIDIA TESLA V100 with 32-GB memory each and with NVLink</td>
</tr>
</tbody>
</table>
High density GPU servers have higher storage for OS M.2 drives for docker volumes on the OS drives.

Table 3 lists the tiered storage reference architecture configuration details for Cisco UCS Integrated Infrastructure for Big Data and Analytics.

Table 3  Tiered Storage (Data Lake Reference Architecture)

<table>
<thead>
<tr>
<th>High capacity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Servers</td>
</tr>
<tr>
<td>CPU</td>
</tr>
<tr>
<td>Memory</td>
</tr>
<tr>
<td>Boot</td>
</tr>
<tr>
<td>Storage</td>
</tr>
<tr>
<td>VIC</td>
</tr>
<tr>
<td>Storage controller</td>
</tr>
<tr>
<td>Network connectivity</td>
</tr>
</tbody>
</table>

Figure 3 illustrates a 12-node starter cluster with all the 3 components in a single rack. The top 8 nodes have Cisco UCS C240 M5 servers as a data lake. Each link in the figure represents a 40 Gigabit Ethernet link from each of the 12 servers directly connected to a Fabric Interconnect. The second 2 x Cisco UCS C480 ML M5 Servers and the last 4 servers illustrate a data tiering on 2xS3260 servers. Every server is connected to both Fabric Interconnects.
As illustrated in Figure 3, a 30-node starter cluster. Rack #1 has sixteen Cisco UCS C240 M5 servers. Each link in the figure represents a 40 Gigabit Ethernet link from each of the sixteen servers directly connected to a Fabric Interconnect. Rack #2 has six Cisco UCS C240 M5 and four Cisco UCS S3260 servers. Every server is connected to both Fabric Interconnects.
Solution Overview

Figure 4 shows an alternate configuration for cases where more GPU capacity is needed. Four of the Cisco UCS C240 M5 servers from the previous configuration in Figure 3 are replaced with Cisco UCS C480 M5 ML M5 server which support up to eight V100 MXM GPUs.

Each Cisco UCS C480 ML M5 has 8 x NVIDIA SXM2 V100 32GB modules with NVLink interconnect. Each Cisco UCS C240 M5 supports up to two PCIe GPU adapters with NVIDIA Tesla V100. For more information about Cisco UCS C240 M5 Sever installation and GPU card configuration rules, go to https://www.cisco.com/c/en/us/td/docs/unified_computing/ucs/c/hw/C240M5/install/C240M5/C240M5_appendix_0101.html

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

Scaling the Solution

Figure 5 illustrates 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 FIs 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.
In the reference architectures discussed in this document, each of the components is scaled separately, and for the purposes of this example, scaling is uniform. Two scale scenarios are discussed here:

- Scaled architecture with 3:1 oversubscription with Cisco fabric interconnects and Cisco ACI
- Scaled architecture with 2:1 oversubscription with Cisco ACI

In the following scenarios, the goal is to populate up to a maximum of 200 leaf nodes in a Cisco ACI domain. Not all cases reach that number because they use the Cisco Nexus® 9508 Switch for this sizing and not the Cisco Nexus 9516 Switch.

**Scaled Architecture with 3:1 Oversubscription with Cisco Fabric Interconnects and Cisco ACI**

The architecture discussed in this document and shown in Figure 6 supports 3:1 network oversubscription from every node to every other node across a multidomain cluster (nodes in a single domain within a pair of Cisco fabric interconnects are locally switched and not oversubscribed).

From the viewpoint of the data lake, 24 Cisco UCS C240 M5 Rack Servers are connected to a pair of Cisco UCS 6332 Fabric Interconnects (with 32 x 40-Gbps throughput). From each fabric interconnect, 8 x 40-Gbps links connect to a pair of Cisco Nexus 9336 Switches. Two pairs of fabric interconnects can connect to a single pair of Cisco Nexus 9336 Switches (8 x 2 40-Gbps links). Each of these Cisco Nexus 9336 Switches connects to a pair of Cisco Nexus 9508 Cisco ACI switches with 6 x 100-Gbps uplinks (connecting to a Cisco NgK-X9736C-FX line card).
Scaled Architecture with 3:1 Oversubscription with Cisco Fabric Interconnects and Cisco ACI

In the scenario discussed here and shown in Figure 7, the Cisco Nexus 9508 Switch with the Cisco N9K-X9736C-FX line card can support up to 36 x 100-Gbps ports, each and 8 such line cards.

For the 2:1 oversubscription, 30 Cisco UCS C240 M5 Rack Servers are connected to a pair of Cisco Nexus 9336 Switches, and each Cisco Nexus 9336 connects to a pair of Cisco Nexus 9508 Switches with three uplinks each. A pair of Cisco Nexus 9336 Switches can support 30 servers and connect to a spine with 6 x 100-Gbps links on each spine. This single pod (pair of Cisco Nexus 9336 Switches connecting to 30 Cisco UCS C240 M5 servers and 6 uplinks to each spine) can be repeated 48 times (288/6) for a given Cisco Nexus 9508 Switch and can support up to 1440 servers.

To reduce the oversubscription ratio (to get 1:1 network subscription from any node to any node), you can use just 15 servers under a pair of Cisco Nexus 9336 Switches and then move to Cisco Nexus 9516 Switches (the number of leaf nodes would double).

To scale beyond this number, multiple spines can be aggregated.
Figure 7 Scaled Architecture with 2:1 Oversubscription with Cisco ACI

- Single pair of spine(9508) - Total Ports = 288
- Total pairs of leaves(9336)/Spine = 288/8 = 48
  - 30 x Server connections/leaf
  - 6 x Spine connections/leaf
- 30 Servers/Pair of leaf
- Total servers x 48 pair of leaves = 1440 Servers

Big Data Cluster

AI Compute Farm

Data Anywhere/Tiered Storage

Cisco Nexus 9508 - A
Cisco Nexus 9508 - B

100 Gbps
40 Gbps
10 Gbps
Cisco UCS Integrated Infrastructure for Big Data and Analytics

The Cisco UCS Integrated Infrastructure for Big Data and Analytics solution for Hortonworks Data Platform on Cisco UCS Integrated Infrastructure for Big Data and Analytics, is 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 Unified Computing System

Cisco Unified Computing System (Cisco UCS) is a next-generation solution for blade and rack server computing. Cisco UCS integrates a low-latency, lossless 10 and 40 Gigabit Ethernet unified network fabric with enterprise-class, x86-architecture servers. Cisco UCS is an integrated, scalable, multi-chassis platform in which all resources participate in a unified management domain. Cisco UCS accelerates the delivery of new services simply, reliably, and securely through end-to-end provisioning and migration support for both virtualized and non-virtualized systems. Cisco UCS fuses access layer networking and servers. This high-performance, next-generation server system provides a data center with a high degree of workload agility and scalability.

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 (UCSM). 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 8 Cisco UCS 6332 UP 32 -Port Fabric Interconnect

Cisco UCS C-Series Rack-Mount Servers

Cisco UCS C-Series Rack-Mount Servers keep pace with Intel Xeon processor innovation by offering the latest processors with increased processor frequency and improved security and availability features. With the increased performance provided by the Intel Xeon Scalable Family Processors, Cisco UCS C-Series servers offer an improved price-to-performance ratio. They also extend Cisco UCS innovations to an industry-standard rack-mount form factor, including a standards-based unified network fabric, Cisco VN-Link virtualization support, and Cisco Extended Memory Technology.

It is designed to operate both in standalone environments and as part of Cisco UCS managed configuration, these servers enable organizations to deploy systems incrementally—using as many or as few servers as needed—on a schedule that best meets the organization’s timing and budget. Cisco UCS C-Series servers offer investment protection through the capability
to deploy them either as standalone servers or as part of Cisco UCS. One compelling reason that many organizations prefer rack-mount servers is the wide range of I/O options available in the form of PCIe adapters. Cisco UCS C-Series servers support a broad range of I/O options, including interfaces supported by Cisco and adapters from third parties.

Cisco UCS C240 M5 Rack-Mount Server

The Cisco UCS C240 M5 Rack-Mount Server (Figure 9) 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 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 2nd Gen 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 9 Cisco UCS C240 M5 Rack-Mount Server – Front View
Cisco UCS C480 M5 Rack-Mount Server

The Cisco UCS C480 M5 Rack-Mount Server is a storage and I/O-optimized enterprise-class rack-mount 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 11) delivers outstanding levels of expandability and performance for standalone or Cisco Unified Computing System managed environments in a 4RU form-factor. 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
- 2933-MHz DDR4 memory and 48 DIMM slots for up to 6 Terabytes (TB) of total memory
- 12 PCI Express (PCIe) 3.0 slots
  - Six x 8 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
Technology Overview

Figure 12 Cisco UCS C480 M5 Rack-Mount Server – Rear View

For more information about Cisco UCS C480 M5 Rack Server, go to:

Cisco UCS C480 ML M5 Rack Server
The Cisco UCS C480 ML M5 Rack Server is a purpose-built server for Deep Learning. It is storage and I/O optimized to deliver an industry-leading performance for training Models. The Cisco UCS C480 ML M5 delivers outstanding levels of storage expandability and performance options for standalone or Cisco Unified Computing System managed environments in a 4RU form factor. Because of its modular design, you pay for only what you need. It offers these capabilities:

- 8 NVIDIA SXM2 V100 32G modules with NVLink interconnect
- Latest Intel Xeon Scalable processors with up to 28 cores per socket and support for two processor configurations
- 2666-MHz DDR4 memory and 24 DIMM slots for up to 3 terabytes (TB) of total memory
- 4 PCI Express (PCIe) 3.0 slots for 100G UCS VIC 1495
- Flexible storage options with support for up to 24 Small-Form-Factor (SFF) 2.5-inch, SAS/SATA Solid-State Disks (SSDs) and Hard-Disk Drives (HDDs)
- Up to 6 PCIe NVMe disk drives
- Cisco 12-Gbps SAS Modular RAID Controller in a dedicated slot
- M.2 boot options
- Dual embedded 10 Gigabit Ethernet LAN-On-Motherboard (LOM) ports
For more information about Cisco UCS C480 ML M5 Server, go to:

Table 4 lists the features and benefits of Cisco UCS C480 ML M5 Server.

<table>
<thead>
<tr>
<th>Feature</th>
<th>Benefits</th>
</tr>
</thead>
<tbody>
<tr>
<td>8 x NVIDIA SXM2 V100 32GB modules with NVLink interconnect</td>
<td>Fast Deep Learning model training</td>
</tr>
<tr>
<td>Modular storage support with up to 24 front accessible hot-swappable Hard Disk Drives (HDDs) and Solid-State Disks (SSDs)</td>
<td>Modularity to right-size storage options to match training requirements Flexibility to expand as storage needs increase</td>
</tr>
<tr>
<td>High-capacity memory support of up to 3 TB using 128-GB DIMMs</td>
<td>Large memory footprint to deliver performance and capacity for large model training</td>
</tr>
<tr>
<td>Up to 6 PCIe NVMe drives</td>
<td>Up to 6 Gen3 x4 lanes NVMe drives for extreme I/O performance for faster model training</td>
</tr>
<tr>
<td>Support for up to 4 PCIe Generation 3.0 slots</td>
<td>Support for up to four 10/25 or 40/100G Cisco VICs</td>
</tr>
<tr>
<td>Hot-swappable, redundant power supplies</td>
<td>Increased high availability</td>
</tr>
</tbody>
</table>
Technology Overview

<table>
<thead>
<tr>
<th>Feature</th>
<th>Benefits</th>
</tr>
</thead>
<tbody>
<tr>
<td>Integrated dual 10-Gbps Ethernet</td>
<td>Increased network I/O performance and additional network options</td>
</tr>
</tbody>
</table>

Cisco UCS Virtual Interface Cards

Cisco UCS Virtual Interface Cards (VICs) 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 VIC 1387](image)


Cisco UCS Manager

Cisco UCS Manager (UCSM) 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 GUI, a CLI, or an XML 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 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 besides rendering graphics.

NVIDIA GPU used in this document, NVIDIA Tesla V100, is an advanced data center GPU built to accelerate AI, HPC, and graphics. It is 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.

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 Enterprise Data Hub and Hortonworks Data Platform

This CVD can be implemented with Cloudera Enterprise Data Hub and also with Hortonworks Data Platform.

Cloudera (CDH 6.2.0)

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 copyrighted, and an open platform to connect with all of the existing investments in technology and skills.
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, and so on. Additionally, CDSW enables access to available GPU resources for deep learning workloads 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, CDSW 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.

Additionally, 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 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; for example, 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.

Data Science Workbench provides the following features:

- **CPU and GPU as a resource:** Data Science Workbench provides basic support for the use of existing general-purpose CPUs for each stage of the workflow and, optionally, accelerates the math-intensive steps with the selective application of special-purpose GPUs all through a Docker container, with Kubernetes scheduling these resources in the back end.

- **Self-service portal:** The Data Science Workbench web user interface console provides a self-service portal for data scientists to create an environment for their workloads (Figure 17). Currently, R, Python, and Scala are supported.

- **Jupyter Notebook:** Most data scientists use Jupyter Notebooks for AI/ML analysis and development. Data Science Workbench provides a Jupyter Notebook environment when data scientists create a portal, and these notebooks can be shared or worked in a collaborative manner.
Hortonworks Data Platform

The Hortonworks Data Platform (HDP 3.1.0) delivers essential capabilities in a completely open, integrated and tested platform that is ready for enterprise usage. With Hadoop YARN at its core, HDP provides flexible enterprise data processing across a range of data processing engines, paired with comprehensive enterprise capabilities for governance, security and operations.

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

Hortonworks Data Platform (HDP) 3.0 delivers significant new features, including the ability to launch apps in a matter of minutes and address new use cases for high-performance deep learning and machine learning apps. In addition, this new version of HDP enables enterprises to gain value from their data faster, smarter, in a hybrid environment.

Apache Ambari

Apache Ambari is a completely open source management platform. It performs provisioning, managing, securing, and monitoring Apache Hadoop clusters. Apache Ambari is a part of Hortonworks Data Platform and it allows enterprises to plan and deploy HDP cluster. It also provides ongoing cluster maintenance and management.

Ambari provides an intuitive Web UI as well as an extensive REST API framework which is very useful for automating cluster operations.

The following are the core benefits that Hadoop operators get with Ambari:

- Simplified Installation, Configuration and Management. Easily and efficiently create, manage and monitor clusters at scale. Takes the guesswork out of configuration with Smart Configs and Cluster Recommendations. Enables repeatable, automated cluster creation with Ambari Blueprints.
- Centralized Security Setup. Reduce the complexity to administer and configure cluster security across the entire platform. Helps automate the setup and configuration of advanced cluster security capabilities such as Kerberos and Apache Ranger.

- Full Visibility into Cluster Health. Ensure your cluster is healthy and available with a holistic approach to monitoring. Configures predefined alerts — based on operational best practices — for cluster monitoring. Captures and visualizes critical operational metrics — using Grafana — for analysis and troubleshooting. Integrated with Hortonworks SmartSense for proactive issue prevention and resolution.

- Highly Extensible and Customizable. Fit Hadoop seamlessly into your enterprise environment. Highly extensible with Ambari Stacks for bringing custom services under management, and with Ambari Views for customizing the Ambari Web UI.

HDP for Data Access

With YARN at its foundation, HDP provides a range of processing engines that allow users to interact with data in multiple and parallel ways, without the need to stand up individual clusters for each data set/application. Some applications require batch while others require interactive SQL or low-latency access with NoSQL. Other applications require search, streaming or in-memory analytics. Apache Solr, Storm and Spark fulfill those needs respectively.

To function as a true data platform, the YARN-based architecture of HDP enables the widest possible range of access methods to coexist within the same cluster avoiding unnecessary and costly data silos.

As shown in Figure 18, HDP Enterprise natively provides for the following data access types:

- Batch – Apache MapReduce has served as the default Hadoop processing engine for years. It is tested and relied upon by many existing applications.

- Interactive SQL Query - Apache Hive is the de facto standard for SQL interactions at petabyte scale within Hadoop. Hive delivers interactive and batch SQL querying across the broadest set of SQL semantics.

- Search - HDP integrates Apache Solr to provide high-speed indexing and sub-second search times across all your HDFS data.

- Scripting - Apache Pig is a scripting language for Hadoop that can run on MapReduce or Apache Tez, allowing you to aggregate, join and sort data.

- Low-latency access via NoSQL - Apache HBase provides extremely fast access to data as a columnar format, NoSQL database. Apache Accumulo also provides high-performance storage and retrieval, but with fine-grained access control to the data.

- Streaming - Apache Storm processes streams of data in real time and can analyze and take action on data as it flows into HDFS.
Deep learning is useful for enterprises tasks in the field of speech recognition, image classification, AI chatbots, machine translation, just to name a few. In order to train deep learning/machine learning models, frameworks such as TensorFlow / MXNet / PyTorch / Caffe / XGBoost can be leveraged. And sometimes these frameworks are used together to solve different problems.

To make distributed deep learning/machine learning applications easily launched, managed and monitored, Hadoop community initiated the Submarine project along with other improvements such as first-class GPU support, Docker container support, container-DNS support, scheduling improvements, and so on.

These improvements make distributed deep learning/machine learning applications run on Apache Hadoop YARN as simple as running it locally, which can let machine-learning engineers focus on algorithms instead of worrying about underlying infrastructure. By upgrading to latest Hadoop, users can now run deep learning workloads with other ETL/streaming jobs running on the same cluster. This can achieve easy access to data on the same cluster and achieve better resource utilization.
Docker Containerization

Hortonworks Data Platform (HDP 3.0) 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 that performs operating-system-level virtualization, also known as "containerization." 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 it 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.

YARN Support For Docker

Containerization provides YARN support for Docker containers, which makes it easier to bundle libraries and dependencies along with their application, allowing third-party applications to run on Apache Hadoop (for example, containerized applications), enabling:

- Faster time to deployment by enabling third-party apps.
- The ability to run multiple versions of an application, enabling users to rapidly create features by developing and testing new versions of services without disrupting old ones.
- Improved resource utilization and increased task throughput for containers, yielding faster time to market for services.
- Orchestration of stateless distributed applications.
- Packaging libraries for Spark application, eliminating the need for operations to deploy those libraries cluster wide.
As shown in Figure 20, YARN Services Framework in addition with Docker containerization, it is now possible to run both existing Hadoop frameworks, such as Hive, Spark, etc., and new containerized workloads on the same underlying infrastructure. Apache Hadoop 3.1 further improved these capabilities to enable advanced use cases such as TensorFlow and HBase.

**NVIDIA Docker**

Docker containers are platform-agnostic, but also hardware-agnostic. This presents a problem when using specialized hardware such as NVIDIA GPUs which require kernel modules and user-level libraries to operate. As a result, Docker does not natively support NVIDIA GPUs within containers.

One of the early workarounds to this problem was to fully install the NVIDIA drivers inside the container and map in the character devices corresponding to the NVIDIA GPUs (for example, /dev/nvidia0) on launch. This solution is brittle because the version of the host driver must exactly match the version of the driver installed in the container. This requirement drastically reduced the portability of these early containers, undermining one of Docker’s more important features.

To enable portability in Docker images that leverage NVIDIA GPUs, NVIDIA developed nvidia-docker, an open-source project hosted on GitHub that provides the two critical components needed for portable GPU-based containers:

- driver-agnostic CUDA images; and a Docker command line wrapper that mounts the user mode components of the driver and the GPUs (character devices) into the container at launch.
- nvidia-docker is essentially a wrapper around the docker command that transparently provisions a container with the necessary components to execute code on the GPU.

---

As of the publishing of this CVD, Hortonworks only supports nvidia-docker version 1.

**GPU Pooling and Isolation**

GPU pooling and isolation allows GPU to be a first-class resource type in Hadoop, making it easier for customers to run machine learning and deep learning workloads.
Technology Overview

- Compute-intensive analytics require not only a large compute pool, but also a fast and expensive processing pool with GPUs in tandem
- Customers can share cluster-wide GPU resources without having to dedicate a GPU node to a single tenant or workload
- GPU isolation dedicates a GPU to an application so that no other application has access to that GPU

When it comes to resource scheduling, it is important to recognize GPU as a resource. YARN extends the resource model to more flexible mode which makes it easier to add new countable resource-types. When GPU is added as resource type, YARN can schedule applications on GPU machines. Furthermore, by specifying the number of requested GPU to containers, YARN can find machines with available GPUs to satisfy container requests.

When GPU scheduling is enabled, YARN can schedule non-GPU applications such as LLAP, Tez, and etc. to servers without GPU. Moreover, YARN can allocate GPU applications such as TensorFlow, Caffe, MXNet, and so on, to servers with GPU.

Red Hat Ansible Automation

Red Hat Ansible Automation is a powerful IT automation tool. It is capable of provisioning numerous types of resources and deploying applications. It can configure and manage devices and operating system components. Due to its simplicity, extensibility, and portability, this solution extensively utilizes Ansible for performing repetitive deployment steps across the nodes.

Solution Design

Requirements

This CVD describes the architecture and deployment procedures for Hortonworks Data Platform (HDP) 3.1.0 on a 31 node cluster based on Cisco UCS Integrated Infrastructure for Big Data and Analytics. The solution goes into detail configuring HDP 3.1.0 on the Cisco UCS Integrated infrastructure for Big Data. In addition, it also details the configuration for Hortonworks Dataflow for various use cases.

The cluster configuration consists of the following:

- 2 Cisco UCS 6332UP Fabric Interconnects
- 22 Cisco UCS C240 M5 Rack-Mount servers
- 8 Cisco UCS C3260 M5 Storage Server
- 12 NVIDIA T4 GPUs
- 2 Cisco R42610 standard racks
- 4 Vertical Power distribution units (PDUs) (Country Specific) per rack

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 6 Cisco UCS C240 M5 Servers and 4 Cisco UCS S3260 Modular Storage Server Chassis connected to each of the vertical PDUs for redundancy; thereby, ensuring availability during power source failure, similar to the first rack.

Port Configuration on Fabric Interconnect

Table 5 lists the port configuration on Cisco UCS FI 6332 Fabric Interconnect.

<table>
<thead>
<tr>
<th>Port Type</th>
<th>Port Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>Server</td>
<td>1-26</td>
</tr>
<tr>
<td>Network</td>
<td>29-32</td>
</tr>
</tbody>
</table>

Please contact your Cisco representative for country specific information.

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 1387 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.

Figure 21 illustrates the port connectivity between the Fabric Interconnect, and Cisco UCS C240 M5 server. Sixteen Cisco UCS C240 M5 servers are used in Master rack configurations.
Figure 21  Cisco UCS C240 M5 and 6300 Series Fabric Interconnect Port Connectivity

For information about physical connectivity and single-wire management, go to:


For more information about physical connectivity illustrations and cluster setup, go to:


Software Distributions and Versions

The software distributions required versions are listed below.

**Hortonworks Data Platform (HDP 3.1.0)**

The Hortonworks Data Platform supported is HDP 3.1.0. For more information, go to: http://www.hortonworks.com.

**Red Hat Enterprise Linux (RHEL)**

The operating system supported is Red Hat Enterprise Linux 7.6. For more information, go to: http://www.redhat.com.

Software Versions

The software versions tested and validated in this document are shown in Table 6.

<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>C240M5.4.0.2a</td>
</tr>
<tr>
<td></td>
<td>Cisco UCS C480 ML M5</td>
<td></td>
</tr>
<tr>
<td>Network</td>
<td>Cisco UCS 6332</td>
<td>UCS 4.0(4b)</td>
</tr>
<tr>
<td></td>
<td>Cisco UCS VIC1387 Firmware</td>
<td>4.3(2a)</td>
</tr>
</tbody>
</table>
The latest drivers can be downloaded from this link: https://software.cisco.com/download/home/283862063/type/283853158/release/3.1%283%29.

The latest supported RAID controller driver is already included with the RHEL 7.6 operating system.

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 the details to configure a fully redundant, highly available Cisco UCS 6332 fabric configuration. The following is the high-level workflow to setup Cisco UCS:

- 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 and uplink ports
- Start discovery process
- Create pools and polices for service profile template
Solution Design

- Create Service Profile template
- Create service profile for each server from service profile template
- Associate Service Profiles to servers

Perform 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

To configure Fabric Interconnect A, follow these steps:

1. Connect to the console port on the first Cisco UCS 6332 Fabric Interconnect.
   
   At the prompt to enter the configuration method, enter console to continue.
   
   If asked to either perform a new setup or restore from backup, enter Setup to continue.
   
   Enter y to continue to set up a new Fabric Interconnect.
   
   Enter y to enforce strong passwords.

2. Enter the password for the admin user.

3. Enter the same password again to confirm the password for the admin user.

4. Enter the cluster name for the system name.

5. Enter the Mgmt0 IPv4 address.

6. Enter the Mgmt0 IPv4 netmask.

7. Enter the IPv4 address of the default gateway.

8. Enter the cluster IPv4 address.

9. To configure DNS, answer y.

10. Enter the DNS IPv4 address.

   Answer y to set up the default domain name.

11. Enter the default domain name.

   Review the settings that were printed to the console, and if they are correct, answer yes to save the configuration.

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

Configure Fabric Interconnect B

To configure Fabric Interconnect B, follow these steps:
1. Connect to the console port on the second Cisco UCS 6332 Fabric Interconnect.

When prompted to enter the configuration method, enter console to continue. The installer detects the presence of the partner Fabric Interconnect and adds this fabric interconnect to the cluster. Enter y to continue the installation.

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

3. Enter the Mgmt0 IPv4 address.

4. Answer yes to save the configuration.

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

For more information about configuring Cisco UCS 6332 Series Fabric Interconnect, go to:


Log Into Cisco UCS Manager

To log into Cisco UCS Manager, follow these 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.

Upgrade Cisco UCS Manager Software to Version 4.0(4b)

This document assumes the use of UCS 4.0(4b). Refer to the Cisco UCS 4.0 Release (upgrade Cisco UCS Manager software and Cisco UCS 6332 Fabric Interconnect software to version 4.0(4b). Also, make sure the Cisco UCS C-Series version 4.0(4b) software bundles are installed on the Fabric Interconnects.

![Warning](image)

Upgrading Cisco UCS firmware is beyond the scope of this document. However for complete Cisco UCS Install and Upgrade Guides, go to: https://www.cisco.com/c/en/us/support/servers-unified-computing/ucs-manager/products-installation-guides-list.html

Add a Block of IP Addresses for KVM Access

To create a block of KVM IP addresses for server access in the Cisco UCS environment, follow these 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.

**Figure 22** Adding a Block of IPv4 Addresses for KVM Access Part 1

5. Enter the starting IP address of the block and number of IPs needed, as well as the subnet and gateway information.

**Figure 23** Adding Block of IPv4 Addresses for KVM Access Part 2

6. **Click OK** to create the IP block.

7. **Click OK** in the message box.

**Enable Uplink Ports**

To enable uplinks ports, follow these 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.
Configure VLANs

VLANs are configured as shown in Table 7.

<table>
<thead>
<tr>
<th>VLAN</th>
<th>NIC Port</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>VLAN13</td>
<td>eth0</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, follow these 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>.

7. Select Common/Global for vlan16.

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.
Enable Server Ports

To enable server ports, follow these steps:

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

2. **Select** Equipment > Fabric Interconnects > Fabric Interconnect A (primary) > Fixed Module.

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.

Figure 29  Enabling Server Ports

After the Server Discovery, Port 29-32 will be a Network Port and 1-28 will be Server Ports.
Create Pools for Service Profile Templates

Create 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, follow these 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.
Create MAC Address Pools

To create MAC address pools, follow these 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.

Create 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, follow these steps:

1. Select the Servers tab in the left pane in the Cisco 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.

9. Click Finish.

10. Click OK and then click Finish.

Create Policies for Service Profile Templates

Create 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, follow these steps:

1. **Select the Servers tab in the left pane in the Cisco 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.**

### Create QoS Policies

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

#### Platinum Policy

To create the Platinum policy, follow these steps:

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 the **Host Control** radio button set to default (none).
10. When the pop-up window appears, click **OK** to complete the creation of the Policy.
Set Jumbo Frames

To set Jumbo frames and enable QoS, follow these 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.
Create the Local Disk Configuration Policy

To create local disk configuration in the Cisco UCS Manager GUI, follow these 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.

Create the 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.
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, follow these 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.

<table>
<thead>
<tr>
<th>BIOS Setting</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>MMU</td>
<td>Platform Default</td>
</tr>
<tr>
<td>CPU Hardware Power Management</td>
<td>Platform Default</td>
</tr>
<tr>
<td>Boot Performance Mode</td>
<td>Platform Default</td>
</tr>
<tr>
<td>CPU Performance</td>
<td>Enterprise</td>
</tr>
<tr>
<td>Core Multi-Processing</td>
<td>All</td>
</tr>
<tr>
<td>DRAM Clock/Threading</td>
<td>Performance</td>
</tr>
<tr>
<td>Direct Cache Access</td>
<td>Enabled</td>
</tr>
<tr>
<td>Energy Performance Tuning</td>
<td>Platform Default</td>
</tr>
<tr>
<td>Enhanced Intel SpeedStep Tech</td>
<td>Disable</td>
</tr>
<tr>
<td>Execute Disable Bit</td>
<td>Platform Default</td>
</tr>
</tbody>
</table>

![Cisco UCS Manager GUI](image-url)
Create the Boot Policy

To create boot policies within the Cisco UCS Manager GUI, follow these steps:

1. Select the Servers tab in the left pane in the Cisco 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.
Create the Power Control Policy

To create Power Control policies within the Cisco UCS Manager GUI, follow these 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.
Create the Service Profile Template

To create the Service Profile Template, follow these 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, follow these 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.
Configure the Storage Provisioning for the Template

To configure storage policies, follow these 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.

Configure Network Settings for the Template

To configure the network settings for the template, follow these steps:

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.
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.

**Configure the vMedia Policy for the Template**

To configure the vMedia policy for the template, follow these steps:

1. Click Next once the vMedia Policy window appears to go to the next section.
Configure the Server Boot Order for the Template

To set the boot order for the servers, follow these 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.
Configure the Server Assignment for the Template

To assign the servers to the pool, in the Server Assignment window, follow these steps:

1. **Select ucs** for the Pool Assignment field.
2. **Select the power state to be Up**.
3. **Keep the Server Pool Qualification field set to <not set>**.
4. **Check the Restrict Migration check box**.
5. **Select ucs in Host Firmware Package**.
Configure the Operational Policies for the Template

To configure the operational policies for the template, in the Operational Policies Window, follow these 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.

![Create Service Profiles From Template window](image)

Association of the Service Profiles will take place automatically.

Install Red Hat Enterprise Linux 7.6

This section provides detailed procedures to install Red Hat Enterprise Linux 7.6 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 the KVM console and virtual media from Cisco UCS Manager.

This installation requires RHEL 7.6 DVD/ISO.

To install the Red Hat Linux 7.6 operating system, follow these steps:

1. Log into 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 OK on the 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 **Choose File** in the Virtual Disk Management windows.

12. The Red Hat Enterprise Linux 7.6 DVD is assumed to be on the client machine.

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

14. Click **Map Drive** after selecting the .iso file.
14. In the KVM window, select the KVM tab to monitor during boot.

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

16. Click OK.

17. Click OK to reboot the system.

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

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

19. On reboot, the machine detects the presence of the Red Hat Enterprise Linux Server 7.6 install media.

21. Skip the Media test and start the installation. Select language of installation and click Continue.
22. Select Date and time, which pops up another window as shown below:
23. Select the location on the map, set the time, and click Done.
24. Click INSTALLATION DESTINATION.

25. This opens a new window with the boot disks. Make the selection and choose I will configure partitioning. Click Done.

26. This opens the new window for creating the partitions. Click the + sign to add a new partition as shown below, boot partition of size 2048 MB.
27. Click Add Mount Point to add the partition.

28. Change the Device type to RAID and make sure the RAID Level is RAID1 (Redundancy) and click Update Settings to save the changes.

29. Click the + sign to create the swap partition of size 2048 MB as shown below.
30. Change the Device type to RAID and RAID level to RAID1 (Redundancy) and click Update Settings.

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

33. Click Done to return to the main screen and continue the Installation.

34. Click SOFTWARE SELECTION.
35. Select Infrastructure Server and select the Add-Ons as noted below. Click Done.

36. Click **NETWORK & HOSTNAME** and configure Hostname and Networking for the Host.
37. Type in the hostname as shown below.

38. Click Configure to open the Network Connectivity window. Click IPV4Settings.
39. Change the Method to **Manual** and click **Add** to enter the IP Address, Netmask, and Gateway details.
40. Click Save and update the hostname and turn Ethernet ON. Click Done to return to the main menu.

41. Click Begin Installation on the Main menu.

42. Select Root Password in the User Settings.
43. Enter the Root Password and click Done.

44. When the installation is complete reboot the system.
45. Repeat the steps to install Red Hat Enterprise Linux 7.6 on the remaining servers.

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 8.

<table>
<thead>
<tr>
<th>Hostname</th>
<th>eth0</th>
</tr>
</thead>
<tbody>
<tr>
<td>rhel1</td>
<td>10.16.1.31</td>
</tr>
<tr>
<td>rhel2</td>
<td>10.16.1.32</td>
</tr>
<tr>
<td>rhel3</td>
<td>10.16.1.33</td>
</tr>
<tr>
<td>rhel4</td>
<td>10.16.1.34</td>
</tr>
<tr>
<td>rhel5</td>
<td>10.16.1.35</td>
</tr>
<tr>
<td>rhel6</td>
<td>10.16.1.36</td>
</tr>
<tr>
<td>rhel7</td>
<td>10.16.1.37</td>
</tr>
<tr>
<td>rhel8</td>
<td>10.16.1.38</td>
</tr>
<tr>
<td>rhel9</td>
<td>10.16.1.39</td>
</tr>
<tr>
<td>rhel10</td>
<td>10.16.1.40</td>
</tr>
<tr>
<td>rhel11</td>
<td>10.16.1.41</td>
</tr>
</tbody>
</table>
### Hostname

<table>
<thead>
<tr>
<th>Hostname</th>
<th>eth0</th>
</tr>
</thead>
<tbody>
<tr>
<td>rhel12</td>
<td>10.16.1.42</td>
</tr>
<tr>
<td>rhel13</td>
<td>10.16.1.43</td>
</tr>
<tr>
<td>rhel14</td>
<td>10.16.1.44</td>
</tr>
<tr>
<td>rhel15</td>
<td>10.16.1.45</td>
</tr>
<tr>
<td>rhel16</td>
<td>10.16.1.46</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>rhel24</td>
<td>10.16.1.54</td>
</tr>
</tbody>
</table>

⚠️ Multi-homing configuration is not recommended in this design, so please assign only one network interface on each host.

⚠️ For simplicity outbound NATing is configured for internet access when desired such as accessing public repos and/or accessing Red Hat Content Delivery Network. However, configuring outbound NAT is beyond the scope of this document.

### Post OS Install Configuration

Choose one of the nodes of the cluster or a separate node as the Admin Node for management such as HDP installation, Ansible, creating a local Red Hat repo and so on. In this document, rhel1 has been used for this purpose.

#### Configure /etc/hosts

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

⚠️ For the purpose of simplicity, /etc/hosts file is configured with hostnames in all the nodes. However, in large scale production grade deployment, DNS server setup is highly recommended. Furthermore, /etc/hosts file is not copied into containers running on the platform.

Below are the sample A records for DNS configuration within Linux environment.

```bash
rhel1   A 10.16.1.31
rhel2    A 10.16.1.32
rhel3    A 10.16.1.33
...      A 10.16.1.33
...      A 10.16.1.59
```

To create the host file on the admin node, follow these steps:

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

3. On Admin Node (rhel1):

```bash
# cat /etc/hosts
127.0.0.1 localhost localhost.localdomain localhost4 localhost4.localdomain4
::1 localhost localhost.localdomain localhost6 localhost6.localdomain6
10.16.1.31 rhel1 rhel1.hdp3.cisco.com
10.16.1.32 rhel2 rhel2.hdp3.cisco.com
10.16.1.33 rhel3 rhel3.hdp3.cisco.com
10.16.1.34 rhel4 rhel4.hdp3.cisco.com
10.16.1.35 rhel5 rhel5.hdp3.cisco.com
10.16.1.36 rhel6 rhel6.hdp3.cisco.com
10.16.1.37 rhel7 rhel7.hdp3.cisco.com
10.16.1.38 rhel8 rhel8.hdp3.cisco.com
10.16.1.39 rhel9 rhel9.hdp3.cisco.com
10.16.1.40 rhel10 rhel10.hdp3.cisco.com
10.16.1.41 rhel11 rhel11.hdp3.cisco.com
10.16.1.42 rhel12 rhel12.hdp3.cisco.com
10.16.1.43 rhel13 rhel13.hdp3.cisco.com
10.16.1.44 rhel14 rhel14.hdp3.cisco.com
10.16.1.45 rhel15 rhel15.hdp3.cisco.com
10.16.1.46 rhel16 rhel16.hdp3.cisco.com
10.16.1.47 rhel17 rhel17.hdp3.cisco.com
```

Set Up the Passwordless Login

To manage all of the cluster nodes from the admin node password-less login needs to be setup. It assists in automating common tasks with Ansible, and shell-scripts without having to use passwords.

To enable a passwordless login across all the nodes when Red Hat Linux is installed across all the nodes in the cluster, follow these steps:

1. Log into the Admin Node (rhel1).

```bash
# ssh 10.13.1.31
```

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

```bash
# ssh-keygen -N '' -f ~/.ssh/id_rsa
```
3. Run the following command from the admin node to copy the public key id_rsa.pub to all the nodes of the cluster.

```
ssh-copy-id appends the keys to the remote-host's .ssh/authorized_keys.
```

```
# for i in {1..17}; do    echo "copying rhel$i.hdp3.cisco.com"; ssh-copy-id -i ~/.ssh/id_rsa.pub root@rhel$i.hdp3.cisco.com; done;
```

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

5. Enter the password of the remote host.

Create the Red Hat Enterprise Linux (RHEL) 7.6 Local Repository

To create the 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 into rhel1. Create a directory that would contain the repository.

```
# mkdir -p /var/www/html/rhelrepo
Copy the contents of the Red Hat DVD to /var/www/html/rhelrepo
```

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

```
Log back into rhel1 and create the mount directory.
# scp rhel-server-7.6-x86_64-dvd.iso rhel1:/root/
# mkdir -p /mnt/rheliso
# mount -t iso9660 -o loop /root/rhel-server-7.6-x86_64-dvd.iso /mnt/rheliso/
Copy the contents of the ISO to the /var/www/html/rhelrepo directory:
# cp -r /mnt/rheliso/* /var/www/html/rhelrepo
```

3. On rhel1 create a .repo file to enable the use of the yum command:
4. Copy rheliso.repo file from /var/www/html/rhelrepo to /etc/yum.repos.d on rhel1:

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

---

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

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

This step is needed to install software on Admin Node (rhel1) using the repo (such as httpd, create-repo, and so on.)

```bash
# vi /etc/yum.repos.d/rheliso.repo
[rhel7.6]
name=Red Hat Enterprise Linux 7.6
baseurl=file:///var/www/html/rhelrepo
gpgcheck=0
enabled=1
```

---

Create the Red Hat Repository Database

To create the Red Hat repository database, follow these steps:

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:

```bash
# yum -y install createrepo
```

2. Run createrepo on the RHEL repository to create the repo database on admin node:

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

---

Figure 33 createrepo

```
[root@rhel1 rhelrepo]# createrepo .
Spawning worker 0 with 3763 pkgs
Workers Finished
Gathering worker results
Saving Primary metadata
Saving file lists metadata
Saving other metadata
Generating sqlite DBs
sqlite DBs complete
```
Set Up Ansible

To set up Ansible, follow these steps:

1. Download Ansible rpm from the following link:

   https://releases.ansible.com/ansible/rpm/release/epel-7-x86_64/ansible-2.4.6.0-1.el7.ans.noarch.rpm

   ```
   # wget https://releases.ansible.com/ansible/rpm/release/epel-7-x86_64/ansible-2.4.6.0-1.el7.ans.noarch.rpm
   ```

   For more information about how to download and install Ansible engine, please follow the link https://access.redhat.com/articles/3174981

2. Run the following command to install ansible:

   ```
   # yum localinstall -y ansible-2.4.6.0-1.el7.ans.noarch.rpm
   ```

3. Verify Ansible installation by running the following commands:

   ```
   # ansible --version
   # ansible localhost -m ping
   
   [WARNING]: provided hosts list is empty, only localhost is available. Note that the implicit localhost does not match 'all'
   
   localhost | SUCCESS => {
   
   "changed": false,
   
   "failed": false,
   
   "ping": "pong"
   }
   ```

4. Prepare the host inventory file for Ansible as shown below. Various host groups have been created based on any specific installation requirements of certain hosts:

   ```
   [root@rhel1 ~]# cat /etc/ansible/hosts
   [admin]
   rhel1.hdp3.cisco.com

   [namenodes]
   rhel1.hdp3.cisco.com
   rhel2.hdp3.cisco.com
   rhel3.hdp3.cisco.com
   rhel4.hdp3.cisco.com
   rhel5.hdp3.cisco.com
   rhel6.hdp3.cisco.com
   rhel7.hdp3.cisco.com
   rhel8.hdp3.cisco.com
   rhel9.hdp3.cisco.com
   rhel10.hdp3.cisco.com
   rhel11.hdp3.cisco.com
   rhel12.hdp3.cisco.com
   rhel13.hdp3.cisco.com
   rhel14.hdp3.cisco.com
   rhel15.hdp3.cisco.com
   rhel16.hdp3.cisco.com
   ```
5. Verify host group by running the following commands. Figure 34 shows the outcome of the ping command:

```bash
# ansible gpunodes -m ping
```
Setting up the RHEL repository on the admin node requires httpd. To set up RHEL repository on the admin node, follow these steps:

1. Install httpd on the admin node to host repositories:

```sh
# yum -y install httpd
```

2. Add ServerName and make the necessary changes to the server configuration file:

```sh
# vi /etc/httpd/conf/httpd.conf
ServerName 10.16.1.31:80
```

3. Start httpd:

```sh
# service httpd start
# chkconfig httpd on
```

### Set Up All Nodes to Use the RHEL Repository

To set up all nodes to use the RHEL repository, follow these steps:

Based on this repository 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:

```sh
# ansible nodes -m copy -a "src=/var/www/html/rhelrepo/rheliso.repo dest=/etc/yum.repos.d/.
```

2. Copy the /etc/hosts file to all nodes:

```sh
# ansible nodes -m copy -a "src=/etc/hosts dest=/etc/hosts"
```
3. Purge the yum caches:

```bash
# ansible nodes -a "yum clean all"
# ansible nodes -a "yum repolist"
```

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 repofiles.

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

### Upgrade 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/283862063/type/283853158/release/4.0.2](https://software.cisco.com/download/home/283862063/type/283853158/release/4.0.2)

1. In the ISO image, the required driver kmod-enic-.....rpm can be located at Network\Cisco\VIC\RHEL\RHEL7.6.

2. From a node connected to the Internet, download, extract and transfer kmod-enic-.rpm to rhel1 (admin node).

3. Copy the rpm on all nodes of the cluster using the following Ansible commands. For this example, the rpm is assumed to be in present working directory of rhel1:

   ```bash
   [root@rhel1 ~]# ansible all -m copy -a "src=/root/kmod-enic-3.2.210.22-738.18.rhel7u6.x86_64.rpm dest=/root/"
   ```

4. Use the yum module to install the enic driver rpm file on all the nodes through Ansible:

   ```bash
   [root@rhel1 ~]# ansible all -m yum -a "name=/root/kmod-enic-3.2.210.22-738.18.rhel7u6.x86_64.rpm state=present"
   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 ~]# ansible all -m command -a "modinfo enic"
   ```

5. 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\RHEL7.6

### Install xfsprogs

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

```bash
# ansible all -m yum -a "name=xfsprogs state=present"
```

### Set Up JAVA

To setup JAVA, follow these steps:

```
HDP 3.1.0 requires JAVA 8.
```
1. **Download** jdk-7u76-linux-x64.rpm and src the rpm to admin node (rhel1) from the link
   (http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html)

2. **Copy JDK rpm to all nodes:**
   
   ```
   # ansible nodes -m copy -a "src=/root/jdk-8u201-linux-x64.rpm dest=/root/.
   
   # ansible all -m command -a "rpm -ivh jdk-8u201-linux-x64.rpm"
   ```

3. **Extract and Install JDK on all nodes:**
   
   ```
   [root@rhel1 ~]# ansible nodes -m command -a "rpm -ivh jdk-8u181-linux-x64.rpm"
   [WARNING]: Consider using yum, dnf or zypper module rather than running rpm
   ```

4. **Create the following files java-set-alternatives.sh and java-home.sh on admin node (rhel1):**
   
   ```
   vi java-set-alternatives.sh
   #!/bin/bash
   for item in java javac javaws jar jps javah javap jcontrol jconsole jdb; do
     rm -f /var/lib/alternatives/$item
     alternatives --install /usr/bin/$item $item /usr/java/jdk1.8.0_181-amd64/bin/$item 9
   alternatives --set $item /usr/java/jdk1.8.0_181-amd64/bin/$item
   done
   vi java-home.sh
   export JAVA_HOME=/usr/java/jdk1.8.0_181-amd64
   ```

5. **Make the two java scripts created above executable:**
   
   ```
   chmod 755 ./java-set-alternatives.sh ./java-home.sh
   ```

6. **Copying java-set-alternatives.sh to all nodes.**
   
   ```
   ansible nodes -m copy -a "src=/root/java-set-alternatives.sh dest=/root/.
   ansible nodes -m file -a "dest=/root/java-set-alternatives.sh mode=755
   ansible nodes -m copy -a "src=/root/java-home.sh dest=/root/.
   ansible nodes -m file -a "dest=/root/java-home.sh mode=755
   ```
7. **Setup Java Alternatives**

   ```bash
   [root@rhel1 ~]# ansible all -m shell -a "/root/java-set-alternatives.sh"
   ```

8. **Make sure correct java is setup on all nodes (should point to newly installed java path):**

   ```bash
   # ansible all -m shell -a "alternatives --display java | head -2"
   ```

9. **Setup JAVA_HOME on all nodes:**

   ```bash
   # ansible all -m shell -a "source /root/java-home.sh"
   ```

10. **Display JAVA_HOME on all nodes:**

    ```bash
    # ansible all -m command -a "echo $JAVA_HOME"
    ```

11. **Display current java –version:**

    ```bash
    # ansible all -m command -a "java -version"
    ```

```
[root@rhel1 ~]# ansible all -m command -a "java -version"
rhel3.hdp3.cisco.com | SUCCESS | rc=0 >>
java version "1.8.0_181"
Java(TM) SE Runtime Environment (build 1.8.0_181-b13)
Java HotSpot(TM) 64-Bit Server VM (build 25.181-b13, mixed mode)
```

**Configure NTP**

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.

```bash
# ansible all -m yum -a "name=ntp state=present"
```

> 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:**

   ```bash
   # vi /etc/ntp.conf
driftfile /var/lib/ntp/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:**
Solution Design

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
# ansible nodes -m copy -a "src=/root/ntp.conf dest=/etc/ntp.conf"
```

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

```bash
# ansible all -m service -a "name=ntpd state=stopped"
# ansible all -m command -a "ntpdate rhel1.hdp3.cisco.com"
# ansible all -m service -a "name=ntpd state=started"
```

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

```bash
# ansible all -m systemctl enable ntpd
```

6. Verify NTP is up and running in all nodes by running the following commands:

```bash
# ansible all -m systemctl status ntpd
```

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

7. Install the Chrony service:

```bash
# ansible all -m yum -a "name=chrony state=present"
```

8. Activate the Chrony service at boot:

```bash
# ansible all -m systemctl enable chronyd
```

9. Start the Chrony service:

```bash
# ansible all -m service -a "name=chronyd state=started" systemctl start chronyd
```

The Chrony configuration is in the /etc/chrony.conf file, configured similar to /etc/ntp.conf.
Enable 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
# ansible all -m command -a "rsyslogd -v"
# ansible all -m command -a "service rsyslog status"
```

Set 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 64000 on every node. Higher values are unlikely to result in an appreciable performance gain.

To set ulimit, follow these steps:

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
   ```

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

   ```bash
   # ansible nodes -m copy -a "src=/etc/security/limits.conf dest=/etc/security/limits.conf"
   ```

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
   ```

   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.

Disable 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.

To disable SELinux, follow these steps:

1. The following command will disable SELINUX on all nodes:

   ```sh
   # ansible nodes -m shell -a "sed -i 's/SELINUX=enforcing/SELINUX=disabled/g' /etc/selinux/config"
   # ansible nodes -m shell -a "setenforce 0"
   ```
   
   The command (above) may fail if SELinux is already disabled. This require reboot to take effect.

2. Query the machine, if needed for SELinux to be disabled in case it does not take effect. It can be checked using the following command:

   ```sh
   # ansible nodes -a "sestatus"
   ```

   ![sestatus output]

**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).

To set TCP retries, follow these steps:

1. On each node, set the number of TCP retries to 5 can help detect unreachable nodes with less latency.

   ```sh
   # ansible nodes -m shell -a "sed -i 's/SELINUX=enforcing/SELINUX=disabled/g' /etc/selinux/config"
   # ansible nodes -m shell -a "setenforce 0"
   ```

   ```sh
   # ansible nodes -m shell -a "sed -i 's/SELINUX=enforcing/SELINUX=disabled/g' /etc/selinux/config"
   # ansible nodes -m shell -a "setenforce 0"
   # ansible nodes -a "sestatus"
   ```

   ![sestatus output]

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

   ```sh
   net.ipv4.tcp_retries2=5
   ```

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

   ```sh
   # ansible nodes -m copy -a "src=/etc/sysctl.conf dest=/etc/sysctl.conf"
   ```

   Load the settings from default sysctl file `/etc/sysctl.conf` by running the following command:

   ```sh
   # ansible nodes -a "sysctl -p"
   ```

**Disable 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.

```sh
# ansible all -m command -a "firewall-cmd --zone=public --add-port=80/tcp --permanent"
# ansible all -m command -a "firewall-cmd --reload"
```
Disable Swapping

To disable swapping, follow these steps:

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

   ```
   # ansible all -m shell -a "echo \'vm.swappiness=1\' >> /etc/sysctl.conf"
   ```

2. Load the settings from default sysctl file /etc/sysctl.conf and verify the content of sysctl.conf:

   ```
   # ansible all -m shell -a "sysctl -p"
   # ansible all -m shell -a "cat /etc/sysctl.conf"
   ```

Disabling Transparent Huge Pages

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

To disable Transparent Huge Pages, follow these steps:

1. The following commands must be run for every reboot, so copy this command to /etc/rc.local so they are executed automatically for every reboot:

   ```
   # ansible all -m shell -a "echo never > /sys/kernel/mm/transparent_hugepage/enabled"
   # ansible all -m shell -a "echo never > /sys/kernel/mm/transparent_hugepage/defrag"
   ```

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:

   ```
   # ansible nodes -m copy -a "src=/root/thp_disable dest=/root/thp_disable"
   ```

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

```bash
# ansible nodes -m shell -a "cat /root/thp_disable >> /etc/rc.local"
```

Disable IPv6 Defaults

To disable IPv6 defaults, follow these steps:

1. Disable IPv6 as the addresses used are IPv4:

   ```bash
   # ansible all -m shell -a "echo 'net.ipv6.conf.all.disable_ipv6 = 1' >> /etc/sysctl.conf"
   # ansible all -m shell -a "echo 'net.ipv6.conf.default.disable_ipv6 = 1' >> /etc/sysctl.conf"
   # ansible all -m shell -a "echo 'net.ipv6.conf.lo.disable_ipv6 = 1' >> /etc/sysctl.conf"
   ```

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

   ```bash
   # ansible all -m shell -a "sysctl -p"
   ```

Configure Data Drives on Name Node and Other Management Nodes

This section describes the steps to configure non-OS disk drives as RAID1 using the StorCli command. All drives are 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.

To configure data drives on Name node and other nodes, follow these steps:


2. Extract the zip file and copy storcli-1.14.12-1.noarch.rpm from the linux directory.

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

   ```bash
   #scp storcli-1.14.12-1.noarch.rpm rhel1:/root/
   ```

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

   ```bash
   # ansible all -m copy -a "src=/root/storcli-1.14.12-1.noarch.rpm dest=/root/.
   ```

5. Run this command to install storcli on all the nodes:

   ```bash
   # ansible all -m shell -a "rpm -ivh storcli-1.14.12-1.noarch.rpm"
   ```

6. Run this command to copy storcli64 to root directory:

   ```bash
   # ansible all -m shell -a "cp /opt/MegaRAID/storcli/storcli64 /root/.
   ```

7. Run this command to check the state of the disks:

   ```bash
   # ansible all -m shell -a ".:/storcli64 /c0 show"
   ```

   If the drive state shows up as JBOD, creating RAID in the subsequent steps will fail with the error "The specified physical disk does not have the appropriate attributes to complete the requested command."

8. If the drive state shows up as JBOD, it can be converted into Unconfigured Good using Cisco UCSM or storcli64 command. Following steps should be performed if the state is JBOD.
9. Get the enclosure id as follows:

   ansible all -m shell -a "/storcli64 pdlist -a0 | grep Enc | grep -v 252 | awk '{print $4}' | sort | uniq -c | awk '{print $(1+2)"\"}"

   It is observed that some earlier versions of storcli64 complains about above mentioned command as if it is deprecated. In this case, please ./storcli64 /c0 show all| awk '{print $1}'| sed -n '/[0-9]:[0-9]/p'|awk '{print substr($1,1,2)|sort -u' command to determine enclosure id.

10. Convert to unconfigured good:

    ansible datanodes -m command -a "/.storcli64 /c0 /e66 /sall set good force"

11. Verify status by running the following command:

    # ansible datanodes -m command -a "/.storcli64 /c0 /e66 /sall show"

12. Run this script as root user on rhel1 to rhel3 to create the virtual drives for the management nodes:

    #vi /root/raid1.sh
The script (above) requires enclosure ID as a parameter.

Run the following command to get enclosure id:

```bash
#!/usr/bin/env bash
do_something
```

13. Run the following command to get enclosure id:

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

14. Run MegaCli script:

```bash
#./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.
StripSize1024: Strip Size of 1024K
```

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

15. Run the following command. State should change to Online:

```bash
ansible namenodes -m command -a "./storcli64 /c0 /e66 /sall show"
```

16. State can also be verified in UCSM as show below in Equipment>Rack Mounts>Servers>Server # under Inventory/Storage/Disk tab:
Configure Data Drives on Data Nodes

To configure non-OS disk drives as individual RAID0 volumes using StorCli command, follow these steps. These volumes will be used for HDFS Data.

1. Issue the following command from the admin node to create the virtual drives with individual RAID 0 configurations on all the data nodes:

```
[root@rhel1 ~]# ansible datanodes -m command -a "./storcli64 -cfgeachdskraid0 WB RA direct
NoCachedBadBBU strpsz1024 -a0"
```

```
rhel7.hdp3.cisco.com | SUCCESS | rc=0 >>
Adapter 0: Created VD 0
  Configured physical device at Encl-66:Slot-7.
Adapter 0: Created VD 1
  Configured physical device at Encl-66:Slot-6.
Adapter 0: Created VD 2
  Configured physical device at Encl-66:Slot-8.
Adapter 0: Created VD 3
  Configured physical device at Encl-66:Slot-5.
Adapter 0: Created VD 4
  Configured physical device at Encl-66:Slot-3.
Adapter 0: Created VD 5
  Configured physical device at Encl-66:Slot-4.
Adapter 0: Created VD 6
  Configured physical device at Encl-66:Slot-1.
Adapter 0: Created VD 7
  Configured physical device at Encl-66:Slot-2.
      ...... Omitted Output
24 physical devices are Configured on adapter 0.
```

Exit Code: 0x00

The command (above) will not override existing configurations. To clear and reconfigure existing configurations, refer to the Embedded MegaRAID Software Users Guide available at www.broadcom.com.
Configure the Filesystem for NameNodes and Datanodes

The following script formats and mounts the available volumes on each node whether it is NameNode or Data node. OS boot partition will be skipped. All drives are mounted based on their UUID as /data/disks, /data/disk2, etc. To configure the filesystem for NameNodes and DataNodes, follow these steps:

1. From the Admin node, create partition tables and file systems on the local disks supplied to each of the nodes, run the following script as the root user on each node:

```bash
#!/bin/bash

#vi /root/driveconf.sh

[[ "-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?
dolist+=$(echo $X " ")
done

for X in /dev/sd??
dolist+=$(echo $X " ")
done

for X in $olist
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##*/}

if [[ -b ${Y} && `/sbin/parted -s ${Y} print quit|/bin/grep -c boot` -ne 0 ]]
then
    echo "${Y} bootable - skipping."
else
    echo "Formatting and Mounting Drive => ${X}"
    /sbin/mkfs.xfs -f ${X}
    (( $? )) && continue
    #Identify UUID
    UUID="/blkid ${X} | cut -d " " -f2 | cut -d "=" -f2 | sed 's/"/\"/g'"
    /sbin/mkdir -p /data/disk$(count)
    (( $? )) && continue
    echo "UUID of ${X} = ${UUID}, mounting ${X} using UUID on /data/disk$(count)"
    /bin/mount -t xfs -o ino64,noatime,nobarrier -U ${UUID} /data/disk$(count)
    (( $? )) && continue
    echo "UUID=${UUID} /data/disk$(count) xfs ino64,noatime,nobarrier 0 0" >> /etc/fstab
    ((count++))
fi

done
```

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

```
```
# chmod 755 /root/driveconf.sh
# ansible datanodes -m copy -a "src=/root/driveconf.sh dest=/root/.
# ansible nodes -m file -a "dest=/root/driveconf.sh mode=755"

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

# ansible datanodes -m shell -a "/root/driveconf.sh"

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

# ansible datanodes -m shell -a "df -h"
# ansible datanodes -m shell -a "mount"
# ansible datanodes -m shell -a "cat /etc/fstab"

Delete Partitions

To delete a partition, follow these steps:

Run the mount command (‘mount’) to identify which drive is mounted to which device /dev/sd<?>
umount the drive for which partition is to be deleted and run fdisk to delete as shown below.

⚠️ Do not delete the OS partition since this will wipe out the OS.

# mount
# umount /data/disk1 (disk1 shown as example)
#(echo d; echo w;) | sudo fdisk /dev/sd<?>

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.

To verify a cluster, follow these steps:

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

```bash
#vi cluster_verification.sh
#!/bin/bash
shopt -s expand_aliases,
# Setting Color codes
green="\e[0;32m"
red="\e[0;31m"
NC="\e[0m" # No Color
echo -e "$(green) === Cisco UCS Integrated Infrastructure for Big Data and Analytics 
Cluster Verification === $(NC)"
echo ""
echo ""
echo -e "$(green) ==== System Information  ==== $(NC)"
echo ""
echo ""
ejcho -e "$(green) System $(NC)"
clush -a -B `which dmidecode` |grep -A2 '^System Information'"}
echo ""
echo ""
```
Solution Design

echo -e "$(green)BIOS ${NC}"
clush -a -B " `which dmidecode` | grep -A3 "^BIOS I"" 

# probe for cpu info #
echo -e "$(green)CPU $(NC)"
clush -a -B "grep "model name" /proc/cpuinfo | sort -u" 

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

# probe for disk info #
echo -e "$\(green\)Storage $(NC)"
clush -a -B " `which lspci` | grep -i ether" 

echo -e "$\(green\) ================ Software ================ $(NC)"

echo -e "$\(green\) =============== Linux Release =============== $\{NC\}" 

echo -e "$\(green\) =============== Linux Version =============== $(NC)"

echo -e "$\(green\) =============== Date =============== $(NC)"

echo -e "$\(green\) =============== NTP Status =============== $(NC)"

echo -e "$\(green\) =============== SELINUX =============== $(NC)"
2. Change permissions to executable:

```bash
# 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:

```bash
#/cluster_verification.sh
```

## Install HDP 3.1.0

HDP is an enterprise grade, hardened Hadoop distribution. HDP combines Apache Hadoop and its related projects into a single tested and certified package. HDP 3.1.0 components are depicted in below. This section details how to install HDP 3.1.0 on the cluster.

![Hortonworks Data Platform](image-url)
Prerequisites for HDP Installation
This section details the prerequisites for the HDP installation, such as setting up HDP repositories.

Hortonworks Repository
1. From a host connected to the Internet, create a Hortonworks folder and download the Hortonworks repositories as shown below, then transfer it to the admin node:

   If the admin node is connected to the internet via outbound NAT, repositories can be downloaded directly into the admin node.

   ```
   # mkdir -p /tmp/Hortonworks/
   # cd /tmp/Hortonworks
   # wget http://public-repo-1.hortonworks.com/HDP/centos7/3.x/updates/3.1.0.0/HDP-3.1.0.0-.centos7-rpm.tar.gz
   --2018-10-13 11:02:02-- http://public-repo-1.hortonworks.com/HDP/centos7/3.x/updates/3.1.0.0/HDP-3.1.0.0-.centos7-rpm.tar.gz
   Resolving public-repo-1.hortonworks.com (public-repo-1.hortonworks.com)... 13.35.121.86, 13.35.121.14, 13.35.121.127, ...
   Connecting to public-repo-1.hortonworks.com (public-repo-1.hortonworks.com)|13.35.121.86|:80... connected.
   HTTP request sent, awaiting response... 200 OK
   Length: 8964079720 (8.3G) [application/x-tar]
   Saving to: 'HDP-3.1.0.0-centos7-rpm.tar.gz'
   100%[==========================================================================================================]
   8,964,079,720 50.3MB/s  in 2m 42s
   2018-10-13 11:04:44 (52.9 MB/s) - 'HDP-3.1.0.0-centos7-rpm.tar.gz' saved [8964079720/8964079720]
   ```

2. Download Hortonworks HDP repo:

   ```
   # wget http://public-repo-1.hortonworks.com/HDP-UTILS-1.1.0.22/repos/centos7/HDP-UTILS-1.1.0.22-centos7.tar.gz
   --2018-10-13 11:05:30-- http://public-repo-1.hortonworks.com/HDP-UTILS-1.1.0.22/repos/centos7/HDP-UTILS-1.1.0.22-centos7.tar.gz
   Resolving public-repo-1.hortonworks.com (public-repo-1.hortonworks.com)... 13.35.121.86, 13.35.121.14, 13.35.121.127, ...
   Connecting to public-repo-1.hortonworks.com (public-repo-1.hortonworks.com)|13.35.121.86|:80... connected.
   HTTP request sent, awaiting response... 200 OK
   Length: 90606616 (86M) [application/x-tar]
   Saving to: 'HDP-UTILS-1.1.0.22-centos7.tar.gz'
   100%[==========================================================================================================]
   90,606,616 45.0MB/s  in 1.9s
   2018-10-13 11:05:33 (45.0 MB/s) - 'HDP-UTILS-1.1.0.22-centos7.tar.gz' saved [90606616/90606616]
   ```

3. Download Hortonworks HDP-Utils repo:

4. Download HDP-GPL repo:
5. Download the Ambari repo:

```bash
# wget http://public-repo-1.hortonworks.com/ambari/centos7/2.x/updates/2.7.1.0/ambari-2.7.1.0-centos7.tar.gz
```

6. Copy the repository directory to the admin node (rhel1):

```bash
# scp -r /tmp/Hortonworks/ rhel1:/var/www/html/
```

7. Extract the tar ball:

```bash
[root@rhel1 Hortonworks]# tar -zxvf HDP-3.1.0.0-centos7-rpm.tar.gz
[root@rhel1 Hortonworks]# tar -zxvf HDP-UTILS-1.1.0.22-centos7.tar.gz
[root@rhel1 Hortonworks]# tar -zxvf HDP-GPL-3.1.0.0-centos7-gpl.tar.gz
[root@rhel1 Hortonworks]# tar -zxvf ambari-2.7.1.0-centos7.tar.gz
```

8. Create HDP repo with the following contents:

```bash
[root@rhel1]# cat /etc/yum.repos.d/hdp.repo
[HDP-3.1.0.0]
name= Hortonworks Data Platform Version - HDP-3.1.0.0
baseurl= http://rhel1.hdp3.cisco.com/Hortonworks/HDP/centos7/3.1.0.0-187
gpgcheck=0
enabled=1
priority=1

[HDP-GPL-3.1.0.0]
name=Hortonworks GPL Version - HDP-GPL-3.1.0.0
baseurl= http://rhel1.hdp3.cisco.com/Hortonworks/GPL/centos7/3.1.0.0-187
gpgcheck=0
enabled=1
priority=1

[HDP-UTILS-1.1.0.22]
name=Hortonworks Data Platform Utils Version - HDP-UTILS-1.1.0.22
baseurl= http://rhel1.hdp3.cisco.com/Hortonworks/HTTPD-UTILS/centos7/1.1.0.22
gpgcheck=0
enabled=1
priority=1
```

To verify the files, go to: [http://rhel1.hdp3.cisco.com/Hortonworks](http://rhel1.hdp3.cisco.com/Hortonworks).
9. Create the Ambari repo:

```bash
vi /etc/yum.repos.d/ambari.repo

[Updates-ambari-2.7.1.0]
name=ambari-2.7.1.0 - Updates
baseurl=http://rhel1.hdp3.cisco.com/Hortonworks/ambari/centos7/2.7.1.0-169
gpgcheck=0
enabled=1
priority=1
```

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

```bash
# ansible nodes -m copy -a "src=/etc/yum.repos.d/hdp.repo dest=/etc/yum.repos.d/"  
# ansible nodes -m copy -a "src=/etc/yum.repos.d/ambari.repo dest=/etc/yum.repos.d/"
```

**Downgrade Snappy on All Nodes**

Downgrade snappy on all data nodes by running this command from admin node:

```bash
# ansible all -m command -a "yum -y downgrade snappy"
```

**HDP Installation**

To install HDP, complete the following the steps:

**Install and Setup Ambari Server on rhel1**

1. Run the following command in rhel1 to install ambari-server:

```bash
#yum -y install ambari-server
```

```
Loaded plugins: langpacks, product-id, search-disabled-repos, subscription-manager
This system is not registered with an entitlement server. You can use subscription-manager to register.
Resolving Dependencies
---> Running transaction check
---> Package ambari-server.x86_64 0:2.7.1.0-169 will be installed
---> Processing Dependency: postgresql-server >= 8.1 for package: ambari-server-2.7.1.0-169.x86_64
    --> Running transaction check
---> Package postgresql-server.x86_64 0:9.2.23-3.el7_4 will be installed
    --> Processing Dependency: postgresql-libs(x86-64) = 9.2.23-3.el7_4 for package: postgresql-server-9.2.23-3.el7_4.x86_64
    --> Processing Dependency: libpq.so.5()(64bit) for package: postgresql-server-9.2.23-3.el7_4.x86_64
    --> Running transaction check
---> Package postgresql.x86_64 0:9.2.23-3.el7_4 will be installed
---> Package postgresql-libs.x86_64 0:9.2.23-3.el7_4 will be installed
---> Finished Dependency Resolution
```

```
Dependencies Resolved
```

```
Package Repository Arch Size Version
============================================= 
```

```bash
```
```
Installing:
  ambari-server  x86_64  2.7.1.0-169

Updates-ambari-2.7.1.0  353 M

Installing for dependencies:
  postgresql  x86_64  9.2.23-3.el7_4
  rhel7.6  3.0 M
  postgresql-libs  x86_64  9.2.23-3.el7_4
  rhel7.6  234 k
  postgresql-server  x86_64  9.2.23-3.el7_4
  rhel7.6  3.8 M

Transaction Summary
=================================================================================================
Install  1 Package (+3 Dependent packages)
Total download size: 360 M
Installed size: 452 M

Installing : postgresql-libs-9.2.23-3.el7_4.x86_64.rpm
  | 234 kB  00:00:00
(2/4): postgresql-9.2.23-3.el7_4.x86_64.rpm
  | 3.0 MB  00:00:00
(3/4): postgresql-server-9.2.23-3.el7_4.x86_64.rpm
  | 3.8 MB  00:00:00
(4/4): ambari-server-2.7.1.0-169.x86_64.rpm
  | 353 MB  00:00:03
-----------------------------------------------------------------------------------------------
Total
  109 MB/s | 360 MB  00:00:03
Running transaction check
Running transaction test
Transaction test succeeded
Running transaction
  Installing : postgresql-libs-9.2.23-3.el7_4.x86_64
  1/4
  Installing : postgresql-9.2.23-3.el7_4.x86_64
  2/4
  Installing : postgresql-server-9.2.23-3.el7_4.x86_64
  3/4
  Installing : ambari-server-2.7.1.0-169.x86_64
  4/4
  Verifying : postgresql-9.2.23-3.el7_4.x86_64
  1/4
  Verifying : postgresql-libs-9.2.23-3.el7_4.x86_64
  2/4
  Verifying : postgresql-server-9.2.23-3.el7_4.x86_64
  3/4
  Verifying : ambari-server-2.7.1.0-169.x86_64
  4/4

Installed:
  ambari-server.x86_64 0:2.7.1.0-169

Dependency Installed:
  postgresql.x86_64 0:9.2.23-3.el7_4
  postgresql-libs.x86_64 0:9.2.23-3.el7_4
  postgresql-server.x86_64 0:9.2.23-3.el7_4

Complete!
Install PostgreSQL in rhel2

The PostgreSQL database is used by Ambari, Hive, and Oozie services.
The rhel2 hosts the Hive and Oozie services and Ambari Server is installed on rhel1. To install, follow these steps:

2. Log into rhel2.

3. Install Red Hat Package Manager (RPM) according to the requirements of your operating system:

   ```
   yum install https://yum.postgresql.org/9.6/redhat/rhel-7-x86_64/pgdg-redhat96-9.6-3.noarch.rpm
   ```

4. Install PostgreSQL version 9.5 or later:

   ```
   yum install postgresql96-server postgresql96-contrib postgresql96
   ```

5. Initialize the database as shown in the below figure by running the following command:

   ```
   /usr/pgsql-9.6/bin/postgresql96-setup initdb
   ```

6. Start PostgreSQL:

   ```
   # systemctl enable postgresql-9.6.service
   # systemctl start postgresql-9.6.service
   ```

7. Open `/var/lib/pgsql/9.6/data/postgresql.conf` and update to the following:

   ```
   listen_addresses = '*'
   ```

8. Update these files on rhel2 in the location chosen to install the databases for Hive, Oozie and Ambari, using the host ip addresses:

   ```
   [root@rhe1 ~]# cat /var/lib/pgsql/9.6/data/pg_hba.conf|tail -n 20
   # TYPE DATABASE USER ADDRESS METHOD
   # "local" is for Unix domain socket connections only
   #local all all peer
   # IPv4 local connections:
   #host all all 127.0.0.1/32 ident
   # IPv6 local connections:
   host all all ::1/128 ident
   # Allow replication connections from localhost, by a user with the
   # replication privilege.
   #local replication postgres peer
   #host replication postgres 127.0.0.1/32 ident
   #host replication postgres ::1/128 ident
   local all postgres peer
   local all all md5
   host all postgres,hive,oozie 10.16.1.32/24 md5
   host all ambari 10.16.1.31/24 md5
   ```
Before adding new entries, comment the old entries as mentioned above.

9. Restart PostgreSQL:

```bash
# systemctl stop postgresql-9.6.service
# systemctl start postgresql-9.6.service
```

10. Run the following:

```bash
sudo -u postgres psql
```

For more information about setting up PostgreSQL, go to: https://docs.hortonworks.com/HDPDocuments/Ambari-2.7.1.0/bk_ambari-installation/content/install-postgres.html

Create Database for Ambari

To create the database for Ambari, follow these steps:

1. Run the following commands mentioned below in bold to create and prepare database for Ambari:

```bash
[root@rhel2 ~]# sudo -u postgres psql
could not change directory to "/root": Permission denied
psql (9.6.10)
Type "help" for help.

dt
No relations found.

postgres=# create database ambari;
CREATE DATABASE

postgres=# create user ambari with password 'bigdata';
CREATE ROLE

postgres=# grant all privileges on database ambari to ambari;
GRANT

postgres=# connect ambari;
You are now connected to database "ambari" as user "postgres".
ambari=# create schema ambari authorization ambari;
CREATE SCHEMA

ambari=# alter schema ambari owner to ambari;
ALTER SCHEMA

ambari=# alter role ambari set search_path to 'ambari', 'public';
ALTER ROLE

ambari=# \
```

2. Restart PostgreSQL:

```bash
[[root@rhel2 ~]# systemctl restart postgresql-9.6.service
```

3. Verify the ambari user by logging into psql:
4. Load the Ambari Server database schema:

Pre-load the Ambari database schema into your PostgreSQL database using the schema script.

5. Find the Ambari-DDL-Postgres-CREATE.sql file in the /var/lib/ambari-server/resources/ directory of the Ambari Server host after you have installed Ambari Server.

Copy /var/lib/ambari-server/resources/ from rhel1 to rhel2:/tmp/.

[root@rhel1 ~]# scp -r /var/lib/ambari-server/resources/* rhel2:/tmp/

6. Run the following command to launch the Ambari-DDL-Postgres-CREATE.sql script:

[root@rhel2 tmp]# cd /tmp
[root@rhel2 tmp]# psql -U ambari -d ambari
Password for user ambari:
psql (9.6.10)
Type "help" for help.
ambari=> \i Ambari-DDL-Postgres-CREATE.sql
CREATE TABLE
CREATE TABLE
CREATE TABLE
.... OUTPUT OMITTED ----

7. Check the table is created by running \dt command:

[root@rhel2 ~]# psql -U ambari -d ambari
psql (9.6.10)
Type "help" for help.
ambari=>
ambari=> \dt

8. Restart PostgreSQL:
Create Database for Hive

Run the following command as shown in bold to create and prepare database for Hive:

```
[root@rhel2 ~]# sudo -u postgres psql
psql (9.6.10)
Type "help" for help.
postgres=# create database hive;
CREATE DATABASE
postgres=# create user hive with password 'bigdata';
CREATE ROLE
postgres=# grant all privileges on database hive to hive;
GRANT
postgres=# \
[root@rhel2 ~]#
```

Create Database for Oozie

Run the following command to create and prepare database for Oozie:

```
[root@rhel2 ~]# sudo -u postgres psql
psql (9.6.10)
Type "help" for help.
postgres=# create database oozie;
CREATE DATABASE
postgres=# create user oozie with password 'bigdata';
CREATE ROLE
postgres=# grant all privileges on database oozie to oozie;
GRANT
postgres=# \
[root@rhel2 ~]#
```

Setup Ambari Server On Admin Node(Rhel1)

To setup the Ambari server, follow these steps:

1. Install the PostgreSQL JDBC driver:

```
[root@rhel1 2.7.1.0-169]# yum -y install postgresql-jdbc*
Loaded plugins: langpacks, product-id, search-disabled-repos, subscription-manager
This system is not registered with an entitlement server. You can use subscription-manager to
register.
Resolving Dependencies
---> Running transaction check
----> Package postgresql-jdbc.noarch 0:9.2.1002-5.el7 will be installed
---> Processing Dependency: jpackage-utils for package: postgresql-jdbc-9.2.1002-5.el7.noarch
---> Processing Dependency: java for package: postgresql-jdbc-9.2.1002-5.el7.noarch
---> Running transaction check
----> Package java-1.8.0-openjdk.x86_64 1:1.8.0.161-2.b14.el7 will be installed
```

2. Configure Ambari server to use the JDBC driver for connectivity to Ambari database in PostgreSQL:

```
[root@rhel1 2.7.1.0-169]# ambari-server setup --jdbc-db=postgresql --jdbc-driver=/usr/share/java/postgresql-jdbc.jar
```
Using python /usr/bin/python
Setup ambari-server
Copying /usr/share/java/postgresql-jdbc.jar to /var/lib/ambari-server/resources/postgresql-jdbc.jar
If you are updating existing jdbc driver jar for postgres with postgresql-jdbc.jar. Please remove the old driver jar, from all hosts. Restarting services that need the driver, will automatically copy the new jar to the hosts.
JDBC driver was successfully initialized.
Ambari Server 'setup' completed successfully.

3. Setup Ambari Server by running the following command:

[root@rhel1 ~]# ambari-server setup -j $JAVA_HOME
Using python /usr/bin/python
Setup ambari-server
Checking SELinux...
SELinux status is 'disabled'
Customize user account for ambari-server daemon [y/n] (n)? n
Adjusting ambari-server permissions and ownership...
Checking firewall status...
Checking JDK...
WARNING: JAVA_HOME /usr/java/jdk1.8.0_181-amd64 must be valid on ALL hosts
WARNING: JCE Policy files are required for configuring Kerberos security. If you plan to use Kerberos, please make sure JCE Unlimited Strength Jurisdiction Policy Files are valid on all hosts.
Check JDK version for Ambari Server...
JDK version found: 8
Minimum JDK version is 8 for Ambari. Skipping to setup different JDK for Ambari Server.
Checking GPL software agreement...
GPL License for LZO: https://www.gnu.org/licenses/old-licenses/gpl-2.0.en.html
Enable Ambari Server to download and install GPL Licensed LZO packages [y/n] (n)? y
Completing setup...
Configuring database...
Enter advanced database configuration [y/n] (n)? y
Configuring database...
==============================================================================
Choose one of the following options:
[1] - PostgreSQL (Embedded)
[2] - Oracle
[3] - MySQL / MariaDB
[4] - PostgreSQL
[5] - Microsoft SQL Server (Tech Preview)
[6] - SQL Anywhere
[7] - BDB
==============================================================================
Enter choice (4):
Hostname (rhel2.hdp3.cisco.com):
Port (5432):
Database name (ambari):
PostgreSQL schema (ambari):
Username (ambari):
Enter Database Password (bigdata):
Configuring ambari database...
Configuring remote database connection properties...
WARNING: Before starting Ambari Server, you must run the following DDL against the database to create the schema: /var/lib/ambari-server/resources/Ambari-DDL-Postgres-CREATE.sql
Proceed with configuring remote database connection properties [y/n] (y)? y
Extracting system views...
.....
Adjusting ambari-server permissions and ownership...
Ambari Server 'setup' completed successfully.
4. Start the Ambari Server:

```
[root@rhel1 ~]# ambari-server start
```

5. To check status of Ambari Server, run the following command:

```
# ambari-server status
```

Launch the Ambari Server

When the Ambari service starts, access the Ambari Install Wizard through the browser. To launch the Ambari server, follow these steps:

1. Point the browser to http://<ip address for rhel1>:8080 or http://rhel1.hdp3.cisco.com:8080

The Ambari Login screen opens.
2. Log into the Ambari Server using the default username/password: admin/admin. This can be changed at a later period of time.

3. When logged in the “Welcome to Apache Ambari” window opens.

Create the Cluster

To create the cluster, follow these steps:

1. To create a cluster, click “LAUNCH INSTALL WIZARD.”

2. From the Get started page type "Cisco_HDP" for the name for the cluster.

3. Click Next.
Select Version

To select the version, follow these steps:

1. In the Select Version section, choose the HDP 3.1.0. version.

2. Under Repositories, select “Use Local Repository.”


Update the Redhat 7 HDP-UTILS-1.1.0.22 to http://rhel1.hdp3.cisco.com/Hortonworks/HDP-UTILS/centos7/1.1.0.22/

---

Make sure there are no trailing spaces after the URLs.

Select Hosts

To build up the cluster, you need to provide the general information about how you want to set up the cluster. This requires providing the Fully Qualified Domain Name (FQDN) of each of the hosts. You also need to provide access to the private key file that was created in Set Up Password-less SSH; this is used to locate all the hosts in the system and to access and interact with them securely.

To select hosts, follow these steps:

1. Use the **Target Hosts** text box to enter the list of host names, one per line. Ranges inside brackets can also be used to indicate larger sets of hosts.

2. Select the option **Provide your SSH Private Key** in the Ambari cluster install wizard.

3. Copy the contents of the file `/root/.ssh/id_rsa` on rhel1 and paste it in the text area provided by the Ambari cluster install wizard.

---

Make sure there is no extra white space after the text......END RSA PRIVATE KEY......

4. Click the **Register and Confirm** to continue.
Hostname Pattern Expressions

1. Click OK on the Host Name Pattern Expressions popup.
Confirm Hosts

Confirm Hosts helps ensure that Ambari has located the correct hosts for the cluster and checks those hosts to make sure they have the correct directories, packages, and processes to continue the install.

To confirm host, follow these steps:

1. If any host was selected in error, remove it by selecting the appropriate checkboxes and clicking the grey Remove Selected button.

2. To remove a single host, click the small white Remove button in the Action column.

3. When the list of hosts is confirmed, click Next.

Choose Services

HDP is made up of a number of components. Go to Hortonworks Understand the Basics for more information.

To choose services, follow these steps:

1. Select all to preselect all items.

2. When you have made your selections, click Next.
Assign Masters

The Ambari installation wizard attempts to assign the master nodes for various services that have been selected to appropriate hosts in the cluster, as listed in Table 9. The right column shows the current service assignments by host, with the hostname and its number of CPU cores and amount of RAM indicated.

1. Reconfigure the service assignments to match Table 9.

<table>
<thead>
<tr>
<th>Service Name</th>
<th>Host</th>
</tr>
</thead>
<tbody>
<tr>
<td>NameNode</td>
<td>rhel1, rhel3 (HA)</td>
</tr>
<tr>
<td>SNNameNode</td>
<td>rhel2</td>
</tr>
<tr>
<td>History Server</td>
<td>rhel2</td>
</tr>
<tr>
<td>App Timeline Server</td>
<td>rhel2</td>
</tr>
<tr>
<td>Resource Manager</td>
<td>rhel2, rhel3 (HA)</td>
</tr>
<tr>
<td>Hive Metastore</td>
<td>rhel2</td>
</tr>
<tr>
<td>WebHCat Server</td>
<td>rhel2</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>Zookeeper</td>
<td>rhel1, rhel2, rhel3</td>
</tr>
<tr>
<td>Spark History Server</td>
<td>rhel2</td>
</tr>
<tr>
<td>SmartSense HST Server</td>
<td>rhel1</td>
</tr>
</tbody>
</table>
Assign Slaves and Clients

The Ambari install wizard attempts to assign the slave components (DataNodes, NFSGateway, NodeManager, RegionServers, Supervisor, and Client) to appropriate hosts in the cluster.

To assign slaves and clients, follow these steps:

1. Reconfigure the service assignment to match the values shown in Table 10.

2. Assign DataNode, NodeManager, RegionServer, and Supervisor on nodes rhel3-rhel28.

3. Assign Client to all nodes.

4. Click Next.

Table 10 Services and Hostnames

<table>
<thead>
<tr>
<th>Client Service Name</th>
<th>Host</th>
</tr>
</thead>
<tbody>
<tr>
<td>DataNode</td>
<td>rhel4-rhel16; rhel24-rhel31</td>
</tr>
<tr>
<td>NFSGateway</td>
<td>rhel1</td>
</tr>
<tr>
<td>NodeManager</td>
<td>rhel4-rhel16; rhel24-rhel31</td>
</tr>
<tr>
<td>RegionServer</td>
<td>rhel4-rhel13; rhel24-rhel31</td>
</tr>
<tr>
<td>Supervisor</td>
<td>rhel4-rhel13; rhel24-rhel31</td>
</tr>
<tr>
<td>Client</td>
<td>All nodes, rhel1-rhel31</td>
</tr>
<tr>
<td>Submarine Nodes</td>
<td>rhel13-rhel16</td>
</tr>
<tr>
<td>CDSW Nodes</td>
<td>rhel18-rhel23</td>
</tr>
</tbody>
</table>
Customize Services

This section shows the tabs that manage configuration settings for Hadoop components. The wizard attempts to set reasonable defaults for each of the options here, but this can be modified to meet specific requirements. The following sections provide configuration guidance that should be refined to meet specific use case requirements.

The following changes need to be made:

- Memory and service level settings for each component and service level tuning.
- Customize the log locations of all the components to make sure growing logs do not cause the SSDs to run out of space.

Credentials

Specify the credentials as per your organizational policies for services in CREDENTIALS tab as shown below:
Databases

In the DATABASES tab, to configure the database for DRUID, HIVE, OOZIE, and RANGER, follow these steps:

1. Configure DRUID as shown below:

   ![DRUID Configuration Screen]

   - **DRUID META DATA STORAGE**
     - **Druid Meta data storage database name**: druid
     - **Druid Meta data storage type**: POSTGRESQL
     - **Meta data storage user**: druid
     - **Meta data storage password**: [Redacted]
     - **Meta data storage hostname**: rhel2.hsp3.cisco.com
     - **Meta data storage port**: 5432
     - **Meta data storage connector url**: jdbc:postgresql://rhel2.hsp3.cisco.com:5432/druid

2. Change the default log location by finding the Log Dir property and modifying the /var prefix to /data/disk1.

   ![Log Directory Configuration]

   - **druid_log_dir**: /data/disk1/log/druid
   - **Druid PID dir**: /var/run/druid
3. Configure HIVE:
   a. Select Existing PostgreSQL database in the Hive Database drop-down list.
   b. Enter Database Name as hive
   c. Enter Database Username as hive
   d. Enter Database URL as jdbc:postgresql://rhel2.hdp3.cisco.com/hive
   e. Enter Database password (use the password created during hive database setup in earlier steps; for example, big-data)
   f. Click TEST CONNECTION to verify the connectivity
   g. In Advanced tab, Change the default log location by filtering the Log Dir property and modifying the /var prefix to /data/disk1.
   h. Change the WebHCat log directory by filtering the Log Dir property and modifying the /var prefix to /data/disk1.

4. Configure OOZIE:
   a. Select Existing PostgreSQL database in the Hive Database drop-down list.
   b. Enter Database Name as oozie.
   c. Enter Database Username as oozie.
   e. Enter Database password (use the password created during hive database setup in earlier steps; for example, big-data).
   f. Click TEST CONNECTION to verify the connectivity.
   g. In Advanced tab, Change the default log location by filtering the Log Dir property and modifying the /var prefix to /data/disk1.
5. Configure RANGER:
   a. Select POSTGRES from DB FLAVOR drop-down list.
   b. Provide a Ranger DB name. For example, ranger.
   c. Provide Ranger DB Username such as rangeradmin.
   d. Enter JDBC connect string for a Ranger Database as jdbc:postgresql://rhel2.dhp3.cisco.com:54323/ranger.
   e. Ranger DB Host as rhel2.hdp3.cisco.com.
   f. Enter Ranger DB password. (Ranger database is not previously created. provide password string that would be configured in the DB. For example, bigdata).
   g. Select “Yes” for Setup Database and Database User.
   h. Enter “postgres” in Database Administrator (DBA) username.
   i. Enter Database Administrator (DBA) password.
   k. Update Ranger Admin Log Dir from /var to /data/disk1.
1. In Ambari, select the HDFS Service tab and use the “Search” box to filter for the properties mentioned in Table 11 and update their values.

2. Update the following HDFS configurations in Ambari.

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>NameNode Java Heap Size</td>
<td>4096</td>
</tr>
<tr>
<td>Hadoop maximum Java heap size</td>
<td>4096</td>
</tr>
<tr>
<td>DataNode maximum Java heap size</td>
<td>4096</td>
</tr>
<tr>
<td>Datanode failed disk toleration</td>
<td>5</td>
</tr>
</tbody>
</table>

3. Change the default log location by filtering the Log Dir property and modifying the /var prefix to /data/disk1.
MapReduce2

1. In Ambari, choose the MapReduce Service tab and update the values as shown below.

2. Under the MapReduce2 tab, change the default log location by finding the Log Dir property and modifying the /var prefix to /data/disk1.

YARN

1. In Ambari, select the YARN Service, and update the following as shown in Table 12.

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>ResourceManager Java heap size</td>
<td>4096</td>
</tr>
<tr>
<td>NodeManager Java heap size</td>
<td>2048</td>
</tr>
<tr>
<td>YARN Java heap size</td>
<td>4096</td>
</tr>
</tbody>
</table>
2. Under YARN tab, change the default log location by filtering the Log Dir property and modifying the /var prefix to /data/disk1.

YARN requires other configurations such as config group, node labeling, enabling docker runtime, CPU/GPU scheduling and isolation, and so on, which can be found in section High Availability for HDFS NameNode and YARN Resource Manager.

High Availability for NameNode and YARN Resource Manager can be configured using Ambari or also on non-Ambari clusters. This deployment guide explains the configuration of high availability using Ambari – Use the Ambari wizard interface to configure HA of the components.

The Ambari web UI provides a wizard-driven user experience that allows to configure high availability of the components in many Hortonworks Data Platform (HDP) stack services. The high availability of the components are achieved by setting up primary and secondary components. In the event that the primary component fails or becomes unresponsive, services failover to secondary component. After configuring the high availability for a service, Ambari enables you to manage and disable (roll back) the high availability of components within that service.

Configure the HDFS NameNode High Availability

The HDFS NameNode high availability feature enables you to run redundant NameNodes in the same cluster in an Active/Passive configuration with a hot standby. This eliminates the NameNode as a potential single point of failure (SPOF) in an HDFS cluster. With the release of Hadoop 3.0, you can configure more than one backup NameNode.

Prior to the release of Hadoop 2.0, the NameNode represented a single point of failure (SPOF) in an HDFS cluster. Each cluster had a single NameNode, and if that machine or process became unavailable, the cluster as a whole would be unavailable until the NameNode was either restarted or brought up on a separate machine. This situation impacted the total availability of the HDFS cluster in two major ways:

- In the case of an unplanned event such as a machine crash, the cluster would be unavailable until an operator restarted the NameNode.
- Planned maintenance events such as software or hardware upgrades on the NameNode machine would result in periods of cluster downtime.
HDFS NameNode High Availability avoids this by facilitating either a fast failover to one or more backup NameNodes during machine crash, or a graceful administrator-initiated failover during planned maintenance.

Secondary NameNode is not required in high availability configuration because the Standby node also performs the tasks of the Secondary NameNode.

HBase
Under the HBase tab, change the default log location by finding the Log Dir property and modifying the /var prefix to /data/disk1.

Zookeeper
Under the Zookeeper tab, change the default log location by filtering the Log Dir property and modifying the /var prefix to /data/disk1.

Storm
Under the Storm tab, change the default log location by finding the Log Dir property and modifying the /var prefix to /data/disk1.

Ambari Metrics
1. Choose the Ambari Metrics Service and expand the General tab and make the changes shown below.
2. Enter the Grafana Admin password as per organizational policy.
3. Change the default log location for Metrics Collector, Metrics Monitor and Metrics Grafana by finding the Log Dir property and modifying the /var prefix to /data/disk1.
4. Change the default data dir location for Metrics Grafana by finding the data Dir property and modifying the /var prefix to /data/disk1.
Accumulo

Select Accumulo Service and change the default log location by finding the Log Dir property and modifying the /var prefix to /data/disk1.

Advanced accumulo-env

Accumulo Log Dir

/data/disk1/log/accumulo

Atlas

Under the Atlas tab, change the default log location by finding the Log Dir property and modifying the /var prefix to /data/disk1.
Kafka

Under the Kafka tab, change the default log location by finding the Log Dir property and modifying the /var prefix to /data/disk1.

---

Advanced kafka-env

<table>
<thead>
<tr>
<th>Property</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kafka Log directory</td>
<td>/data/disk1/log/kafka</td>
</tr>
</tbody>
</table>

---

Knox

1. Select the Knox Service tab and expand the Knox gateway tab and make the changes shown below.
2. Enter the Knox Master Secret password as per organizational policy.
3. For Knox, change the gateway port to 8444 to ensure no conflicts with local HTTP server.

---

Knox Gateway

<table>
<thead>
<tr>
<th>Property</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Knox Gateway host</td>
<td>rhel.hdp3.cisco.com</td>
</tr>
<tr>
<td>Knox Master Secret</td>
<td>........................</td>
</tr>
</tbody>
</table>

---

Advanced gateway site

<table>
<thead>
<tr>
<th>Property</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>gateway port</td>
<td>8444</td>
</tr>
</tbody>
</table>

---

SmartSense

The SmartSense account requires the Hortonworks support subscription. Subscribers can populate the properties as shown below:
Spark

Select the Spark tab, change the default log location by finding the Log Dir property and modifying the /var prefix to /data/disk1.

Advanced livy2-env

- Livy2 Log directory: /data/disk1/log/livy2
- Livy2 PID directory: /var/run/livy2

Advanced spark2-env

- Spark Log directory: /data/disk1/log/spark2
- Spark PID directory: /var/run/spark2

Review

The assignments that have been made are displayed. Check to make sure all is correct before clicking the Deploy button. If any changes are necessary, use the left navigation bar to return to the appropriate screen.
Deploy

1. When the review is complete, click **DEPLOY**.

2. Follow the onscreen installation process. Watch for warnings and failures by clicking the link as shown below:
Summary of the Installation Process

1. On the Summary page click “COMPLETE.”
High Availability for HDFS NameNode and YARN ResourceManager

High availability for NameNode and YARN Resource Manager can be configured using Ambari or also on non-Ambari clusters. This deployment guide covers the configuration of high availability using Ambari – Use the Ambari wizard interface to configure HA of the components.

The Ambari web UI provides a wizard-driven user experience that allows to configure high availability of the components in many Hortonworks Data Platform (HDP) stack services. The high availability of the components are achieved by setting up primary and secondary components. In the event that the primary component fails or becomes unresponsive, services failover to secondary component. After configuring the high availability for a service, Ambari enables you to manage and disable (roll back) the high availability of components within that service.

Configure the HDFS NameNode High Availability

The HDFS NameNode high availability feature enables you to run redundant NameNodes in the same cluster in an Active/Passive configuration with a hot standby. This eliminates the NameNode as a potential single point of failure (SPOF) in an HDFS cluster. With the release of Hadoop 3.0, you can configure more than one backup NameNode.

Prior to the release of Hadoop 2.0, the NameNode represented a single point of failure (SPOF) in an HDFS cluster. Each cluster had a single NameNode, and if that machine or process became unavailable, the cluster as a whole would be unavailable until the NameNode was either restarted or brought up on a separate machine. This situation impacted the total availability of the HDFS cluster in two major ways:

- In the case of an unplanned event such as a machine crash, the cluster would be unavailable until an operator restarted the NameNode.
- Planned maintenance events such as software or hardware upgrades on the NameNode machine would result in periods of cluster downtime.

HDFS NameNode HA avoids this by facilitating either a fast failover to one or more backup NameNodes during machine crash, or a graceful administrator-initiated failover during planned maintenance.

Secondary NameNode is not required in high availability configuration because the Standby node also performs the tasks of the Secondary NameNode.

Active NameNode honors all the client requests and the Standby NameNode acts as a backup. The Standby NameNode keeps its state synchronized with Active NameNode through a group of JournalNodes (JNs). When the Active node performs any namespace modification, the Active node durably logs a modification record to a majority of these JNs. The Standby node reads the edits from the JNs and continuously watches the JNs for changes to the edit log.

Prerequisites for NameNode High Availability

The following are the prerequisites for NameNode high availability:

- NameNode Machine: Hardware for Active and Standby node should be exactly identical.
- JournalNodes Machine: JournalNode daemon is relatively lightweight, therefore it can be co-located on machines with other Hadoop daemons; it is typically located on the management nodes.
- There MUST be at least three JournalNodes, because the edit log modifications must be written to a majority of JNs. This allows the system tolerate failure of a single machine. You may also run more than three JournalNodes, but in order to increase the number of failures that the system can tolerate, you must run an odd number of JNs (3, 5, 7, and so on).
- **ZooKeeper Machines:** For automatic failover capability, an existing Zookeeper cluster must exist. The Zookeeper service can also co-exist with other Hadoop daemons.

- **In HA Cluster,** the Standby NameNode also performs the checkpoints of the namespace state, therefore do not deploy a Secondary NameNode, CheckpointNode, or BackupNode in an high availability cluster.

**Deploy the NameNode High Availability Cluster**

To deploy the NameNode high availability on a Ambari managed cluster, follow these steps:

---

High availability cannot accept HDFS cluster names that include underscore (_).

1. Log into Ambari. Click HDFS > CONFIGS. Click the ACTIONS drop-down list and click Enable NameNode HA to launch the wizard.

Figure 36  Enable NameNode HA

2. Step 1 launches the Enable NameNode HA wizard. On the Get Started page, specify the Nameservice ID as shown below. Click Next.
3. On the Select Hosts page, select the Additional NameNode and JournalNode. Click Next.

4. On the Review page, confirm the selection. To change any values, click Back, or to continue click Next.
Figure 39   Enable NameNode HA Wizard – Review

Figure 40   Enable NameNode HA Wizard – Create Checkpoint

5. Create a checkpoint on the NameNode on the Linux server (rhel1.hdp3.cisco.com) as shown below:

6. SSH to current NameNode, rhel1.hdp3.cisco.com and run the following commands:

```
[root@rhel1 ~]# sudo su hdfs -l -c 'hdfs dfsadmin -safemode enter'
Safe mode is ON
[root@rhel1 ~]# sudo su hdfs -l -c 'hdfs dfsadmin -saveNamespace'
```
Save namespace successful
[root@rhel1 ~]#

Figure 41  Current NameNode – Safe Mode and Create Checkpoint Command

7. Return to the Ambari web UI, verify that the Checkpoint was created. Click Next.

8. See the progress bar on the Configure Components page. When the configuration steps are completed, click Next.

Figure 42  Enable NameNode HA Wizard – Configure Components

9. Initialize the JournalNodes as shown below:
10. SSH to the current NameNode, for example rhel1.hdp3.cisco.com.

11. Run the following command:

```
[root@rhel ~]# sudo su hdfs -l -c 'hdfs namenode -initializeSharedEdits'
```

12. Return to the Ambari UI, when Ambari detects success, click Next.

13. On the Start Components page, when completed, click Next.
14. On the Initialize Metadata page, add the information as shown below:

15. SSH to rhel1.hdp3.cisco.com and run the following command:

```
[root@rhel1 ~]# sudo su hdfs -l -c 'hdfs zkfc -formatZK'
```
16. SSH to an additional NameNode, for example, rhel3.hdp3.cisco.com and run the following command:

```
[root@rhel3 ~]# sudo su hdfs -l -c 'hdfs namenode -bootstrapStandby'
```

17. Return to the Ambari web UI, click NEXT. Click OK on the confirmation pop-up window. Make sure the initialization of metadata was performed in NameNode and an additional NameNode as mentioned in step 15 and 16.

18. On the Finalize HA Setup page, you can see the progress bar as the high availability completes.
19. Click COMPLETE when done.

20. Click HDFS > SUMMARY tab, verify the Active and Standby NameNode. The Quick Links pane also shows that rhel1.hdp3.cisco.com is running the Active NameNode and rhel3.hdp3.cisco.com is running in Standby NameNode.

Configure the YARN ResourceManger HA

This section provides instructions on setting up the ResourceManager (RM) HA feature in a HDFS cluster. The Active and Standby ResourceManagers embed the ZooKeeper based ActiveStandbyElector to determine which RM should be active.

Prerequisites for ResourceManager HA

The following are the prerequisites for ResourceManager HA:
- The servers where Active and Standby RMs are run should have identical hardware.
- For automated failover configurations, there must be an existing Zookeeper cluster available. The ZooKeeper service nodes can be co-located with the other Hadoop daemons.

At least three ZooKeeper servers must be running.

Deploy the ResourceManager HA

ResourceManager HA can be configured manually or through Ambari. These instructions are based on configuring ResourceManager HA using Ambari.

To setup ResourceManager HA, follow these steps:

1. From the Ambari web UI, click Services > YARN. Click the ACTIONS drop-down list and select Enable ResourceManager HA.

   ![Services/YARN - Enable ResourceManager HA](image)

   Figure 52 Services/YARN - Enable ResourceManager HA

2. This launches the ResourceManager HA wizard as shown below. Click NEXT.

   ![Enable ResourceManager HA Wizard – Get Started](image)

   Figure 53 Enable ResourceManager HA Wizard – Get Started

3. From the Select Host page, select the host for Additional Resource Manager. Click NEXT.
4. Proceed to the Review page.

5. The Configure Components page shows the progress bar as the Additional ResourceManager is being deployed.
6. Click COMPLETE when done.

It was observed that in certain circumstances, services might fail to restart. Click COMPLETE and restart the services in Ambari dashboard.

7. Verify the ResourceManager HA setup by clicking Services > YARN > SUMMARY tab. The Quick Links pane identifies Active and Standby ResourceManager.
Figure 57  Services/YARN/Summary – Verify ResourceManger HA

Summary

Components

- Started TIMELINE SERVICES 1.5
- Started ACTIVE RESOURCE MANAGER
- Started STANDBY RESOURCE MANAGER
- Started TIMELINE SERVICES 2.0

16/16 Started NODERMANAGERS
18 Installed YARN CLIENTS

13m
RESOURCE MANAGER uptime

NODERMANAGER STATUS

- Active: 16
- Lost: 0
- Unhealthy: 0
- Reconnect: 0

Quick Links

- App Engine UI
- Resource Manager UI
- Resource Manager logs
- Resource Manager UI logs
- Resource Manager JMX
- Thread Stacks
- YARN UI Engine (sandbox)
Bill of Materials

This section provides the BOM for the 16 Nodes Hadoop Base Rack, 6 Nodes Data Science (AI/ML) Rack and 8 Nodes Hadoop tiered storage. See Table 13 for BOM for the Hadoop Base rack, Table 14 for BOM for Data Science(AI/ML) Expansion Rack, Table 15 for BOM for Hadoop Tiered Storage. Table 16, Table 17, and Table 18 for software components. Table 19 lists Cloudera SKUs available from Cisco.

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>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>16</td>
</tr>
<tr>
<td>CON-OSP-C240M5A2</td>
<td>SNTC 24X7X4OS UCS C240 M5 A2</td>
<td>16</td>
</tr>
<tr>
<td>UCS-CPU-I6230</td>
<td>2.1 GHz 6230/125W 20C/27.5MB Cache/DDR4 2933MHz</td>
<td>32</td>
</tr>
<tr>
<td>UCS-MR-X32G2RT-H</td>
<td>32GB DDR4-2933-MHz RDIMM/2Rx4/1.2v</td>
<td>192</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>16</td>
</tr>
<tr>
<td>UCSC-MLOM-C40Q-03</td>
<td>Cisco VIC 1387 Dual Port 40Gb QSFP CNA MLOM</td>
<td>16</td>
</tr>
<tr>
<td>UCSC-PSU1-1600W</td>
<td>Cisco UCS 1600W AC Power Supply for Rack Server</td>
<td>32</td>
</tr>
<tr>
<td>CAB-9K12A-NA</td>
<td>Power Cord, 125VAC 13A NEMA 5-15 Plug, North America</td>
<td>32</td>
</tr>
<tr>
<td>UCSC-RAILB-M4</td>
<td>Ball Bearing Rail Kit for C220 &amp; C240 M4 &amp; M5 rack servers</td>
<td>16</td>
</tr>
<tr>
<td>CIMC-LATEST</td>
<td>IMC SW (Recommended) latest release for C-Series Servers.</td>
<td>16</td>
</tr>
<tr>
<td>UCSC-HS-C240M5</td>
<td>Heat sink for UCS C240 M5 rack servers 150W CPUs &amp; below</td>
<td>32</td>
</tr>
<tr>
<td>UCSC-BBLKD-S2</td>
<td>UCS C-Series M5 SFF drive blanking panel</td>
<td>416</td>
</tr>
<tr>
<td>UCSC-PCIF-240M5</td>
<td>C240 M5 PCIe Riser Blanking Panel</td>
<td>16</td>
</tr>
<tr>
<td>CBL-SC-MR12GM5P</td>
<td>Super Cap cable for UCSC-RAID-M5HD</td>
<td>16</td>
</tr>
<tr>
<td>UCSC-SCAP-M5</td>
<td>Super Cap for UCSC-RAID-M5, UCSC-MRAID2GB-KIT</td>
<td>16</td>
</tr>
<tr>
<td>UCSC-RAID-M5HD</td>
<td>Cisco 12G Modular RAID controller with 4GB cache</td>
<td>16</td>
</tr>
<tr>
<td>UCS-SP-FI6332-2X</td>
<td>UCS SP Select 2 x 6332 FI</td>
<td>1</td>
</tr>
<tr>
<td>UCS-SP-FI6332</td>
<td>(Not sold standalone) UCS 6332 1RU FI/12 QSFP+</td>
<td>2</td>
</tr>
<tr>
<td>CON-OSP-SPI6332</td>
<td>ONSITE 24X7X4 (Not sold standalone) UCS 6332 1RU FI/No PSU/3</td>
<td>2</td>
</tr>
<tr>
<td>UCS-PSU-6332-AC</td>
<td>UCS 6332 Power Supply/100-240VAC</td>
<td>4</td>
</tr>
<tr>
<td>CAB-9K12A-NA</td>
<td>Power Cord, 125VAC 13A NEMA 5-15 Plug, North America</td>
<td>4</td>
</tr>
<tr>
<td>QSFP-H40G-CU3M</td>
<td>40GBASE-CR4 Passive Copper Cable, 3m</td>
<td>16</td>
</tr>
</tbody>
</table>

155
<table>
<thead>
<tr>
<th>Part Number</th>
<th>Description</th>
<th>Qty</th>
</tr>
</thead>
<tbody>
<tr>
<td>QSFP-40G-SR-BD</td>
<td>QSFP40G BiDi Short-reach Transceiver</td>
<td>8</td>
</tr>
<tr>
<td>N50-MGT015</td>
<td>UCS Manager v3.2(1)</td>
<td>2</td>
</tr>
<tr>
<td>UCS-ACC-6332</td>
<td>UCS 6332 Chassis Accessory Kit</td>
<td>2</td>
</tr>
<tr>
<td>UCS-FAN-6332</td>
<td>UCS 6332 Fan Module</td>
<td>8</td>
</tr>
<tr>
<td>QSFP-H40G-CU3M=</td>
<td>40GBase-CR4 Passive Copper Cable, 3m</td>
<td>32</td>
</tr>
<tr>
<td>UCS-HD24TB10K4KN</td>
<td>2.4 TB 12G SAS 10K RPM SFF HDD (4K)</td>
<td>312</td>
</tr>
<tr>
<td>UCS-SP-H1P8TB</td>
<td>1.8 TB 12G SAS 10K RPM SFF HDD (4K)</td>
<td>384</td>
</tr>
<tr>
<td>UCS-SP-HD-1P8T-2</td>
<td>1.8TB 12G SAS 10K RPM SFF HDD (4K) 2 Pack</td>
<td>15</td>
</tr>
</tbody>
</table>

Table 14  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>6</td>
</tr>
<tr>
<td>CON-OSP-C240M5A2</td>
<td>SNTEC 2x7X4OS UCS C240 M5 A2</td>
<td>6</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>UCS-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>UCS-MLOM-C40O-03</td>
<td>Cisco VIC 1387 Dual Port 40Gb QSFP CNA MLOM</td>
<td>8</td>
</tr>
<tr>
<td>UCS-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>UCS-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>UCS-HS-C240M5</td>
<td>Heat sink for UCS C240 M5 rack servers 150W CPUs and below</td>
<td>16</td>
</tr>
<tr>
<td>UCS-SCBLOKD-S2</td>
<td>UCS C-Series M5 SFF drive blanking panel</td>
<td>208</td>
</tr>
<tr>
<td>UCS-PCIF-C240M5</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-M5HD</td>
<td>8</td>
</tr>
<tr>
<td>UCS-SCAP-M5</td>
<td>Super Cap for UCSC-RAID-M5, UCSC-MRAID2GB-KIT</td>
<td>8</td>
</tr>
<tr>
<td>UCS-RAID-M5HD</td>
<td>Cisco 12G Modular RAID controller with 4GB cache</td>
<td>8</td>
</tr>
<tr>
<td>UCS-SP-H2P8TB-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-H2P8TB</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 15  Bill of Materials for Hadoop Nodes Tiered Storage Rack

<table>
<thead>
<tr>
<th>Part Number</th>
<th>Description</th>
<th>Qty</th>
</tr>
</thead>
<tbody>
<tr>
<td>UCSS-S3260</td>
<td>Cisco UCS S3260 Storage Server Base Chassis</td>
<td>4</td>
</tr>
<tr>
<td>CON-OSP-UCSS3260</td>
<td>SNTEC 24x7x4OS, Cisco UCS S3260 Storage Server Base Chassis</td>
<td>4</td>
</tr>
<tr>
<td>UCSC-PSU1-1050W</td>
<td>Cisco UCS 1050W AC Power Supply for Rack Server</td>
<td>16</td>
</tr>
<tr>
<td>CAB-N5K6A-NA</td>
<td>Power Cord, 200/240V 6A North America</td>
<td>16</td>
</tr>
<tr>
<td>CIMC-LATEST</td>
<td>IMC SW (Recommended) latest release for C-Series Servers.</td>
<td>4</td>
</tr>
<tr>
<td>UCSC-C3X60-RAIL</td>
<td>UCS C3X60 Rack Rails Kit</td>
<td>4</td>
</tr>
<tr>
<td>UCSS-S3260-BBEZEL</td>
<td>Cisco UCS S3260 Bezel</td>
<td>4</td>
</tr>
<tr>
<td>N20-BBLKD-7MM</td>
<td>UCS 7MM SSD Blank Filler</td>
<td>8</td>
</tr>
<tr>
<td>N20-BKVM</td>
<td>KVM local IO cable for UCS servers console port</td>
<td>8</td>
</tr>
<tr>
<td>UCSC-C3260-SIOC</td>
<td>Cisco UCS C3260 System IO Controller with VIC 1300 incl.</td>
<td>4</td>
</tr>
<tr>
<td>UCSC-C3260-SIOC</td>
<td>Cisco UCS C3260 System IO Controller with VIC 1300 incl.</td>
<td>4</td>
</tr>
<tr>
<td>UCS-S3260-M5SRB</td>
<td>UCS S3260 M5 Server Node for Intel Scalable CPUs</td>
<td>4</td>
</tr>
<tr>
<td>UCS-S3260-DRAID</td>
<td>UCS S3260 Dual Raid based on LSI 3316</td>
<td>4</td>
</tr>
<tr>
<td>UCS-S3260-M5HS</td>
<td>UCS S3260 M5 Server Node HeatSink</td>
<td>8</td>
</tr>
<tr>
<td>UCS-S3260-M5SRB</td>
<td>UCS S3260 M5 Server Node for Intel Scalable CPUs</td>
<td>4</td>
</tr>
<tr>
<td>UCS-S3260-DRAID</td>
<td>UCS S3260 Dual Raid based on LSI 3316</td>
<td>4</td>
</tr>
<tr>
<td>UCS-S3260-M5HS</td>
<td>UCS S3260 M5 Server Node HeatSink</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>48</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>UCS-C3K-HD4TB</td>
<td>UCS C3000 4TB NL-SAS 7200 RPM 12Gb HDD w Carrier- Top Load</td>
<td>224</td>
</tr>
<tr>
<td>UCS-S3260-G3SD48</td>
<td>UCS S3260 480G Boot SSD (Micron 6G SATA)</td>
<td>16</td>
</tr>
<tr>
<td>UCS-S3260-56HD4</td>
<td>Cisco UCS C3X60 Four row of drives containing 56 x 4TB</td>
<td>4</td>
</tr>
<tr>
<td>UCS-CPU-I5220</td>
<td>Intel 5220 2.2GHz/125W 18C/24.75MB DCP DDR4 2666 MHz</td>
<td>8</td>
</tr>
<tr>
<td>UCS-CPU-I5220</td>
<td>Intel 5220 2.2GHz/125W 18C/24.75MB DCP DDR4 2666 MHz</td>
<td>8</td>
</tr>
<tr>
<td>RACK2-UCS2</td>
<td>Cisco R42612 standard rack, w/side panels</td>
<td>1</td>
</tr>
</tbody>
</table>
### Table 16  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>CON-SNT-RCK2UCS2</td>
<td>SNTC 8X5XNBD, Cisco R42612 standard rack, w side panels</td>
<td>1</td>
</tr>
<tr>
<td>UCS-SP-C240M5-A2</td>
<td>SP C240 M5SX w/2x6132,6x32GB mem, VIC1387</td>
<td>6</td>
</tr>
<tr>
<td>CON-OSP-C240M5A2</td>
<td>SNTC 24X7X4OS UCS C240 M5 A2</td>
<td>6</td>
</tr>
<tr>
<td>UCS-CPU-i6230</td>
<td>2.1 GHz 6230/125W 20C/27.5MB Cache/DDR4 2933MHz</td>
<td>12</td>
</tr>
<tr>
<td>UCS-MR-X32G2RT-H</td>
<td>32GB DDR4-2933-MHz RDIMM/2Rx4/1.2v</td>
<td>72</td>
</tr>
<tr>
<td>UCSC-MLOM-C40Q-03</td>
<td>Cisco VIC 1387 Dual Port 40Gb QSFP CNA MLOM</td>
<td>6</td>
</tr>
<tr>
<td>UCSC-PSU1-1600W</td>
<td>Cisco UCS 1600W AC Power Supply for Rack Server</td>
<td>12</td>
</tr>
<tr>
<td>CAB-9K12A-NA</td>
<td>Power Cord, 125VAC 13A NEMA 5-15 Plug, North America</td>
<td>12</td>
</tr>
<tr>
<td>UCSC-RAILB-M4</td>
<td>Ball Bearing Rail Kit for C220, C240 M4 and M5 rack servers</td>
<td>6</td>
</tr>
<tr>
<td>CIMC-LATEST</td>
<td>IMC SW (Recommended) latest release for C-Series Servers.</td>
<td>6</td>
</tr>
<tr>
<td>UCSC-HS-C240M5</td>
<td>Heat sink for UCS C240 M5 rack servers 150W CPUs &amp; below</td>
<td>12</td>
</tr>
<tr>
<td>UCSC-BBLKD-S2</td>
<td>UCS C-Series M5 SFF drive blanking panel</td>
<td>156</td>
</tr>
<tr>
<td>CBL-SC-MR12GM5P</td>
<td>Super Cap cable for UCSC-RAID-M5HD</td>
<td>6</td>
</tr>
<tr>
<td>UCSC-SCAP-M5</td>
<td>Super Cap for UCSC-RAID-M5, UCSC-MRAID1GB-KIT</td>
<td>6</td>
</tr>
<tr>
<td>UCSC-RAID-M5HD</td>
<td>Cisco 12G Modular RAID controller with 4GB cache</td>
<td>6</td>
</tr>
<tr>
<td>UCSC-PCI-2A-240M5</td>
<td>Riser 2A including 3 PCIe slots (x8, x16, x16) supports GPU</td>
<td>6</td>
</tr>
<tr>
<td>UCS-SP-SD-1P6TB-4X</td>
<td>UCS SP 1.6TB 2.5inch Ent. Perf 12G SAS SSD (10Xendurance)4Pk</td>
<td>6</td>
</tr>
<tr>
<td>UCS-SP-SD-1P6TB</td>
<td>1.6TB 2.5inch Ent. Performance 12G SAS SSD (10X endurance)</td>
<td>24</td>
</tr>
<tr>
<td>UCSC-GPU-T4-16=</td>
<td>NVIDIA Tesla T4 16GB</td>
<td>12</td>
</tr>
<tr>
<td>CON-SNT-CGPUT416</td>
<td>SNTC-24X7X4OS NVIDIA T4 PCIE 75W 16GB</td>
<td>12</td>
</tr>
</tbody>
</table>

### Table 17  Red Hat Enterprise Linux License

<table>
<thead>
<tr>
<th>Part Number</th>
<th>Description</th>
<th>Qty</th>
</tr>
</thead>
<tbody>
<tr>
<td>RHEL-2S2V-3A</td>
<td>Red Hat Enterprise Linux</td>
<td>30</td>
</tr>
<tr>
<td>CON-ISV1-EL2S2V3A</td>
<td>3 year Support for Red Hat Enterprise Linux</td>
<td>30</td>
</tr>
</tbody>
</table>

### Table 18  Cloudera Data Science Workbench Software

<table>
<thead>
<tr>
<th>Part Number</th>
<th>Description</th>
<th>Qty</th>
</tr>
</thead>
<tbody>
<tr>
<td>UCS-BD-CDSWB=</td>
<td>UCS-BD-CDSWB-1Y Cloudera Data Science Work Bench, 10-user pack - 1 Year</td>
<td></td>
</tr>
</tbody>
</table>
Table 19  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>
</tbody>
</table>
Appendix – A

Configure Data Drives on Name Node and Other Management Nodes

This section describes the steps needed to configure non-OS disk drives as RAID1 using the StorCli command. All drives are 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.

To configure data drives on Name node and other nodes, If the drive state shows up as JBOD, creating RAID in the subsequent steps will fail with the error “The specified physical disk does not have the appropriate attributes to complete the requested command.”

To configure data drives on Name Node and others, follow these steps:

1. If the drive state shows up as JBOD, it can be converted into Unconfigured Good using Cisco UCSM or storcli64 command. Following steps should be performed if the state is JBOD.

2. Get the enclosure id as follows:

   ```
   ansible all -m shell -a './storcli64 pdlist -a0 | grep Enc | grep -v 252 | awk '{print $4}' | sort | uniq -c | awk '{print $2}''
   ```

   It is observed that some earlier versions of storcli64 complains about above mentioned command as if it is deprecated. In this case, please use "./storcli64 /c0 show all| awk '{print $1}'| sed -n '/[0-9]:[0-9]/p'|awk '{print substr($1,1,2)}'|sort -u" command to determine enclosure id.

3. Convert to unconfigured good:

   ```
   ansible datanodes -m command -a './storcli64 /c0 /e66 /sall set good force'
   ```

4. Verify status by running the following command:

   ```
   # ansible datanodes -m command -a './storcli64 /c0 /e66 /sall show'
   ```

5. Run this script as root user on rhel01 to rhel3 to create the virtual drives for the management nodes:

   ```
   #vi /root/raid1.sh
   ```

The script (above) requires enclosure ID as a parameter.

6. 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}'
```

7. Run MegaCli script:

```
./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
```

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

8. Run the following command. State should change to Online:

```
ansible namenodes -m command -a "./storcli64 /c0 /e66 /sall show"
```

9. State can also be verified in UCSM as show below in Equipment>Rack-Mounts>Servers>Server # under Inventory/Storage/Disk tab:

![UCSM Configuration Screen](image)

Configure Data Drives on Data Nodes

To configure non-OS disk drives as individual RAID0 volumes using StorCli command, follow these steps. These volumes will be used for HDFS Data.

1. Issue the following command from the admin node to create the virtual drives with individual RAID 0 configurations on all the data nodes:

```
[root@rhel01 ~]# ansible datanodes -m command -a "./storcli64 -cfgeachdskraid0 WB RA direct NoCachedBadBBU strpsz1024 -a0"
```

```
rhel7.hdp3.cisco.local | SUCCESS | rc=0 >>
Adapter 0: Created VD 0
```
The command (above) will not override existing configurations. To clear and reconfigure existing configurations, refer to the Embedded MegaRAID Software Users Guide available at www.broadcom.com.

Cloudera Data Science Workbench (CDSW)

Cloudera Data Science Workbench runs on one or more dedicated gateway / edge hosts on HDP clusters. A gateway host is one that does not have any cluster services running on them. They only run the clients for cluster services (such as the HDFS Client, YARN Client, Spark2 Client, and so on). These clients ensure that Cloudera Data Science Workbench has all the libraries and configuration files necessary to securely access the HDP cluster and their respective services.

Cloudera Data Science Workbench does not support running any other services on these gateway hosts. Each gateway host must be dedicated solely to Cloudera Data Science Workbench. This is because user workloads require dedicated CPU and memory, which might conflict with other services running on these hosts.

From the gateway hosts assigned to Cloudera Data Science Workbench, one will serve as the master host, which also runs the CDSW web application, while others will serve as worker hosts. You should note that worker hosts are not required for a fully-functional Cloudera Data Science Workbench deployment. For proof-of-concept deployments you can deploy a 1-host cluster with just a Master host. The Master host can run user workloads just as a worker can.

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

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

This CVD incorporates 6x Cisco UCS C240 M5 with 2x T4 GPUs; the following steps in this section detail how to enable GPU as a Hadoop resource.

For supported platforms and requirements, go to: https://www.cloudera.com/documentation/data-science-workbench/latest/topics/cdsw_hdp.html

Figure 58 Cisco UCS C240 M5 resource inventory as seen through Cisco UCS Manager

Install the Prerequisites for CUDA

To install the prerequisites for CUDA, follow these steps:

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

These commands are run as root or sudo.

1. List GPUs and CPUs installed:

   [root@rhel1 ~]# ansible gpunodes -m shell -a "lspci | grep -i nvidia"
Install GCC

To install GCC, follow these steps:

1. Make sure gcc is installed in the system:

   ```bash
   [root@rhel1 ~]# ansible gpunodes -m shell -a "gcc --version"
   ```
Install Kernel Headers and Installation Packages

The CUDA Driver requires that the kernel headers and development packages for the running version of the kernel are installed at the time of the driver installation, as well as whenever the driver is rebuilt. For example, if your system is running kernel version 3.17.4-301, the 3.17.4-301 kernel headers and development packages must also be installed.

```bash
ansible gpunodes -m shell -a "uname -r"
```

```bash
[root@rhel1 ~]# ansible gpunodes -m shell -a "uname -r"
rhel22.hdp3.cisco.com | SUCCESS | rc=0 >>
3.10.0-957.el7.x86_64
rhel23.hdp3.cisco.com | SUCCESS | rc=0 >>
3.10.0-957.el7.x86_64
rhel18.hdp3.cisco.com | SUCCESS | rc=0 >>
3.10.0-957.el7.x86_64
```

```bash
[root@rhel1 ~]# ansible gpunodes -m yum -a "name=kernel-devel-$\{uname -r\} state=present"
```
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.
To install DKMS, follow these steps:

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-2.7.1-1.el7.noarch.rpm

# wget http://rpmfind.net/linux/epel/7/x86_64/Packages/d/dkms-2.7.1-1.el7.noarch.rpm

1. Copy dkms rpm to all the GPU servers:

   [root@rhel1 ~]# ansible gpunodes -m copy -a "src=/root/dkms-2.7.1-1.el7.noarch.rpm dest=/root/"

2. Install dkms with yum install:

   [root@rhel1 ~]# ansible gpunodes -m command -a "yum install -y /root/dkms-2.7.1-1.el7.noarch.rpm"

Install NVIDIA GPU Drivers

To install the NVIDIA GPU drivers, follow these steps:


2. For the NVIDIA driver download, select the product type, Series, Product, OS, and CUDA toolkit.

   For this deployment, select 10.1 for CUDA Toolkit.

3. Click SEARCH. The selected driver is shown below:
4. Click DOWNLOAD. The download link can be captured by right-clicking AGREE & DOWNLOAD as shown below.

5. Copy the .rpm file in all the GPU nodes as shown below.

6. Install the driver by running the following command:

Install CUDA

To install CUDA, follow these steps:

1. Download CUDA 10.1.
TensorFlow needs CUDA; make sure that the version of CUDA is supported by TensorFlow before installing CUDA. Earlier versions of CUDA are here: https://developer.nvidia.com/cuda-toolkit-archive.

The latest version of CUDA is available here:

2. Copy .rpm file to all GPU nodes as follows:

    [root@rhel ~]# ansible gpunodes -m copy -a "src=/root/cuda-repo-rhel7-10-1-local-10.1.168-1.x86_64.rpm dest=/root/.

3. Install CUDA in all GPU nodes by running the following set of commands:

    [root@rhel ~]# ansible gpunodes -m shell -a "rpm -i cuda-repo-rhel7-10-1-local-10.1.168-1.x86_64.rpm"
    [root@rhel ~]# ansible gpunodes -m shell -a "yum clean all"
    [root@rhel ~]# ansible gpunodes -m shell -a "yum -y install cuda"
Appendix – A

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

Download cuDNN 10.1
To download and set up NVIDIA cuDNN, follow these steps:


2. Copy cuDNN into all the GPU servers.

https://developer.nvidia.com/rdp/cudnn-archive

This step may require joining the NVIDIA developer community.

3. Run the following Ansible commands in all GPU nodes to setup cuDNN:

```
[root@rhel1 ~]# ansible gpunodes -m copy -a "src=/root/cudnn-10.1-linux-x64-v7.1.tgz dest=/root/"
[root@rhel1 ~]# ansible gpunodes -m shell -a "tar -xzf cudnn-10.1-linux-x64-v7.1.tgz"
[root@rhel1 ~]# ansible gpunodes -m shell -a "cp /root/cuda/include/cudnn.h /usr/local/cuda-10.1/include"
[root@rhel1 ~]# ansible gpunodes -m shell -a "cp /root/cuda/lib64/libcudnn* /usr/local/cuda-10.1/lib64"
[root@rhel1 ~]# ansible gpunodes -m shell -a "chmod a+r /usr/local/cuda-10.1/include/cudnn.h /usr/local/cuda-10.1/lib64/libcudnn*"
```

Post Installation Steps

1. Add CUDA in PATH and LD_LIBRARY_PATH variable in all the GPU nodes.

2. The PATH and LD_LIBRARY_PATH variable need to include /usr/local/<cuda>/bin:

```
export PATH=/usr/local/cuda-10.1/bin$PATH:+$PATH
export LD_LIBRARY_PATH=/usr/local/cuda-10.1/lib64$LD_LIBRARY_PATH:+$LD_LIBRARY_PATH
```

3. Reboot the GPU nodes:

```
# ansible gpunodes -a "/sbin/reboot"
```

The SSH and Ansible connection will disconnect with this command.

For more information, go to:

Verify Drivers
To verify the drivers have been installed, run the following commands in all GPU nodes as shown below:

```
[root@rhel1 ~]# ansible gpunodes -a "nvidia-smi"
Or on specific node
[root@rhel1 ~]# ansible gpunodes -a "nvidia-smi"
```
Installation Prerequisites for CDSW

To install the prerequisites for CDSW, complete the steps in the following sections on all CDSW nodes:

A

For more information, refer to: https://www.cloudera.com/documentation/data-science-workbench/latest/topics/cdsw_requirements_supported_versions.html#cdsw_requirements_supported_versions and https://www.cloudera.com/documentation/data-science-workbench/1-6-x/topics/cdsw_install.html

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.

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. For more information, refer to: https://www.cloudera.com/documentation/data-science-workbench/1-6-x/topics/cdsw_install.html#wildcard_dns

For the Solution validation we had installed DNS server on windows 2012 R2 and setup DNS wild card configuration on the same.

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

1. Update /etc/dnsmasq.conf to enable Wildcard entry:

   address=/cdsw/10.15.1.53
   #service dnsmasq restart

2. 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

For more information, go to: https://www.cloudera.com/documentation/data-science-workbench/latest/topics/cdsw_install.html#pre_install.

Supported JDK Version

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

Spark 2 which is needed by CDSW to run Spark jobs (on Hadoop nodes) requires JDK 1.8. On CSD-based deployments, the version of Java installed on Cloudera Data Science Workbench gateway hosts and also updating the JAVA home directory in the “cdsw.conf” file located in /etc/cdsw/config directory.

To update the JDK version, go to “Changing the JDK version on an Existing HDP Cluster”

For more details, go to: https://www.cloudera.com/documentation/enterprise/release-notes/topics/rn_consolidated_pcm.html#pcm_jdk

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.

```bash
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

For more information about Cloudera Data Science Workbench 1.5.x Requirements and Supported Platforms, go to:
https://www.cloudera.com/documentation/data-science-workbench/latest/topics/cdsw_requirements_supported_versions.html

Please refer to section Disable SELinux if SELinux is enabled.

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 HDP 3.1.0.

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 HDP 3.1.0

To download and install CDSW, complete the following steps:

1. Log into the Ambari Server
2. Go to the Hosts page and select Actions > + Add New Hosts.
3. On the Install Options page, enter the fully-qualified domain names for your new hosts.
The wizard also needs the private key file you created when you set up password-less SSH. Using the host names and key file information, the wizard can locate, access, and interact securely with all the hosts in the cluster. Alternatively, you can manually install and start the Ambari agents on all the new hosts.

4. Click Register and Confirm. For more detailed instructions, refer to "https://docs.hortonworks.com/HDPDocuments/Ambari-2.6.2.2/bk_ambari-installation/content/install_options.html"

5. The Confirm Hosts page prompts you to confirm that Ambari has located the correct hosts for your cluster and to check those hosts to make sure they have the correct directories, packages, and processes required to continue the install. When you are satisfied with the list of hosts, click Next.

For detailed instructions, go to https://docs.hortonworks.com/HDPDocuments/Ambari-2.6.2.2/bk_ambari-installation/content/confirm_hosts.html.

6. On the Assign Slaves and Clients page, select the Clients that should be installed on the new hosts. To install clients on all hosts, select the Client checkbox for every host. You can use the all option for each available client to expedite this.

Make sure no other services are running on these hosts. To make this easier, select the none option for all other services.

7. On the Configurations page, select the “configuration groups” for the new hosts.

8. The Review page displays the host assignments you have made. Check to make sure everything is correct. If you need to make changes, use the left navigation bar to return to the appropriate screen.

9. Click Deploy.

10. The Install, Start and Test page displays progress as the clients are installed and deployed on each host. When the process is complete, click Next.

11. The Summary page provides you a list of the accomplished tasks. Click Complete and you will be directed back to the Hosts page.

Create HDFS User Directories

To run workloads that leverage HDP cluster services, make sure that HDFS directories (/user/<username>) are created for each user so that they can seamlessly connect to HDP from Cloudera Data Science Workbench.

Follow these steps for each user directory that must be created:

1. SSH to a host in the cluster that includes the HDFS client.

2. Switch to the hdfs system account user:

   su - hdfs

3. Create an HDFS directory for the user. For example, you would create the following directory for the default user admin:

   hdfs dfs -mkdir /user/admin
4. Assign ownership of the new directory to the user. For example, for the new `/user/admin` directory, make the `admin` user the owner of the directory:

```
hdfs dfs -chown admin:hadoop /user/admin
```

**Install Cloudera Data Science Workbench on the Master Host**

To install Cloudera Data Science Workbench on the master host, follow these steps:

1. **Download Cloudera Data Science Workbench:**
   
   For non-airgapped installations, download this file and save it to `/etc/yum.repos.d/`: [cloudera-cdsw.repo](https://archive.cloudera.com/cdsw1/1.5.0/redhat7/yum/RPM-GPG-KEY-cloudera)

2. The Cloudera Public GPG repository key verifies that you are downloading genuine packages. Add the repository key:

   ```
sudo rpm --import https://archive.cloudera.com/cdsw1/1.5.0/redhat7/yum/RPM-GPG-KEY-cloudera
```

3. **Install the latest RPM with the following command:**

   ```
sudo yum install cloudera-data-science-workbench
```

4. **Initialize and start Cloudera Data Science Workbench:**

   ```
cdsw start
```

   The application will take a few minutes to bootstrap. You can watch the status of the application installation and startup with `watch cdsw status`.

**Install Cloudera Data Science Workbench on Worker Hosts**

To install Cloudera Data Science Workbench on worker hosts, follow these steps:

1. **Download Cloudera Data Science Workbench:**
   
   For non-airgapped installations, download this file and save it to `/etc/yum.repos.d/`: [cloudera-cdsw.repo](https://archive.cloudera.com/cdsw1/1.5.0/redhat7/yum/RPM-GPG-KEY-cloudera)

2. The Cloudera Public GPG repository key verifies that you are downloading genuine packages. Add the repository key:

   ```
sudo rpm --import https://archive.cloudera.com/cdsw1/1.5.0/redhat7/yum/RPM-GPG-KEY-cloudera
```

3. **Install the latest RPM with the following command:**

   ```
sudo yum install cloudera-data-science-workbench
```
4. Copy cdsw.conf file from the master host:

```
scp root@<cdsw-master-hostname.your_domain.com>:~/etc/cdsw/config/cdsw.conf /etc/cdsw/config/cdsw.conf
```

After initialization, the cdsw.conf file includes a generated bootstrap token that allows worker hosts to securely join the cluster. You can get this token by copying the configuration file from master and ensuring it has 600 permissions.

If your hosts have heterogeneous block device configurations, modify the Docker block device settings in the worker host configuration file after you copy it. Worker hosts do not need application block devices, which store the project files and database state, and this configuration option is ignored.

5. Create /var/lib/cdsw on the worker host. This directory must exist on all worker hosts. Without it, the next step that registers the worker host with the master will fail.

6. Unlike the master host, the /var/lib/cdsw directory on worker hosts does not need to be mounted to an Application Block Device. It is only used to store client configuration for HDP services on workers.

7. Initialize and start Cloudera Data Science Workbench:

```
cdsw join
```

This causes the worker hosts to register themselves with the Cloudera Data Science Workbench master host and increase the available pool of resources for workloads.

8. Return to the master host and verify the host is registered with this command:

```
cdsw status
```

**Create the Administrator Account**

After your installation is complete, set up the initial administrator account.

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

In this CVD, Kerberos is not enabled.

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.

```bash
# sudo -u hdfs hdfs dfs -mkdir /user/admin
# sudo -u hdfs hdfs dfs -chown admin:admin /user/admin
```

4. Go to the Settings on Cloudera Data Science WorkBench, Click on Hadoop Authentication > Hadoop Username Override. Enter admin. Click on Authenticate.
5. Restart CDSW from command-line using "cdsw stop" followed by "cdsw start".

**Use 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.

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

**Enable 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. Create your own custom CUDA-capable engine image using the instructions described in this topic.
- Cloudera Data Science Workbench gateway hosts. These depend on your GPU hardware and will have to be installed by your system administrator. The steps provided in this topic are generic guidelines that will help you evaluate your setup.
- The instructions described in this topic require Internet access. If you have an airgapped deployment, you will be required to manually download and load the resources onto your hosts.
- For a list of known issues associated with this feature, refer Known Issues - GPU Support.

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 removed and since by default CDSW version 1.5 does not come with nvidia-docker 1.0 plugins which is a thin wrapper around the Docker CLI and a Docker plugin, shall be installed separately.

1. Set the following parameter in /etc/cdsw/config/cdsw.conf on all Cloudera Data Science Workbench hosts. You must make sure that cdsw.conf is consistent across all hosts, irrespective of whether they have GPU hardware installed on them.

   | NVIDIA_GPU_ENABLE | Set this property to true to enable GPU support for Cloudera Data Science Workbench workloads. When this property is enabled on a host that is equipped with GPU hardware, the GPU(s) will be available for use by Cloudera Data Science Workbench.

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.

---

CDSW 1.5 does not ships with nvidia-docker version 1.0 and shall be installed independently.

2. Run a small container to create the Docker volume structure

```
#docker run --rm nvidia/cuda nvidia-smi
```

![CUDA SMIs](image)

Use the following Docker command to verify that Cloudera Data Science Workbench can access the GPU:

```
# vi nvidia-gpu-access.sh
mkdir /var/lib/nvidia-docker/
mkdir /var/lib/nvidia-docker/volumes/
mkdir /var/lib/nvidia-docker/volumes/nvidia_driver/
mkdir /var/lib/nvidia-docker/volumes/nvidia_driver/418.67
mkdir /var/lib/nvidia-docker/volumes/nvidia_driver/418.67/bin
mkdir /var/lib/nvidia-docker/volumes/nvidia_driver/418.67/lib64
cp /usr/bin/nvidia* /var/lib/nvidia-docker/volumes/nvidia_driver/418.67/bin
cp /usr/lib64/libcuda* /var/lib/nvidia-docker/volumes/nvidia_driver/418.67/lib64
cp /usr/lib64/libnvidia* /var/lib/nvidia-docker/volumes/nvidia_driver/418.67/lib64
```
Please replace the version of NVidia driver to one that matches your environment. Run "nvidia-smi" to capture the running version of NVidia driver.

3. In CDSW, local mount point (/usr/local/nvidia) for the docker containers will be established manually while creating the directory and copying the files from the CUDA library locations:

```bash
# docker run --net host \\
--device=/dev/nvidiactl \\
--device=/dev/nvidia-uvm \\
--device=/dev/nvidia0 \\
-v /var/lib/nvidia-docker/volumes/nvidia_driver/418.67/:/usr/local/nvidia/ \\
-it nvidia/cuda \\
usr/local/nvidia/bin/nvidia-smi
```

Test Cloudera Data Science Workbench to Detect GPUs

Once Cloudera Data Science Workbench has successfully restarted, if NVIDIA drivers have been installed on the Cloudera Data Science Workbench hosts, Cloudera Data Science Workbench will now be able to detect the GPUs available on its hosts.

Create a Custom CUDA-Capable Engine Image

The base engine image (docker.repository.cloudera.com/cdsw/engine:8) 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, go to [https://www.cloudera.com/documentation/data-science-workbench/latest/topics/cdsw_gpu.html](https://www.cloudera.com/documentation/data-science-workbench/latest/topics/cdsw_gpu.html).
Run a Local Registry

To start the registry container, run the following command:

```bash
# docker run -d -p 5000:5000 --restart=always --name registry registry:2
```

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 1.14.0 uses CUDA 10.0 and requires cuDNN 7.4.

1. To create the cuda.Dockerfile, run the following command:

```bash
FROM  docker.repository.cloudera.com/cdsw/engine:8
RUN NVIDIA_GPGKEY_SUM=d1be581509378368edeec8c1eb2958702feef33d71a76d458f66bac9a291290f6df4c3254d4a6 &&
    echo "$NVIDIA_GPGKEY_SUM" | sha256sum --strict
```

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 1.14.0 uses CUDA 10.0 and requires cuDNN 7.4.

1. To create the cuda.Dockerfile, run the following command:
ENV CUDA_VERSION 10.1.168
LABEL com.nvidia.cuda.version="\${CUDA_VERSION}\"

ENV CUDA_PKG_VERSION 10-1-\${CUDA_VERSION}-1
RUN apt-get update \&\& apt-get install -y --no-install-recommends \  
    cuda-cudart-\${CUDA_PKG_VERSION} \&\& \  
    ln -s cuda-10.1 /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
ENV CUDA_PKG_VERSION 10-1-\${CUDA_VERSION}-1\n
ENV CUDNN_VERSION 7.6.2.24
LABEL com.nvidia.cudnn.version="\${CUDNN_VERSION}\"
RUN apt-get update \&\& apt-get install -y --no-install-recommends \  
    libcudnn7=\${CUDNN_VERSION}-1+cuda10.1 \&\& \  
    rm -rf /var/lib/apt/lists/*

Build a **custom engine image** out of cuda.Dockerfile using the following sample command:

docker build --network host -t <company-registry>/cdsw-cuda:8 . -f cuda.Dockerfile
3. Push this new engine image to a public Docker registry so that it can be made available for Cloudera Data Science Workbench workloads. For example:

```
docker push <company-registry>/cdsw-cuda:8
```

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

### Image Management

Images can be preloaded on all NodeManager hosts or they can be implicitly pulled at runtime if they are available in a public Docker registry, such as Docker hub. If the image does not exist on the NodeManager and cannot be pulled, the container will fail.

For AI framework specific images, it is recommend using Docker images from NG Cloud, which is described in subsequent sections of this document.

It is also recommended to have a private Docker image repository. This CVD details the setup of a private registry for simplicity but doesn’t go into details on a registry setup with high availability and is provided only as an example.

On a Multi-GPU, server, the output of this command will show all the GPUs for some reason. This behavior needs to be further validated with the vendors.

### Configure CDSW to Run Docker Containers

#### Set Up Docker Registry

Private trusted registry is required to provision YARN container. This topic provides basic information about deploying and configuring a registry.

This is a sample registry to showcase the use-case and not recommended for Production grade setup.

**Designate a Server for Docker and Start the Registry**

To designate a server for Docker and start the registry, follow these steps:

1. Designate a server in the cluster for the Docker registry. Minimal resources are required, but sufficient disk space is needed to store the images and metadata. Docker must be installed and running.

2. Optional: By default, data will only be persisted within the container. If you would like to persist the data on the host, you can customize the bind mounts using the `-v` option.
3. Create /var/lib/registry folder:

```bash
# mkdir /var/lib/registry
```

4. Configure Docker to allow pulling from this insecure registry. Modify /etc/docker/daemon.json on all nodes in the cluster to include the following configuration options:

```bash
# cat /etc/docker/daemon.json
{
   "live-restore": true,
   "debug": true,
   "insecure-registries": ["linuxjh.hdp3.cisco.com:5000"]
}
```

5. Restart Docker on all nodes.

6. Provision the registry container by running the following command:

```bash
docker run -d -p 5000:5000 --restart=always --name registry -v /mnt/registry:/var/lib/registry registry:2
```

7. Verify the registry container is provisioned by running docker ps command:

```
CONTAINER ID   IMAGE                  COMMAND                  CREATED           STATUS          PORTS                    NAMES
037d176d2576   registry:2          "/entrypoint.sh /etc..." 14 minutes ago    Up 4 minutes    0.0.0.0:5000->5000/tcp registry
```

Create a Custom CUDA-capable Engine Image

```
[root@rhel17 ~]# docker run -d -p 5000:5000 --restart=always --name registry registry:2
Unable to find image 'registry:2' locally
2: Pulling from library/registry
c87736221ed0: Pull complete
1cc8e0bb44df: Pull complete
54d33bcb37f5: Pull complete
e8af0c91c717: Pull complete
b4541f6d3db6: Pull complete
Digest: sha256:8004747f1e8cd820a148fb7499d71a76d45ff66bac6a29129bfdbf0c054d146
Status: Downloaded newer image for registry:2
1e294ea97d773d296c2d6f51a744c76b24048e82c2d665b248c304d58158607
[root@rhel17 ~]# docker build --network host -t localhost/cdh-cdip:demo:1 . -f cuda.Dockerfile
Sending build context to Docker daemon 7.017 GB
Step 1/14 : FROM docker.repository.cloudera.com/cdservices:8
 ---> a583d1499187
Step 2/14 : RUN NVIDIA_GPGKEY_SUM=d1be58159378368e0e8c1e8c1eb2958702feedf3bc3d17131adbf24efacce4ab5 &&
NVIDIA_GPGKEY_FPR=ae0f6e4bb223a842b0ccf3e3f60fb3d7fa2a08c &&
apt-key adv --fetch-keys http://developer.download.nvidia.com/compute/cuda/repos/ubuntu1604/x86_64/7fa2a08c.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
--- Running in la0d2e21e2f2
Executing: /tmp/tmp.YMCjemm0nM/gpg.1.sh --fetch-keys
```
http://developer.download.nvidia.com/compute/cuda/repos/ubuntu1604/x86_64/7fa2af80.pub

gpg: key 7FA2AF80: public key "cudatools <cudatools@nvidia.com>" imported
gpg: Total number processed: 1
gpg: imported: 1 (RSA: 1)
cudasign.pub: OK
Selecting previously unselected package cuda-license-10-1.
(Reading database ... 106755 files and directories currently installed.)
Preparing to unpack .../cuda-license-10-1_10.1.243-1_amd64.deb ...
Unpacking cuda-license-10-1 (10.1.243-1) ...
Selecting previously unselected package cudart-10-1.
Preparing to unpack .../cuda-cudart-10-1_10.1.168-1_amd64.deb ...
Unpacking cuda-cudart-10-1 (10.1.168-1) ...
Setting up cuda-license-10-1 (10.1.243-1) ...
*** LICENSE AGREEMENT ***
By using this software you agree to fully comply with the terms and
conditions of the EULA (End User License Agreement). The EULA is located
at /usr/local/cuda-10.1/doc/EULA.txt. The EULA can also be found at
http://docs.nvidia.com/cuda/eula/index.html. If you do not agree to the
terms and conditions of the EULA, do not use the software.

Get:27 http://archive.ubuntu.com/ubuntu xenial-updates/universe amd64 Packages [983 kB]
Fetched 16.3 MB in 7s (2,304 kB/s)
Reading package lists...
Signature by key F36A89E33CC1BD0F710790073275754EE02A818DD uses weak digest algorithm (SHA1)
Reading package lists...
Building dependency tree...
Reading state information...
The following NEW packages will be installed:
  libcudnn7
0 upgraded, 1 newly installed, 0 to remove and 53 not upgraded.
Need to get 181 MB of archives.
After this operation, 435 MB of additional disk space will be used.
libcudnn7 7.6.2.24-1+cuda10.1 [181 MB]
Fetched 181 MB in 3s (48.4 MB/s)
Selecting previously unselected package libcudnn7.
(Reading database ... 106775 files and directories currently installed.)
Preparing to unpack .../libcudnn7_7.6.2.24-1+cuda10.1_amd64.deb ...
Unpacking libcudnn7 (7.6.2.24-1+cuda10.1) ...
Processing triggers for libc-bin (2.23-0ubuntu11) ...
Setting up libcudnn7 (7.6.2.24-1+cuda10.1) ...
Processing triggers for libc-bin (2.23-0ubuntu11) ...
libcudnn7 set on hold.
--- elf6c9eb21ae
Removing intermediate container 0359fa10d1b7
Success fully built elf6c9eb21ae
[root@rhel17 ~]#

Push Image to the Local Registry

[root@rhel17 ~]# docker push localhost:5000/cdh-cdip-demo
The push refers to a repository [localhost:5000/cdh-cdip-demo]
cb2ec429a23f: Preparing
2b90248b8b5b: Preparing
38aad9b5250: Preparing
bd674e93042a: Preparing
219140170d8f: Preparing
f1baf9cfd99: Pushed
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.

To allocate GPUs for sessions and jobs, follow these steps:

1. Sign into 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. Enter description and Tag created for repository. Click Add.
Appendix – A

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 existing project or create a new project and choose Settings.
3. Go to the Engines tab.
4. Under Engine Image, add the CUDA-capable engine image.
5. Go to the Sessions tab and click an existing session or create a new session and allocate resources.

6. Click Terminal Access.
About the Authors


Yogesh Ramesh is a Big Data Solutions Architect in the Computing Systems Product Group. He is part of the solution engineering team focusing on big data infrastructure, solutions, and performance.


Muhammad Afzal is an Engineering Architect and Technical Marketing Engineer in Cisco UCS Product Management and Datacenter Solutions Engineering. He is currently responsible for designing, developing, and producing validated architectures for Big Data and analytics while working collaboratively with product partners. Previously, Afzal had been a lead architect for various cloud and data center solutions in Solution Development Unit at Cisco. Prior to this, Afzal has been a Solutions Architect in Cisco's Advanced Services group, where he worked closely with Cisco's large enterprise and service provider customers delivering data center and cloud solutions. Afzal holds an MBA in Finance and a BS in Computer Engineering.

Acknowledgements

For their support and contribution to the design, validation, and creation of this Cisco Validated Design, the authors would like to thank:

- Ali Bajwa, Hortonworks
- Wangda Tan, Hortonworks
- Harsh Shah, Hortonworks
- Sicong Ji, NVIDIA