



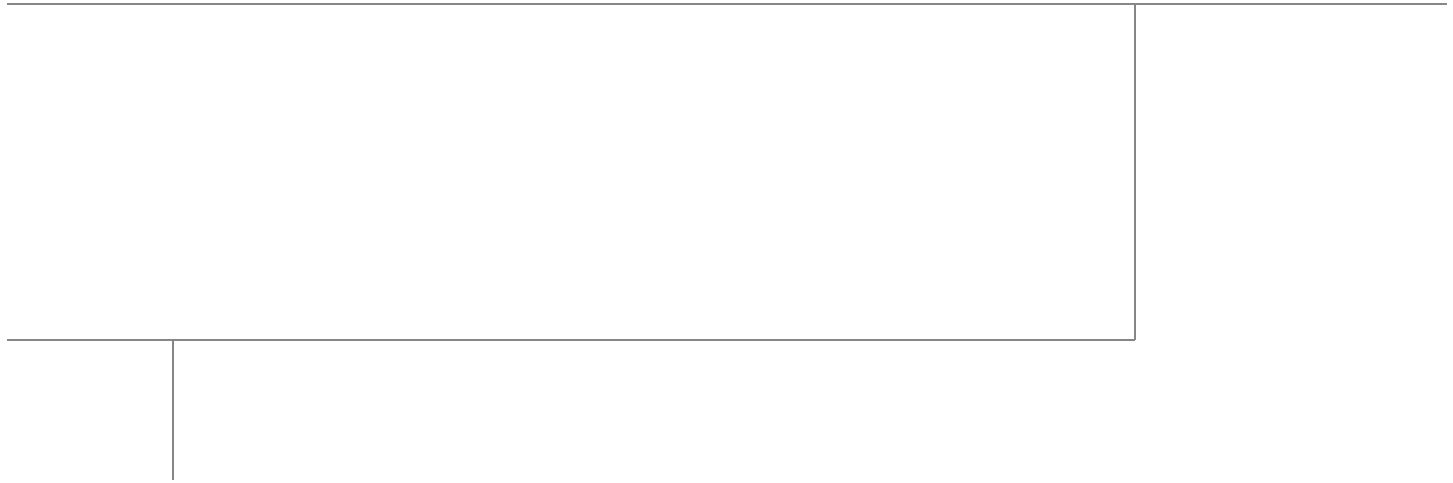
# Cisco Application Centric Infrastructure for Big Data with Cloudera

Highly Scalable Architecture for Big Data Clusters

Last Updated: October 28, 2014



Building Architectures to Solve Business Problems



## About the Authors



Raghunath Nambiar

### **Raghunath Nambiar, Distinguished Engineer, Cisco Systems**

Raghunath Nambiar is a Distinguished Engineer at Cisco's Data Center Business Group. His current responsibilities include emerging technologies and big data strategy.



Ranga Rao

### **Ranga Rao, Technical Marketing Engineer, Cisco Systems**

Ranga Rao is a Technical Marketing Engineer working on the Cisco Application Centric Infrastructure and Nexus 9000 Series Switches. He leads the Cisco ACI solution engineering team, focusing on big data and other high performance infrastructure like SAP HANA.



Karthik Kulkarni

### **Karthik Kulkarni, Technical Marketing Engineer, Cisco Systems**

Karthik Kulkarni is a Technical Marketing Engineer with role as a big data Solutions Architect in the Data Center Solutions Group. His main focus is on architecting and building solutions on big data related technologies, infrastructure and performance.



Samuel Kommu

### **Samuel Kommu, Technical Marketing Engineer, Cisco Systems**

Samuel Kommu is a Technical Marketing Engineer and is part of Cisco Application Centric Infrastructure solution engineering team focusing on big data infrastructure and performance.

## About Cisco Validated Design (CVD) Program

---

The CVD program consists of systems and solutions designed, tested, and documented to facilitate faster, more reliable, and more predictable customer deployments. For more information visit: <http://www.cisco.com/go/designzone>.

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.

The Cisco implementation of TCP header compression is an adaptation of a program developed by the University of California, Berkeley (UCB) as part of UCB's public domain version of the UNIX operating system. All rights reserved. Copyright © 1981, Regents of the University of California.

Cisco and the Cisco logo are trademarks or registered trademarks of Cisco and/or its affiliates in the U.S. and other countries. To view a list of Cisco trademarks, go to this URL:

<http://www.cisco.com/go/trademarks>. Third-party trademarks mentioned 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. (1110R).

Any Internet Protocol (IP) addresses and phone numbers used in this document are not intended to be actual addresses and phone numbers. Any examples, command display output, network topology diagrams, and other figures included in the document are shown for illustrative purposes only. Any use of actual IP addresses or phone numbers in illustrative content is unintentional and coincidental.

Cisco Application Centric Infrastructure for Big Data with Cloudera

© 2014 Cisco Systems, Inc. All rights reserved.

# **Acknowledgment**

The authors acknowledge contributions of Amrit Kharel, Manankumar Trivedi, Ashwin Manjunatha, and Sindhu Sudhir for their contributions in developing this document.



# Cisco Application Centric Infrastructure for Big Data with Cloudera

---

## Audience

This document describes the architecture and deployment procedures of Cloudera on a big data cluster based on Cisco Application Centric Infrastructure. The intended audience of this document includes, but is not limited to, sales engineers, field consultants, professional services, IT managers, partner engineering and customers who want to deploy Cloudera on the Cisco ACI for big data.

## Introduction

Hadoop has become a strategic data platform embraced by mainstream enterprises as it offers the fastest path for businesses to unlock value in big data while maximizing existing investments. Cloudera is the leading provider of enterprise-grade Hadoop infrastructure software and services, and the leading contributor to the Apache Hadoop project overall. Cloudera provides an enterprise-ready Hadoop-based solution known as Cloudera Enterprise, which includes their market leading open source Hadoop distribution (CDH), their comprehensive management system (Cloudera Manager), and technical support. The combination of Cisco Application Centric Infrastructure (ACI), Cloudera and Cisco UCS Servers provides industry-leading platform for Hadoop based applications.

## Cisco Application Centric Infrastructure (ACI) Overview

ACI provides network the ability to deploy and respond to the needs of applications, both in the data center and in the cloud. The network must be able to deliver the right levels of connectivity, security, compliance, firewalls, and load balancing, and it must be able to do this dynamically and on-demand.

This is accomplished through centrally defined policies and application profiles. The profiles are managed by new Application Policy Infrastructure Controller [APIC] and distributed to switches like the Cisco Nexus 9000 Series. Cisco Nexus 9000 Series Switches and the Cisco Application Policy Infrastructure Controller (APIC) are the building blocks for ACI.



---

Corporate Headquarters:  
Cisco Systems, Inc., 170 West Tasman Drive, San Jose, CA 95134-1706 USA

Copyright © 2014 Cisco Systems, Inc. All rights reserved.

ACI is software-defined networking (SDN) plus a whole lot more. Most SDN models stop at the network. ACI extends the promise of SDN—namely agility and automation—to the applications themselves. Through a policy-driven model, the network can cater to the needs of each application, with security, network segmentation, and automation at scale. And it can do so across physical and virtual environments, with a single pane of management.

The ACI fabric supports more than 64,000 dedicated tenant networks. A single fabric can support more than one million IPv4/IPv6 endpoints, more than 64,000 tenants, and more than 200,000 10G ports. The ACI fabric enables any service (physical or virtual) anywhere with no need for additional software or hardware gateways to connect between the physical and virtual services and normalizes encapsulations for Virtual Extensible Local Area Network (VXLAN) / VLAN / Network Virtualization using Generic Routing Encapsulation (NVGRE).

The ACI fabric decouples the endpoint identity and associated policy from the underlying forwarding graph. It provides a distributed Layer 3 gateway that ensures optimal Layer 3 and Layer 2 forwarding. The fabric supports standard bridging and routing semantics without standard location constraints (any IP address anywhere), and removes flooding requirements for the IP control plane Address Resolution Protocol (ARP) / Generic Attribute Registration Protocol (GARP). All traffic within the fabric is encapsulated within VXLAN.

## Solution Benefits with Cisco ACI

Some of the key benefits with Cisco ACI is discussed in the following sections.

### Centralized Management for the Entire Network

Cisco ACI treats the network as a single entity rather than a collection of switches. It uses a central controller to implicitly automate common practices such as Cisco ACI fabric startup, upgrades, and individual element configuration. The Cisco Application Policy Infrastructure Controller (Cisco APIC) is this unifying point of automation and management for the Application Centric Infrastructure (ACI) fabric. This architectural approach dramatically increases the operational efficiency of networks, by reducing the time and effort needed to make modifications to the network and, also, for root cause analysis and issue resolution.

### Performance Oriented Fabric

The Cisco ACI Fabric incorporates numerous capabilities that can help provide performance improvements to applications.

- **Dynamic Load Balancing (DLB):** The ACI fabric provides several load balancing options for balancing the traffic among the available uplink links. Static hash load balancing is the traditional load balancing mechanism used in networks where each flow is allocated to an uplink based on a hash of its 5-tuple. This load balancing gives a distribution of flows across the available links that is roughly even. Usually, with a large number of flows, the even distribution of flows results in an even distribution of bandwidth as well. However, if a few flows are much larger than the rest, static load balancing might give suboptimal results. Dynamic load balancing (DLB) adjusts the traffic allocations according to congestion levels. It measures the congestion across the available paths and places the flows on the least congested paths, which results in an optimal or near optimal placement of the data.

- **Dynamic Packet Prioritization (DPP):** while not a load balancing technology, uses some of the same mechanisms as DLB in the switch. DPP configuration is exclusive of DLB. DPP prioritizes short flows higher than long flows; a short flow is less than approximately 15 packets. Short flows are more sensitive to latency than long ones. DPP can improve overall application performance.

Together these technologies provide performance enhancements to applications, including big data workloads. More information on these technologies and results associated with performance analysis can be found in the following paper that recently won the best paper award at ACM SIGCOMM 2014: [CONGA: Distributed Congestion-Aware Load Balancing for data centers](#) (M. Alizadeh, T. Edsall, et al.)

## Application Centric Infrastructure Policy Model

The Cisco ACI policy model is designed top down using a promise theory model to control a scalable architecture of defined network and service objects. This model provides robust repeatable controls, multi tenancy, and minimal requirements for detailed knowledge by the control system known as the Cisco APIC. The model is designed to scale beyond current needs to the needs of private clouds, public clouds, and software-defined data centers.

The policy enforcement model within the fabric is built from the ground up in an application-centric object model. This provides a logical model for laying out applications, which will then be applied to the fabric by the Cisco APIC. This helps to bridge the gaps in communication between application requirements and the network constructs that enforce them. The Cisco APIC model is designed for rapid provisioning of applications on the network that can be tied to robust policy enforcement while maintaining a workload anywhere approach.

## Multi-tenant and Mixed Workload Support

Cisco ACI is built to incorporate secure multi-tenancy capabilities. The fabric enables customers to host multiple concurrent big data workloads on a shared infrastructure. ACI provides the capability to enforce proper isolation and SLA's for workloads of different tenants. These benefits extend beyond multiple big data workloads – Cisco ACI allows the same cluster to run a variety of different application workloads, not just big data, with the right level of security and SLA for each workload.

## Extensibility and Openness

ACI supports an open ecosystem embracing open APIs, open source, and open standards. This provides the broadest choice in data center management and infrastructure. ACI supports embracing open APIs, open source, and open standards. This provides the broadest choice in data center management and infrastructure.

## Easy Migration to 40 Gbps in the Network

Cisco QSFP BiDi technology removes 40 Gbps cabling cost barriers for migration from 10 Gbps to 40 Gbps connectivity in data center networks. Cisco QSFP BiDi transceivers provide 40 Gbps connectivity with immense savings and simplicity compared to other 40 Gbps QSFP transceivers. The Cisco QSFP BiDi transceiver allows organizations to migrate the existing 10 Gbps cabling infrastructure to 40 Gbps at no cost and to expand the infrastructure with low capital investment. Together with Cisco Nexus 9000 Series Switches, which introduce attractive pricing for networking devices, Cisco QSFP BiDi technology provides a cost-effective solution for migration from 10 Gbps to 40 Gbps infrastructure.



# Cisco ACI Building blocks

Cisco ACI consists of:

- The Cisco Nexus 9000 Series Switches.
- A centralized policy management and Cisco Application Policy Infrastructure Controller (APIC).

## Nexus 9000 Series Switches

The 9000 Series Switches offer both modular (9500 switches) and fixed (9300 switches) 1/10/40/100 GigabitEthernet switch configurations designed to operate in one of two modes:

- Cisco NX-OS mode for traditional architectures and consistency across the Cisco Nexus portfolio.
- ACI mode to take full advantage of the policy-driven services and infrastructure automation features of ACI.

The ACI-Ready Cisco Nexus 9000 Series provides:

- Accelerated migration to 40 GB: zero cabling upgrade cost with Cisco QSFP+ BiDi Transceiver Module innovation.
- Switching platform integration: Nexus 9000 Series enables a highly scalable architecture and is software upgradable to ACI.
- Streamline Application Management: Drastically reduce application deployment time and get end to end application visibility.

This architecture consists of Nexus 9500 series switches acting as the spine and Nexus 9300 series switches as leaves.

## Cisco Nexus 9508 Spine Switch

The Cisco Nexus 9508 Switch offers a comprehensive feature set, high resiliency, and a broad range of 1/10/40 GigabitEthernet line cards to meet the most demanding requirements of enterprise, service provider, and cloud data centers. The Cisco Nexus 9508 Switch is an ACI modular spine device enabled by a non-blocking 40 GigabitEthernet line card, supervisors, system controllers, and power supplies.

The Cisco Nexus 9500 platform internally uses a Cisco fabric design that interconnects the line cards with rear-mounted fabric modules. The Cisco Nexus 9500 platform supports up to six fabric modules, each of which provides up to 10.24 Tbps line-rate packet forwarding capacity. All fabric cards are directly connected to all line cards. With load balancing across fabric cards, the architecture achieves optimal bandwidth distribution within the chassis.

*Figure 1 Cisco Nexus 9508 Switch*



## ACI Spine Line Card for Cisco Nexus 9508

There are a multiple spine line cards supported on Cisco Nexus 9508. This architecture uses Cisco N9K-X9736PQ, with a 40 GigabitEthernet ACI spine line card. Following are the features of the 40 GbE ACI spine line card:

- 36 port 40 GigabitEthernet QSFP+ line card
- Non-blocking
- Designed for use in an ACI spine switch role
- Works only in ACI mode
- Cannot be used with non-spine line cards
- Supported in 8-slot chassis

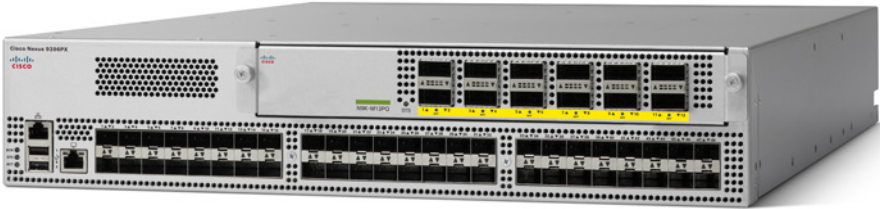
*Figure 2 Cisco N9K-X9736PQ ACI Spine Line Card*



## Cisco Nexus 9396 Leaf Switch

The Cisco Nexus 9396X Switch delivers comprehensive line-rate, layer 2 and layer 3 features in a two-rack-unit (2RU) form factor. It supports line rate 1/10/40 GE with 960 Gbps of switching capacity. It is ideal for top-of-rack and middle-of-row deployments in both traditional and Cisco Application Centric Infrastructure (ACI)-enabled enterprise, service provider, and cloud environments.

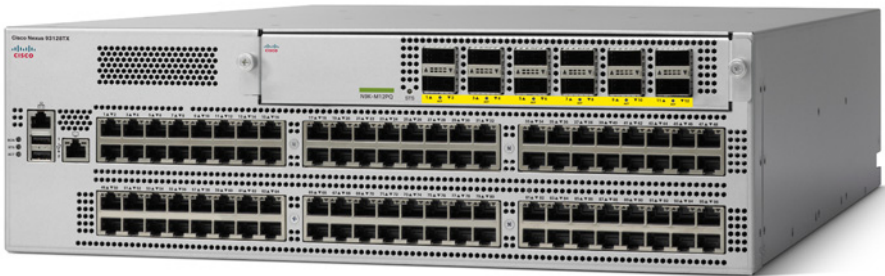
*Figure 3 Cisco Nexus 9396PX Switch*



## Cisco Nexus 93128 Leaf Switch

The Cisco Nexus 93128 Switch offers comprehensive layer 2 and layer 3 support and up to 1.28 Tbps of switching capacity in a three-rack-unit from factor. The Nexus 93128TX is ideal for top-of-rack and middle-of-row deployments in both traditional and Cisco Application Centric Infrastructure (ACI)-enabled enterprise, service provider, and cloud environments. The Cisco Nexus 93128TX Switch is a 3RU switch that supports 2.56 Tbps across 96 fixed 1/10GBASE-T ports and 8 fixed 40 Gbps QSFP ports.

*Figure 4 Cisco Nexus 93128TX Switch*



## Application Policy Infrastructure Controller (APIC)

The Application Centric Infrastructure is a distributed, scalable, multi-tenant infrastructure with external end-point connectivity controlled and grouped through application-centric policies. The APIC is the unified point of automation, management, monitoring, and programmability for the Cisco Application Centric Infrastructure. The APIC supports the deployment, management, and monitoring of any application anywhere, with a unified operations model for physical and virtual components of the infrastructure. The APIC programmatically automates network provisioning and Control that is based on the application requirements and policies. It is the central control engine for the broader cloud network; it simplifies management and allows flexibility in how application networks are defined and automated. It also provides northbound REST APIs. The APIC is a distributed system that is implemented as a cluster of many controller instances.

*Figure 5 Cisco APIC Appliance*

Front View



Rear View



## Solution Components

The components used in this solution and the topology is discussed in the following section.

### ACI Fabric with Cisco Nexus 9000 Series Switches

ACI topology is spine-leaf architecture. Each leaf is connected to each spine. It uses internal routing protocol; Intermediate System to Intermediate System (IS-IS) to establish IP connectivity throughout the fabric among all the nodes including spine and leaf. To transport tenant traffic across the IP fabric, integrated VxLAN overlay is used. The broadcast ARP traffic coming from the end point or hosts to the leaf are translated to unicast ARP in the fabric.

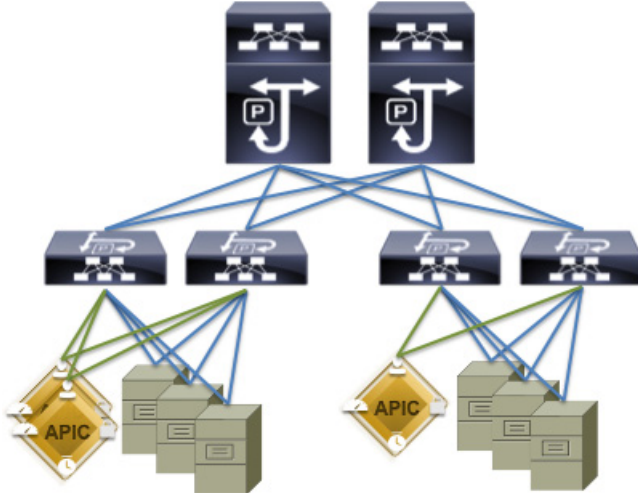
The forwarding is done as a host based forwarding. In the leaf layer the user information such as user name, IP address, locations, policy groups etc., are decoupled from the actual forwarding path and encode them into the fabric VxLAN header and is forwarded to the desired destination.

Each spine has the complete forwarding information about the end hosts that are connected to the fabric and on every leaf have the cached forwarding information. The leaf only needs to know the hosts it needs to talk to. For example if Server Rack-1 has to send some information to Server Rack-2, When packet comes in the ingress leaf (LEAF\_1) it will encapsulate the information into the VxLAN header and forward that information to LEAF\_2. If the LEAF\_1 does not have information about the LEAF\_2, it uses Spine as a proxy and since Spine has all the complete information about the entire end host connected to the fabric, it will resolve the egress leaf and forward the packet to the destination.

To the outside world, routing protocols can be used to learn outside prefixes or static routing can be used instead. The outside learned routes will be populated into the fabric or to the other leafs with Multi protocol BGP (M-BGP). In M-BGP topology the spine nodes acts as route reflectors.

The Network architecture of the solution is as depicted below:

**Figure 6** Network Topology Based on Cisco ACI Model



The Cisco ACI infrastructure incorporates the following components:

- Two Cisco Nexus 9508 Spine Switch
  - ACI Spine Line Card for Nexus 9508
- Cisco Nexus 9396 Leaf Switch for Data Traffic
- Cisco Nexus 93128 Leaf Switch for Management Traffic
- Cisco APIC-L1-Cluster with three APIC-L1 appliances

## Cisco Unified Computing System

The Cisco UCS C240 M3 servers has been used in this validated design as an example of a compute platform for Big Data architecture based on Cisco Application Centric Infrastructure (ACI). The Cisco UCS Integrated Infrastructure with ACI brings together a highly scalable architecture designed to meet a variety of scale-out application demands with seamless data integration and management integration capabilities.

The following Cisco UCS configuration could be used for Big Data work loads depending on the compute and storage requirements.

**Table 1** Compute Nodes used for the Big Data Cluster with ACI

Performance and Capacity Balanced	Capacity Optimized
42 Cisco UCS C240 M3S Rack Servers, each with: <ul style="list-style-type: none"> <li>• 2 Intel Xeon processors E5-2660 v2</li> <li>• 256 GB of memory</li> <li>• LSI MegaRaid 9271CV 8i card</li> <li>• 24 1-TB 7.2K SFF SAS drives (1080 TB total)</li> </ul>	42 Cisco UCS C240 M3S Rack Servers, each with: <ul style="list-style-type: none"> <li>• 2 Intel Xeon processors E5-2640 v2</li> <li>• 128 GB of memory</li> <li>• LSI MegaRaid 9271CV 8i card</li> <li>• 12 4-TB 7.2 LFF SAS drives (2160 TB total)</li> </ul>

This CVD uses Performance and Capacity Balanced configuration.

## Cloudera (CDH 5.0)

CDH is popular enterprise-grade, hardened distribution of Apache Hadoop and related projects. CDH is 100% Apache-licensed open source and offers unified [batch processing](#), [interactive SQL](#), and [interactive search](#), and [role-based access controls](#).

CDH delivers the core elements of Hadoop – scalable storage and distributed computing – along with additional components such as a [user interface](#), plus necessary enterprise capabilities such as security, and [integration with a broad range of hardware and software solutions](#).

For more information about the projects that are included in CDH, see CDH Version and Packaging information at:

<http://www.cloudera.com/content/cloudera-content/cloudera-docs/CDH5/latest/CDH-Version-and-Packaging-Information/CDH-Version-and-Packaging-Information.html>

## Physical Layout for the Solution

Each rack consists of two vertical PDUs. The solution consists of three Cisco R42610 racks. The first and the third rack consist of 21 UCS C240 M3S Servers each. The middle rack consists of two Cisco Nexus 9508, two Cisco Nexus 9396PX, one Cisco Nexus 93128TX and three Cisco APIC Appliances and three Cisco UCS C240 M3S servers. All the Switches and UCS Servers are dual connected to vertical PDUs for redundancy; thereby, ensuring availability during power source failure.



Note

Contact your Cisco representative for country specific information.

*Table 2 Rack 1 to 3*

Slot No:	Rack 1	Rack 2	Rack 3
1	N9K-C9396PX	N9K-C9396PX	N9K-C93128TX
2			
3			
4	APIC-L1	APIC-L1	APIC-L1
5	UCS C240M3		UCS C240M3
6			
7	UCS C240M3		UCS C240M3
8			
9	UCS C240M3	UCS C240M3	UCS C240M3
10			
11	UCS C240M3	UCS C240M3	UCS C240M3
12			

Table 2 Rack 1 to 3

Slot No:	Rack 1	Rack 2	Rack 3	
13	UCS C240M3	UCS C240M3	UCS C240M3	
14				
15	UCS C240M3	UCS C240M3	UCS C240M3	
16				
17	UCS C240M3	N9k-C9508	UCS C240M3	
18				
19	UCS C240M3			UCS C240M3
20				
21	UCS C240M3			UCS C240M3
22				
23	UCS C240M3			UCS C240M3
24				
25	UCS C240M3			UCS C240M3
26				
27	UCS C240M3			UCS C240M3
28				
29	UCS C240M3			UCS C240M3
30			N9k-C9508	
31	UCS C240M3			UCS C240M3
32				
33	UCS C240M3			UCS C240M3
34				
35	UCS C240M3			UCS C240M3
36				
37	UCS C240M3			UCS C240M3
38				
39	UCS C240M3			UCS C240M3
40				
41	UCS C240M3			UCS C240M3
42				

## Software Distributions and Versions

The versions of all software distributions required are listed below:

## Cloudera Enterprise

The Cloudera software for Cloudera Distribution for Apache Hadoop is version 5.0. For more information see: [www.cloudera.com](http://www.cloudera.com)

## Red Hat Enterprise Linux (RHEL)

The operating system supported is Red Hat Enterprise Linux 6.4. For more information, see, <http://www.redhat.com>

## Software Versions

The software versions tested and validated in this document are shown in [Table 3](#):

**Table 3** *Tested and Validated Software Versions*

Layer	Component	Version or Release
Network	Cisco ACI OS	11.0 (1b)
	APIC OS	1.0 (1e)
Compute	Cisco UCS C240 M3	1.5.4f
	Cisco UCS VIC 1225 Firmware	2.2(1b)
	Cisco UCS VIC 1225 Driver	2.1.1.41
Storage	LSI 9271-8i Firmware	23.12.0-0021
	LSI 9271-8i Driver	06.602.03.00
Software	Red Hat Enterprise Linux Server	6.4 (x86_64)
	CDH	5.0b2



**Note**

The latest drivers can be downloaded from the link:  
<http://software.cisco.com/download/release.html?mdfid=284296254&flowid=31743&softwareid=283853158&release=1.5.1&relind=AVAILABLE&rellifecycle=&reltype=latest>



Figure 7 System Architecture

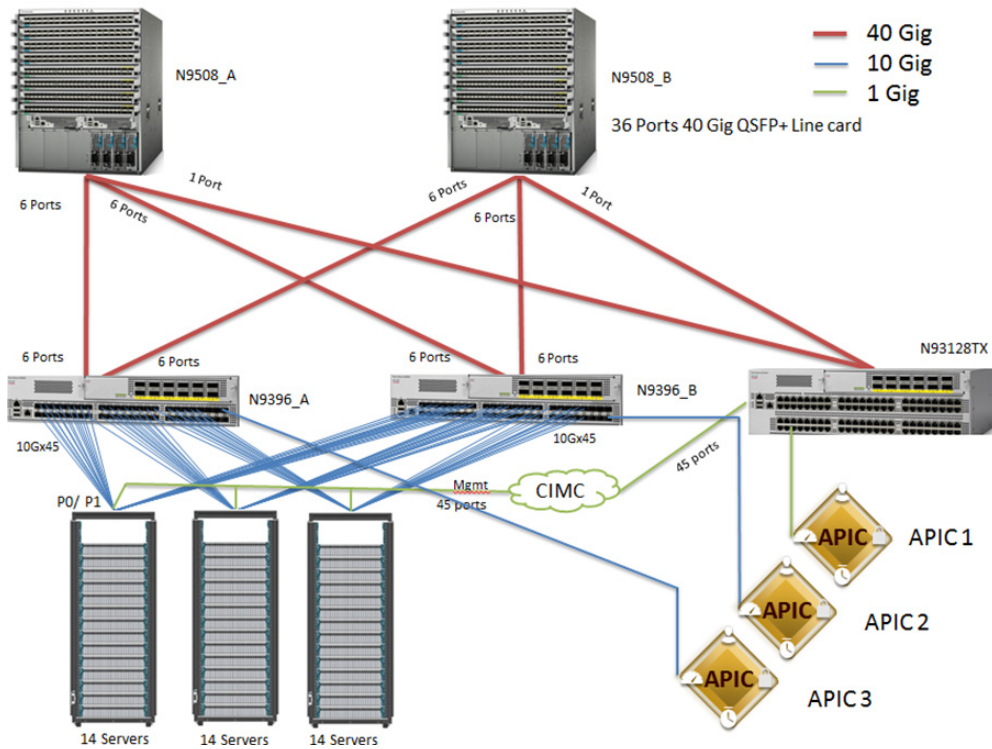
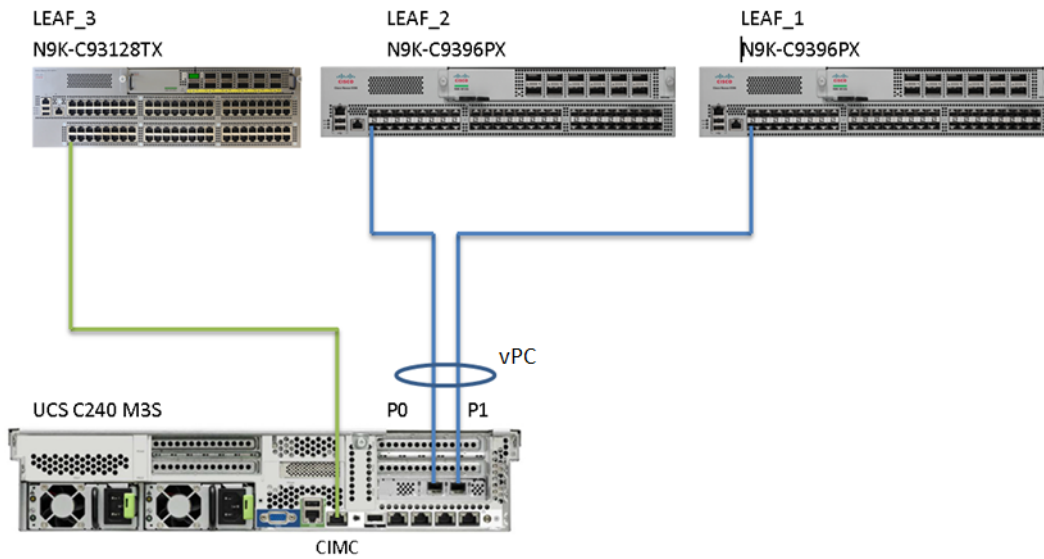


Figure 8 Server Connectivity



**Table 4** Required Hardware Details

Hardware	Role	Quantity
NK-C9508	Spine	2
N9K-X9736PQ	36 ports 40 Gig QSFP+ Spine Line Card	2
N9K-C9396PX	Leaf	2
N9K-C93128TX	Leaf	1
APIC-L1	APIC Appliance	3
UCS C240M3S	Rack Server	42
QSFP-H40G	40 GigabitEthernet connectivity	26
SFP-H10GB	10 GigabitEthernet Connectivity	84 (Servers) + 3 (APICs)

The ACI fabric is designed from the foundation to support emerging industry demands while maintaining a migration path for architecture already in place. The fabric is designed to support the industry move to management automation, programmatic policy, and dynamic “workload-anywhere” models. The ACI fabric accomplishes this with a combination of hardware, policy-based control systems, and software closely coupled to provide advantages not possible in other models.

The fabric consists of three major components: the Application Policy Infrastructure Controller, spine switches, and leaf switches. These three components handle both the application of network policy and the delivery of packets.

The system architecture consists of two Cisco Nexus 9508 switches acting as spine, two Cisco Nexus 9396 and one Cisco Nexus 93128 switches acting as leaf switches, three APIC-L1 as an APIC appliance.

The following explains the system architecture:

- The 45 server nodes are rack mounted and are cross connected across two leaf switches (Nexus 9396). Each of the leaf switches are connected to spine switches via 6 x 40 GigabitEthernet connectivity cables and each servers are connected via 10 GigaE connectivity cables.
- The two APICs are connected to two leaves (Nexus 9396) via 10 GigE SFP cable and the third APIC is connected to Nexus 93128 via copper cable using special converter (Cisco MGBT1 Gigabit 1000 Base-T Mini-GBIC SFP Transceiver).
- The servers are connected to two leaves for redundancy and at the server these two 10 GigE connectivity is bonded to act as one 20 GigE, giving sufficient bandwidth for data transfers.
- The Cisco Integrated Management Controller (CIMC) management ports of all the servers are connected to Nexus 93128 via 1 GigabitEthernet.



Note

For more details on NIC bonding, see [Server NIC Bonding](#) section.

## Scaling the Architecture

In this particular design, we leveraged two Cisco ACI spine switches. Each of Cisco Nexus 9396 connects 6 x 40G ports to both the Cisco Nexus 9508 line card Cisco N9K-X9736PQ, hence with the pair of Cisco Nexus 9396, 12 ports are consumed. Further, Cisco Nexus 93128 needs to be connected to

Cisco Nexus 9508 similarly but, only one port gets consumed on the line card as this is only for management traffic. So roughly for every 3 racks, 13 ports on the line card on Cisco Nexus 9508 get consumed. Single Cisco Nexus 93128 can support up to 90 Servers. A single line card can support slightly over 8 racks (that is, two full pods and one pod with 2 racks) which are 128 servers. Cisco Nexus 9508 needs to add more line card to support additional pods.

*Figure 9 Two Spine Network Architecture*

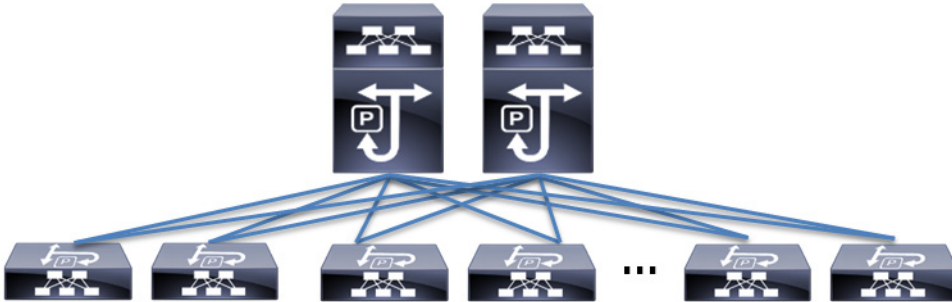


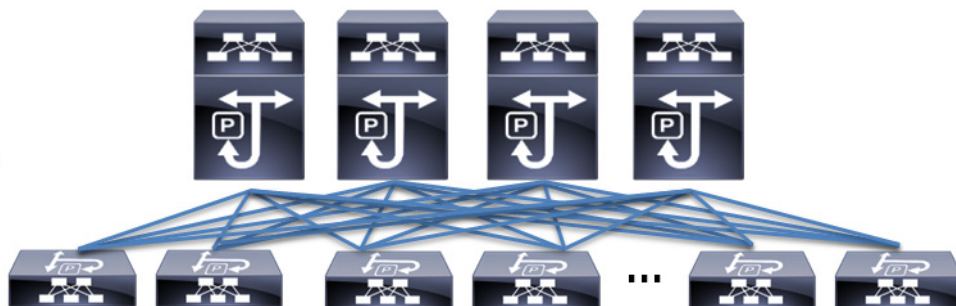
Table 5 Scalability of 2 Spine Architecture

Spine	Line Card Pair	Ports Used	POD	Leaf	Ports Used	Server
N9508_A	Line Card 1	1-6	1	9396_1A	1-42	1-42
	Line Card 1	7-12		9396_1B	1-42	
	Line Card 1	13		93128_1A	1-42	
	Line Card 1	14-19	2	9396_2A	1-42	43-84
	Line Card 1	20-25		9396_2B	1-42	
	Line Card 1			93128_1A	43-84	
	Line Card 1	26-31	3	9396_3A	1-42	43-84
	Line Card 1	32-36		9396_3B	1-42	
	Line Card 2	1		9396_3B	-	
	Line Card 2	2		93128_2A	1-42	
	Line Card 2	3-8	4	9396_4A	1-42	127-168
	Line Card 2	9-14		9396_4B	1-42	
	Line Card 2			93128_2A	1-42	
	Line Card 2	15-20	5	9396_5A	1-42	211-252
	Line Card 2	21-26		9396_5B	1-42	
	Line Card 2	27		93128_3A	1-42	
	Line Card 2	28-33	6		9396_6A	
	Line Card 2	34-36			9396_6B	
	Line Card 3	1-3			9396_6B	
	Line Card 3				93128_3A	
	..	...	...	...	...	..
Line Card 7	36	21	9396_21A	1-42	841-882	
Line Card 8	1-5		9396_21A	-		
Line Card 8	6-11		9396_21B	1-42		
Line Card 8	12		93128_11A	1-42		
Line Card 8	13-18	22	9396_22A	1-42	884-924	
Line Card 8	19-24		9396_22B	1-42		
Line Card 8			93128_11A	43-84		
Line Card 8	25-29	23	9396_23A	1-40	924-964	
Line Card 8	30-34		9396_23B	1-40		
Line Card 8	35		93128_12A	1-40		

## Scaling the architecture further with additional Spines Switches

The physical network of the Cisco Application Centric Infrastructure is built around leaf-spine architecture. It is possible to scale this infrastructure, immensely, by adding additional Spine switches. The ACI infrastructure supports upto 12 spine switches.

*Figure 10 Cisco ACI Fabric with Multiple Spine Switches*



With a 12-spine design, each leaf switch can be connected up to 12 spine switches. Allowing for upwards of 12,960 nodes to be part of this infrastructure – being interconnected by a non-blocking fabric.

## Future Scalability of the Architecture

In the future, once Cisco Nexus 9516 switches are supported by Cisco ACI OS, this design can be scaled further to upwards of 25,920 nodes – with the nodes being interconnected by a completely non-blocking fabric.

## Scaling through Fabric Interconnect (FI)

Here the UCS Servers are directly connected to Fabric Interconnect (FI) which in-turn connects to N9K switches. This mode allows using the UCS Manager capabilities in FI for provisioning the servers. Up to 5 Racks are connected to a Pair of FI forming a single domain and three such domains are connected to a pair of Leaf 9396. This topology has no network over-subscription within a domain (servers under a pair of FI) and 1:5.7 over-subscription between domains and can scale up to 5760 servers for a fully populated pair of Nexus 9508 with all the 8 linecards.

Figure 11 Scaling Through Fabric Interconnect (FI)

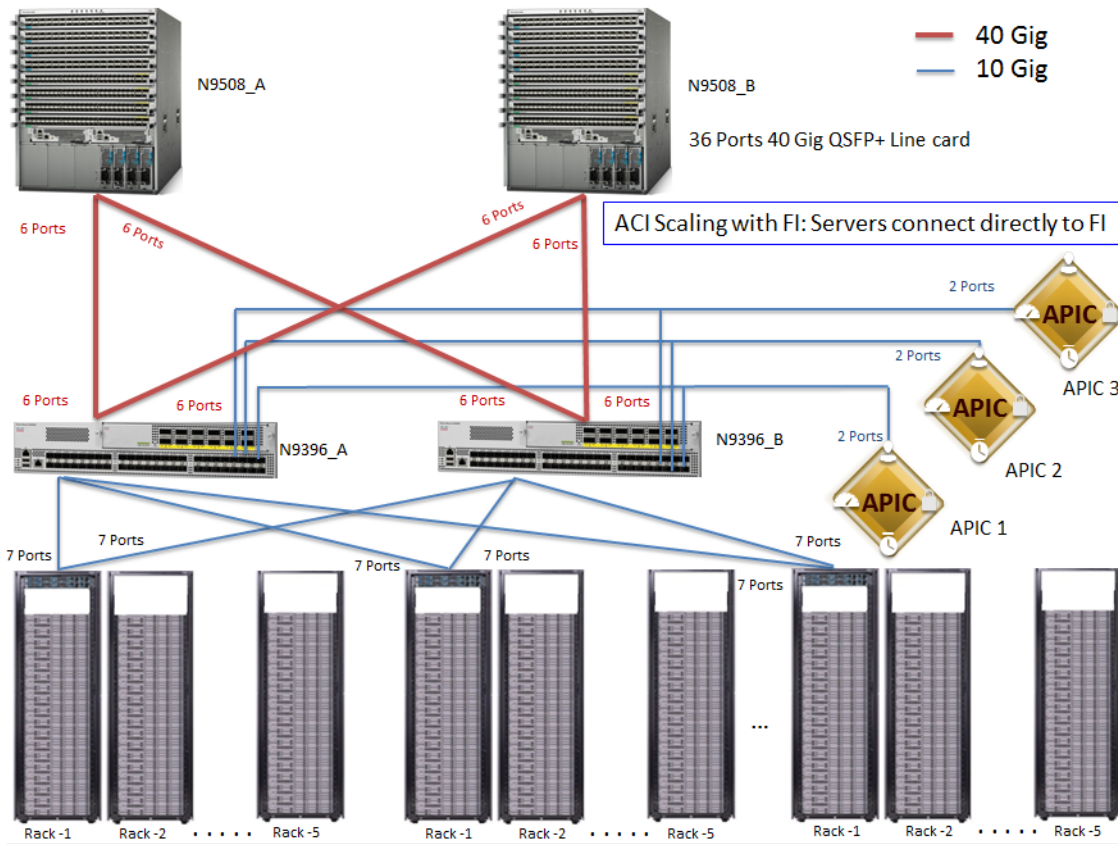


Table 6 Cisco Nexus 9508 – Nexus 9396PX Connectivity

Spine	Line Card Pair	Ports Used	POD	Servers	LEAF	
N9508_A	Line card 1	1-6	1	240	9396_1A	
	Line card 1	7-12			9396_1B	
	Line card 1	13-18	2	480	9396_2A	
	Line card 1	19-24			9396_2B	
	Line card 1	25-30	3	720	9396_3A	
	Line card 1	31-36			9396_3B	
	Line card 8	1-6	22	5280	9396_22A	
	Line card 8	7-12			9396_22B	
	Line card 8	13-18	23	5520	9396_23A	
	Line card 8	19-24			9396_23B	
	Line card 8	25-30	24	5760	9396_24A	
Line card 8	31-36			9396_24B		

Spine	Line Card Pair	Ports Used	POD	Servers	LEAF
N9508_B	Line card 1	1-6	1	240	9396_1A
	Line card 1	7-12			9396_1B
	Line card 1	13-18	2	480	9396_2A
	Line card 1	19-24			9396_2B
	Line card 1	25-30	2	720	9396_3B
	Line card 1	31-36			
	Line card 8	1-6	22	5280	9396_22A
	Line card 8	7-12			9396_22B
	Line card 8	13-18	23	5520	9396_23A
Line card 1	19-24	9396_23B			
Line card 8	25-30	24	5760	9396_24A	
Line card 8	31-36			9396_24B	

*Table 7 FI Connectivity*

LEAF	Ports Used	FI	Servers
9396_1A	1-14	FI_1A	1-80
9396_1A	15-28	FI_2A	81-160
9396_1A	29-42	FI_3A	161-240
9396_1A	43	APIC	
	44-48	Unused	
9396_1B	1-14	FI_1B	1-80
9396_1B	15-28	FI_2B	81-160
9396_1B	29-42	FI_3B	161-240
9396_1B	43	APIC	
	44-48	Unused	

## Network Configuration

This section describes loading and configuring the APIC.

## Configuration of APIC

Once the APIC appliance is booted for the first time, the APIC console presents a series of initial setup options. For many options, you can press Enter to choose the default setting that is displayed in brackets. At any point in the setup dialog, you can restart the dialog from the beginning by pressing Ctrl-C.

Shown below is the initial configuration of the APIC

```
Enter the fabric name [ACI Fabric1]:
Enter the number of controllers in the fabric (1-9) [3]:3
Enter the controller ID (1-3) [1]:1
Enter the controller name [apic1]:APIC_1
Enter address pool for TEP addresses [10.0.0.0/16]:
Enter the VLAN ID for infra network (1-4094) [4]: 145

Out-of-band management configuration...
Enter the IP address for out-of-band management: 10.0.130.71/24
Enter the IP address of the default gateway [None]: 10.0.130.1
Administrator user configuration...
Enable strong passwords? [Y]
Enter the password for admin:
```

**Figure 12** *APIC Initial Configuration*

```
Reenter the password for admin:

Cluster configuration ...
Fabric name: BIG_DATA
Number of controllers: 3
Controller name: APIC
Controller ID: 1
TEP address pool: 10.0.0.0/16
Infra VLAN ID: 130
Multicast address pool: 225.0.0.0/15

Out-of-band management configuration ...
Management IP address: 10.0.130.71/24
Default gateway: 10.0.130.1
Interface speed/duplex mode: auto

admin user configuration ...
Strong Passwords: N
User name: admin
Password: *****

The above configuration will be applied ...

Would you like to edit the configuration? (y/n) [n]:
```

Once the configuration is completed, the APIC will boot its APIC IOS Image and will ask for the login information. The default username is “admin” and the password is the one that was set during the initial configuration.



*Figure 13 APIC Login Screen*

```

Application Policy Infrastructure Controller
Version 1.0(1e)

APIC login: admin
Password: _

```

## Switch Discovery with the APIC

The APIC is a central point of automated provisioning and management for all the switches that are part of the ACI fabric. A single data center might include multiple ACI fabrics, each with their own APIC cluster and Cisco Nexus 9000 Series switches that are part of the fabric. To ensure that a switch is managed only by a single APIC cluster, each switch must be registered with that specific APIC cluster that manages the fabric. The APIC discovers new switches that are directly connected to any switch it currently manages. Each APIC instance in the cluster first discovers only the leaf switch to which it is directly connected. After the leaf switch is registered with the APIC, the APIC discovers all spine switches that are directly connected to the leaf switch. As each spine switch is registered, that APIC discovers all the leaf switches that are connected to that spine switch. This cascaded discovery allows the APIC to discover the entire fabric topology in a few simple steps.

## Switch Registration with the APIC Cluster

Once the switch is discovered by the APIC cluster it needs to be registered in the APIC to make it a part of the fabric.

### Prerequisite

All switches must be physically connected and booted with the correct ACI Image.

### Procedure

Using a web browser connect to the out-of-band management ip address [10.0.130.71] configured in the initial configuration.

1. On the menu bar, choose **FABRIC > INVENTORY**. In the Navigation pane, choose the appropriate pod.
2. In the Navigation pane, expand the pod, and click **Fabric Membership**. In the **Work** pane, in the Fabric Membership table, a single leaf switch is displayed with an ID of 0. It is the leaf switch that is connected to APIC.

Figure 14 Switch Discovery

SERIAL NUMBER	NODE ID	NODE NAME	RACK NAME	MODEL	ROLE	IP	DECOMMISSIONED	SUPPORTED MODEL
SAL181950RY	0			N9K-C9396PX	leaf	0.0.0.0	False	True

- To configure the ID, double-click the leaf switch row, and perform the following actions:
  - In the **ID** field, add the appropriate ID (leaf1 is ID 101, leaf2 is ID 102 and leaf3 is ID103). The ID must be a number that is greater than 100 because the first 100 IDs are for APIC appliance nodes.
  - In the **Switch Name** field, add the name of the switch, and click **Update**. After an ID is assigned, it cannot be updated. The switch name can be updated by double-clicking the name and updating the **Switch Name** field.

**Note**

The Success dialog box is displayed. An IP address gets assigned to the switch, and in the Navigation pane, the switch is displayed under the pod.

Figure 15 Switch Registration

SERIAL NUMBER	NODE ID	NODE NAME	RACK NAME	MODEL	ROLE	IP	DECOMMISSIONED	SUPPORTED MODEL
SAL181950RY	101	LEAF_1	BIG_DATA	N9K-C9396PX	leaf	10.0.47.255/32	False	True

- Monitor the **Work** pane until one or more spine switches appear.
- To configure the ID, double-click the spine switch row and perform the following actions:
  - In the **ID** field, add the appropriate ID (spine1 is ID 201 and spine 2 is ID 202). The ID must be a number that is greater than 100.
  - In the **Switch Name** field, add the name of the switch, and click **Update**. The Success dialog box is displayed. An IP address gets assigned to the switch, and in the **Navigation** pane, the switch is displayed under the pod. Wait until all remaining switches appear in the Node Configurations table.
- For each switch listed in the *Fabric Membership* table, perform the following steps:
  - Double-click the switch, enter an ID and a Name, and click **Update**.
  - Repeat for the next switch in the list.

## Switch Registration Using the Rest API

```

method: undefined
url:
https://10.0.130.71/api/node/mo/uni/fabric/site-default/building-default/floor-default/room-default.json
response:
{"totalCount":"1","imdata":[{"geoRoom":{"attributes":{"childAction":"","descr":"","dn":"uni/fabric/site-default/building-default/floor-default/room-default","lcOwn":"local","modTs":"2014-09-16T10:45:42.857+00:00","name":"default","status":"","uid":"0"}}}]}
10:53:00 DEBUG -
method: GET
url: https://10.0.130.71/api/node/mo/info.json
response:
{"totalCount":"0","imdata":[{"topInfo":{"attributes":{"childAction":"","currentTime":"2014-09-16T10:48:50.132+00:00","dn":"info","id":"1","role":"controller","status":""}}}]}
10:53:06 DEBUG -
method: Event Channel Message
response:
{"subscriptionId":["72057598349672454","72057598349672455"],"imdata":[{"geoRack":{"attributes":{"childAction":"","descr":"","dn":"uni/fabric/site-default/building-default/floor-default/room-default/rack-Big_Data","lcOwn":"local","modTs":"2014-09-16T10:48:55.700+00:00","name":"Big_Data","rn":"","status":"created","uid":"15374"}}}]}
10:53:06 DEBUG -
method: POST
url:
https://10.0.130.71/api/node/mo/uni/fabric/site-default/building-default/floor-default/room-default/rack-Big_Data.json
payload{"geoRack":{"attributes":{"dn":"uni/fabric/site-default/building-default/floor-default/room-default/rack-Big_Data","name":"Big_Data","rn":"rack-Big_Data","status":"created"},"children":[]}}
response: {"imdata":[]}
10:53:06 DEBUG -
method: GET
url: https://10.0.130.71/api/node/class/geoRack.json?subscription=yes
response:
{"totalCount":"2","subscriptionId":"72057598349672457","imdata":[{"geoRack":{"attributes":{"childAction":"","descr":"","dn":"uni/fabric/site-default/building-default/floor-default/room-default/rack-default","lcOwn":"local","modTs":"2014-09-16T10:45:42.857+00:00","name":"default","status":"","uid":"0"}},{geoRack":{"attributes":{"childAction":"","descr":"","dn":"uni/fabric/site-default/building-default/floor-default/room-default/rack-Big_Data","lcOwn":"local","modTs":"2014-09-16T10:48:55.700+00:00","name":"Big_Data","status":"","uid":"15374"}}}]}
10:53:10 DEBUG -
method: Event Channel Message
response:
{"subscriptionId":["72057598349672454"],"imdata":[{"geoRsNodeLocation":{"attributes":{"childAction":"","dn":"uni/fabric/site-default/building-default/floor-default/room-default/rack-Big_Data/rsnodeLocation-[topology/pod-1/node-101]","forceResolve":"no","lcOwn":"local","modTs":"2014-09-16T10:48:59.495+00:00","rType":"mo","rn":"","state":"unformed","stateQual":"none","status":"created","tCl":"fabricNode","tDn":"topology/pod-1/node-101","tType":"mo","uid":"15374"}}}]}
10:53:10 DEBUG -
method: Event Channel Message
response:
{"subscriptionId":["72057598349672456"],"imdata":[{"dhcpClient":{"attributes":{"childAction":"","dn":"client-[SAL1819S0RY]","modTs":"2014-09-16T10:48:59.503+00:00","name":"LEAF_1","nodeId":"101","rn":"","status":"modified"}}}]}
10:53:10 DEBUG -
method: POST
url:
https://10.0.130.71/api/node/mo/uni/fabric/site-default/building-default/floor-default/room-default/rack-Big_Data.json

```

```

payload{"geoRsNodeLocation":{"attributes":{"tDn":"topology/pod-1/node-101","status":"created,modified"},"children":[]}}
response: {"imdata": []}
10:53:10 DEBUG -
method: POST
url: https://10.0.130.71/api/node/mo/uni/controller/nodeidentpol.json
payload{"fabricNodeIdentP":{"attributes":{"dn":"uni/controller/nodeidentpol/nodep-SAL1819S0RY","serial":"SAL1819S0RY","nodeId":"101","name":"LEAF_1","status":"created,modified"},"children":[]}}
response: {"imdata": []}
10:53:10 DEBUG -
method: undefined
url:
https://10.0.130.71//api/node/class/geoRack.json?query-target=subtree&target-subtree-class=geoRack,geoRsNodeLocation&subscription=yes
response:
{"totalCount":"3","subscriptionId":"72057598349672458","imdata":[{"geoRack":{"attributes":{"childAction":"","descr":"","dn":"uni/fabric/site-default/building-default/floor-default/room-default/rack-default","lcOwn":"local","modTs":"2014-09-16T10:45:42.857+00:00","name":"default","status":"","uid":"0"}},{geoRack":{"attributes":{"childAction":"","descr":"","dn":"uni/fabric/site-default/building-default/floor-default/room-default/rack-Big_Data","lcOwn":"local","modTs":"2014-09-16T10:48:55.700+00:00","name":"Big_Data","status":"","uid":"15374"}},{geoRsNodeLocation":{"attributes":{"childAction":"","dn":"uni/fabric/site-default/building-default/floor-default/room-default/rack-Big_Data/rsnodeLocation-[topology/pod-1/node-101]","forceResolve":"no","lcOwn":"local","modTs":"2014-09-16T10:48:59.495+00:00","rType":"mo","state":"unformed","stateQual":"none","status":"","tCl":"fabricNode","tDn":"topology/pod-1/node-101","tType":"mo","uid":"15374"}}}]}]}
10:53:10 DEBUG -
method: undefined
url:
https://10.0.130.71//api/node/class/geoRack.json?query-target=subtree&target-subtree-class=geoRack,geoRsNodeLocation&subscription=yes
response:
{"totalCount":"3","subscriptionId":"72057598349672459","imdata":[{"geoRack":{"attributes":{"childAction":"","descr":"","dn":"uni/fabric/site-default/building-default/floor-default/room-default/rack-default","lcOwn":"local","modTs":"2014-09-16T10:45:42.857+00:00","name":"default","status":"","uid":"0"}},{geoRack":{"attributes":{"childAction":"","descr":"","dn":"uni/fabric/site-default/building-default/floor-default/room-default/rack-Big_Data","lcOwn":"local","modTs":"2014-09-16T10:48:55.700+00:00","name":"Big_Data","status":"","uid":"15374"}},{geoRsNodeLocation":{"attributes":{"childAction":"","dn":"uni/fabric/site-default/building-default/floor-default/room-default/rack-Big_Data/rsnodeLocation-[topology/pod-1/node-101]","forceResolve":"no","lcOwn":"local","modTs":"2014-09-16T10:48:59.495+00:00","rType":"mo","state":"unformed","stateQual":"none","status":"","tCl":"fabricNode","tDn":"topology/pod-1/node-101","tType":"mo","uid":"15374"}}}]}]}
10:53:10 DEBUG -
method: GET
url:
https://10.0.130.71/api/node/class/topology/pod-1/node-1/dhcpClient.json?query-target-filter=or(eq(dhcpClient.nodeRole,"leaf"),eq(dhcpClient.nodeRole,"spine"),eq(dhcpClient.nodeRole,"unsupported"))&subscription=yes
response:
{"totalCount":"1","subscriptionId":"72057598349672460","imdata":[{"dhcpClient":{"attributes":{"childAction":"","clientEvent":"denied","decommissioned":"no","dn":"client-[SAL1819S0RY]","fabricId":"1","fwVer":"","hwAddr":"00:00:00:00:00:00","id":"SAL1819S0RY","ip":"0.0.0.0","lcOwn":"local","modTs":"2014-09-16T10:48:59.503+00:00","model":"N9K-C9396PX","name":"LEAF_1","nodeId":"101","nodeRole":"leaf","podId":"1","spineLevel":"0","status":"","supported":"yes"}}}]}
10:53:10 DEBUG -
method: GET
url:
https://10.0.130.71/api/node/class/topology/pod-1/node-1/dhcpClient.json?query-target-filter=or(eq(dhcpClient.nodeRole,"leaf"),eq(dhcpClient.nodeRole,"spine"),eq(dhcpClient.nodeRole,"unsupported"))&subscription=yes

```

```

response:
{ "totalCount": "1", "subscriptionId": "72057598349672461", "imdata": [{"dhcpClient": {"attributes": {"childAction": "", "clientEvent": "denied", "decomissioned": "no", "dn": "client- [SAL1819SORY ]", "fabricId": "1", "fwVer": "", "hwAddr": "00:00:00:00:00:00", "id": "SAL1819SORY", "ip": "0.0.0.0", "lcOwn": "local", "modTs": "2014-09-16T10:48:59.503+00:00", "model": "N9K-C9396PX", "name": "LEAF_1", "nodeId": "101", "nodeRole": "leaf", "podId": "1", "spineLevel": "0", "status": "", "supported": "yes"}}}}]}

```

## Validating the Switches

### Procedure

1. On the menu bar, choose **FABRIC > INVENTORY**, and in the navigation pane, under **Pod 1**, expand **Fabric Membership**.
2. The switches in the fabric are displayed with their node IDs. In the **Work** pane, all the registered switches are displayed with the IP addresses that are assigned to them.

**Figure 16** Switch Validation

SERIAL NUMBER	NODE ID	NODE NAME	RACK NAME	MODEL	ROLE	IP	DECOMMISSIONED	SUPPORTED MODEL
FGE18200AW0	201	SPINE_1	BIG_DATA	N9K-C9508	spine	10.0.168.94/32	False	True
FGE18200AWL	202	SPINE_2	BIG_DATA	N9K-C9508	spine	10.0.168.65/32	False	True
SAL1816QWFA	103	LEAF_3	BIG_DATA	N9K-C93128TX	leaf	10.0.168.64/32	False	True
SAL181950M8	102	LEAF_2	BIG_DATA	N9K-C9396PX	leaf	10.0.168.93/32	False	True
SAL181950RY	101	LEAF_1	BIG_DATA	N9K-C9396PX	leaf	10.0.168.95/32	False	True

## Validating Switch Using Rest API

```

method: GET
url: https://10.0.130.71/api/node/class/fabricPod.json?subscription=yes
response:
{ "totalCount": "1", "subscriptionId": "72057598349672517", "imdata": [{"fabricPod": {"attributes": {"childAction": "", "dn": "topology/pod-1", "id": "1", "lcOwn": "local", "modTs": "2014-09-16T10:45:42.798+00:00", "monPolDn": "uni/fabric/monfab-default", "status": ""}}}}]}
11:10:48 DEBUG -
method: GET
url: https://10.0.130.71/api/node/mo/sys.json
response: {"totalCount": "0", "imdata": []}
11:10:52 DEBUG -
method: undefined
url:
https://10.0.130.71//api/node/class/geoRack.json?query-target=subtree&target-subtree-class=geoRack,geoRsNodeLocation&subscription=yes
response:
{ "totalCount": "6", "subscriptionId": "72057598349672518", "imdata": [{"geoRack": {"attributes": {"childAction": "", "descr": "", "dn": "uni/fabric/site-default/building-default/floor-default/room-default/rack-default", "lcOwn": "local", "modTs": "2014-09-16T10:45:42.857+00:00", "name": "default", "status": "", "uid": "0"}}}, {"geoRack": {"attributes": {"childAction": "", "descr": "", "dn": "uni/fabric/site-default/building-default/floor-default/room-default/rack-Big_Data", "l

```

```

cOwn": "local", "modTs": "2014-09-16T10:48:55.700+00:00", "name": "Big_Data", "status": "", "uid":
"15374"}}, {"geoRsNodeLocation": {"attributes": {"childAction": "", "dn": "uni/fabric/site-defa
ult/building-default/floor-default/room-default/rack-Big_Data/rsnodeLocation-[topology/pod
-1/node-101]", "forceResolve": "no", "lcOwn": "local", "modTs": "2014-09-16T10:48:59.495+00:00",
"rType": "mo", "state": "unformed", "stateQual": "none", "status": "", "tCl": "fabricNode", "tDn": "t
opology/pod-1/node-101", "tType": "mo", "uid": "15374"}}, {"geoRsNodeLocation": {"attributes": {
"childAction": "", "dn": "uni/fabric/site-default/building-default/floor-default/room-default
/rack-Big_Data/rsnodeLocation-[topology/pod-1/node-201]", "forceResolve": "no", "lcOwn": "loca
l", "modTs": "2014-09-16T10:52:58.316+00:00", "rType": "mo", "state": "unformed", "stateQual": "no
ne", "status": "", "tCl": "fabricNode", "tDn": "topology/pod-1/node-201", "tType": "mo", "uid": "153
74"}}, {"geoRsNodeLocation": {"attributes": {"childAction": "", "dn": "uni/fabric/site-default/
building-default/floor-default/room-default/rack-Big_Data/rsnodeLocation-[topology/pod-1/n
ode-102]", "forceResolve": "no", "lcOwn": "local", "modTs": "2014-09-16T10:55:14.513+00:00", "rTy
pe": "mo", "state": "unformed", "stateQual": "none", "status": "", "tCl": "fabricNode", "tDn": "topol
ogy/pod-1/node-102", "tType": "mo", "uid": "15374"}}, {"geoRsNodeLocation": {"attributes": {"chi
ldAction": "", "dn": "uni/fabric/site-default/building-default/floor-default/room-default/rac
k-Big_Data/rsnodeLocation-[topology/pod-1/node-103]", "forceResolve": "no", "lcOwn": "local", "
modTs": "2014-09-16T10:55:24.330+00:00", "rType": "mo", "state": "unformed", "stateQual": "none",
"status": "", "tCl": "fabricNode", "tDn": "topology/pod-1/node-103", "tType": "mo", "uid": "15374"}
}}}}
11:10:52 DEBUG -
method: GET
url: https://10.0.130.71/api/node/class/geoRack.json?subscription=yes
response:
{"totalCount": "2", "subscriptionId": "72057598349672519", "imdata": [{"geoRack": {"attributes":
{"childAction": "", "descr": "", "dn": "uni/fabric/site-default/building-default/floor-default/
room-default/rack-default", "lcOwn": "local", "modTs": "2014-09-16T10:45:42.857+00:00", "name":
"default", "status": "", "uid": "0"}}, {"geoRack": {"attributes": {"childAction": "", "descr": "", "
dn": "uni/fabric/site-default/building-default/floor-default/room-default/rack-Big_Data", "lc
Own": "local", "modTs": "2014-09-16T10:48:55.700+00:00", "name": "Big_Data", "status": "", "uid":
"15374"}}}]}
11:10:52 DEBUG -
method: GET
url:
https://10.0.130.71/api/node/class/topology/pod-1/node-1/dhcpClient.json?query-target=filte
r=or(eq(dhcpClient.nodeRole, "leaf"), eq(dhcpClient.nodeRole, "spine"), eq(dhcpClient.nodeRol
e, "unsupported"))&subscription=yes
response:
{"totalCount": "4", "subscriptionId": "72057598349672520", "imdata": [{"dhcpClient": {"attribute
s": {"childAction": "", "clientEvent": "assigned", "decomissioned": "no", "dn": "client-[SAL1819S0
RY]", "fabricId": "1", "fwVer": "", "hwAddr": "00:00:00:00:00:00", "id": "SAL1819S0RY", "ip": "10.0.
48.95/32", "lcOwn": "local", "modTs": "2014-09-16T10:49:30.559+00:00", "model": "N9K-C9396PX", "n
ame": "LEAF_1", "nodeId": "101", "nodeRole": "leaf", "podId": "1", "spineLevel": "0", "status": "", "s
upported": "yes"}}, {"dhcpClient": {"attributes": {"childAction": "", "clientEvent": "assigned",
"decomissioned": "no", "dn": "client-[FGE18200AWL]", "fabricId": "1", "fwVer": "", "hwAddr": "00:00
:00:00:00:00", "id": "FGE18200AWL", "ip": "10.0.48.94/32", "lcOwn": "local", "modTs": "2014-09-16T
10:53:22.329+00:00", "model": "N9K-C9508", "name": "SPINE_2", "nodeId": "201", "nodeRole": "spine",
"podId": "1", "spineLevel": "1", "status": "", "supported": "yes"}}, {"dhcpClient": {"attributes":
{"childAction": "", "clientEvent": "assigned", "decomissioned": "no", "dn": "client-[SAL1819S0M8
]", "fabricId": "1", "fwVer": "", "hwAddr": "00:00:00:00:00:00", "id": "SAL1819S0M8", "ip": "10.0.48
.93/32", "lcOwn": "local", "modTs": "2014-09-16T10:55:34.254+00:00", "model": "N9K-C9396PX", "nam
e": "LEAF_2", "nodeId": "102", "nodeRole": "leaf", "podId": "1", "spineLevel": "0", "status": "", "sup
ported": "yes"}}, {"dhcpClient": {"attributes": {"childAction": "", "clientEvent": "assigned", "d
ecomissioned": "no", "dn": "client-[SAL1816QWFA]", "fabricId": "1", "fwVer": "", "hwAddr": "00:00:0
0:00:00:00", "id": "SAL1816QWFA", "ip": "10.0.48.92/32", "lcOwn": "local", "modTs": "2014-09-16T10
:55:46.332+00:00", "model": "N9K-C93128TX", "name": "LEAF_3", "nodeId": "103", "nodeRole": "leaf",
"podId": "1", "spineLevel": "0", "status": "", "supported": "yes"}}}]}

```

## Validating Fabric Topology

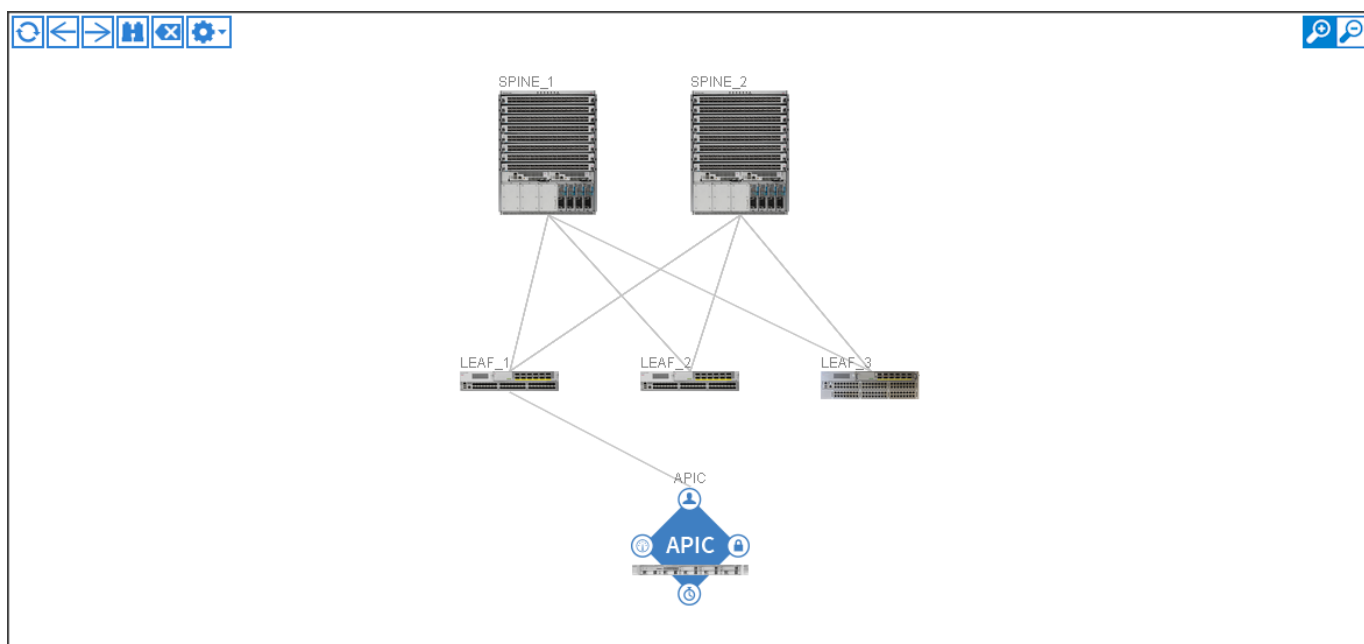
### Procedure

1. On the menu bar, choose **FABRIC > INVENTORY**.
2. In the **Navigation** pane, choose the pod that you want to view.
3. In the **Work** pane, click the **TOPOLOGY** tab. The displayed diagram shows all attached switches, APIC instances, and links.
4. (Optional) To view the port-level connectivity of a leaf switch or spine switch, double-click its icon in the topology diagram.

To return to the topology diagram, in the upper left corner of the **Work** pane click the **Previous View** icon.

5. (Optional) To refresh the topology diagram, in the upper left corner of the **Work** pane, click the **Refresh** icon.

Figure 17 Fabric Topology



## Validating Fabric Topology using Rest API

```
method: GET
url: https://10.0.130.71/api/node/class/fabricPod.json?subscription=yes
response:
{"totalCount": "1", "subscriptionId": "72057598349672524", "imdata": [{"fabricPod": {"attributes": {"childAction": "", "dn": "topology/pod-1", "id": "1", "lcOwn": "local", "modTs": "2014-09-16T10:45:42.798+00:00", "monPolDn": "uni/fabric/monfab-default", "status": ""}}}]
11:13:11 DEBUG -
method: GET
url: https://10.0.130.71/api/node/mo/sys.json
response: {"totalCount": "0", "imdata": []}
11:13:14 DEBUG -
method: GET
url: https://10.0.130.71/api/node/mo/sys.json
```

```

response: {"totalCount":"0","imdata":[]}
11:13:14 DEBUG -
method: GET
url: https://10.0.130.71/api/node/class/fabricNode.json?&subscription=yes
response:
{"totalCount":"5","subscriptionId":"72057598349672525","imdata":[{"fabricNode":{"attributes":{"adSt":"on","childAction":"","delayedHeartbeat":"no","dn":"topology/pod-1/node-101","fabricSt":"active","id":"101","lcOwn":"local","modTs":"2014-09-16T10:49:42.856+00:00","model":"N9K-C9396PX","monPolDn":"uni/fabric/monfab-default","name":"LEAF_1","role":"leaf","serial":"SAL1819S0RY","status":"","uid":"0","vendor":"Cisco Systems, Inc","version":""}},{"fabricNode":{"attributes":{"adSt":"on","childAction":"","delayedHeartbeat":"no","dn":"topology/pod-1/node-103","fabricSt":"active","id":"103","lcOwn":"local","modTs":"2014-09-16T10:56:42.972+00:00","model":"N9K-C93128TX","monPolDn":"uni/fabric/monfab-default","name":"LEAF_3","role":"leaf","serial":"SAL1816QWFA","status":"","uid":"0","vendor":"Cisco Systems, Inc","version":""}},{"fabricNode":{"attributes":{"adSt":"on","childAction":"","delayedHeartbeat":"no","dn":"topology/pod-1/node-201","fabricSt":"active","id":"201","lcOwn":"local","modTs":"2014-09-16T10:53:42.921+00:00","model":"N9K-C9508","monPolDn":"uni/fabric/monfab-default","name":"SPINE_2","role":"spine","serial":"FGE18200AWL","status":"","uid":"0","vendor":"Cisco Systems, Inc","version":""}},{"fabricNode":{"attributes":{"adSt":"on","childAction":"","delayedHeartbeat":"no","dn":"topology/pod-1/node-1","fabricSt":"unknown","id":"1","lcOwn":"local","modTs":"2014-09-16T10:46:47.175+00:00","model":"APIC","monPolDn":"uni/fabric/monfab-default","name":"APIC_1","role":"controller","serial":"","status":"","uid":"0","vendor":"Cisco Systems, Inc","version":""}},{"fabricNode":{"attributes":{"adSt":"on","childAction":"","delayedHeartbeat":"no","dn":"topology/pod-1/node-102","fabricSt":"active","id":"102","lcOwn":"local","modTs":"2014-09-16T10:56:42.971+00:00","model":"N9K-C9396PX","monPolDn":"uni/fabric/monfab-default","name":"LEAF_2","role":"leaf","serial":"SAL1819S0M8","status":"","uid":"0","vendor":"Cisco Systems, Inc","version":""}}]}]}
11:13:14 DEBUG -
method: GET
url:
https://10.0.130.71/api/node/class/fabricLink.json?rsp-subtree-include=health,fault-count&subscription=yes
response:
{"totalCount":"7","subscriptionId":"72057598349672526","imdata":[{"fabricLink":{"attributes":{"childAction":"","dn":"topology/pod-1/lkcnt-201/lnk-103-1-97-to-201-1-4","lcOwn":"local","linkState":"ok","modTs":"2014-09-16T10:55:50.721+00:00","monPolDn":"uni/fabric/monfab-default","n1":"103","n2":"201","p1":"97","p2":"4","s1":"1","s2":"1","status":"","wiringIssues":"","children":[{"faultCounts":{"attributes":{"childAction":"","crit":"0","maj":"0","minor":"0","rn":"fltCnts","status":"","warn":"0"}}}]}},{"fabricLink":{"attributes":{"childAction":"","dn":"topology/pod-1/lkcnt-201/lnk-102-1-52-to-201-1-2","lcOwn":"local","linkState":"ok","modTs":"2014-09-16T10:55:51.073+00:00","monPolDn":"uni/fabric/monfab-default","n1":"102","n2":"201","p1":"52","p2":"2","s1":"1","s2":"1","status":"","wiringIssues":"","children":[{"faultCounts":{"attributes":{"childAction":"","crit":"0","maj":"0","minor":"0","rn":"fltCnts","status":"","warn":"0"}}}]}},{"fabricLink":{"attributes":{"childAction":"","dn":"topology/pod-1/lkcnt-101/lnk-201-1-6-to-101-1-50","lcOwn":"local","linkState":"ok","modTs":"2014-09-16T10:53:26.370+00:00","monPolDn":"uni/fabric/monfab-default","n1":"201","n2":"101","p1":"6","p2":"50","s1":"1","s2":"1","status":"","wiringIssues":"","children":[{"faultCounts":{"attributes":{"childAction":"","crit":"0","maj":"0","minor":"0","rn":"fltCnts","status":"","warn":"0"}}}]}},{"fabricLink":{"attributes":{"childAction":"","dn":"topology/pod-1/lkcnt-102/lnk-201-1-2-to-102-1-52","lcOwn":"local","linkState":"ok","modTs":"2014-09-16T10:55:34.874+00:00","monPolDn":"uni/fabric/monfab-default","n1":"201","n2":"102","p1":"2","p2":"52","s1":"1","s2":"1","status":"","wiringIssues":"","children":[{"faultCounts":{"attributes":{"childAction":"","crit":"0","maj":"0","minor":"0","rn":"fltCnts","status":"","warn":"0"}}}]}},{"fabricLink":{"attributes":{"childAction":"","dn":"topology/pod-1/lkcnt-103/lnk-201-1-4-to-103-1-97","lcOwn":"local","linkState":"ok","modTs":"2014-09-16T10:55:47.250+00:00","monPolDn":"uni/fabric/monfab-default","n1":"201","n2":"103","p1":"4","p2":"97","s1":"1","s2":"1","status":"","wiringIssues":"","children":[{"faultCounts":{"attributes":{"childAction":"","crit":"0","maj":"0","minor":"0","rn":"fltCnts","status":"","warn":"0"}}}]}},{"fabricLink":{"attributes":{"childAction":"","dn":"topology/pod-1/lnk-101-1-48-to-1-2-2","lcOwn":"local","linkState":"ok","modTs":"2014-09-16T10:49:33.226+00:00","monPolDn":"uni/fabric/monfab-default","n1":"101","n2":"1","p1":"48","p2":"2",

```



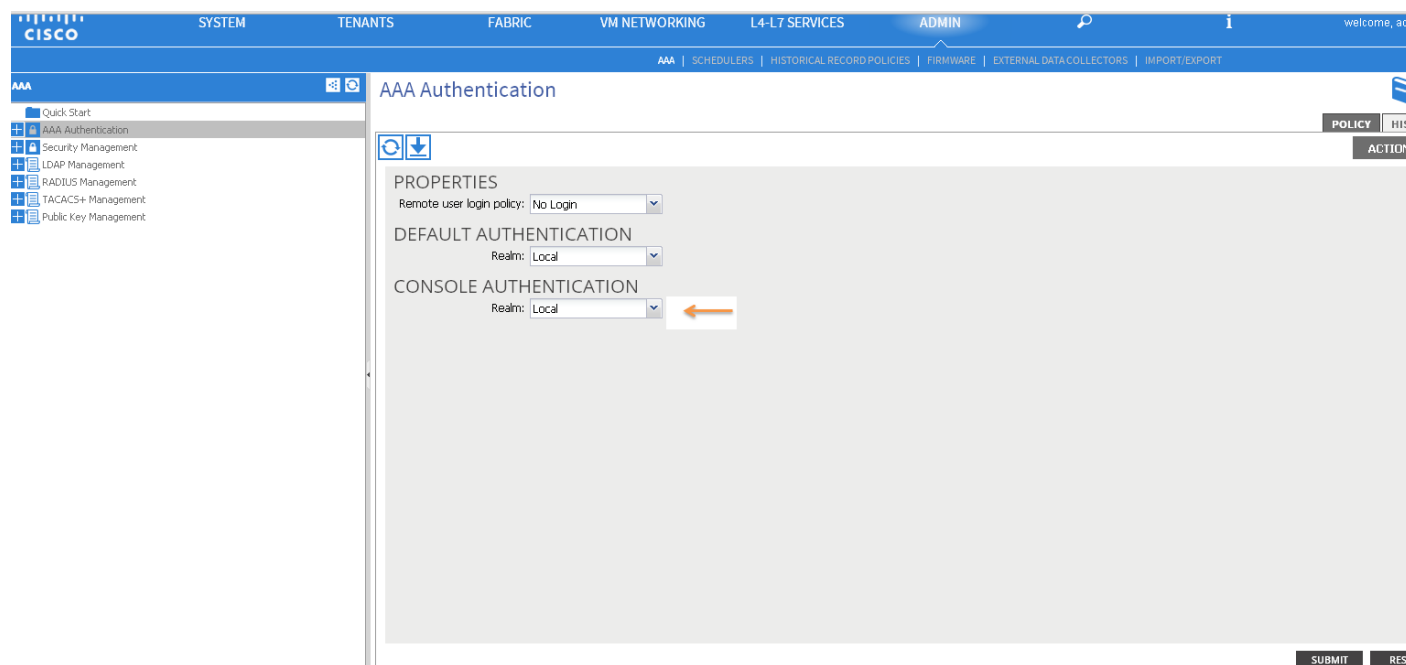
```
"s1": "1", "s2": "2", "status": "", "wiringIssues": "", "children": [{"faultCounts": {"attributes": {"childAction": "", "crit": "0", "maj": "0", "minor": "0", "rn": "fltCnts", "status": "", "warn": "0"}}}], {"fabricLink": {"attributes": {"childAction": "", "dn": "topology/pod-1/lnkcnt-201/lnk-101-1-50-to-201-1-6", "lcOwn": "local", "linkState": "ok", "modTs": "2014-09-16T10:53:22.741+00:00", "monPolDn": "uni/fabric/monfab-default", "n1": "101", "n2": "201", "p1": "50", "p2": "6", "s1": "1", "s2": "1", "status": "", "wiringIssues": ""}, "children": [{"faultCounts": {"attributes": {"childAction": "", "crit": "0", "maj": "0", "minor": "0", "rn": "fltCnts", "status": "", "warn": "0"}}}]}]}}
```

## Creating User Accounts

### Procedure

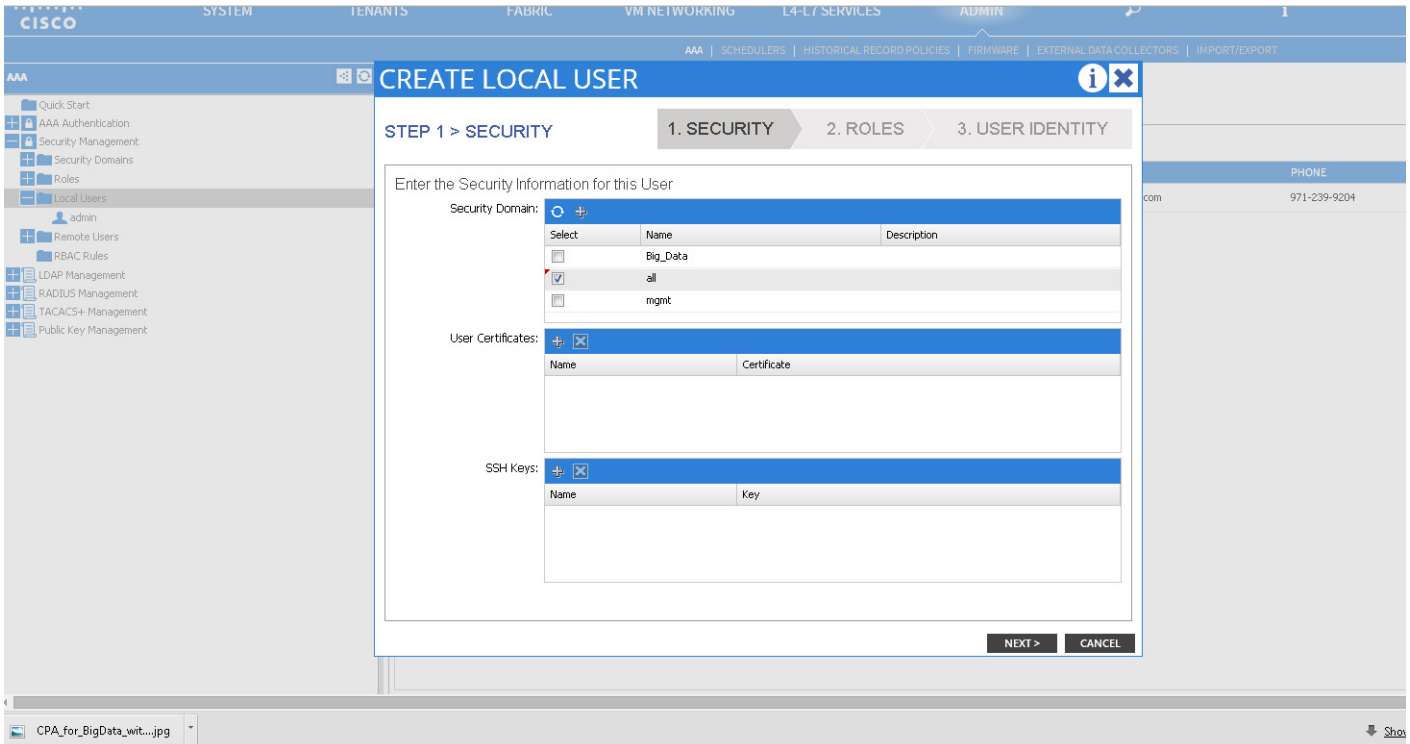
1. On the menu bar, choose **ADMIN > AAA**
2. In the **Navigation** pane, click **AAA Authentication**. In the **Work** pane, the **AAA Authentication** dialog box is displayed.
3. Verify that in the default Authentication field, the **Realm** field displays as **Local**.

**Figure 18** AAA Authentication



4. In the Navigation pane, expand **Security Management > Local Users**. The admin user is present by default.
5. In the navigation pane, right-click **Create Local User**. The **Create Local User** dialog box opens.
6. Under the **Security** dialog box, choose the desired security domain for the user, and click **Next**.

Figure 19 Creating Local User



The **Roles** dialog box opens.

7. In the **Roles** dialog box, click the radio buttons to choose the roles for your user, and click **Next**. You can provide read-only or read/write privileges.
8. In the **User Identity** dialog box, perform the following actions:
  - a. In the **Login ID** field, add an ID.
  - b. In the **Password** field, type the password.
  - c. In the **Confirm Password** field, confirm the password.
  - d. Click **Finish**.
  - e. Type other parameters if desired.

Figure 20 User Identity

The screenshot shows the Cisco Configuration Assistant (CCA) interface for creating a local user. The main window is titled 'CREATE LOCAL USER' and is in 'STEP 3 > USER IDENTITY'. The form contains the following fields and options:

- Login ID:
- Password:
- Confirm Password:
- First Name:
- Last Name:
- Phone:
- Email:
- Description:
- Account Status:  Inactive  Active
- Account Expires:  Yes  No

Buttons for 'FINISH' and 'CANCEL' are located at the bottom right of the form.

- In the **Navigation** pane, click the name of the user that you created. In the **Work** pane, expand the + sign next to your user in the Security Domains area. The access privileges for your user are displayed.

## Using Rest API

```

12:31:37 DEBUG -
12:31:43 DEBUG -
method: GET
url:
https://10.0.130.71/api/node/class/polUni.json?target-subtree-class=aaaUserEp,aaaLdapEp,aaaRadiusEp,aaaTacacsPlusEp,aaaAuthRealm&query-target=subtree&subscription=yes
response:
{"totalCount": "5", "subscriptionId": "72057619824508933", "imdata": [{"aaaUserEp": {"attributes": {"childAction": "", "descr": "", "dn": "uni/userext", "lcOwn": "local", "modTs": "2014-09-16T10:45:42.891+00:00", "monPolDn": "uni/fabric/monfab-default", "name": "", "ownerKey": "", "ownerTag": "", "pwdStrengthCheck": "no", "status": "", "uid": "0"}}, {"aaaAuthRealm": {"attributes": {"childAction": "", "defRolePolicy": "no-login", "descr": "", "dn": "uni/userext/authrealm", "lcOwn": "local", "modTs": "2014-09-16T10:45:42.891+00:00", "name": "", "ownerKey": "", "ownerTag": "", "status": "", "uid": "0"}}, {"aaaRadiusEp": {"attributes": {"childAction": "", "descr": "", "dn": "uni/userext/radiusext", "lcOwn": "local", "modTs": "2014-09-16T10:45:42.891+00:00", "monPolDn": "uni/fabric/monfab-default", "name": "", "ownerKey": "", "ownerTag": "", "retries": "1", "status": "", "timeout": "5", "uid": "0"}}, {"aaaLdapEp": {"attributes": {"attribute": "CiscoAVPair", "basedn": "", "childAction": "", "descr": "", "dn": "uni/userext/ldapext", "filter": "cn=$userid", "lcOwn": "local", "modTs": "2014-09-16T10:45:42.891+00:00", "monPolDn": "uni/fabric/monfab-default", "name": "", "ownerKey": "", "ownerTag": "", "retries": "1", "status": "", "timeout": "30", "uid": "0"}}, {"aaaTacacsPlusEp": {"attributes": {"childAction": "", "descr": "", "dn": "uni/userext/tacacsExt", "lcOwn": "local", "modTs": "2014-09-16T10:45:42.891+00:00", "monPolDn": "uni/fabric/monfab-default", "name": "", "ownerKey": "", "ownerTag": "", "retries": "1", "status": "", "timeout": "5", "uid": "0"}}}]}
12:31:47 DEBUG -
method: GET

```

```

url:
https://10.0.130.71/api/node/class/aaaDomain.json?query-target-filter=ne (aaaDomain.name, "c
ommon")&subscription=yes
response:
{"totalCount": "3", "subscriptionId": "72057619824508934", "imdata": [{"aaaDomain": {"attributes
": {"childAction": "", "descr": "", "dn": "uni/userext/domain-mgmt", "lcOwn": "local", "modTs": "201
4-09-16T10:45:42.891+00:00", "name": "mgmt", "ownerKey": "", "ownerTag": "", "status": "", "uid": "0
"}}, {"aaaDomain": {"attributes": {"childAction": "", "descr": "", "dn": "uni/userext/domain-all"
, "lcOwn": "local", "modTs": "2014-09-16T10:45:42.891+00:00", "name": "all", "ownerKey": "", "owner
Tag": "", "status": "", "uid": "0"}}, {"aaaDomain": {"attributes": {"childAction": "", "descr": "",
dn": "uni/userext/domain-Big_Data", "lcOwn": "local", "modTs": "2014-09-16T12:18:31.616+00:00",
"name": "Big_Data", "ownerKey": "", "ownerTag": "", "status": "", "uid": "15374"}}}]}
12:31:53 DEBUG -
method: GET
url: https://10.0.130.71/api/node/class/aaaRole.json?subscription=yes
response:
{"totalCount": "10", "subscriptionId": "72057619824508935", "imdata": [{"aaaRole": {"attributes
": {"childAction": "", "descr": "", "dn": "uni/userext/role-aaa", "lcOwn": "local", "modTs": "2014-09
-16T10:45:42.891+00:00", "name": "aaa", "ownerKey": "", "ownerTag": "", "priv": "aaa", "status": "",
"uid": "0"}}, {"aaaRole": {"attributes": {"childAction": "", "descr": "", "dn": "uni/userext/role-
access-admin", "lcOwn": "local", "modTs": "2014-09-16T10:45:42.891+00:00", "name": "access-admin
", "ownerKey": "", "ownerTag": "", "priv": "access-connectivity-l1,access-connectivity-l2,access
-connectivity-l3,access-connectivity-mgmt,access-connectivity-util,access-equipment,access
-protocol-l1,access-protocol-l2,access-protocol-l3,access-protocol-mgmt,access-protocol-ops
,access-protocol-util,access-qos", "status": "", "uid": "0"}}, {"aaaRole": {"attributes": {"chi
ldAction": "", "descr": "", "dn": "uni/userext/role-admin", "lcOwn": "local", "modTs": "2014-09-16T
10:45:42.891+00:00", "name": "admin", "ownerKey": "", "ownerTag": "", "priv": "admin", "status": "",
"uid": "0"}}, {"aaaRole": {"attributes": {"childAction": "", "descr": "", "dn": "uni/userext/role-
fabric-admin", "lcOwn": "local", "modTs": "2014-09-16T10:45:42.891+00:00", "name": "fabric-admin
", "ownerKey": "", "ownerTag": "", "priv": "fabric-connectivity-l1,fabric-connectivity-l2,fabric
-connectivity-l3,fabric-connectivity-mgmt,fabric-connectivity-util,fabric-equipment,fabric-
protocol-l1,fabric-protocol-l2,fabric-protocol-l3,fabric-protocol-mgmt,fabric-protocol-ops
,fabric-protocol-util", "status": "", "uid": "0"}}, {"aaaRole": {"attributes": {"childAction": "
", "descr": "", "dn": "uni/userext/role-nw-svc-admin", "lcOwn": "local", "modTs": "2014-09-16T10:4
5:42.891+00:00", "name": "nw-svc-admin", "ownerKey": "", "ownerTag": "", "priv": "nw-svc-device,nw
-svc-devshare,nw-svc-policy", "status": "", "uid": "0"}}, {"aaaRole": {"attributes": {"childActi
on": "", "descr": "", "dn": "uni/userext/role-ops", "lcOwn": "local", "modTs": "2014-09-16T10:45:42
.891+00:00", "name": "ops", "ownerKey": "", "ownerTag": "", "priv": "ops", "status": "", "uid": "0"}
}, {"aaaRole": {"attributes": {"childAction": "", "descr": "", "dn": "uni/userext/role-tenant-admin
", "lcOwn": "local", "modTs": "2014-09-16T10:45:42.891+00:00", "name": "tenant-admin", "ownerKey"
: "", "ownerTag": "", "priv": "access-connectivity-l1,access-connectivity-l2,access-connectivit
y-l3,access-connectivity-mgmt,access-connectivity-util,access-equipment,access-protocol-l1
,access-protocol-l2,access-protocol-l3,access-protocol-mgmt,access-protocol-ops,access-pro
tocol-util,access-qos,fabric-connectivity-l1,fabric-connectivity-l2,fabric-connectivity-l3
,fabric-connectivity-mgmt,fabric-connectivity-util,fabric-equipment,fabric-protocol-l1,fab
ric-protocol-l2,fabric-protocol-l3,fabric-protocol-mgmt,fabric-protocol-ops,fabric-protoco
l-util,nw-svc-device,nw-svc-devshare,nw-svc-policy,ops,tenant-connectivity-l1,tenant-conne
ctivity-l2,tenant-connectivity-l3,tenant-connectivity-mgmt,tenant-connectivity-util,tenant
-epg,tenant-ext-connectivity-l1,tenant-ext-connectivity-l2,tenant-ext-connectivity-l3,tena
nt-ext-connectivity-mgmt,tenant-ext-connectivity-util,tenant-ext-protocol-l1,tenant-ext-pr
otocol-l2,tenant-ext-protocol-l3,tenant-ext-protocol-mgmt,tenant-ext-protocol-util,tenant-
network-profile,tenant-protocol-l1,tenant-protocol-l2,tenant-protocol-l3,tenant-protocol-m
gmt,tenant-protocol-ops,tenant-protocol-util,tenant-qos,tenant-security,vmm-connectivity,v
mm-ep,vmm-policy,vmm-protocol-ops,vmm-security", "status": "", "uid": "0"}}, {"aaaRole": {"attr
ibutes": {"childAction": "", "descr": "", "dn": "uni/userext/role-tenant-ext-admin", "lcOwn": "loc
al", "modTs": "2014-09-16T10:45:42.891+00:00", "name": "tenant-ext-admin", "ownerKey": "", "owner
Tag": "", "priv": "tenant-ext-connectivity-l1,tenant-ext-connectivity-l2,tenant-ext-connectiv
ity-l3,tenant-ext-connectivity-mgmt,tenant-ext-connectivity-util,tenant-ext-protocol-l1,te
nant-ext-protocol-l2,tenant-ext-protocol-l3,tenant-ext-protocol-mgmt,tenant-ext-protocol-u
til", "status": "", "uid": "0"}}, {"aaaRole": {"attributes": {"childAction": "", "descr": "", "dn": "
uni/userext/role-vmm-admin", "lcOwn": "local", "modTs": "2014-09-16T10:45:42.891+00:00", "name":
"vmm-admin", "ownerKey": "", "ownerTag": "", "priv": "vmm-connectivity,vmm-ep,vmm-policy,vmm-pr
otocol-ops,vmm-security", "status": "", "uid": "0"}}, {"aaaRole": {"attributes": {"childAction":
", "descr": "", "dn": "uni/userext/role-read-all", "lcOwn": "local", "modTs": "2014-09-16T10:45:4

```

```

2.891+00:00", "name": "read-all", "ownerKey": "", "ownerTag": "", "priv": "access-connectivity-11,
access-connectivity-12, access-connectivity-13, access-connectivity-mgmt, access-connectivity-
util, access-equipment, access-protocol-11, access-protocol-12, access-protocol-13, access-pro
tocol-mgmt, access-protocol-ops, access-protocol-util, access-qos, fabric-connectivity-11, fabr
ic-connectivity-12, fabric-connectivity-13, fabric-connectivity-mgmt, fabric-connectivity-uti
l, fabric-equipment, fabric-protocol-11, fabric-protocol-12, fabric-protocol-13, fabric-protoco
l-mgmt, fabric-protocol-ops, fabric-protocol-util, nw-svc-device, nw-svc-devshare, nw-svc-polici
y, ops, tenant-connectivity-11, tenant-connectivity-12, tenant-connectivity-13, tenant-connecti
vity-mgmt, tenant-connectivity-util, tenant-epg, tenant-ext-connectivity-11, tenant-ext-connecti
vity-12, tenant-ext-connectivity-13, tenant-ext-connectivity-mgmt, tenant-ext-connectivity-
util, tenant-ext-protocol-11, tenant-ext-protocol-12, tenant-ext-protocol-13, tenant-ext-proto
col-mgmt, tenant-ext-protocol-util, tenant-network-profile, tenant-protocol-11, tenant-protoco
l-12, tenant-protocol-13, tenant-protocol-mgmt, tenant-protocol-ops, tenant-protocol-util, tena
nt-qos, tenant-security, vmm-ep, vmm-policy, vmm-protocol-ops, vmm-security", "
status": "", "uid": "0"}]}}
12:31:54 DEBUG -
response:
{"totalCount": "0", "imdata": [{"topInfo": {"attributes": {"childAction": "", "currentTime": "2014
-09-17T12:28:07.798+00:00", "dn": "info", "id": "1", "role": "controller", "status": ""}}]}]}
12:32:28 DEBUG -
method: POST
url: https://10.0.130.71/api/node/mo/uni/userext/user-guest.json
payload{"aaaUser": {"attributes": {"dn": "uni/userext/user-guest", "name": "guest", "pwd": "passw
ord", "rn": "user-guest", "status": "created"}, "children": [{"aaaUserDomain": {"attributes": {"dn
": "uni/userext/user-guest/userdomain-all", "name": "all", "rn": "userdomain-all", "status": "cre
ated, modified"}, "children": [{"aaaUserRole": {"attributes": {"dn": "uni/userext/user-guest/use
rdomain-all/role-aaa", "name": "aaa", "rn": "role-aaa", "status": "created, modified"}, "children
": []}}, {"aaaUserRole": {"attributes": {"dn": "uni/userext/user-guest/userdomain-all/role-acces
s-admin", "name": "access-admin", "rn": "role-access-admin", "status": "created, modified"}, "chil
dren": []}}, {"aaaUserRole": {"attributes": {"dn": "uni/userext/user-guest/userdomain-all/role-
admin", "name": "admin", "rn": "role-admin", "status": "created, modified"}, "children": []}}, {"aaa
UserRole": {"attributes": {"dn": "uni/userext/user-guest/userdomain-all/role-fabric-admin", "n
ame": "fabric-admin", "rn": "role-fabric-admin", "status": "created, modified"}, "children": []}},
{"aaaUserRole": {"attributes": {"dn": "uni/userext/user-guest/userdomain-all/role-nw-svc-admi
n", "name": "nw-svc-admin", "rn": "role-nw-svc-admin", "status": "created, modified"}, "children":
[]}}, {"aaaUserRole": {"attributes": {"dn": "uni/userext/user-guest/userdomain-all/role-ops", "
name": "ops", "rn": "role-ops", "status": "created, modified"}, "children": []}}, {"aaaUserRole": {"
attributes": {"dn": "uni/userext/user-guest/userdomain-all/role-read-all", "name": "read-all",
"rn": "role-read-all", "status": "created, modified"}, "children": []}}, {"aaaUserRole": {"attribu
tes": {"dn": "uni/userext/user-guest/userdomain-all/role-tenant-admin", "name": "tenant-admin",
"rn": "role-tenant-admin", "status": "created, modified"}, "children": []}}, {"aaaUserRole": {"at
tributes": {"dn": "uni/userext/user-guest/userdomain-all/role-tenant-ext-admin", "name": "tena
nt-ext-admin", "rn": "role-tenant-ext-admin", "status": "created, modified"}, "children": []}}, {"
aaaUserRole": {"attributes": {"dn": "uni/userext/user-guest/userdomain-all/role-vmm-admin", "n
ame": "vmm-admin", "rn": "role-vmm-admin", "status": "created, modified"}, "children": []}}]}]}]}
response: {"imdata": []}

```

## Adding Management Access

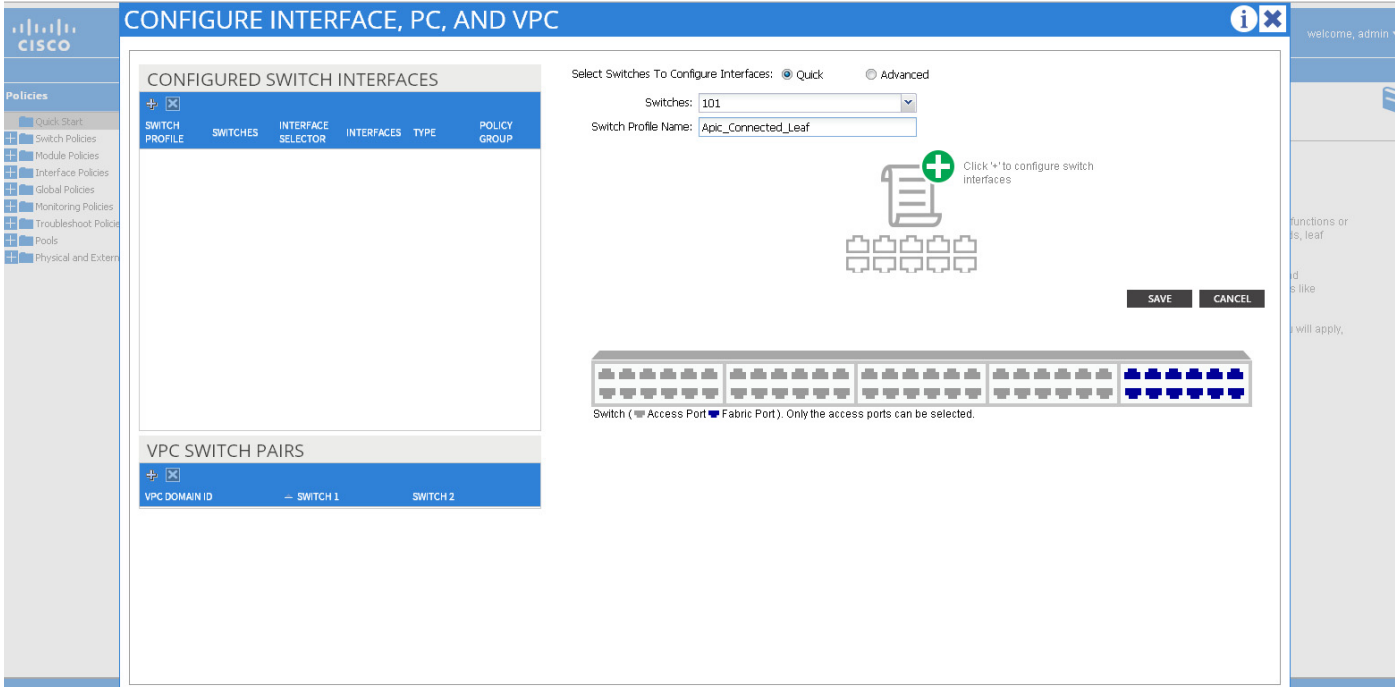
### Configuring In-Band Management Access using GUI

#### Procedure

1. On the menu bar, choose **FABRIC > Access Policies**. In the **Work** pane, click **Configure an Interface, PC and VPC**.
2. In the **Configure Interface, PC, and VPC** dialog box, click the **large +** icon next to the switch diagram to create a new profile and configure VLANs for the APIC.

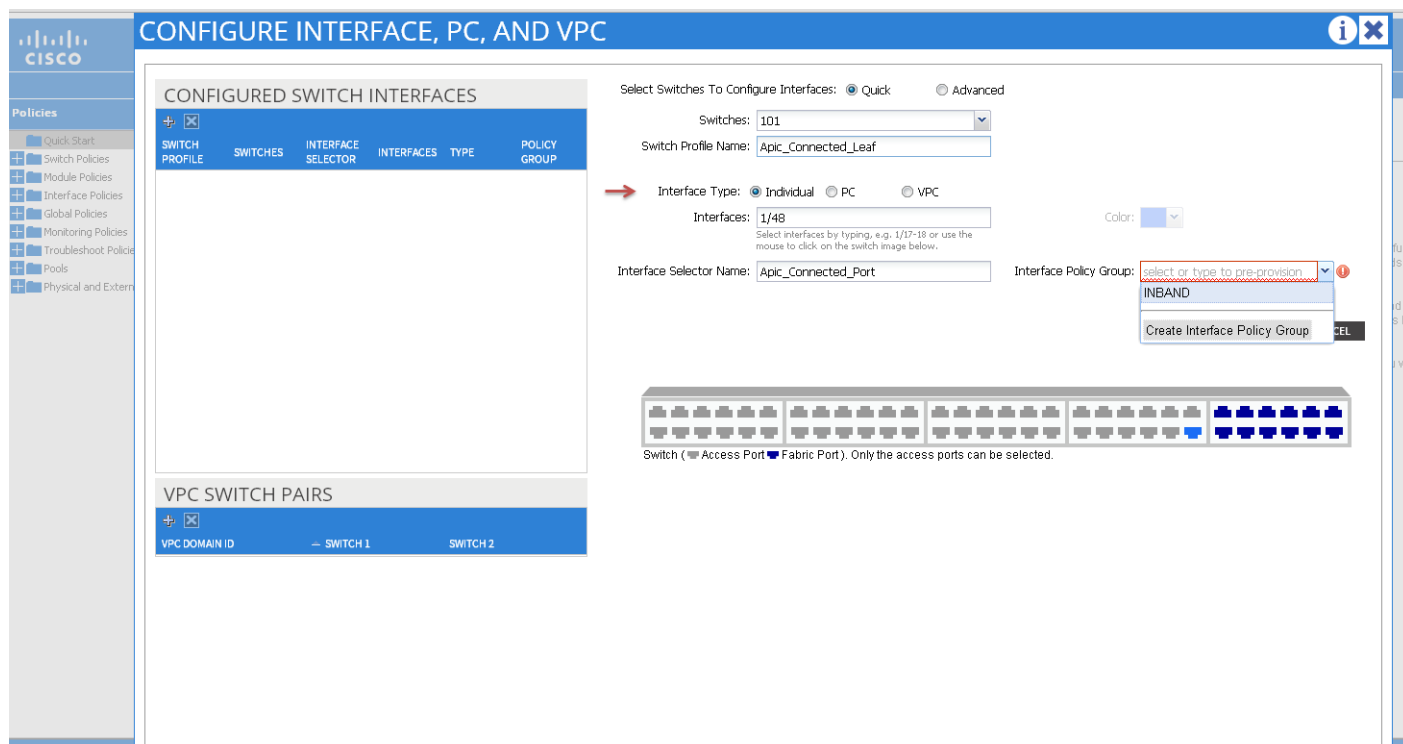
In the **Switches** field, from drop-down list, check the check boxes for the switches to which the APICs are connected. (leaf1, leaf2 and leaf3).

*Figure 21*      *Configuring Interface, PC, and VPC*



3. In the Switch **Profile Name** field, enter a name for the profile (Apic\_Connected\_Leaf).
4. Click the + icon to configure the ports.
5. Verify that in the Interface Type area, the Individual radio button is selected.

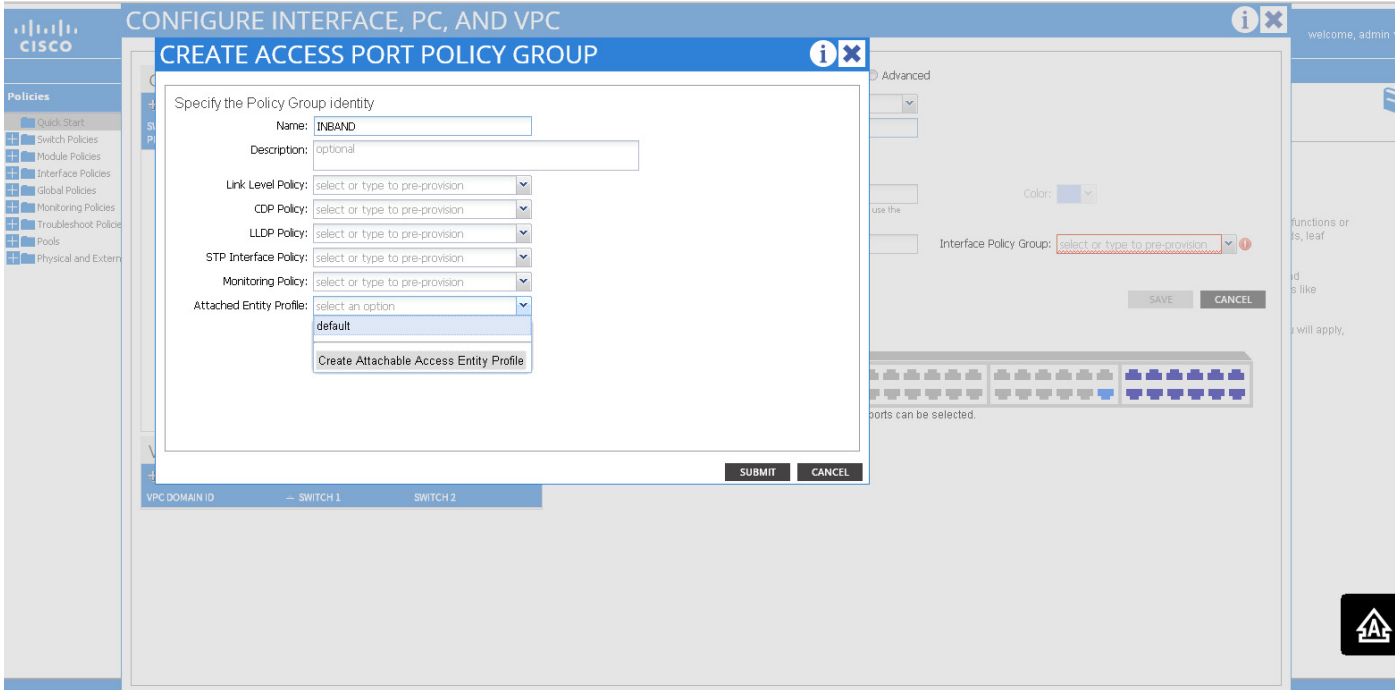
**Figure 22** *Configuring APIC Interface*



6. In the **Interfaces** field, enter the ports to which APICs are connected (1/48).
7. In the **Interface Selector Name** field, enter the name of the port profile (Apic\_Connected\_Port).
8. In the **Interface Policy Group** field, from drop-down list, choose **Create Interface Policy Group**.
9. In the **Create Access Port Policy Group** dialog box, perform the following actions:
  - a. In the **Name** field, enter the name of the policy group (INBAND).
  - b. You can leave the default values in the rest of the fields as they are. In the **Attached Entity Profile** field, choose **Create Attachable Access Entity Profile**.

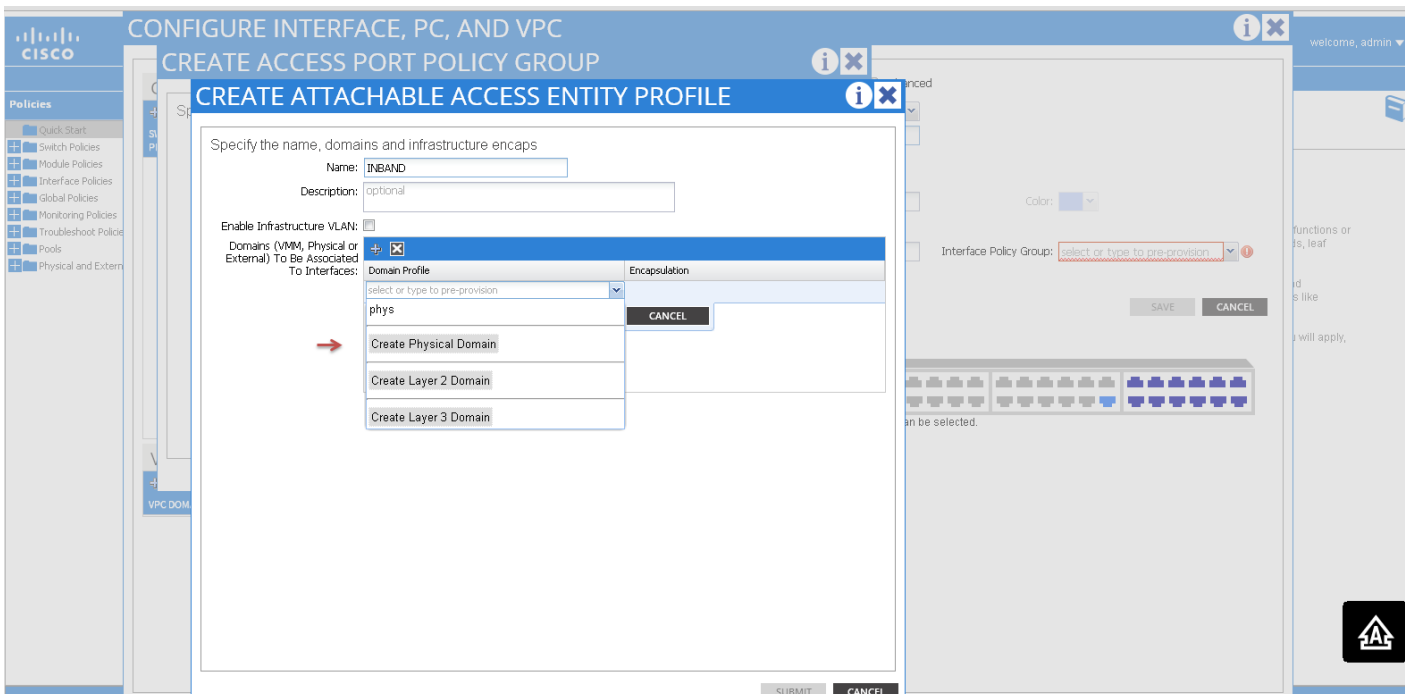
These new attach entity profile enables you to specify the VLAN ranges that will be used.

Figure 23 Creating Access Port Policy Group



10. In the **Create Attachable Access Entity Profile** dialog box, perform the following actions:
  1. In the **Name** field, enter a name (INBAND).
  2. Expand Domains to be **Associated to Interfaces** field. In the **Domain Profile** field, from the drop-down list, choose **Create Physical Domain**.

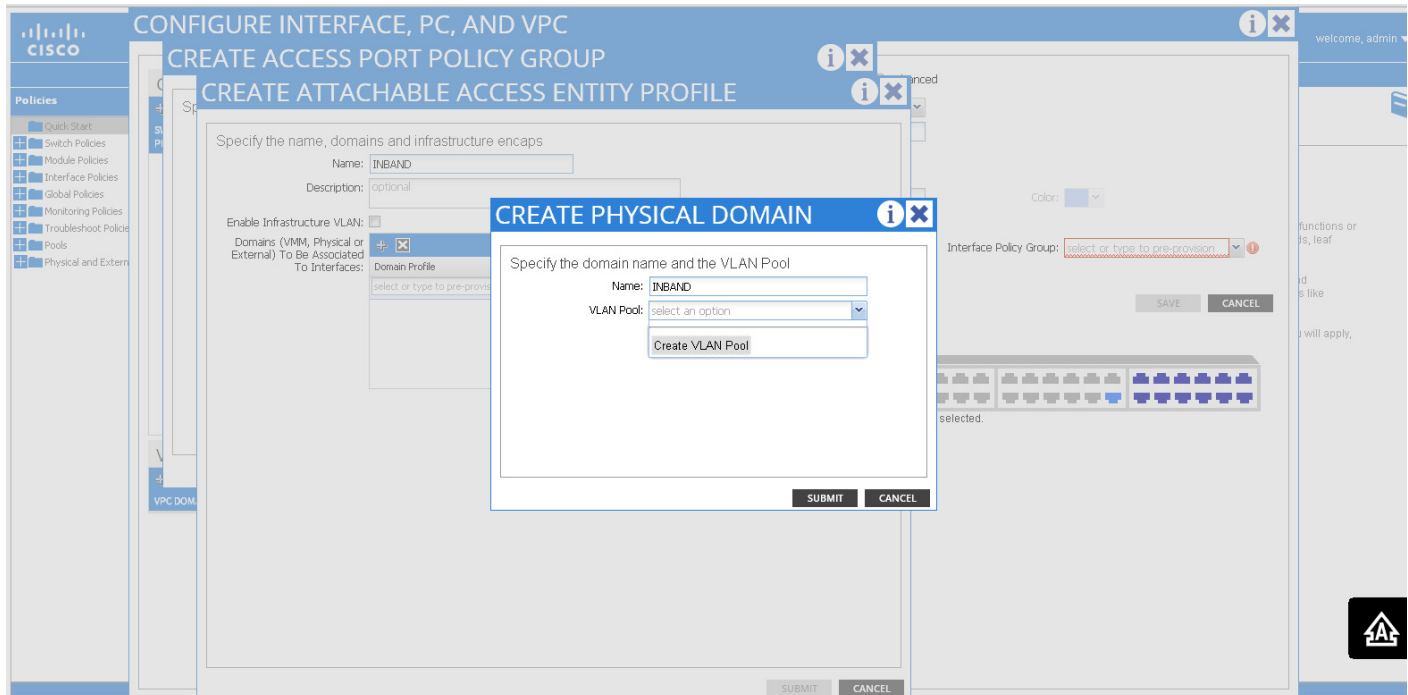
Figure 24 Creating Attachable Access Entity Profile





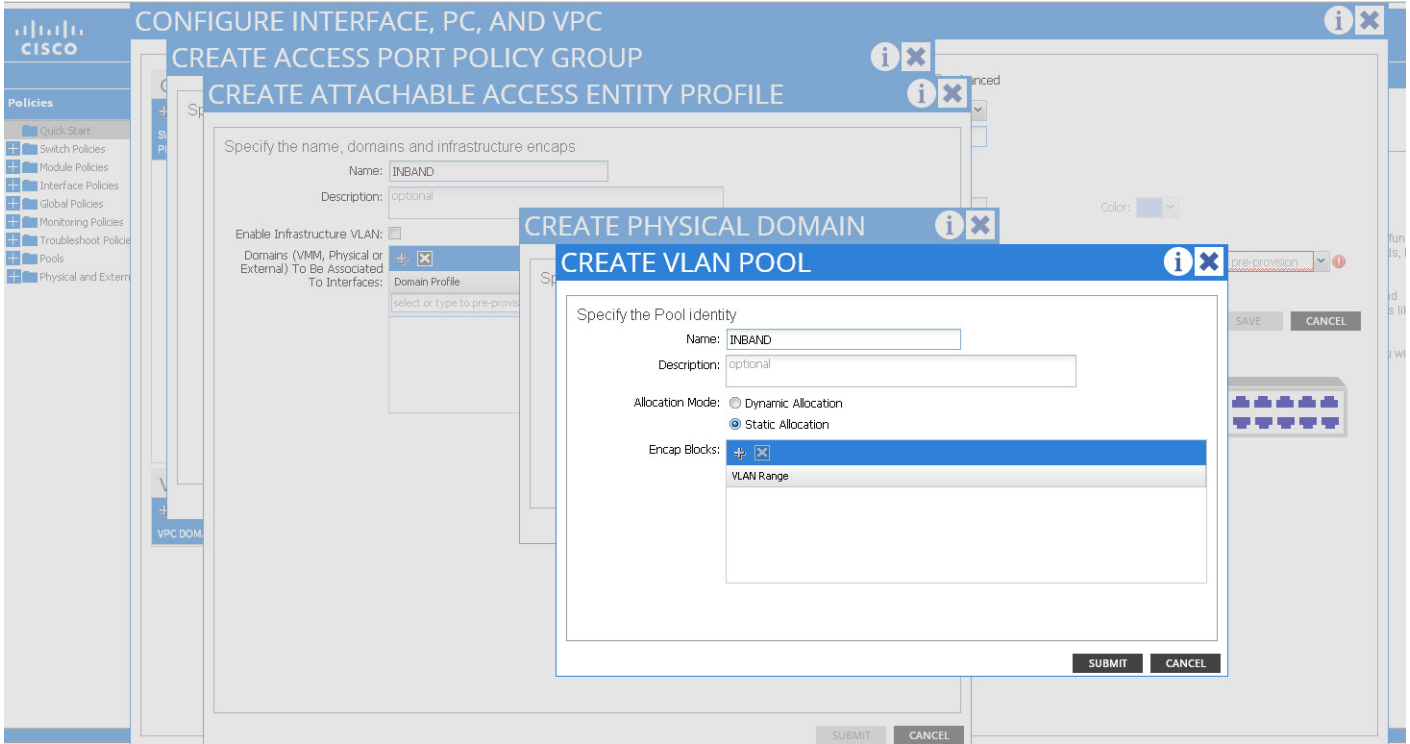
3. In the **Create Physical Domain** dialog box, in the **Name** field, enter the name (INBAND).
4. In the **VLAN Pool** field, from the drop-down list, choose **Create VLAN Pool**.

**Figure 25**      *Creating Physical Domain*



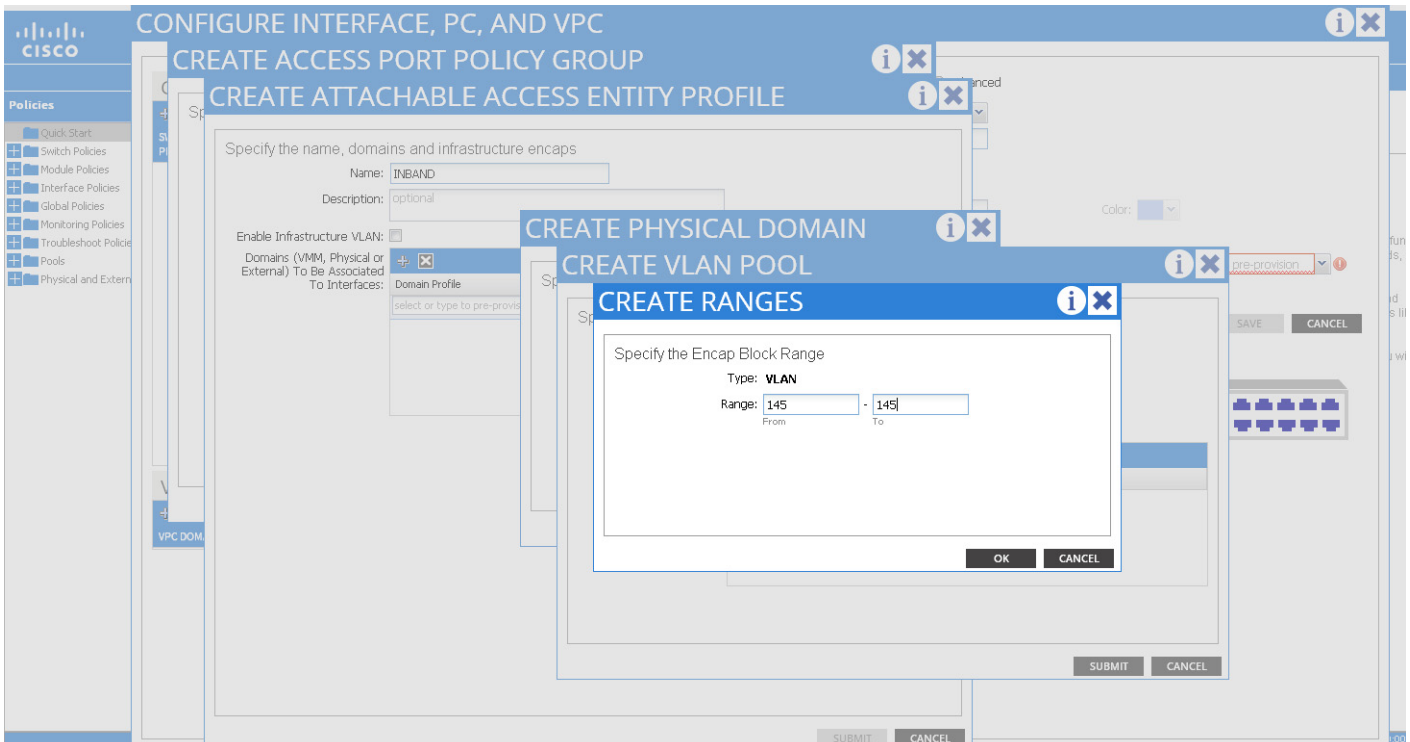
5. In the Create VLAN Pool dialog box, in the Name field, enter the pool name (INBAND).
6. In the Allocation Mode area, click the Static Allocation radio button.

Figure 26 Creating Vlan Pool



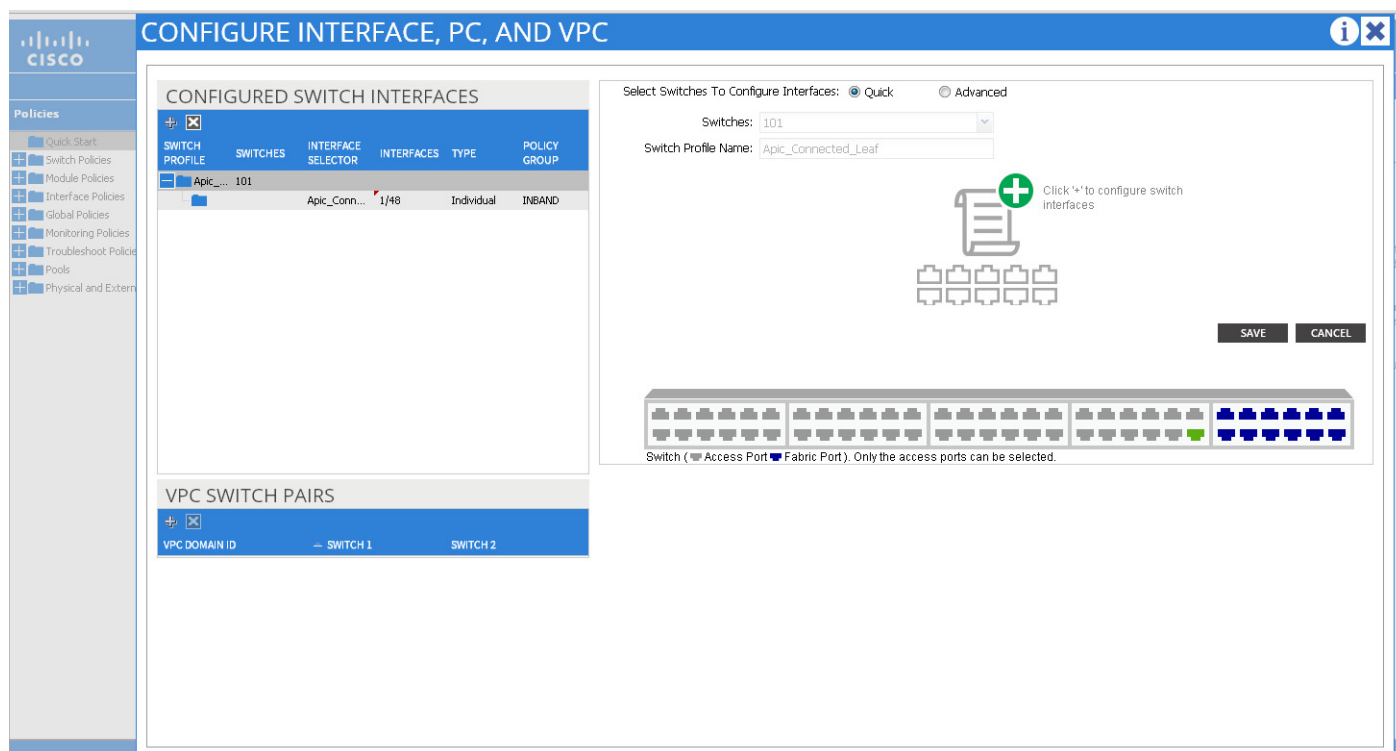
7. Expand Encap Blocks. In the Create Ranges dialog box, in the Range fields, add a VLAN range (145-145).

Figure 27 Creating vlan Range



8. In the **Create VLAN Pool** dialog box, click **Submit**.
9. In the **Create Physical Domain** dialog box, click **Submit**.
10. In the **Create Attachable Access Entity Profile** dialog box, click **Update** and then **Submit**.
11. In the **Create Access Port Policy Group** dialog box, click **Submit**.
12. In the **Configure Interface, PC, and VPC** dialog box, click **Save**.

**Figure 28** Saving the Configuration



The VLAN and the ports to which the APIC is connected are now configured.

## Using REST API

```
11:16:18 DEBUG - method: GET url:
https://10.0.130.71/api/node/class/fabricPod.json?subscription=yes response:
{"totalCount": "1", "subscriptionId": "72057598349672544", "imdata": [{"fabricPod": {"attributes": {"childAction": "", "dn": "topology/pod-1", "id": "1", "lcOwn": "local", "modTs": "2014-09-16T10:45:42.798+00:00", "monPolDn": "uni/fabric/monfab-default", "status": ""}}}]
11:16:18 DEBUG - method: GET url: https://10.0.130.71/api/node/mo/sys.json response:
{"totalCount": "0", "imdata": []}
11:16:21 DEBUG - method: GET url: https://10.0.130.71/api/node/mo/uni/infra.json response:
{"totalCount": "1", "imdata": [{"infraInfra": {"attributes": {"childAction": "", "dn": "uni/infra", "lcOwn": "local", "modTs": "2014-09-16T10:45:42.855+00:00", "monPolDn": "uni/fabric/monfab-default", "name": "infra", "ownerKey": "", "ownerTag": "", "status": "", "uid": "0"}}}]
11:16:25 DEBUG - method: undefined url:
https://10.0.130.71/api/node/class/infraNodeP.json?query-target=subtree&target-subtree-class=infraNodeP,infraRsAccPortP,infraLeafS,infraNodeBlk,infraRsAccNodePGrp&subscription=yes
response: {"totalCount": "0", "subscriptionId": "72057598349672545", "imdata": []}
```

```

11:16:25 DEBUG - method: undefined url:
https://10.0.130.71/api/node/class/infraAccPortP.json?query-target=subtree&target-subtree-
class=infraAccPortP,infraRtAccPortP,infraHPortS,infraPortBlk,infraRsAccBaseGrp,infraRsAccB
ndlSubgrp&subscription=yes response:
{"totalCount":"0","subscriptionId":"72057598349672546","imdata":[]}
11:16:25 DEBUG - method: undefined url:
https://10.0.130.71/api/node/class/infraAccBndlGrp.json?subscription=yes response:
{"totalCount":"0","subscriptionId":"72057598349672547","imdata":[]}
11:16:25 DEBUG - method: undefined url:
https://10.0.130.71/api/node/class/infraFexBndlGrp.json?subscription=yes response:
{"totalCount":"0","subscriptionId":"72057598349672549","imdata":[]}
11:16:25 DEBUG - method: undefined url:
https://10.0.130.71/api/node/class/infraAccPortGrp.json?subscription=yes response:
{"totalCount":"0","subscriptionId":"72057598349672548","imdata":[]}
11:16:25 DEBUG - method: undefined url:
https://10.0.130.71/api/node/mo/uni/fabric/protpol.json?query-target=subtree&target-subtr
ee-class=fabricProtPol,fabricExplicitGEp,fabricAutoGEp,fabricNodePEp&subscription=yes
response:
{"totalCount":"1","subscriptionId":"72057598349672551","imdata":[{"fabricProtPol":{"attrib
utes":{"childAction":"","descr":"","dn":"uni/fabric/protpol","lcOwn":"local","modTs":"2014
-09-16T10:45:42.857+00:00","monPolDn":"uni/fabric/monfab-default","name":"default","ownerK
ey":"","ownerTag":"","pairT":"explicit","status":"","uid":"0"}}}]
11:16:25 DEBUG - method: undefined url:
https://10.0.130.71/api/node/class/infraFexP.json?query-target=subtree&target-subtree-clas
s=infraFexP,infraFexBndlGrp,infraRtAccBaseGrp,infraHPortS,infraPortBlk,infraRsAccBaseGrp,i
nfraRsAccBndlSubgrp&subscription=yes response:
{"totalCount":"0","subscriptionId":"72057598349672550","imdata":[]}
11:16:25 DEBUG - method: undefined url:
https://10.0.130.71/api/node/class/fabricNode.json?subscription=yes response:
{"totalCount":"5","subscriptionId":"72057598349672552","imdata":[{"fabricNode":{"attribute
s":{"adSt":"on","childAction":"","delayedHeartbeat":"no","dn":"topology/pod-1/node-101","f
abricSt":"active","id":"101","lcOwn":"local","modTs":"2014-09-16T10:49:42.856+00:00","mode
l":"N9K-C9396PX","monPolDn":"uni/fabric/monfab-default","name":"LEAF_1","role":"leaf","ser
ial":"SAL1819S0RY","status":"","uid":"0","vendor":"Cisco Systems,
Inc","version":""}},{"fabricNode":{"attributes":{"adSt":"on","childAction":"","delayedHea
rtbeat":"no","dn":"topology/pod-1/node-103","fabricSt":"active","id":"103","lcOwn":"local"
,"modTs":"2014-09-16T10:56:42.972+00:00","model":"N9K-C93128TX","monPolDn":"uni/fabric/mon
fab-default","name":"LEAF_3","role":"leaf","serial":"SAL1816QWFA","status":"","uid":"0","v
endor":"Cisco Systems,
Inc","version":""}},{"fabricNode":{"attributes":{"adSt":"on","childAction":"","delayedHea
rtbeat":"no","dn":"topology/pod-1/node-201","fabricSt":"active","id":"201","lcOwn":"local"
,"modTs":"2014-09-16T10:53:42.921+00:00","model":"N9K-C9508","monPolDn":"uni/fabric/monfab
-default","name":"SPINE_2","role":"spine","serial":"FGE18200AWL","status":"","uid":"0","v
endor":"Cisco Systems,
Inc","version":""}},{"fabricNode":{"attributes":{"adSt":"on","childAction":"","delayedHea
rtbeat":"no","dn":"topology/pod-1/node-1","fabricSt":"unknown","id":"1","lcOwn":"local","m
odTs":"2014-09-16T10:46:47.175+00:00","model":"APIC","monPolDn":"uni/fabric/monfab-default
","name":"APIC_1","role":"controller","serial":"","status":"","uid":"0","vendor":"Cisco
Systems,
Inc","version":""}},{"fabricNode":{"attributes":{"adSt":"on","childAction":"","delayedHea
rtbeat":"no","dn":"topology/pod-1/node-102","fabricSt":"active","id":"102","lcOwn":"local"
,"modTs":"2014-09-16T10:56:42.971+00:00","model":"N9K-C9396PX","monPolDn":"uni/fabric/monf
ab-default","name":"LEAF_2","role":"leaf","serial":"SAL1819S0M8","status":"","uid":"0","v
endor":"Cisco Systems, Inc","version":""}}}]
11:16:30 DEBUG - method: GET url:
https://10.0.130.71/api/subscriptionRefresh.json?id=72057598349672452 response:
{"imdata":[]}
11:16:30 DEBUG - method: GET url:
https://10.0.130.71/api/subscriptionRefresh.json?id=72057598349672545 response:
{"imdata":[]}
11:16:30 DEBUG - method: GET url:
https://10.0.130.71/api/subscriptionRefresh.json?id=72057598349672546 response:
{"imdata":[]}

```

```

11:16:30 DEBUG - method: GET url:
https://10.0.130.71/api/subscriptionRefresh.json?id=72057598349672547 response:
{"imdata": []}
11:16:30 DEBUG - method: GET url:
https://10.0.130.71/api/subscriptionRefresh.json?id=72057598349672548 response:
{"imdata": []}
11:16:30 DEBUG - method: GET url:
https://10.0.130.71/api/subscriptionRefresh.json?id=72057598349672550 response:
{"imdata": []}
11:16:30 DEBUG - method: GET url:
https://10.0.130.71/api/subscriptionRefresh.json?id=72057598349672549 response:
{"imdata": []}
11:16:30 DEBUG - method: GET url:
https://10.0.130.71/api/subscriptionRefresh.json?id=72057598349672551 response:
{"imdata": []}
11:16:30 DEBUG - method: GET url:
https://10.0.130.71/api/subscriptionRefresh.json?id=72057598349672552 response:
{"imdata": []}
11:16:53 DEBUG - method: GET url:
https://10.0.130.71/api/node/class/infraAccPortGrp.json?subscription=yes response:
{"totalCount": "0", "subscriptionId": "72057598349672553", "imdata": []}
11:16:53 DEBUG - method: GET url:
https://10.0.130.71/api/node/class/infraAccBndlGrp.json?query-target-filter=eq(infraAccBndlGrp.lagT, "link")&query-target=subtree&target-subtree-class=infraAccBndlGrp,infraAccBndlSubgrp&subscription=yes response:
{"totalCount": "0", "subscriptionId": "72057598349672554", "imdata": []}
11:16:53 DEBUG - method: GET url:
https://10.0.130.71/api/node/class/infraAccBndlGrp.json?query-target-filter=eq(infraAccBndlGrp.lagT, "node")&query-target=subtree&target-subtree-class=infraAccBndlGrp,infraAccBndlSubgrp&subscription=yes response:
{"totalCount": "0", "subscriptionId": "72057598349672555", "imdata": []}
11:16:53 DEBUG - method: GET url:
https://10.0.130.71/api/node/class/infraAccPortGrp.json?subscription=yes response:
{"totalCount": "0", "subscriptionId": "72057598349672556", "imdata": []}
11:16:53 DEBUG - method: GET url:
https://10.0.130.71/api/node/class/infraAccBndlGrp.json?query-target-filter=eq(infraAccBndlGrp.lagT, "link")&query-target=subtree&target-subtree-class=infraAccBndlGrp,infraAccBndlSubgrp&subscription=yes response:
{"totalCount": "0", "subscriptionId": "72057598349672557", "imdata": []}
11:16:53 DEBUG - method: GET url:
https://10.0.130.71/api/node/class/infraAccBndlGrp.json?query-target-filter=eq(infraAccBndlGrp.lagT, "node")&query-target=subtree&target-subtree-class=infraAccBndlGrp,infraAccBndlSubgrp&subscription=yes response:
{"totalCount": "0", "subscriptionId": "72057598349672558", "imdata": []}
11:17:00 DEBUG - method: GET url:
https://10.0.130.71/api/subscriptionRefresh.json?id=72057598349672452 response:
{"imdata": []}
11:17:00 DEBUG - method: GET url:
https://10.0.130.71/api/subscriptionRefresh.json?id=72057598349672545 response:
{"imdata": []}
11:17:00 DEBUG - method: GET url:
https://10.0.130.71/api/subscriptionRefresh.json?id=72057598349672546 response:
{"imdata": []}
11:17:00 DEBUG - method: GET url:
https://10.0.130.71/api/subscriptionRefresh.json?id=72057598349672547 response:
{"imdata": []}
11:17:00 DEBUG - method: GET url:
https://10.0.130.71/api/subscriptionRefresh.json?id=72057598349672556 response:
{"imdata": []}
11:17:00 DEBUG - method: GET url:
https://10.0.130.71/api/subscriptionRefresh.json?id=72057598349672549 response:
{"imdata": []}

```

```

11:17:00 DEBUG - method: GET url:
https://10.0.130.71/api/subscriptionRefresh.json?id=72057598349672550 response:
{"imdata": []}
11:17:00 DEBUG - method: GET url:
https://10.0.130.71/api/subscriptionRefresh.json?id=72057598349672552 response:
{"imdata": []}
11:17:00 DEBUG - method: GET url:
https://10.0.130.71/api/subscriptionRefresh.json?id=72057598349672551 response:
{"imdata": []}
11:17:00 DEBUG - method: GET url:
https://10.0.130.71/api/subscriptionRefresh.json?id=72057598349672557 response:
{"imdata": []}
11:17:00 DEBUG - method: GET url:
https://10.0.130.71/api/subscriptionRefresh.json?id=72057598349672558 response:
{"imdata": []}
11:17:03 DEBUG - method: GET url: https://10.0.130.71/api/node/mo/info.json response:
{"totalCount": "0", "imdata": [{"topInfo": {"attributes": {"childAction": "", "currentTime": "2014-09-16T11:12:53.334+00:00", "dn": "info", "id": "1", "role": "controller", "status": ""}}]}]
11:17:12 DEBUG - method: GET url: https://10.0.130.71/api/aaaRefresh.json response:
{"imdata": [{"aaaLogin": {"attributes": {"token": "nbkFhU2RM9qwWViy9rqwuDJdPuGHc7R1fEuGLhgdG0Lkm1TzJfcLfDGqklQBUBwmh4mywG01zBA75Cj5GlypQX5RVhgXcFto4VsXrsr3Q5IALJ6lo2HxvJUiuEgMG0FLREce7+2a5JCo0DrNdaJr5ypEBRRiKwC8h8FwSInVtVM=", "siteFingerprint": "9q9Iiud2vywcjIRw", "urlToken": "", "refreshTimeoutSeconds": "600", "maximumLifetimeSeconds": "86400", "guiIdleTimeoutSeconds": "1200", "restTimeoutSeconds": "90", "creationTime": "1410865766", "firstLoginTime": "1410864469", "userName": "admin", "remoteUser": "false", "unixUserId": "15374", "sessionId": "mAnzMcKBT22geEb+uXjwaQ==", "lastName": "", "firstName": "", "version": "1.0(1e)", "buildTime": "Mon Aug 04 08:29:29 PDT 2014", "node": "topology/pod-1/node-1"}, "children": [{"aaaUserDomain": {"attributes": {"name": "all", "readRoleBitmask": "0", "writeRoleBitmask": "1"}}}]}]}]}]
11:17:13 DEBUG - method: GET url:
https://10.0.130.71/api/node/class/fabricNode.json?query-target-filter=eq(fabricNode.role, "leaf")&subscription=yes response:
{"totalCount": "3", "subscriptionId": "72057598349672559", "imdata": [{"fabricNode": {"attributes": {"adSt": "on", "childAction": "", "delayedHeartbeat": "no", "dn": "topology/pod-1/node-101", "fabricSt": "active", "id": "101", "lCOwn": "local", "modTs": "2014-09-16T10:49:42.856+00:00", "model": "N9K-C9396PX", "monPolDn": "uni/fabric/monfab-default", "name": "LEAF_1", "role": "leaf", "serial": "SAL1819S0RY", "status": "", "uid": "0", "vendor": "Cisco Systems, Inc", "version": ""}}, {"fabricNode": {"attributes": {"adSt": "on", "childAction": "", "delayedHeartbeat": "no", "dn": "topology/pod-1/node-103", "fabricSt": "active", "id": "103", "lCOwn": "local", "modTs": "2014-09-16T10:56:42.972+00:00", "model": "N9K-C93128TX", "monPolDn": "uni/fabric/monfab-default", "name": "LEAF_3", "role": "leaf", "serial": "SAL1816QWFA", "status": "", "uid": "0", "vendor": "Cisco Systems, Inc", "version": ""}}, {"fabricNode": {"attributes": {"adSt": "on", "childAction": "", "delayedHeartbeat": "no", "dn": "topology/pod-1/node-102", "fabricSt": "active", "id": "102", "lCOwn": "local", "modTs": "2014-09-16T10:56:42.971+00:00", "model": "N9K-C9396PX", "monPolDn": "uni/fabric/monfab-default", "name": "LEAF_2", "role": "leaf", "serial": "SAL1819S0M8", "status": "", "uid": "0", "vendor": "Cisco Systems, Inc", "version": ""}}]}]}]
11:17:30 DEBUG - method: GET url:
https://10.0.130.71/api/subscriptionRefresh.json?id=72057598349672452 response:
{"imdata": []}
11:17:30 DEBUG - method: GET url:
https://10.0.130.71/api/subscriptionRefresh.json?id=72057598349672546 response:
{"imdata": []}
11:17:30 DEBUG - method: GET url:
https://10.0.130.71/api/subscriptionRefresh.json?id=72057598349672556 response:
{"imdata": []}
11:17:30 DEBUG - method: GET url:
https://10.0.130.71/api/subscriptionRefresh.json?id=72057598349672547 response:
{"imdata": []}
11:17:30 DEBUG - method: GET url:
https://10.0.130.71/api/subscriptionRefresh.json?id=72057598349672545 response:
{"imdata": []}

```

```

11:17:30 DEBUG - method: GET url:
https://10.0.130.71/api/subscriptionRefresh.json?id=72057598349672549 response:
{"imdata": []}
11:17:30 DEBUG - method: GET url:
https://10.0.130.71/api/subscriptionRefresh.json?id=72057598349672551 response:
{"imdata": []}
11:17:30 DEBUG - method: GET url:
https://10.0.130.71/api/subscriptionRefresh.json?id=72057598349672552 response:
{"imdata": []}
11:17:30 DEBUG - method: GET url:
https://10.0.130.71/api/subscriptionRefresh.json?id=72057598349672557 response:
{"imdata": []}
11:17:30 DEBUG - method: GET url:
https://10.0.130.71/api/subscriptionRefresh.json?id=72057598349672558 response:
{"imdata": []}
11:17:30 DEBUG - method: GET url:
https://10.0.130.71/api/subscriptionRefresh.json?id=72057598349672559 response:
{"imdata": []}
11:17:30 DEBUG - method: GET url:
https://10.0.130.71/api/subscriptionRefresh.json?id=72057598349672550 response:
{"imdata": []}
11:17:34 DEBUG - method: undefined url:
https://10.0.130.71/api/node/mo/uni/infra/funcprof.json response:
{"totalCount": "1", "imdata": [{"infraFuncP": {"attributes": {"childAction": "", "descr": "", "dn": "uni/infra/funcprof", "lcOwn": "local", "modTs": "2014-09-16T10:45:42.855+00:00", "monPolDn": "uni/fabric/monfab-default", "name": "default", "ownerKey": "", "ownerTag": "", "status": "", "uid": "0"}}}]}
11:17:34 DEBUG - method: GET url:
https://10.0.130.71/api/node/class/fabricHIfPol.json?subscription=yes response:
{"totalCount": "1", "subscriptionId": "72057598349672560", "imdata": [{"fabricHIfPol": {"attributes": {"autoNeg": "on", "childAction": "", "descr": "", "dn": "uni/infra/hintfpol-default", "lcOwn": "local", "linkDebounce": "100", "modTs": "2014-09-16T10:45:42.855+00:00", "name": "default", "ownerKey": "", "ownerTag": "", "speed": "10G", "status": "", "uid": "0"}}}]}
11:17:34 DEBUG - method: GET url:
https://10.0.130.71/api/node/mo/uni/infra.json?query-target=children&target-subtree-class=cdpIfPol&subscription=yes response:
{"totalCount": "1", "subscriptionId": "72057598349672561", "imdata": [{"cdpIfPol": {"attributes": {"adminSt": "disabled", "childAction": "", "descr": "", "dn": "uni/infra/cdpIfP-default", "lcOwn": "local", "modTs": "2014-09-16T10:45:42.855+00:00", "name": "default", "ownerKey": "", "ownerTag": "", "status": "", "uid": "0"}}}]}
11:17:34 DEBUG - method: GET url:
https://10.0.130.71/api/node/mo/uni/infra.json?query-target=children&target-subtree-class=lldpIfPol&subscription=yes response:
{"totalCount": "1", "subscriptionId": "72057598349672562", "imdata": [{"lldpIfPol": {"attributes": {"adminRxSt": "enabled", "adminTxSt": "enabled", "childAction": "", "descr": "", "dn": "uni/infra/lldpIfP-default", "lcOwn": "local", "modTs": "2014-09-16T10:45:42.855+00:00", "name": "default", "ownerKey": "", "ownerTag": "", "status": "", "uid": "0"}}}]}
11:17:34 DEBUG - method: GET url:
https://10.0.130.71/api/node/class/stpIfPol.json?subscription=yes response:
{"totalCount": "1", "subscriptionId": "72057598349672563", "imdata": [{"stpIfPol": {"attributes": {"childAction": "", "ctrl": "", "descr": "", "dn": "uni/infra/ifPol-default", "lcOwn": "local", "modTs": "2014-09-16T10:45:42.855+00:00", "name": "default", "ownerKey": "", "ownerTag": "", "status": "", "uid": "0"}}}]}
11:17:34 DEBUG - method: GET url:
https://10.0.130.71/api/node/class/monInfraPol.json?subscription=yes response:
{"totalCount": "1", "subscriptionId": "72057598349672564", "imdata": [{"monInfraPol": {"attributes": {"childAction": "", "descr": "", "dn": "uni/infra/moninfra-default", "lcOwn": "local", "modTs": "2014-09-16T10:45:42.855+00:00", "monPolDn": "uni/infra/moninfra-default", "name": "default", "ownerKey": "", "ownerTag": "", "status": "", "uid": "0"}}}]}
11:17:34 DEBUG - method: GET url:
https://10.0.130.71/api/node/class/infraAttEntityP.json?subscription=yes response:
{"totalCount": "1", "subscriptionId": "72057598349672565", "imdata": [{"infraAttEntityP": {"attr

```

```

attributes":{"childAction":"","configIssues":"","descr":"","dn":"uni/infra/attentp-default","lcOwn":"local","modTs":"2014-09-16T10:45:42.855+00:00","monPolDn":"uni/fabric/monfab-default","name":"default","ownerKey":"","ownerTag":"","status":"","uid":"0"}}}]
11:17:42 DEBUG - method: GET url:
https://10.0.130.71/api/node/class/infraDomP.json?subscription=yes response:
{"totalCount":"1","subscriptionId":"72057598349672566","imdata":[{"physDomP":{"attributes":{"childAction":"","configIssues":"","dn":"uni/phys-phys","lcOwn":"local","modTs":"2014-09-16T10:45:42.859+00:00","monPolDn":"uni/fabric/monfab-default","name":"phys","ownerKey":"","ownerTag":"","status":"","uid":"0"}}}]}}
11:17:50 DEBUG - method: GET url:
https://10.0.130.71/api/node/mo/uni/infra.json?query-target=subtree&target-subtree-class=fvnsVlanInstP&target-subtree-class=fvnsEncapBlk&query-target=subtree&subscription=yes response: {"totalCount":"0","subscriptionId":"72057598349672567","imdata":[]}
11:18:00 DEBUG - method: GET url:
https://10.0.130.71/api/subscriptionRefresh.json?id=72057598349672452 response: {"imdata":[]}
11:18:00 DEBUG - method: GET url:
https://10.0.130.71/api/subscriptionRefresh.json?id=72057598349672545 response: {"imdata":[]}
11:18:00 DEBUG - method: GET url:
https://10.0.130.71/api/subscriptionRefresh.json?id=72057598349672556 response: {"imdata":[]}
11:18:00 DEBUG - method: GET url:
https://10.0.130.71/api/subscriptionRefresh.json?id=72057598349672547 response: {"imdata":[]}
11:18:00 DEBUG - method: GET url:
https://10.0.130.71/api/subscriptionRefresh.json?id=72057598349672546 response: {"imdata":[]}
11:18:00 DEBUG - method: GET url:
https://10.0.130.71/api/subscriptionRefresh.json?id=72057598349672549 response: {"imdata":[]}
11:18:00 DEBUG - method: GET url:
https://10.0.130.71/api/subscriptionRefresh.json?id=72057598349672557 response: {"imdata":[]}
11:18:00 DEBUG - method: GET url:
https://10.0.130.71/api/subscriptionRefresh.json?id=72057598349672551 response: {"imdata":[]}
11:18:00 DEBUG - method: GET url:
https://10.0.130.71/api/subscriptionRefresh.json?id=72057598349672550 response: {"imdata":[]}
11:18:00 DEBUG - method: GET url:
https://10.0.130.71/api/subscriptionRefresh.json?id=72057598349672552 response: {"imdata":[]}
11:18:00 DEBUG - method: GET url:
https://10.0.130.71/api/subscriptionRefresh.json?id=72057598349672558 response: {"imdata":[]}
11:18:00 DEBUG - method: GET url:
https://10.0.130.71/api/subscriptionRefresh.json?id=72057598349672560 response: {"imdata":[]}
11:18:00 DEBUG - method: GET url:
https://10.0.130.71/api/subscriptionRefresh.json?id=72057598349672561 response: {"imdata":[]}
11:18:00 DEBUG - method: GET url:
https://10.0.130.71/api/subscriptionRefresh.json?id=72057598349672562 response: {"imdata":[]}
11:18:00 DEBUG - method: GET url:
https://10.0.130.71/api/subscriptionRefresh.json?id=72057598349672563 response: {"imdata":[]}
11:18:00 DEBUG - method: GET url:
https://10.0.130.71/api/subscriptionRefresh.json?id=72057598349672564 response: {"imdata":[]}
11:18:00 DEBUG - method: GET url:
https://10.0.130.71/api/subscriptionRefresh.json?id=72057598349672565 response: {"imdata":[]}

```



```

11:18:00 DEBUG - method: GET url:
https://10.0.130.71/api/subscriptionRefresh.json?id=72057598349672566 response:
{"imdata": []}
11:18:00 DEBUG - method: GET url:
https://10.0.130.71/api/subscriptionRefresh.json?id=72057598349672559 response:
{"imdata": []}
11:18:00 DEBUG - method: GET url:
https://10.0.130.71/api/subscriptionRefresh.json?id=72057598349672567 response:
{"imdata": []}
11:18:03 DEBUG - method: GET url: https://10.0.130.71/api/node/mo/info.json response:
{"totalCount": "0", "imdata": [{"topInfo": {"attributes": {"childAction": "", "currentTime": "2014-09-16T11:13:53.343+00:00", "dn": "info", "id": "1", "role": "controller", "status": ""}}]}]}
11:18:06 DEBUG - method: Event Channel Message response:
{"subscriptionId": ["72057598349672567"], "imdata": [{"fvnsVlanInstP": {"attributes": {"allocMode": "static", "childAction": "", "configIssues": "", "descr": "", "dn": "uni/infra/vlanns-[INBAND]-static", "lcOwn": "local", "modTs": "2014-09-16T11:13:55.575+00:00", "monPolDn": "uni/fabric/monfab-default", "name": "INBAND", "ownerKey": "", "ownerTag": "", "rn": "", "status": "created", "uid": "15374"}}]}]}
11:18:06 DEBUG - method: Event Channel Message response:
{"subscriptionId": ["72057598349672567"], "imdata": [{"fvnsEncapBlk": {"attributes": {"childAction": "", "descr": "", "dn": "uni/infra/vlanns-[INBAND]-static/from-[vlan-145]-to-[vlan-145]", "from": "vlan-145", "lcOwn": "local", "modTs": "2014-09-16T11:13:55.575+00:00", "name": "", "rn": "", "status": "created", "to": "vlan-145", "uid": "15374"}}]}]}
11:18:06 DEBUG - method: POST url:
https://10.0.130.71/api/node/mo/uni/infra/vlanns-[INBAND]-static.json
payload{"fvnsVlanInstP":{"attributes":{"dn":"uni/infra/vlanns-[INBAND]-static","name":"INBAND","allocMode":"static","rn":"vlanns-[INBAND]-static","status":"created"},"children":[{"fvnsEncapBlk":{"attributes":{"dn":"uni/infra/vlanns-[INBAND]-static/from-[vlan-145]-to-[vlan-145]","from":"vlan-145","to":"vlan-145","rn":"from-[vlan-145]-to-[vlan-145]","status":"created"},"children":[]}}]}]} response: {"imdata": []}
11:18:06 DEBUG - method: GET url:
https://10.0.130.71/api/node/mo/uni/infra.json?query-target=subtree&target-subtree-class=fvnsVlanInstP&target-subtree-class=fvnsEncapBlk&query-target=subtree&subscription=yes
response:
{"totalCount": "2", "subscriptionId": "72057598349672568", "imdata": [{"fvnsEncapBlk": {"attributes": {"childAction": "", "descr": "", "dn": "uni/infra/vlanns-[INBAND]-static/from-[vlan-145]-to-[vlan-145]", "from": "vlan-145", "lcOwn": "local", "modTs": "2014-09-16T11:13:55.575+00:00", "name": "", "status": "", "to": "vlan-145", "uid": "15374"}}, {"fvnsVlanInstP": {"attributes": {"allocMode": "static", "childAction": "", "configIssues": "", "descr": "", "dn": "uni/infra/vlanns-[INBAND]-static", "lcOwn": "local", "modTs": "2014-09-16T11:13:55.575+00:00", "monPolDn": "uni/fabric/monfab-default", "name": "INBAND", "ownerKey": "", "ownerTag": "", "status": "", "uid": "15374"}}]}]}
11:18:07 DEBUG - method: Event Channel Message response:
{"subscriptionId": ["72057598349672566"], "imdata": [{"physDomP": {"attributes": {"childAction": "", "configIssues": "", "dn": "uni/phys-INBAND", "lcOwn": "local", "modTs": "2014-09-16T11:13:57.172+00:00", "monPolDn": "uni/fabric/monfab-default", "name": "INBAND", "ownerKey": "", "ownerTag": "", "rn": "", "status": "created", "uid": "15374"}}]}]}
11:18:07 DEBUG - method: POST url: https://10.0.130.71/api/node/mo/uni/phys-INBAND.json
payload{"physDomP":{"attributes":{"dn":"uni/phys-INBAND","name":"INBAND","rn":"phys-INBAND","status":"created"},"children":[{"infraRsVlanNs":{"attributes":{"tDn":"uni/infra/vlanns-[INBAND]-static","status":"created"},"children":[]}}]}]} response: {"imdata": []}
11:18:07 DEBUG - method: GET url:
https://10.0.130.71/api/node/class/infraDomP.json?subscription=yes response:
{"totalCount": "2", "subscriptionId": "72057598349672569", "imdata": [{"physDomP": {"attributes": {"childAction": "", "configIssues": "", "dn": "uni/phys-phys", "lcOwn": "local", "modTs": "2014-09-16T10:45:42.859+00:00", "monPolDn": "uni/fabric/monfab-default", "name": "phys", "ownerKey": "", "ownerTag": "", "status": "", "uid": "0"}}, {"physDomP": {"attributes": {"childAction": "", "configIssues": "", "dn": "uni/phys-INBAND", "lcOwn": "local", "modTs": "2014-09-16T11:13:57.172+00:00", "monPolDn": "uni/fabric/monfab-default", "name": "INBAND", "ownerKey": "", "ownerTag": "", "status": "", "uid": "15374"}}]}]}
11:18:09 DEBUG - method: undefined url:
https://10.0.130.71//api/node/mo/uni/phys-INBAND/rsvlanNs.json response:
{"totalCount": "1", "imdata": [{"infraRsVlanNs": {"attributes": {"childAction": "", "dn": "uni/phys-INBAND/rsvlanNs", "forceResolve": "no", "lcOwn": "local", "modTs": "2014-09-16T11:13:57.180+00

```

```

:00", "monPolDn": "uni/fabric/monfab-default", "rType": "mo", "state": "formed", "stateQual": "non
e", "status": "", "tCl": "fvnsVlanInstP", "tDn": "uni/infra/vlanns- [INBAND] -static", "tType": "mo
", "uid": "15374"}]]}}
11:18:09 DEBUG - method: undefined url:
https://10.0.130.71/api/node/mo/uni/infra/vlanns- [INBAND] -static.json?rsp-subtree=full&rs
p-subtree-class=fvnsEncapBlk response:
{"totalCount": "1", "imdata": [{"fvnsVlanInstP": {"attributes": {"allocMode": "static", "childAct
ion": "", "configIssues": "", "descr": "", "dn": "uni/infra/vlanns- [INBAND] -static", "lcOwn": "loca
l", "modTs": "2014-09-16T11:13:55.575+00:00", "monPolDn": "uni/fabric/monfab-default", "name": "
INBAND", "ownerKey": "", "ownerTag": "", "status": "", "uid": "15374"}, "children": [{"fvnsEncapBlk"
: {"attributes": {"childAction": "", "descr": "", "from": "vlan-145", "lcOwn": "local", "modTs": "201
4-09-16T11:13:55.575+00:00", "name": "", "rn": "from- [vlan-145] -to- [vlan-145]", "status": "", "to
": "vlan-145", "uid": "15374"}]]}}]}
11:18:09 DEBUG - method: undefined url:
https://10.0.130.71/api/node/mo/uni/infra/vlanns- [INBAND] -static.json?rsp-subtree=full&rs
p-subtree-class=fvnsEncapBlk response:
{"totalCount": "1", "imdata": [{"fvnsVlanInstP": {"attributes": {"allocMode": "static", "childAct
ion": "", "configIssues": "", "descr": "", "dn": "uni/infra/vlanns- [INBAND] -static", "lcOwn": "loca
l", "modTs": "2014-09-16T11:13:55.575+00:00", "monPolDn": "uni/fabric/monfab-default", "name": "
INBAND", "ownerKey": "", "ownerTag": "", "status": "", "uid": "15374"}, "children": [{"fvnsEncapBlk"
: {"attributes": {"childAction": "", "descr": "", "from": "vlan-145", "lcOwn": "local", "modTs": "201
4-09-16T11:13:55.575+00:00", "name": "", "rn": "from- [vlan-145] -to- [vlan-145]", "status": "", "to
": "vlan-145", "uid": "15374"}]]}}]}
11:18:10 DEBUG - method: Event Channel Message response:
{"subscriptionId": ["72057598349672565"], "imdata": [{"infraAttEntityP": {"attributes": {"child
Action": "", "configIssues": "", "descr": "", "dn": "uni/infra/attentp-INBAND", "lcOwn": "local", "m
odTs": "2014-09-16T11:14:00.123+00:00", "monPolDn": "uni/fabric/monfab-default", "name": "INBAN
D", "ownerKey": "", "ownerTag": "", "rn": "", "status": "created", "uid": "15374"}]]}}
11:18:10 DEBUG - method: POST url:
https://10.0.130.71/api/node/mo/uni/infra/attentp-INBAND.json
payload{"infraAttEntityP": {"attributes": {"dn": "uni/infra/attentp-INBAND", "name": "INBAND", "
rn": "attentp-INBAND", "status": "created"}, "children": [{"infraRsDomP": {"attributes": {"tDn": "
uni/phys-INBAND", "status": "created"}, "children": []}}]} response: {"imdata": []}
11:18:10 DEBUG - method: GET url:
https://10.0.130.71/api/node/class/infraAttEntityP.json?subscription=yes response:
{"totalCount": "2", "subscriptionId": "72057598349672570", "imdata": [{"infraAttEntityP": {"attr
ibutes": {"childAction": "", "configIssues": "", "descr": "", "dn": "uni/infra/attentp-default", "lc
Own": "local", "modTs": "2014-09-16T10:45:42.855+00:00", "monPolDn": "uni/fabric/monfab-default", "name": "default", "ownerKey": "", "ownerTag": "", "status": "", "uid": "0"}}, {"infraAttEntityP
": {"attributes": {"childAction": "", "configIssues": "", "descr": "", "dn": "uni/infra/attentp-INB
AND", "lcOwn": "local", "modTs": "2014-09-16T11:14:00.123+00:00", "monPolDn": "uni/fabric/monfab
-default", "name": "INBAND", "ownerKey": "", "ownerTag": "", "status": "", "uid": "15374"}]]}}
11:18:12 DEBUG - method: Event Channel Message response:
{"subscriptionId": ["72057598349672553", "72057598349672556", "72057598349672548"], "imdata": [
{"infraAccPortGrp": {"attributes": {"childAction": "", "descr": "", "dn": "uni/infra/funcprof/acc
portgrp-INBAND", "lcOwn": "local", "modTs": "2014-09-16T11:14:01.587+00:00", "monPolDn": "uni/fa
bric/monfab-default", "name": "INBAND", "ownerKey": "", "ownerTag": "", "rn": "", "status": "created
", "uid": "15374"}]]}}
11:18:12 DEBUG - method: POST url:
https://10.0.130.71/api/node/mo/uni/infra/funcprof/accportgrp-INBAND.json
payload{"infraAccPortGrp": {"attributes": {"dn": "uni/infra/funcprof/accportgrp-INBAND", "name
": "INBAND", "rn": "accportgrp-INBAND", "status": "created"}, "children": [{"infraRsAttEntP": {"at
tributes": {"tDn": "uni/infra/attentp-INBAND", "status": "created,modified"}, "children": []}}]}
} response: {"imdata": []}
11:18:12 DEBUG - method: GET url:
https://10.0.130.71/api/node/class/infraAccPortGrp.json?subscription=yes response:
{"totalCount": "1", "subscriptionId": "72057598349672571", "imdata": [{"infraAccPortGrp": {"attr
ibutes": {"childAction": "", "descr": "", "dn": "uni/infra/funcprof/accportgrp-INBAND", "lcOwn": "
local", "modTs": "2014-09-16T11:14:01.587+00:00", "monPolDn": "uni/fabric/monfab-default", "nam
e": "INBAND", "ownerKey": "", "ownerTag": "", "status": "", "uid": "15374"}]]}}
11:18:16 DEBUG - method: Event Channel Message response:
{"subscriptionId": ["72057598349672546"], "imdata": [{"infraAccPortP": {"attributes": {"childAc
tion": "", "descr": "GUI Interface Selector Generated PortP Profile:
APIC_Connected_Leaf", "dn": "uni/infra/accportprof-APIC_Connected_Leaf_ifselector", "lcOwn": "

```

```
local", "modTs": "2014-09-16T11:14:06.060+00:00", "monPolDn": "uni/fabric/monfab-default", "name": "APIC_Connected_Leaf_ifselector", "ownerKey": "", "ownerTag": "", "rn": "", "status": "created", "uid": "15374"}]]}]
```

11:18:16 DEBUG - method: Event Channel Message response:

```
{ "subscriptionId": ["72057598349672546"], "imdata": [{"infraHPortS": {"attributes": {"childAction": "", "descr": "", "dn": "uni/infra/accportprof-APIC_Connected_Leaf_ifselector/hports-APIC_Connected_Port-typ-range", "lcOwn": "local", "modTs": "2014-09-16T11:14:06.060+00:00", "monPolDn": "uni/fabric/monfab-default", "name": "APIC_Connected_Port", "ownerKey": "", "ownerTag": "", "rn": "", "status": "created", "type": "range", "uid": "15374"}]]}]
```

11:18:16 DEBUG - method: Event Channel Message response:

```
{ "subscriptionId": ["72057598349672546"], "imdata": [{"infraRsAccBaseGrp": {"attributes": {"childAction": "", "dn": "uni/infra/accportprof-APIC_Connected_Leaf_ifselector/hports-APIC_Connected_Port-typ-range/rsaccBaseGrp", "fexId": "101", "forceResolve": "no", "lcOwn": "local", "modTs": "2014-09-16T11:14:06.060+00:00", "monPolDn": "uni/fabric/monfab-default", "rType": "mo", "rn": "", "state": "formed", "stateQual": "none", "status": "created", "tCl": "infraAccPortGrp", "tDn": "uni/infra/funcprof/accportgrp-INBAND", "tType": "mo", "uid": "15374"}]]}]
```

11:18:16 DEBUG - method: Event Channel Message response:

```
{ "subscriptionId": ["72057598349672546"], "imdata": [{"infraPortBlk": {"attributes": {"childAction": "", "dn": "uni/infra/accportprof-APIC_Connected_Leaf_ifselector/hports-APIC_Connected_Port-typ-range/portblk-block1", "fromCard": "1", "fromPort": "48", "lcOwn": "local", "modTs": "2014-09-16T11:14:06.060+00:00", "monPolDn": "uni/fabric/monfab-default", "name": "block1", "rn": "", "status": "created", "toCard": "1", "toPort": "48", "uid": "15374"}]]}]
```

11:18:16 DEBUG - method: Event Channel Message response:

```
{ "subscriptionId": ["72057598349672545"], "imdata": [{"infraNodeP": {"attributes": {"childAction": "", "descr": "GUI Interface Selector Generated Profile: APIC_Connected_Leaf", "dn": "uni/infra/nprof-APIC_Connected_Leaf", "lcOwn": "local", "modTs": "2014-09-16T11:14:06.060+00:00", "monPolDn": "uni/fabric/monfab-default", "name": "APIC_Connected_Leaf", "ownerKey": "", "ownerTag": "", "rn": "", "status": "created", "uid": "15374"}]]}]
```

11:18:16 DEBUG - method: Event Channel Message response:

```
{ "subscriptionId": ["72057598349672545"], "imdata": [{"infraLeafS": {"attributes": {"childAction": "", "descr": "", "dn": "uni/infra/nprof-APIC_Connected_Leaf/leaves-APIC_Connected_Leaf_selector_101-typ-range", "lcOwn": "local", "modTs": "2014-09-16T11:14:06.060+00:00", "monPolDn": "uni/fabric/monfab-default", "name": "APIC_Connected_Leaf_selector_101", "ownerKey": "", "ownerTag": "", "rn": "", "status": "created", "type": "range", "uid": "15374"}]]}]
```

11:18:16 DEBUG - method: Event Channel Message response:

```
{ "subscriptionId": ["72057598349672545"], "imdata": [{"infraNodeBlk": {"attributes": {"childAction": "", "dn": "uni/infra/nprof-APIC_Connected_Leaf/leaves-APIC_Connected_Leaf_selector_101-typ-range/nodeblk-single0", "from_": "101", "lcOwn": "local", "modTs": "2014-09-16T11:14:06.060+00:00", "name": "single0", "rn": "", "status": "created", "to_": "101", "uid": "15374"}]]}]
```

11:18:16 DEBUG - method: Event Channel Message response:

```
{ "subscriptionId": ["72057598349672545"], "imdata": [{"infraRsAccPortP": {"attributes": {"childAction": "", "dn": "uni/infra/nprof-APIC_Connected_Leaf/rsaccPortP-[uni/infra/accportprof-APIC_Connected_Leaf_ifselector]", "forceResolve": "no", "lcOwn": "local", "modTs": "2014-09-16T11:14:06.060+00:00", "monPolDn": "uni/fabric/monfab-default", "rType": "mo", "rn": "", "state": "formed", "stateQual": "none", "status": "created", "tCl": "infraAccPortP", "tDn": "uni/infra/accportprof-APIC_Connected_Leaf_ifselector", "tType": "mo", "uid": "15374"}]]}]
```

11:18:16 DEBUG - method: Event Channel Message response:

```
{ "subscriptionId": ["72057598349672546"], "imdata": [{"infraRtAccPortP": {"attributes": {"childAction": "", "dn": "uni/infra/accportprof-APIC_Connected_Leaf_ifselector/rtaccPortP-[uni/infra/nprof-APIC_Connected_Leaf]", "lcOwn": "local", "modTs": "2014-09-16T11:14:06.060+00:00", "rn": "", "status": "created", "tCl": "infraNodeP", "tDn": "uni/infra/nprof-APIC_Connected_Leaf"}]]}]
```

11:18:16 DEBUG - method: POST url: <https://10.0.130.71/api/node/mo/uni/infra.json>

```
payload{"infraInfra": {"attributes": {"dn": "uni/infra", "status": "created,modified"}, "children": [{"infraAccPortP": {"attributes": {"dn": "uni/infra/accportprof-APIC_Connected_Leaf_ifselector", "name": "APIC_Connected_Leaf_ifselector", "descr": "GUI Interface Selector Generated PortP Profile: APIC_Connected_Leaf", "status": "created,modified"}, "children": [{"infraHPortS": {"attributes": {"dn": "uni/infra/accportprof-APIC_Connected_Leaf_ifselector/hports-APIC_Connected_Port-typ-range", "name": "APIC_Connected_Port", "type": "range", "status": "created,modified"}, "children": [{"infraPortBlk": {"attributes": {"dn": "uni/infra/accportprof-APIC_Connected_Leaf_ifselector/hports-APIC_Connected_Port-typ-range/portblk-block1", "fromPort": "48", "toPort": "48", "name": "block1", "status": "created,modified"}, "children": [{"infraRsAccBaseGrp": {"attributes": {"tDn": "uni/infra/funcprof/accportgrp-INBAND", "status": "created,modified"}, "children": []}}]]}}]}], {"infraRsAccBaseGrp": {"attributes": {"tDn": "uni/infra/funcprof/accportgrp-INBAND", "status": "created,modified"}, "children": []}}]}], {"infraNodeP": {"attributes": {"dn": "uni/infra/nprof-APIC_Connected_Leaf", "status": "created,modified"}, "children": [{"infraLeafS": {"attributes": {"dn": "uni/infra/nprof-APIC_Connected_Leaf/leaves-APIC_Connected_Leaf_selector_101-typ-range", "name": "APIC_Connected_Leaf_selector_101", "type": "range", "status": "created,modified"}, "children": [{"infraNodeBlk": {"attributes": {"dn": "uni/infra/nprof-APIC_Connected_Leaf/leaves-APIC_Connected_Leaf_selector_101-typ-range/nodeblk-single0", "name": "single0", "type": "single", "status": "created,modified"}, "children": [{"infraRsAccPortP": {"attributes": {"dn": "uni/infra/nprof-APIC_Connected_Leaf/rsaccPortP-[uni/infra/accportprof-APIC_Connected_Leaf_ifselector]", "name": "APIC_Connected_Leaf_ifselector", "type": "range", "status": "created,modified"}, "children": [{"infraRtAccPortP": {"attributes": {"dn": "uni/infra/accportprof-APIC_Connected_Leaf_ifselector/rtaccPortP-[uni/infra/nprof-APIC_Connected_Leaf]", "name": "APIC_Connected_Leaf_ifselector", "type": "range", "status": "created,modified"}, "children": [{"infraRsAccBaseGrp": {"attributes": {"tDn": "uni/infra/funcprof/accportgrp-INBAND", "status": "created,modified"}, "children": []}}]]}}]]}}]}], {"infraNodeP": {"attributes": {"dn": "uni/infra/nprof-APIC_Connected_Leaf", "status": "created,modified"}, "children": [{"infraLeafS": {"attributes": {"dn": "uni/infra/nprof-APIC_Connected_Leaf/leaves-APIC_Connected_Leaf_selector_101-typ-range", "name": "APIC_Connected_Leaf_selector_101", "type": "range", "status": "created,modified"}, "children": [{"infraNodeBlk": {"attributes": {"dn": "uni/infra/nprof-APIC_Connected_Leaf/leaves-APIC_Connected_Leaf_selector_101-typ-range/nodeblk-single0", "name": "single0", "type": "single", "status": "created,modified"}, "children": [{"infraRsAccPortP": {"attributes": {"dn": "uni/infra/nprof-APIC_Connected_Leaf/rsaccPortP-[uni/infra/accportprof-APIC_Connected_Leaf_ifselector]", "name": "APIC_Connected_Leaf_ifselector", "type": "range", "status": "created,modified"}, "children": [{"infraRtAccPortP": {"attributes": {"dn": "uni/infra/accportprof-APIC_Connected_Leaf_ifselector/rtaccPortP-[uni/infra/nprof-APIC_Connected_Leaf]", "name": "APIC_Connected_Leaf_ifselector", "type": "range", "status": "created,modified"}, "children": [{"infraRsAccBaseGrp": {"attributes": {"tDn": "uni/infra/funcprof/accportgrp-INBAND", "status": "created,modified"}, "children": []}}]]}}]]}}]}], {"infraNodeP": {"attributes": {"dn": "uni/infra/nprof-APIC_Connected_Leaf", "status": "created,modified"}, "children": [{"infraLeafS": {"attributes": {"dn": "uni/infra/nprof-APIC_Connected_Leaf/leaves-APIC_Connected_Leaf_selector_101-typ-range", "name": "APIC_Connected_Leaf_selector_101", "type": "range", "status": "created,modified"}, "children": [{"infraNodeBlk": {"attributes": {"dn": "uni/infra/nprof-APIC_Connected_Leaf/leaves-APIC_Connected_Leaf_selector_101-typ-range/nodeblk-single0", "name": "single0", "type": "single", "status": "created,modified"}, "children": [{"infraRsAccPortP": {"attributes": {"dn": "uni/infra/nprof-APIC_Connected_Leaf/rsaccPortP-[uni/infra/accportprof-APIC_Connected_Leaf_ifselector]", "name": "APIC_Connected_Leaf_ifselector", "type": "range", "status": "created,modified"}, "children": [{"infraRtAccPortP": {"attributes": {"dn": "uni/infra/accportprof-APIC_Connected_Leaf_ifselector/rtaccPortP-[uni/infra/nprof-APIC_Connected_Leaf]", "name": "APIC_Connected_Leaf_ifselector", "type": "range", "status": "created,modified"}, "children": [{"infraRsAccBaseGrp": {"attributes": {"tDn": "uni/infra/funcprof/accportgrp-INBAND", "status": "created,modified"}, "children": []}}]]}}]]}}]}], {"infraNodeP": {"attributes": {"dn": "uni/infra/nprof-APIC_Connected_Leaf", "status": "created,modified"}, "children": [{"infraLeafS": {"attributes": {"dn": "uni/infra/nprof-APIC_Connected_Leaf/leaves-APIC_Connected_Leaf_selector_101-typ-range", "name": "APIC_Connected_Leaf_selector_101", "type": "range", "status": "created,modified"}, "children": [{"infraNodeBlk": {"attributes": {"dn": "uni/infra/nprof-APIC_Connected_Leaf/leaves-APIC_Connected_Leaf_selector_101-typ-range/nodeblk-single0", "name": "single0", "type": "single", "status": "created,modified"}, "children": [{"infraRsAccPortP": {"attributes": {"dn": "uni/infra/nprof-APIC_Connected_Leaf/rsaccPortP-[uni/infra/accportprof-APIC_Connected_Leaf_ifselector]", "name": "APIC_Connected_Leaf_ifselector", "type": "range", "status": "created,modified"}, "children": [{"infraRtAccPortP": {"attributes": {"dn": "uni/infra/accportprof-APIC_Connected_Leaf_ifselector/rtaccPortP-[uni/infra/nprof-APIC_Connected_Leaf]", "name": "APIC_Connected_Leaf_ifselector", "type": "range", "status": "created,modified"}, "children": [{"infraRsAccBaseGrp": {"attributes": {"tDn": "uni/infra/funcprof/accportgrp-INBAND", "status": "created,modified"}, "children": []}}]]}}]]}}]}]]}]}
```

```

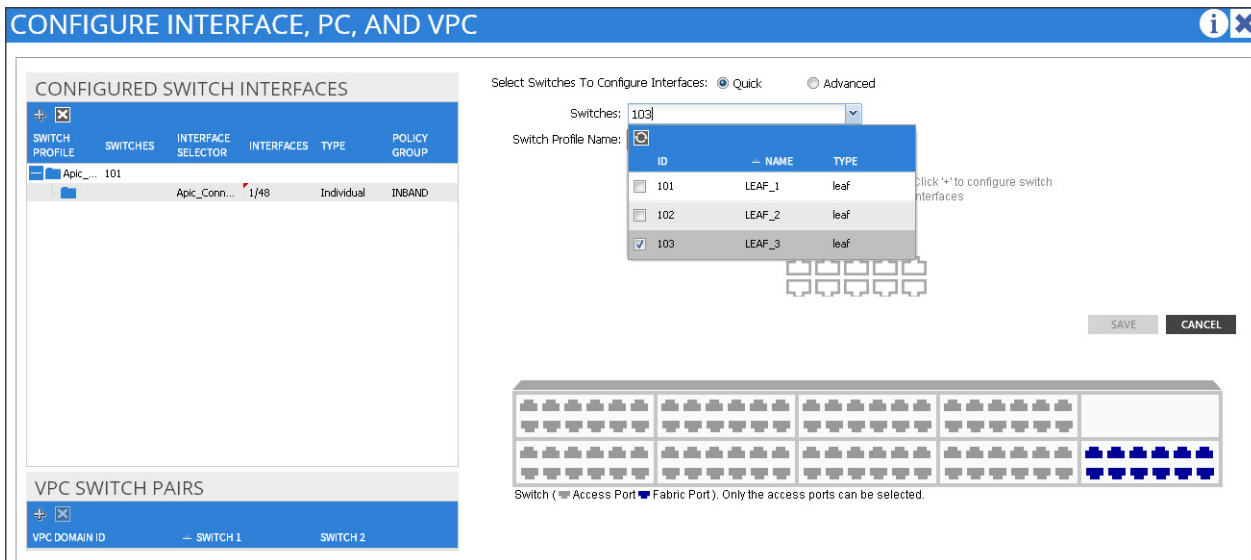
nected_Leaf", "name": "APIC_Connected_Leaf", "descr": "GUI Interface Selector Generated
Profile:
APIC_Connected_Leaf", "status": "created,modified", "children": [{"infraLeafS": {"attributes":
{"dn": "uni/infra/nprof-APIC_Connected_Leaf/leaves-APIC_Connected_Leaf_selector_101-typ-ran
ge", "name": "APIC_Connected_Leaf_selector_101", "type": "range", "status": "created"}, "children
": [{"infraNodeBlk": {"attributes": {"dn": "uni/infra/nprof-APIC_Connected_Leaf/leaves-APIC_Co
nnected_Leaf_selector_101-typ-range/nodeblk-single0", "status": "created", "from_": "101", "to_
": "101", "name": "single0", "rn": "nodeblk-single0"}, "children": []}}]}, {"infraRsAccPortP": {"a
ttributes": {"tDn": "uni/infra/accportprof-APIC_Connected_Leaf_ifselector", "status": "created
,modified"}, "children": []}}]}]} response: {"imdata": []}

```

## Configuring a Switch for CIMC

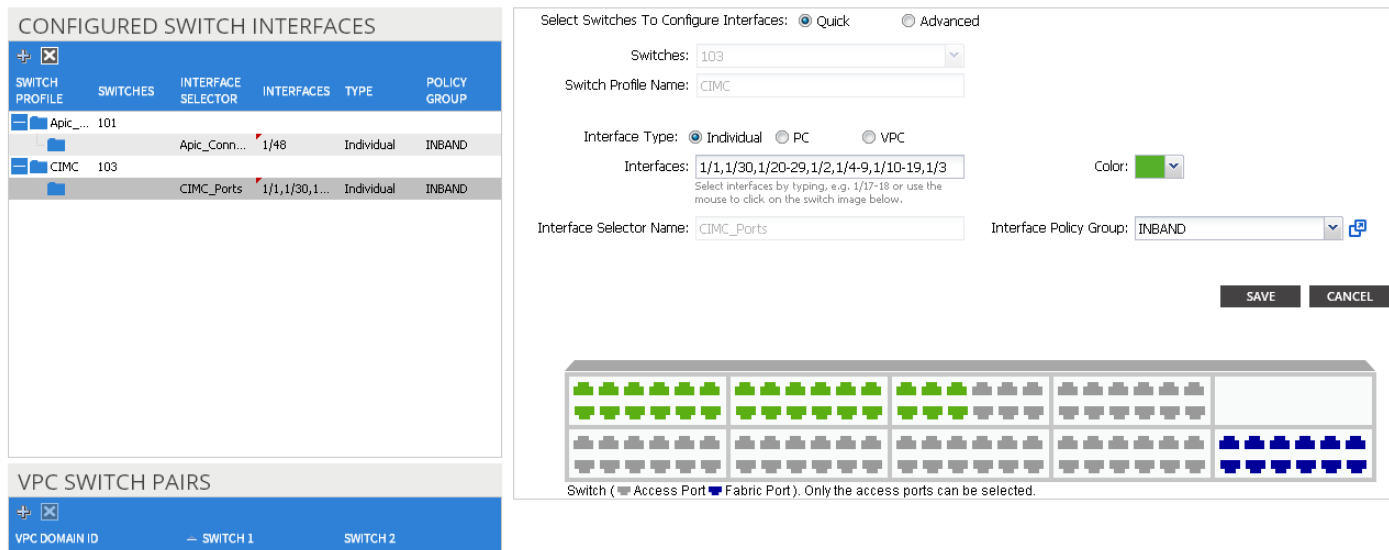
1. Expand the Configured Switch Interfaces area to configure the CIMC for the server ports, and perform the following actions:
  1. In the Switches drop-down list, check the check boxes for the switches that you want to connect to the Servers for CIMC (LEAF\_3).
  2. In the Switch Profile Name field, enter a name for the profile (CIMC).

Figure 29 Configuring CIMC Node



3. Click the + icon to configure the ports.
4. In the **Interface Type** area, verify the Individual radio button is selected.
5. In the **Interfaces** field, enter the ports to which the servers are connected. You can select multiple ports by clicking on the individual switch ports shown in the GUI.
6. In the **Interface Selector Name** field, enter the name of the port profile (CIMC\_Ports).
7. In the **Interface Policy Group** field, from the drop-down list, choose the policy group that was created earlier (INBAND). Click **Save**, and click **Save again**.
8. In the **Configure Interface, PC, and VPC** dialog box, click **Submit**.

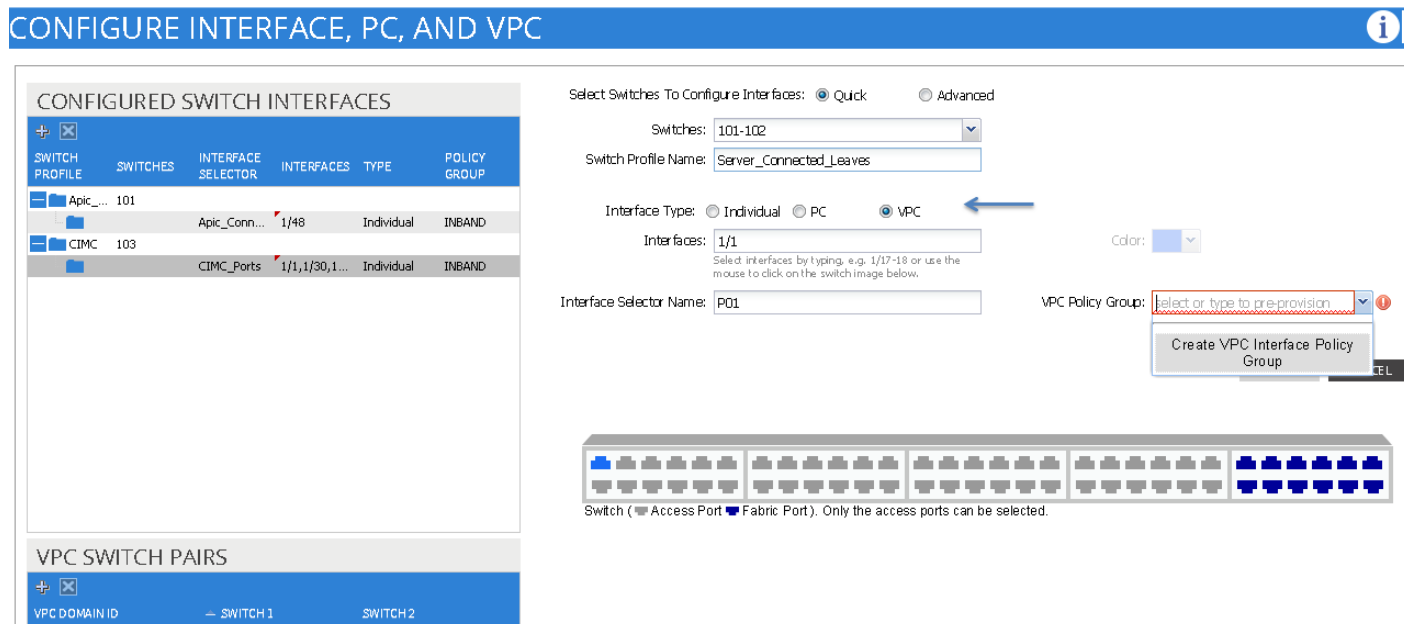
Figure 30 Configuring CIMC Interface



## Configuring vPC Ports for Servers

1. Expand the Configured Switch Interfaces area to configure the VLANs for the server ports, and perform the following actions:
  1. In the Switches drop-down list, check the check boxes for the switches that you want to connect to the Servers. (LEAF\_1 & LEAF\_2).
  2. In the Switch Profile Name field, enter a name for the profile (Server\_Connected\_Leaves).

Figure 31 Configuring vPC Ports



3. Click the + icon to configure the ports.

4. In the Interface Type area, verify the VPC radio button is selected.
5. In the Interfaces field, enter the ports to which the servers are connected.
6. In the Interface Selector Name field, enter the name of the port profile (P01).
7. In the VPC Policy Group field, from the drop-down list, choose Create VPC Interface Policy Group.
8. In Create VPC Interface Policy Group window, enter the name “VPC1”.



**Note** Create separate VPC interface policy group for each VPC link.

*Figure 32 Creating VPC Interface Policy Group*

9. In the **LLDP Policy** field, from the drop-down list, choose “**default**”.
10. In the **LACP policy** field, from the drop-down list choose “**Create LACP Policy**”.
11. In **Create LACP Policy** window, enter the name “LACP\_Active”. In the mode select the “**Active**” radio button and click submit.

**Figure 33**      *Creating LACP Policy*

12. In the **Create VPC Interface Policy Group** window click **submit**.
13. In the **Configure Interface, PC, and VPC** dialog box, click **Save** and click **save again**.
14. In the **Configure Interface, PC, and VPC** dialog box, click **Submit**.
15. Repeat step 3 to 13 to create VPC port for all the servers. Once all the server vPC port is configured, the configured switch interface window should look like fig below.

**Figure 34**      *Configure vPC Interfaces*

SWITCH PROFILE	SWITCHES	INTERFACE SELECTOR	INTERFACES	TYPE	POLICY GROUP
Serv...	101,102	P03	1/3	VPC	VPC3
		P04	1/4	VPC	VPC4
		P01	1/1	VPC	VPC1
		P02	1/2	VPC	VPC2
		P05	1/5	VPC	VPC5
		P08	1/8	VPC	VPC8
		P07	1/7	VPC	VPC7
		P06	1/6	VPC	VPC6
		P13	1/13	VPC	VPC13
		P14	1/14	VPC	VPC14
		P15	1/15	VPC	VPC15
		P10	1/10	VPC	VPC10
		P16	1/16	VPC	VPC16
		P17	1/17	VPC	VPC17
		P09	1/9	VPC	VPC9

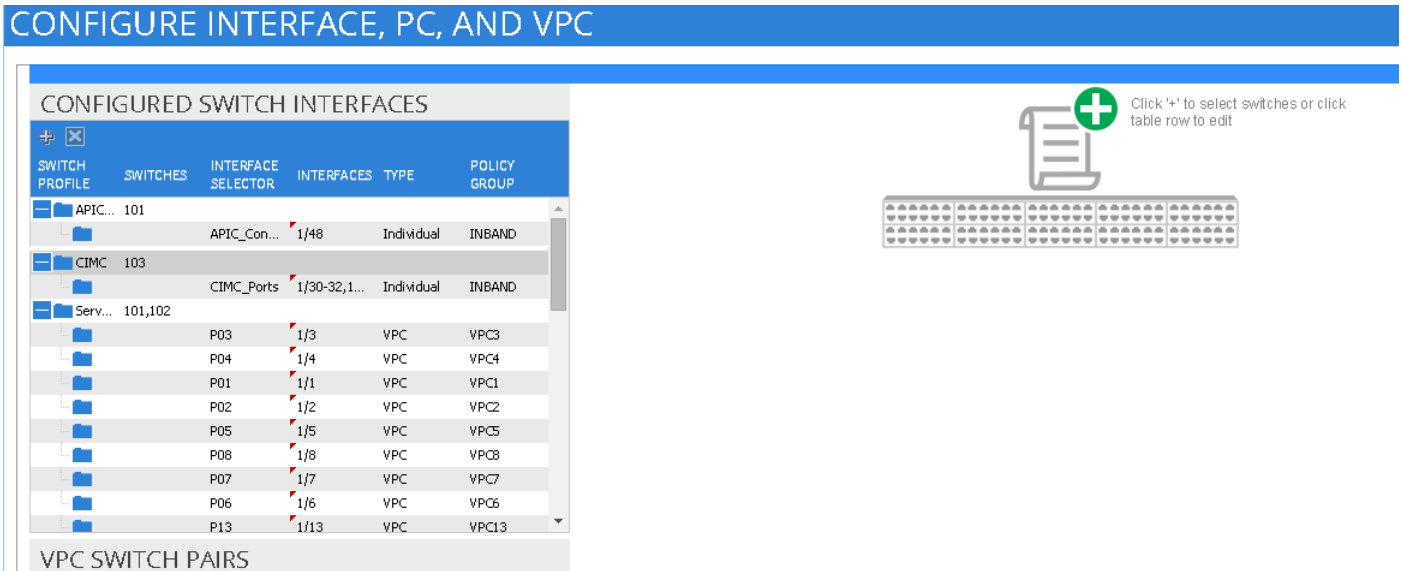
VPC SWITCH PAIRS

VPC DOMAIN ID      SWITCH 1      SWITCH 2

## Configuring vPC Leaf Pairing

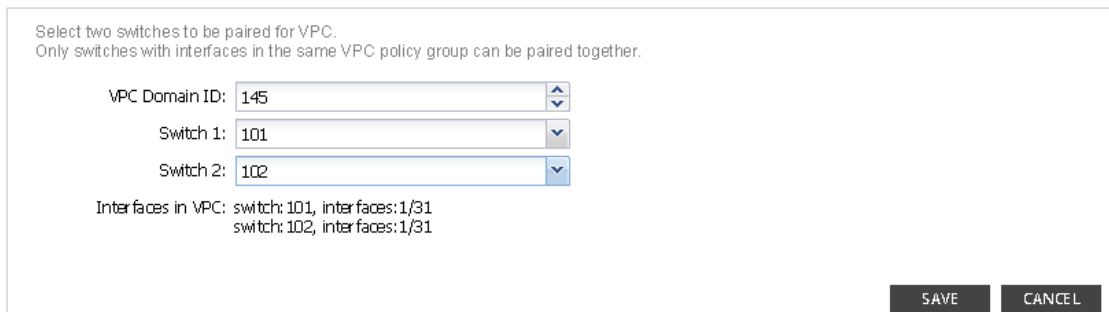
1. In the **Configure Interface, PC, and VPC** dialog box, click on the “+” on VPC DOMAIN ID.

Figure 35 Creating vPC Domain



2. In the VPC Domain ID field, enter “145”.
3. In the “Switch A” drop down box, select node “101”.
4. In the “Switch B” drop down box, select node “102” and click save and submit.

Figure 36 Creating vPC Peer



## Creating Attachable Entity Profile

1. On the menu bar, choose FABRIC > Access Policies. In the Work pane, expand Global Policies.
2. Select Attachable Access Entity Profile and right click on it and select “Create Attachable Access Entity Profile”.



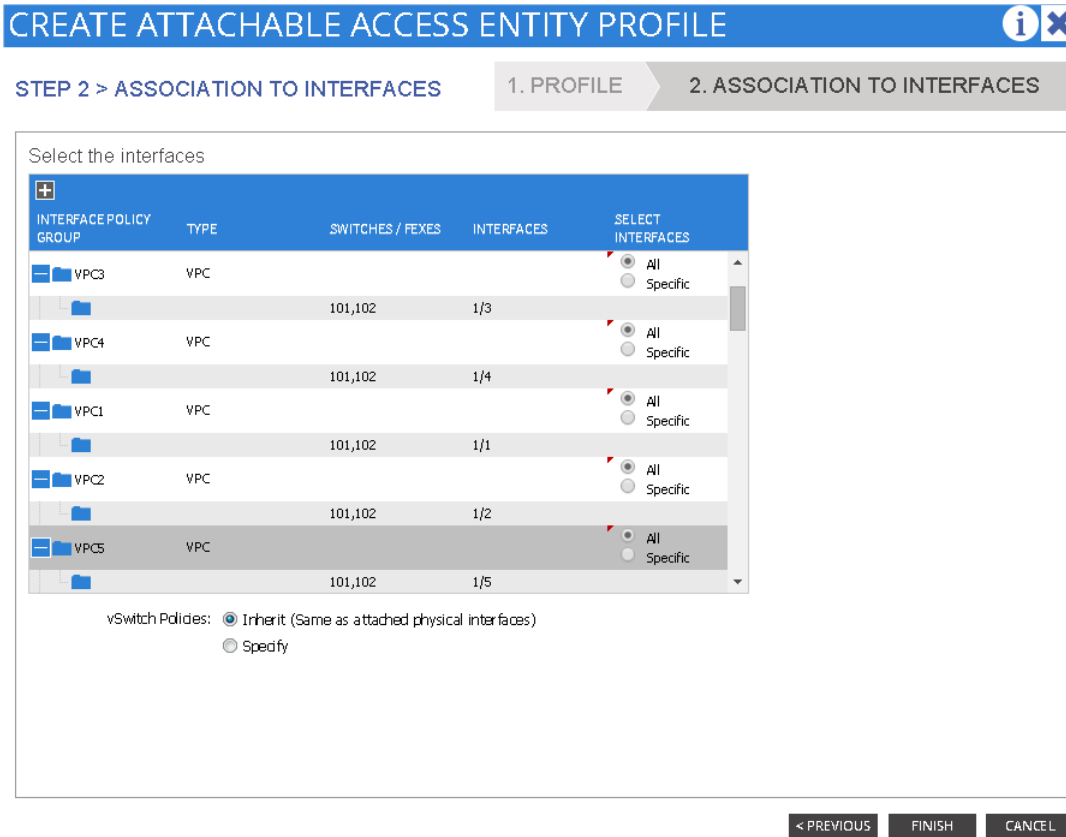
Figure 37 Fabric Window

3. Create **Attachable Access Entity Profile** window opens, in the name field enter **VPC\_AEP** and click **Next**.

Figure 38 Creating Attachable Access Entity Profile for vPC

4. In each of the “**Interface Policies**” that was created for the vPC, select the radio button “**All**” and click on “**Finish**”

Figure 39 Associating the Interface to AEP



## Creating Tenants, Private Network, and Bridge Domains

### Tenants Overview

- A tenant contains policies that enable qualified users domain-based access control. Qualified users can access privileges such as tenant administration and networking administration.
- A user requires read/write privileges for accessing and configuring policies in a domain. A tenant user can have specific privileges into one or more domains.
- In a multi-tenancy environment, a tenant provides group user access privileges so that resources are isolated from one another (such as for endpoint groups and networking). These privileges also enable different users to manage different tenants.

### Creating a Tenant, Private Network, and Bridge Domain Using the GUI

#### Procedure

1. On the menu bar, choose **TENANTS**, and perform the following actions:
  - Click **Add Tenant**.
  - The **Create Tenant** dialog box opens.
  - In the Name field, add the tenant name (Big\_Data), and click Next.

**Note**

Create a security domain so that it allows only users in that security domain to have access.

2. Click the + sign next to Security Domains to open the Create Security Domain dialog box, and perform the following actions:
  1. In the **Name** field, specify the security domain name. (Big\_Data)
  2. Click **Submit**. In the **Create Tenant** dialog box, check the check box for the security domain that you created, and click **Next**.

*Figure 40 Creating Tenant*

## STEP 1 &gt; TENANT

## 1. TENANT

Tenant Identity

Specify tenant details

Name:

Description:

Tags:  ▼  
enter tags separated by comma

Monitoring Policy:  ▼

Security Domains: ↻ +

Select	Name	Description
<input checked="" type="checkbox"/>	Big_Data	
<input type="checkbox"/>	all	
<input type="checkbox"/>	mgmt	

3. In the **Network** window, perform the following actions:
  - Click the + sign to add the network.
  - In the **Create New Network** area, specify the private tenant network name (PVN\_1) and click **Next**.

You are prompted to specify a bridge domain in the network.

4. Specify the bridge domain in the **Name** field (BD\_1), click **OK**. Click **Next**, and perform the following actions:

Figure 41 Creating Tenant Network

# CREATE TENANT


STEP 2 > NETWORK

1. TENANT
2. NETWORK

## TENANT BIG\_DATA

CREATE NEW NETWORK

NETWORK > BRIDGE DOMA



Specify Tenant Network

Name:

Policy Enforcement:  Enforced  
 Unenforced

Description:

BGP Timers:  ▼


OSPF Timers:  ▼

Monitoring Policy:  ▼

DNS Labels:   
enter names separated by comma

Create A Bridge Domain:

**Figure 42**      **Creating Bridge Domain**

**TENANT BIG\_DATA**      NETWORK > BRIDGE DOMAIN  
 CREATE NEW NETWORK      

Specify Bridge Domain for the Network



Name:

Description:



Forwarding:

IGMP Snoop Policy:

Config BD MAC Address:

Subnets:  

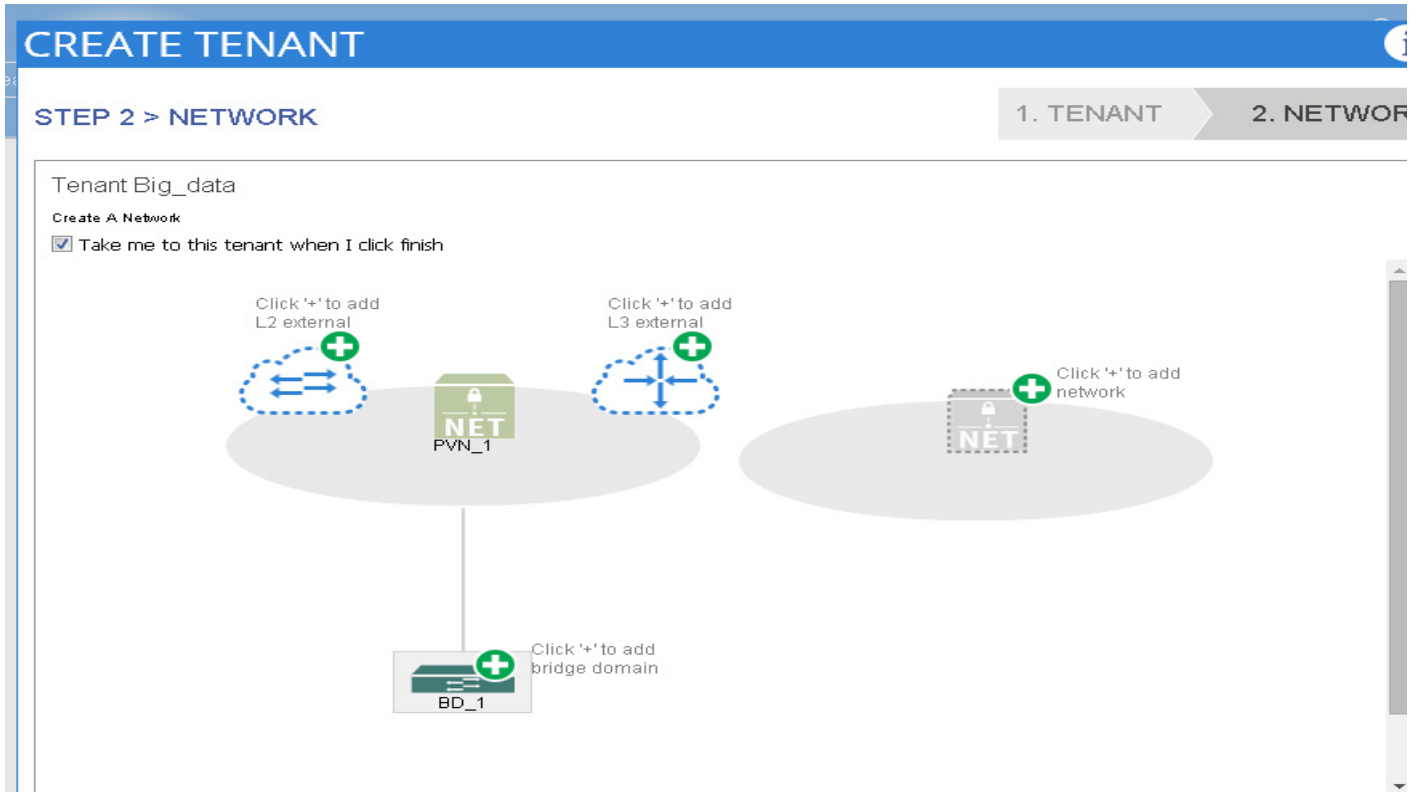
Gateway Address	Scope	Subnet Control

DHCP Labels:  

Name	Scope	DHCP Option Policy

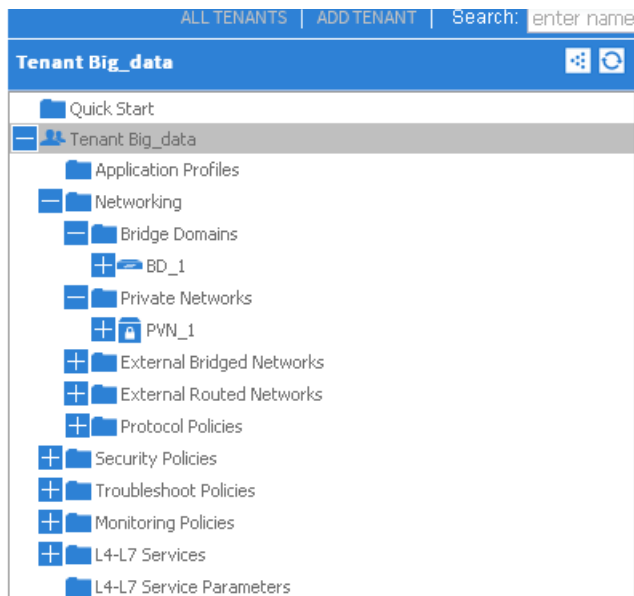
- a. Confirm that the private network (PVN\_1) is created and is associated with the bridge domain (BD\_1).
- b. In the **Application Profile** window, click **Finish**.

Figure 43 Confirming the Association



- To validate that the tenant has a private network and bridge domain, in the submenu bar under the **Tenants** tab, click the new tenant name that you created. In the **Navigation** pane, expand the tenant name. **Under Bridge Domains**, the new bridge domain is displayed. **Under Private Networks**, the new network is displayed.

Figure 44 Validating the Bridge Domain and Private Network



6. Select the bridge domain created earlier (BD\_1) and check the **ARP Flooding** checkbox and click **Submit**.

Figure 45 ARP Flooding

The screenshot displays the Cisco ACI GUI for configuring a Bridge Domain (BD\_1). The navigation pane on the left shows the hierarchy: Tenant Big\_Data > Bridge Domains > BD\_1. The main panel shows the 'Bridge Domain - BD\_1' configuration page with a 'POLICY' tab. The 'PROPERTIES' section includes the following settings:

- Name: **BD\_1**
- Description: optional
- Unknown Unicast Traffic Class ID: **32770**
- Segment: **16187318**
- Multicast Address: **225.0.48.16**
- Network: **PVN\_1**
- Custom MAC Address: **00:22:BD:F8:19:FF**
- L2 Unknown Unicast:  Hardware Proxy
- Unknown Multicast Flooding:  Flood
- ARP Flooding:
- Unicast Routing:
- IGMP Snoop Policy: select or type to pre-
- End Point Retention Policy: select or type to pre-
- Associated L3 Outs: - L3 OUT

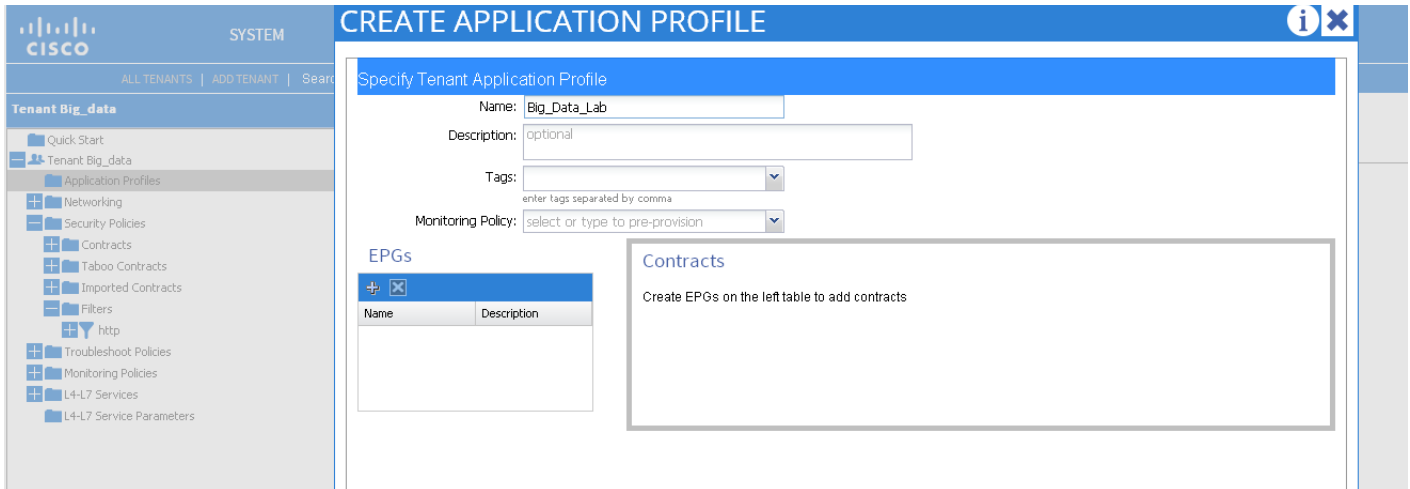
A status bar at the top right of the configuration area shows a green '100' and a 'POLICY' tab. At the bottom right, a message states: 'No items have been found. Select Actions to create a new item.'

## Creating an Application Profile Using the GUI

### Procedure

1. On the menu bar, choose **TENANTS**. In the Navigation pane, expand the tenant, right-click **Application Profiles**, and click **Create Application Profile**.
2. In the **Create Application Profile** dialog box, in the **Name** field, add the application profile name (Big\_Data\_Lab).

Figure 46 Creating Application Profile



## Creating EPGs Using the GUI

### Procedure

Expand EPGs. In the **Create Application EPG** dialog box, perform the following actions:

1. In the **Name** field, add the EPG name (Cloudera).
2. In the **Bridge Domain** field, choose the bridge domain from the drop-down list (BD\_1).
3. Expand **Associated Domain Profiles** and from the drop-down list, choose the **Domain Profile** name (INBAND).
4. Click **Update**, and click **OK**.



**Figure 47**      **Creating EPG**

**CREATE APPLICATION PROFILE**  
**CREATE APPLICATION EPG**

**STEP 1 > IDENTITY**      1. IDENTITY

Specify the EPG Identity

Name:

Description:

Tags:   
enter tags separated by comma

QoS class:

Custom QoS:

Bridge Domain:

Monitoring Policy:

Associated Domain Profiles (VMs or bare metals):

Domain Profile	Deployment Immediacy	Resolution Immediacy
<input type="text" value="INBAND"/>	<input type="text" value="On Demand"/>	<input type="text" value="On Demand"/>

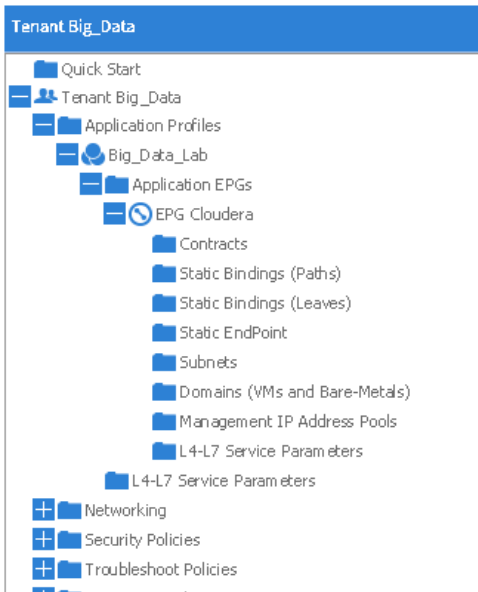
Statically Link with Leaves/Paths:

< PREVIOUS    OK    CANCEL

## Creating the Static Binding for the Leaves and Paths

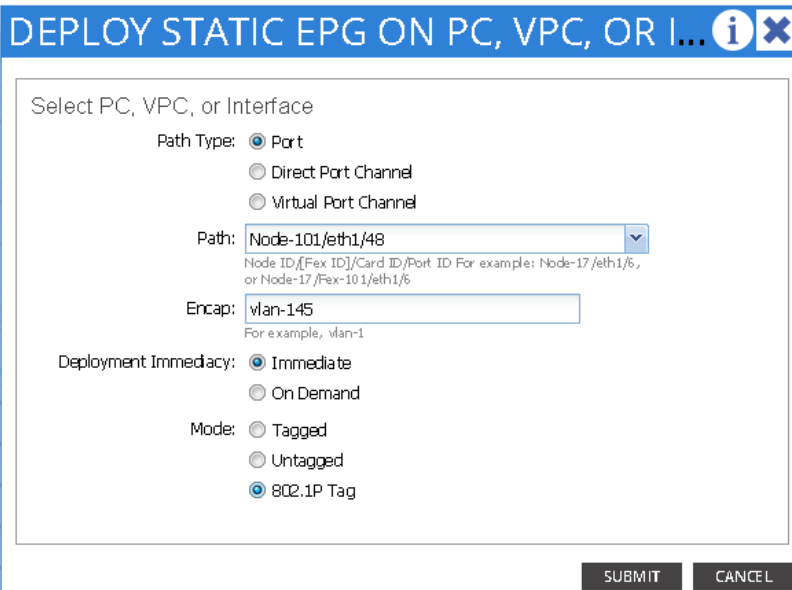
1. On the menu bar, choose **TENANTS** and the tenant name on which you want to operate. In the Navigation pane, expand the **tenant > Application Profiles > Big\_Data\_Lab > Application EPGs > EPG Cloudera** and select **Static Bindings (Paths)**.

Figure 48 Exploring the EPG



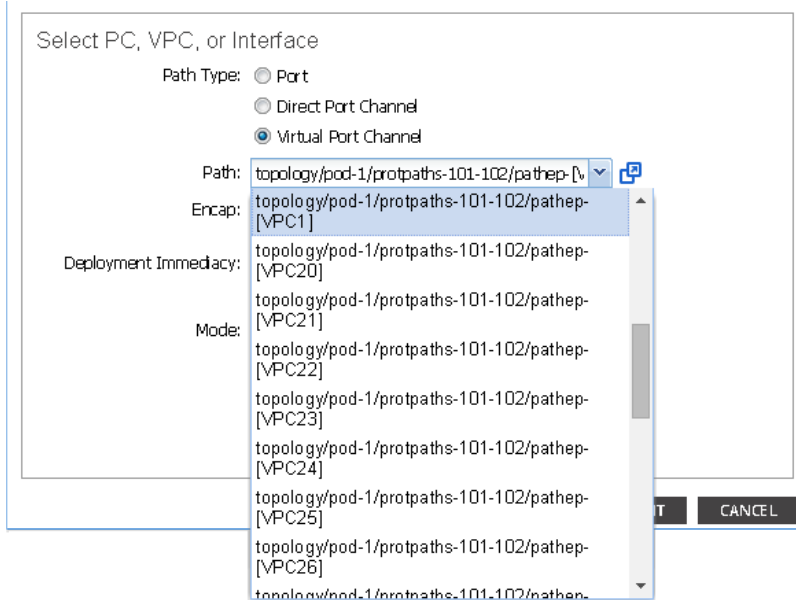
2. Right click on **Static Bindings (Paths)** and select **Deploy Static EPG on PC, VPC, or Interface**.
3. From the Path: drop down list select the appropriate nodes and port where the APIC is connected. **On Encap** field use vlan-145, on Deployment Immediacy select the **Immediate** radio button and on Mode select the 802.1p Tag and click **Submit**.

Figure 49 Deploying Static EPG on Interface



4. Right click on **Static Bindings (paths)** and select **Deploy Static EPG on PC, VPC, or Interface**.
5. In the Path Type: select the **Virtual Port Channel** radio button.
6. From the Path: drop down list select the appropriate nodes and port where the servers are connected. On Encap field use vlan-145, on Deployment Immediacy select the **Immediate** radio button and on Mode select the 802.1p Tag and click **Submit**.

**Figure 50** Deploying Static EPG on vPC



7. Repeat step 4, 5 & 6 for all the vPC ports created.

The network is now fully configured for the servers and for CIMC management. The servers should be able to communicate after configuring IP and other parameters as desired.

## Configuring Disk Drives for OS on Name Nodes

Namenode and Secondary Namenode have a different RAID configuration compared to Data nodes. This section details the configuration of disk drives for OS on these nodes (rhel1 and rhel2). The disk drives are configured as RAID1, read ahead cache is enabled and write cache is enabled while battery is present. The first two disk drives are used for operating system and remaining 22 disk drives are using for HDFS as described in the following sections.

There are several ways to configure RAID: using LSI WebBIOS Configuration Utility embedded in the MegaRAID BIOS, booting DOS and running MegaCLI commands, using Linux based MegaCLI commands, or using third party tools that have MegaCLI integrated. For this deployment, the first two disk drives are configured using LSI WebBIOS Configuration Utility and rests are configured using Linux based MegaCLI commands after the completion of the Operating system Installation.

Figure 51 Configuring Disk Drives



Disk drives 1 and 2  
configure RAID 1  
using WebBIOS

Disk Drives 3 to 24  
configure RAID 1  
using Megacli

Follow these steps to create RAID1 on the first two disk drives to install the operating system:

## Configure CIMC IP Address using console

Follow the steps given below to configure CIMC IP Address:

1. Press the **Power** button to boot the server. Watch for the prompt to press F8.
2. During bootup, press F8 when prompted to open the **Cisco IMC Configuration Utility**.

Figure 52 Selecting KVM Console Option

```

Cisco IMC Configuration Utility Version 2.0 Cisco Systems, Inc.
*****
NIC Properties
NIC mode          NIC redundancy
Dedicated:       [X]          None:             [ ]
Shared LOM:      [ ]          Active-standby:  [ ]
Cisco Card:      [ ]          Active-active:   [X]
Shared LOM Ext:  [ ]

IP (Basic)
IPV4:            [X]          IPV6:             [ ]
DHCP enabled    [ ]
CIMC IP:         10.0.145.201
Prefix/Subnet:  255.255.255.0
Gateway:        10.0.145.1
Pref DNS Server: ::

VLAN (Advanced)
VLAN enabled:   [ ]
VLAN ID:        1
Priority:        0

*****
<Up/Down>Selection <F10>Save <Space>Enable/Disable <F5>Refresh <ESC>Exit
<F1>Additional settings
    
```

3. Select the NIC Mode as **Dedicated** [x].
4. On IP (Basic), select IPv4 [x], define the desired IP address, subnet and gateway.



Note

The IP Subnet should be the same throughout the fabric. Here we have used 10.0.145.201/24

5. Under NIC Redundancy, select **Active-active [x]**.
6. Press f1 for additional settings.

*Figure 53*      *Configuring Additional Settings*

```

Cisco IMC Configuration Utility Version 2.0 Cisco Systems, Inc.
*****
Common Properties
  Hostname:
  Dynamic DNS: [ ]
  DDNS Domain:
FactoryDefaults
  Factory Default: [ ]
Default User(Basic)
  Default password:
  Reenter password:
Port Properties
  Auto Negotiation: [ ]
  Speed[1000/100 Mbps]: 1000
  Duplex mode[half/full]: full
Port Profiles
  Reset: [ ]
  Name:

*****
<Up/Down>Selection <F10>Save <Space>Enable/Disable <F5>Refresh <ESC>Ex
<F2>PreviousPage

```

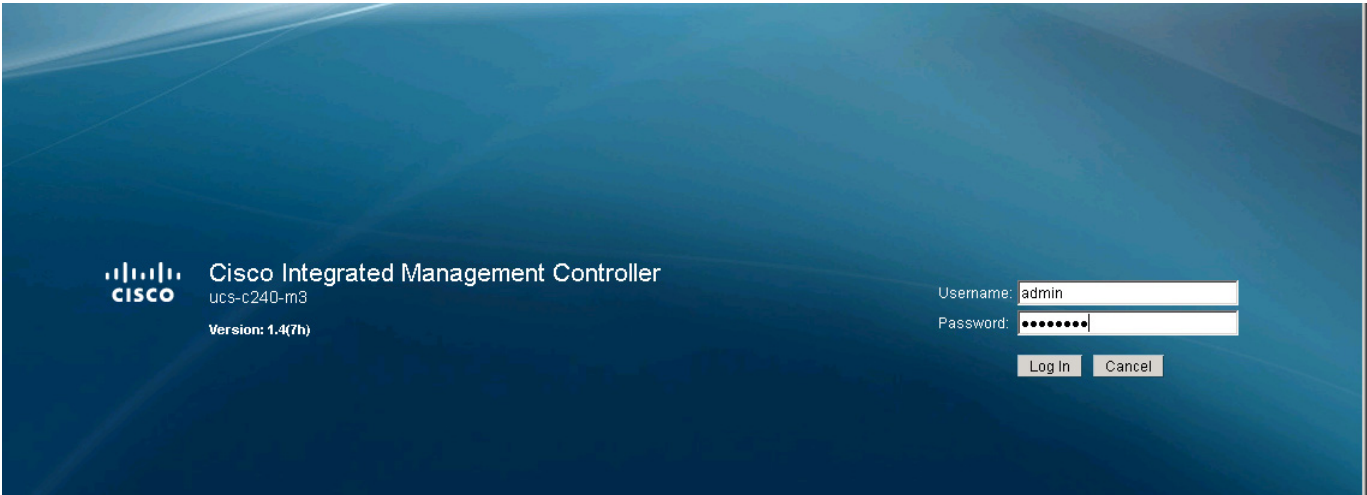
7. Under Port Properties select the **speed as 1000**.
8. You can leave rest of the field as default.
9. Press **F10** to save the configuration and reboot the system (ctrl+alt+del).
10. Repeat step a through I for all the nodes.

## Connecting Servers Through CIMC and Launching KVM

Follow the steps given below to launch KVM:

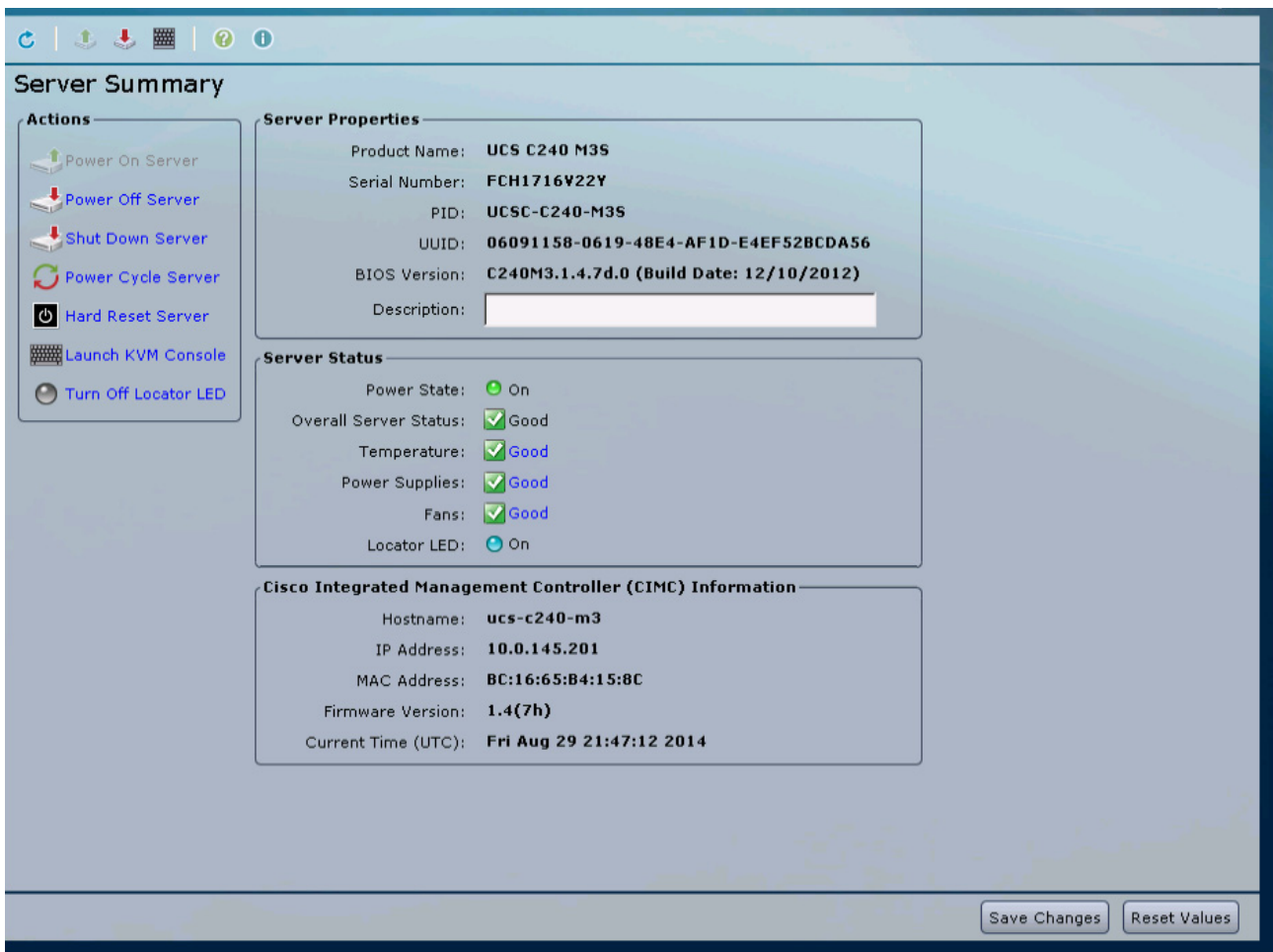
1. Using web browsers connect to CIMC IP address of the nodes assigned above.

Figure 54 KVM Login Page



2. The default username is “admin” and the password is “password”.

Figure 55 Server Summary



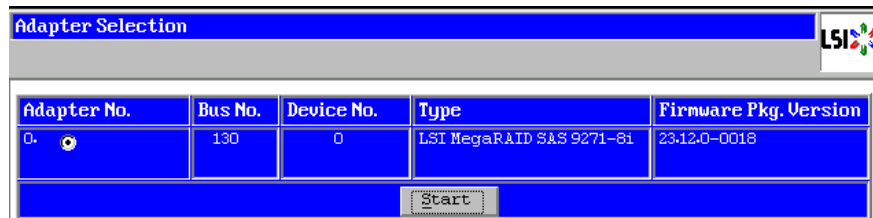
3. Click on **Launch KVM Console**.

4. On the security warning dialogue box check the “Always trust content from this publisher” checkbox and select **Yes**.

## Configuring RAID on Namenode

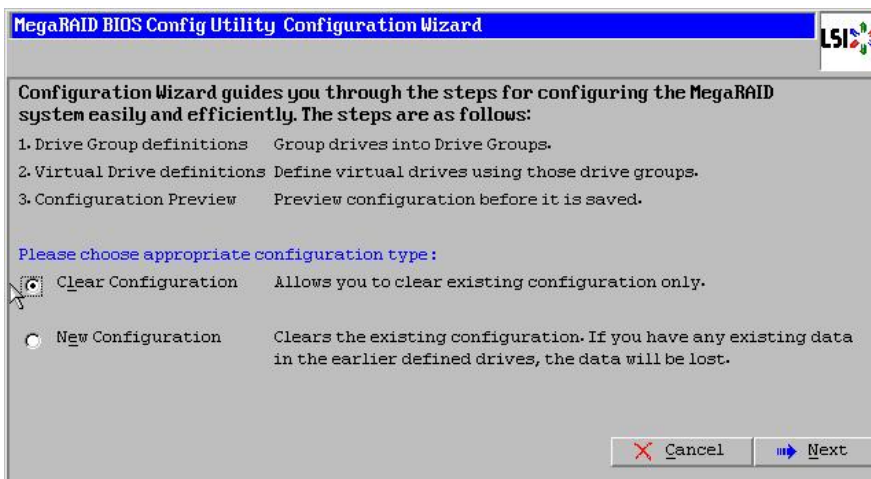
1. When the server is booting, the following text appears on the screen:
  - a. Press <Ctrl><H> to launch the WebBIOS.
  - b. Press **Ctrl+H** immediately.
 The **Adapter Selection** window appears.
2. Click **Start** to continue.
3. Click **Configuration Wizard**.

Figure 56 Adapter Selection for RAID Configuration



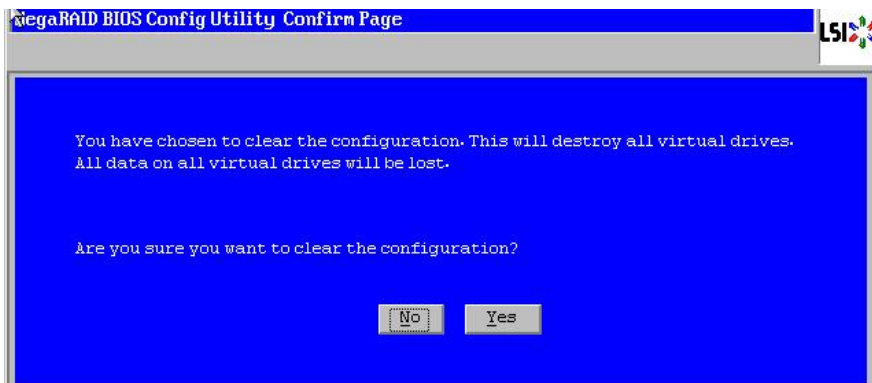
4. In the configuration wizard window, choose **Clear Configuration** and click **Next** to clear the existing configuration.

Figure 57 Clearing Current configuration on the controller



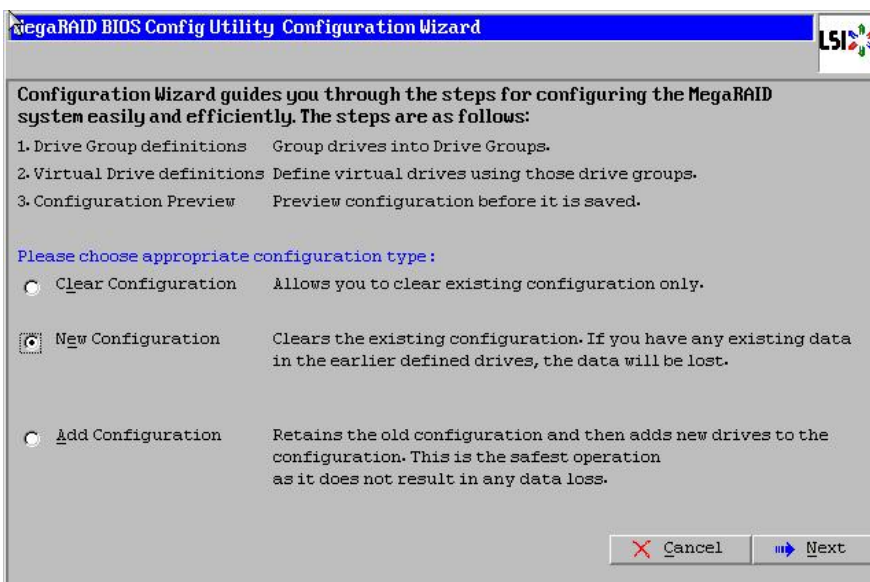
5. Choose **Yes** when asked to confirm the wiping of the current configuration.
6. In the **Physical View**, ensure that all the drives are **Unconfigured Good**.
7. Click **Configuration Wizard**.

Figure 58 Confirming Clearance of the previous configuration on the controller



8. In the **Configuration Wizard** window choose the configuration type to be **New Configuration** and click **Next**.

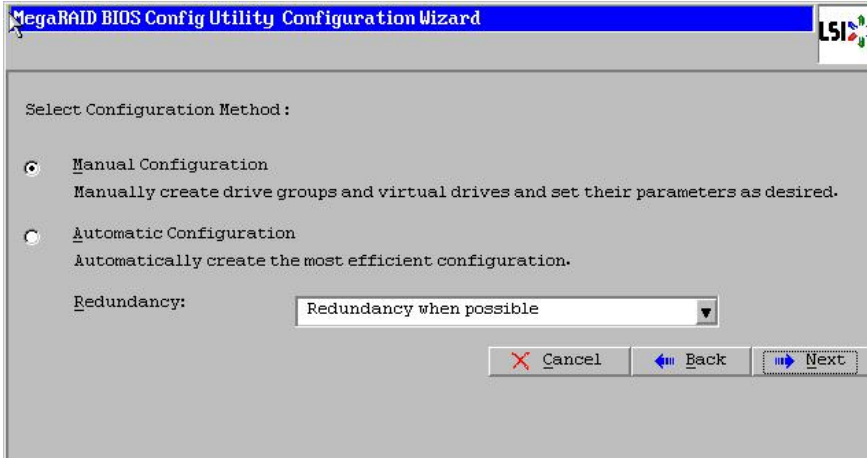
Figure 59 Choosing to create a New Configuration



9. Select the configuration method to be **Manual Configuration**; this enables you to have complete control over all attributes of the new storage configuration, such as, the drive groups, virtual drives and the ability to set their parameters.
10. Click **Next**.

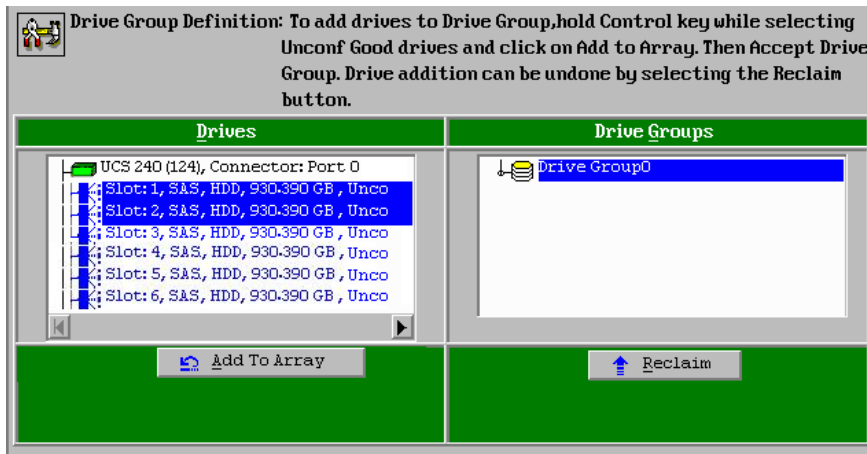


Figure 60 Choosing Manual Configuration Method



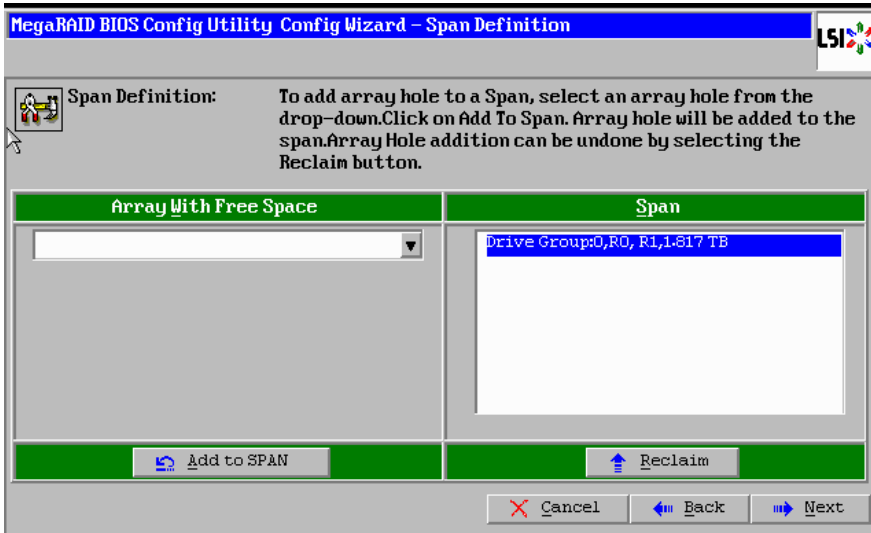
11. The **Drive Group Definition** window appears. Use this window to choose the first two drives to create drive group.
12. Click **Add** to Array to move the drives to a proposed drive group configuration in the **Drive Groups** pane. Click **Accept DG** and then, click **Next**.

Figure 61 Selecting first drive and Adding to Drive Group



13. In the **Span definitions** Window, Click **Add to SPAN** and click **Next**.

Figure 62 Span Definition Window



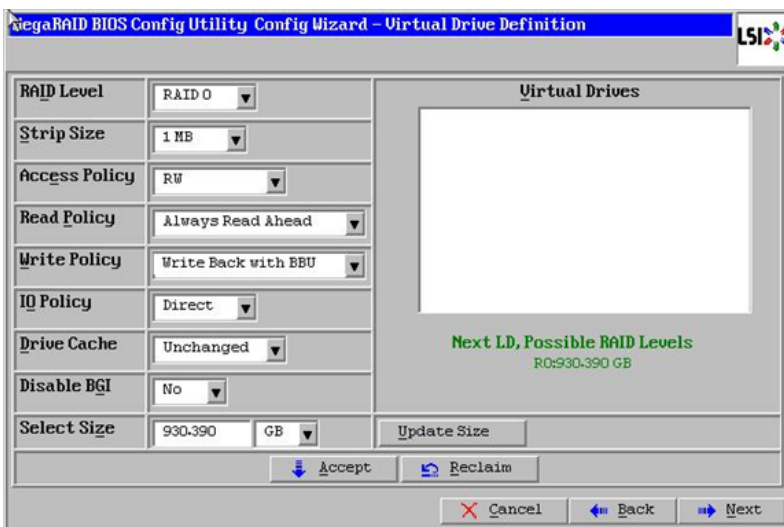
14. In the Virtual Drive definitions window,
  - a. Click on **Update Size**.
  - b. Change Strip Size to 64 KB. A larger strip size produces higher read performance
  - c. From the **Read Policy** drop-down list, choose **Always Read Ahead**.
  - d. From the **Write Policy** drop-down list, choose **Write Back** with BBU.
  - e. Make sure RAID Level is set to **RAID1**.
  - f. Click **Accept** to accept the changes to the virtual drive definitions.
  - g. Click **Next**.



**Note**

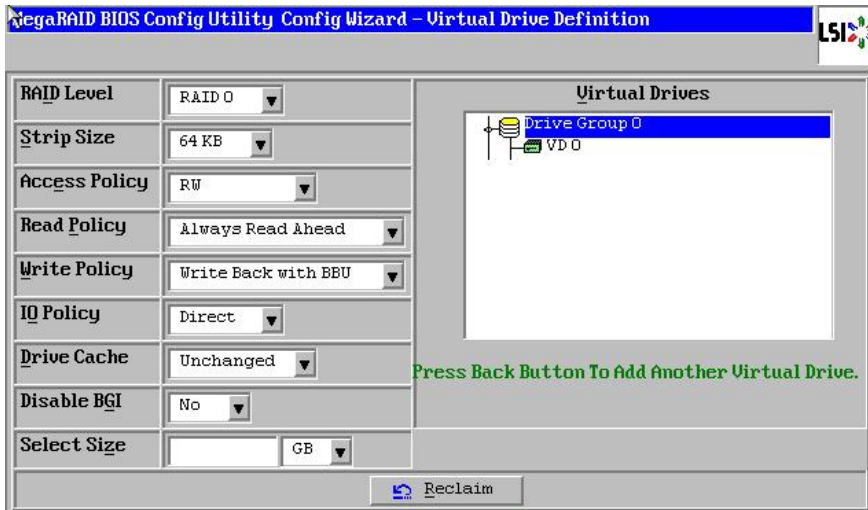
Clicking on Update Size might change some of the settings in the window. Make sure all settings are correct before accepting.

Figure 63 Virtual Drive Definition Window



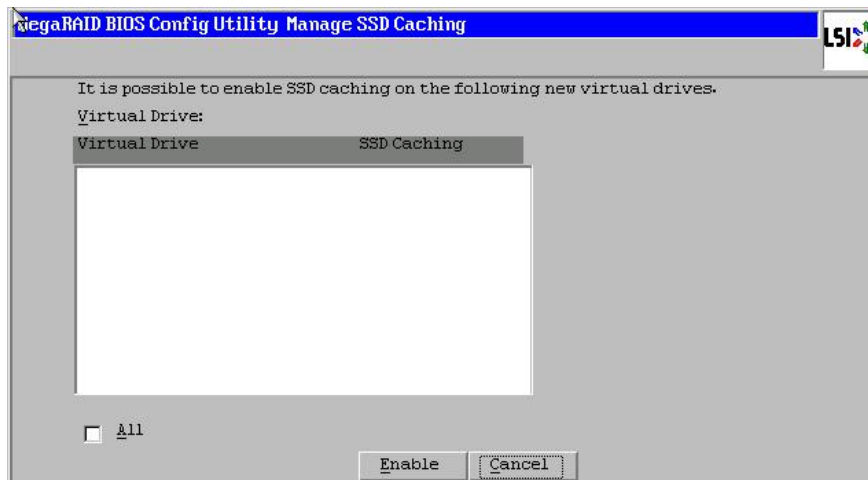
15. After you finish the virtual drive definitions, click **Next**. The **Configuration Preview** window appears showing VDO.
16. Check the virtual drive configuration in the **Configuration Preview** window and click **Accept** to save the configuration.

Figure 64 Completed Virtual Drive Definition



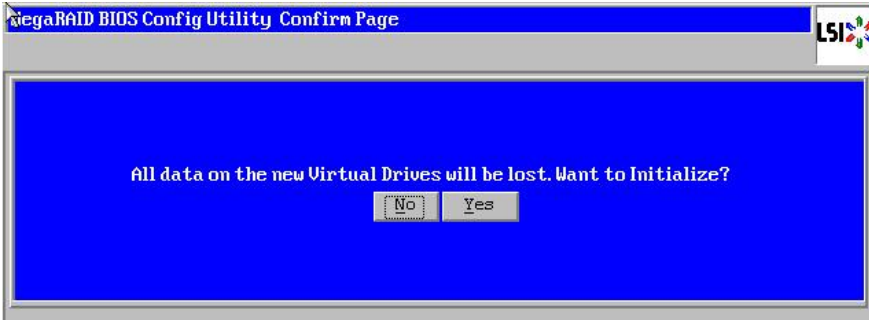
17. Click **Yes** to save the configuration.
18. In the **Managing SSD Caching Window**, Click **Cancel**.

Figure 65 SSD Caching Window



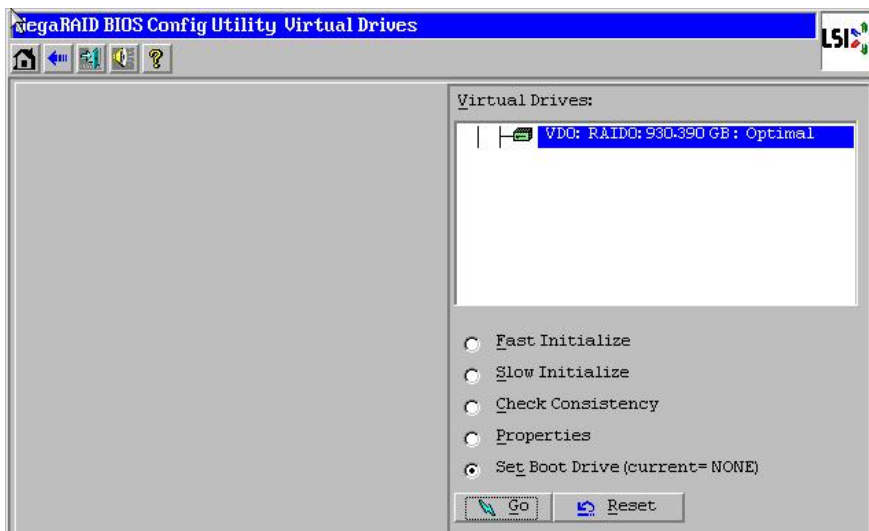
19. Click **Yes**. When asked to confirm the initialization.

Figure 66 *Initializing Virtual Drive Window*



1. Set **VD0** as the **Boot Drive** and click **Go**.
2. Click **Home**.
3. Review the Configuration and Click **Exit**.

Figure 67 *Setting Virtual Drive as Boot Drive*



Configuration of disks 3-24 are done using Linux based MegaCLI command as described in Section on Configuring Data Drives for Namenode later in this document.

## Configuring Disk Drives for OS on Data Nodes

Nodes 3 through 64 are configured as data nodes. This section details the configuration of disk drives for OS on the data nodes. As stated above, the focus of this CVD is the High Performance Configuration featuring 24 1TB SFF disk drives. The disk drives are configured as individual RAID0 volumes with 1MB stripe size. Read ahead cache and write cache is enabled while battery is present. The first disk drive is used for operating system and remaining 23 disk drives are using for HDFS as described in the following sections.

**Note**

In the case of High Capacity Configuration featuring 12 4TB LFF disk drives, the disk drives are configured as individual RAID0 volumes with 1MB stripe size. Read ahead cached is enable and write cache is enabled while battery is present. Two partitions of 1TB and 3TB are created on the first disk drive, the 1TB partition is used for operating system and the 3TB partition is used for HDFS along with disk drives 2 through 12.

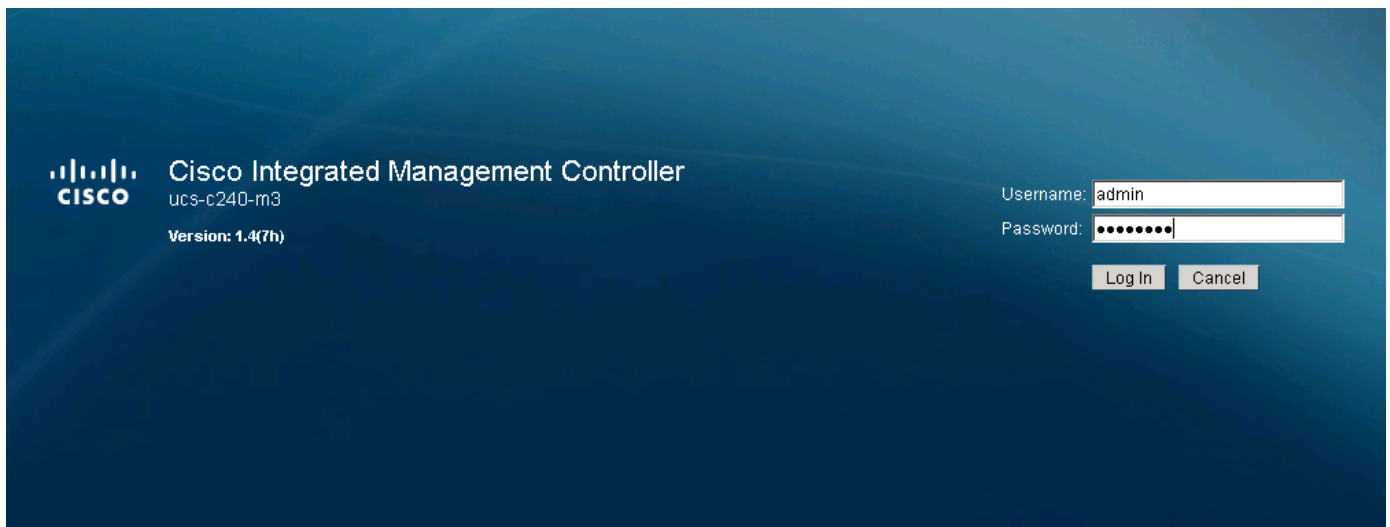
There are several ways to configure RAID: using LSI WebBIOS Configuration Utility embedded in the MegaRAID BIOS, booting DOS and running MegaCLI commands, using Linux based MegaCLI commands, or using third party tools that have MegaCLI integrated. For this deployment, the first disk drive is configured using LSI WebBIOS Configuration Utility and rest is configured using Linux based MegaCLI commands after the OS is installed.

Follow these steps to create RAID0 on the first disk drive to install the operating system:

## Launching KVM

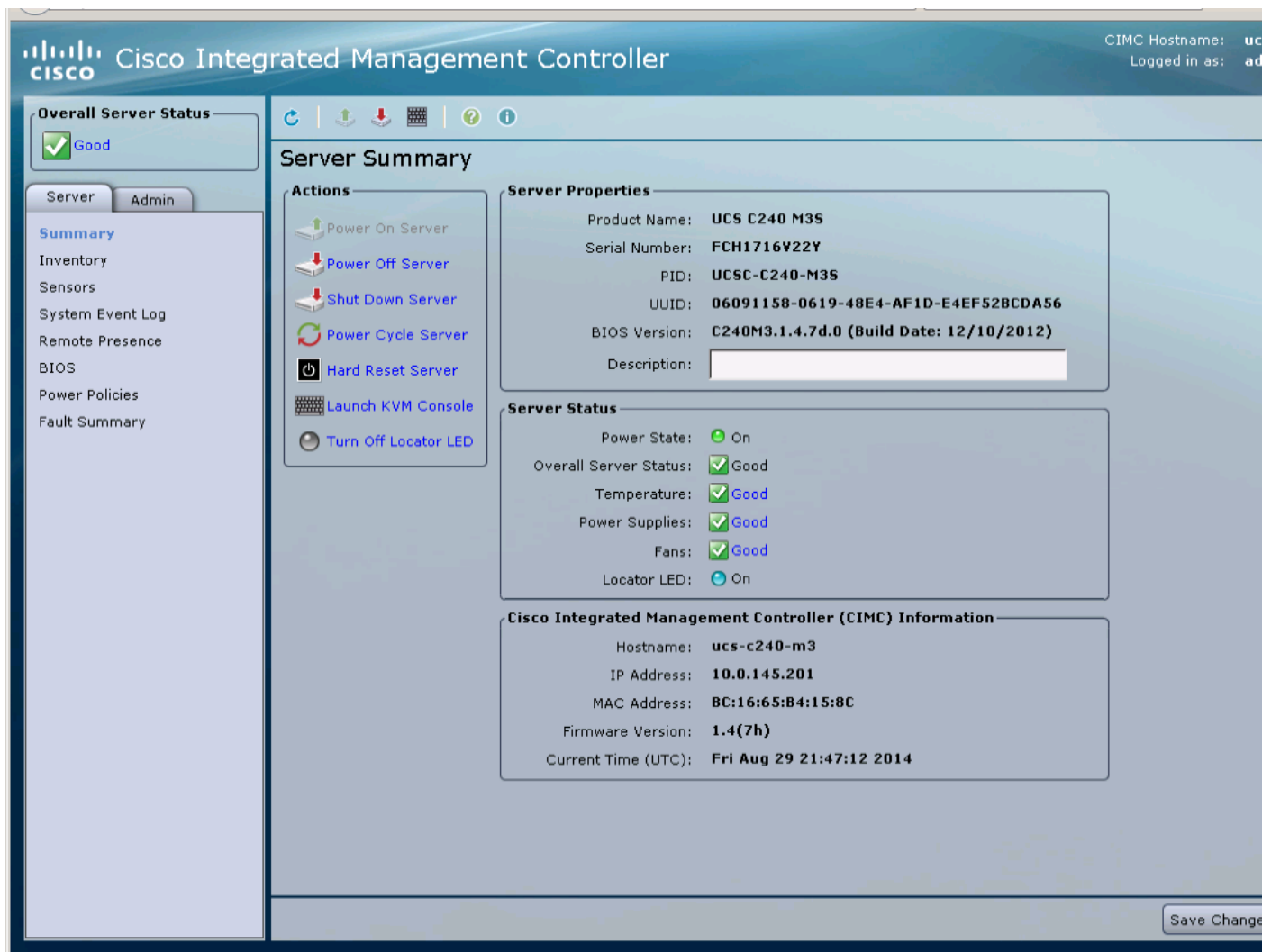
1. Using web browser connect to CIMC IP address of the nodes assigned above.

*Figure 68 KVM Login Screen*



2. The default username is “**admin**” and the password is “**password**”.

Figure 69 Server Summary Page



3. Click on **Launch KVM Console**.
4. On the security warning dialogue box check the “**Always trust content from this publisher**” checkbox and select **Yes**.

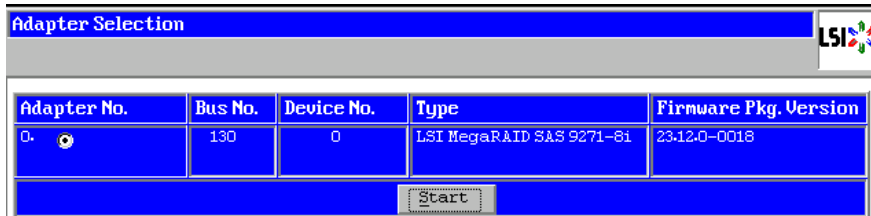
## Configuring RAID for Data Nodes

1. When the server is booting, the following text appears on the screen:
  - a. Press <Ctrl><H> to launch the **WebBIOS**.
  - b. Press **Ctrl+H** immediately.

The Adapter Selection window appears.

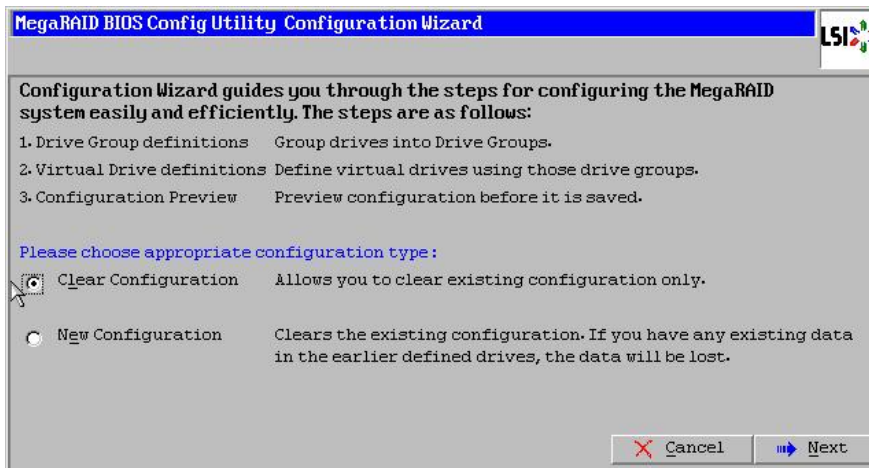
2. Click **Start** to continue.
3. Click **Configuration Wizard**.

Figure 70 Adapter Selection for RAID Configuration



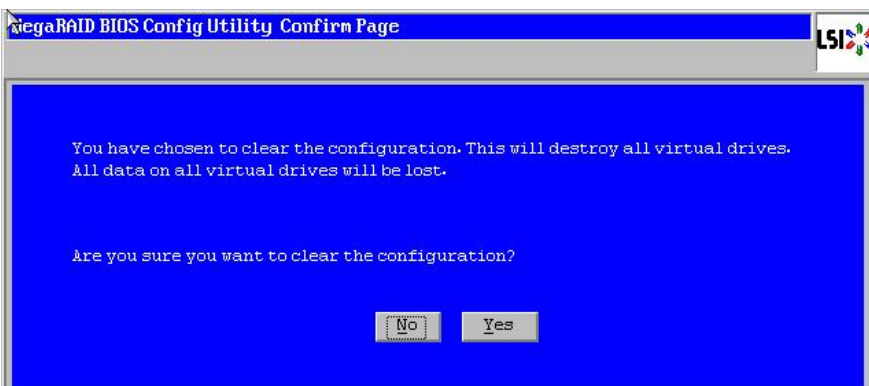
- In the configuration wizard window, choose **Clear Configuration** and click **Next** to clear the existing configuration.

Figure 71 Clearing Current configuration on the controller



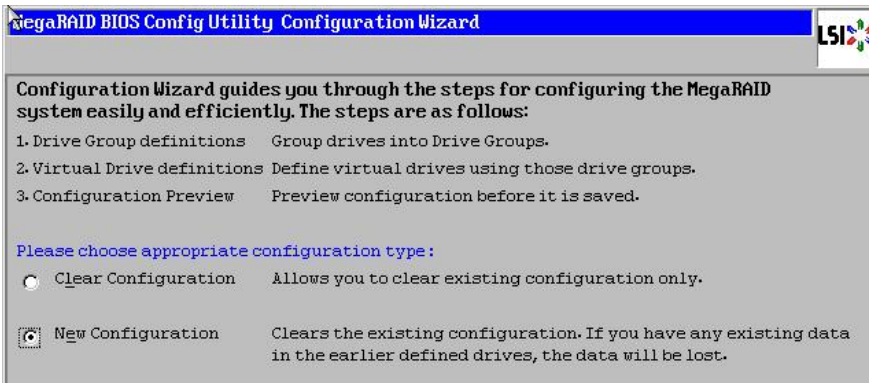
- Choose **Yes** when asked to confirm the wiping of the current configuration.
- In the **Physical View**, ensure that all the drives are **Unconfigured Good**.
- Click **Configuration Wizard**.

Figure 72 Confirming Clearance of the previous configuration on the controller



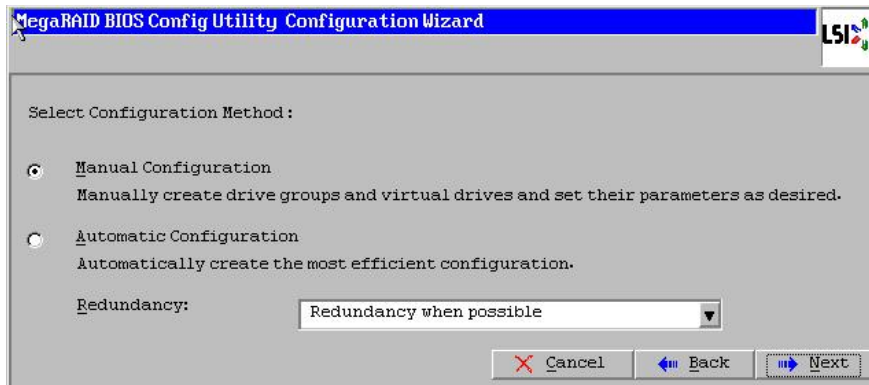
- In the **Configuration Wizard** window choose the configuration type to be **New Configuration** and click **Next**.

Figure 73 Choosing to create a New Configuration



9. Select the configuration method to be **Manual Configuration**; this enables you to have complete control over all attributes of the new storage configuration, such as, the drive groups, virtual drives and the ability to set their parameters.
10. Click **Next**.

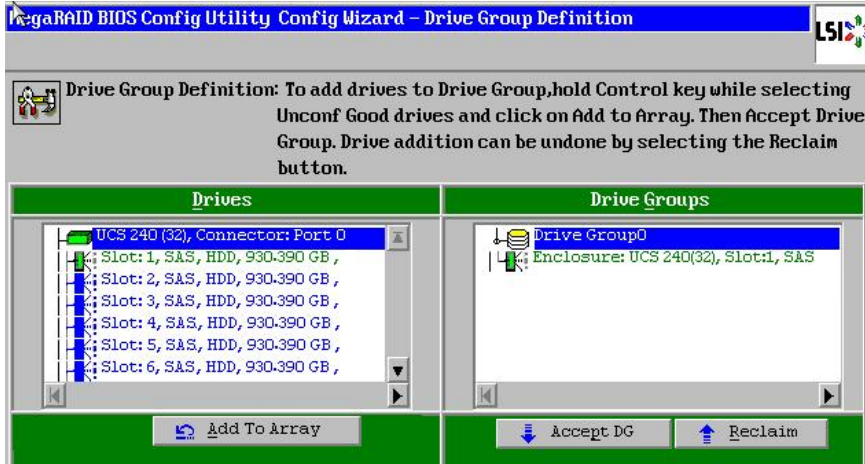
Figure 74 Choosing Manual Configuration Method



11. The **Drive Group Definition** window appears. Use this window to choose the first drive to create drive groups.
12. Click **Add to Array** to move the drives to a proposed drive group configuration in the **Drive Groups** pane. Click **Accept DG** and then, click **Next**.

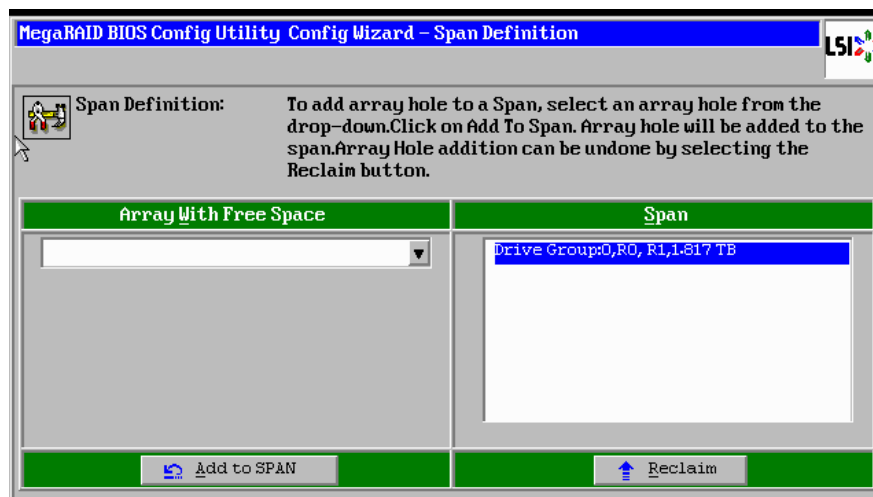


Figure 75 Selecting first drive and Adding to Drive Group



13. In the **Span definitions** Window, Click **Add to SPAN** and click **Next**.

Figure 76 Span Definition Window\_2



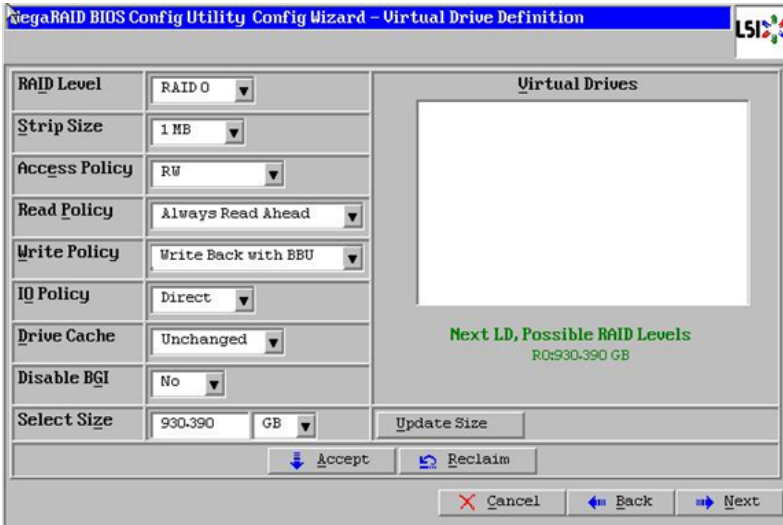
14. In the **Virtual Drive** definitions window,
- Click on **Update Size**.
  - Change Strip Size to 1MB. A larger strip size produces higher read performance
  - From the **Read Policy** drop-down list, choose **Always Read Ahead**.
  - From the **Write Policy** drop-down list, choose **Write Back with BBU**.
  - Make sure RAID Level is set to **RAID0**.
  - Click **Accept** to accept the changes to the virtual drive definitions.
  - Click **Next**.



Note

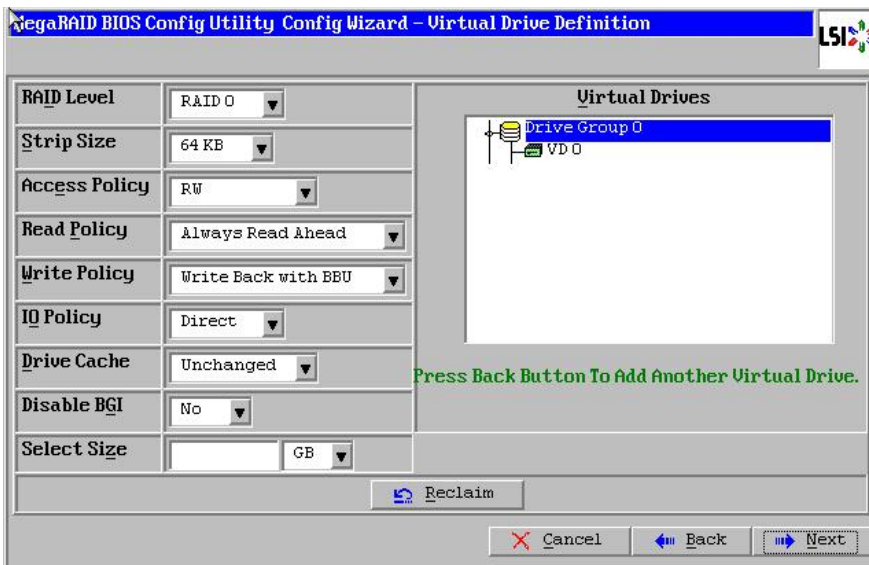
Clicking on Update Size might change some of the settings in the window. Make sure all settings are correct before accepting.

Figure 77 Virtual Drive Definition Window



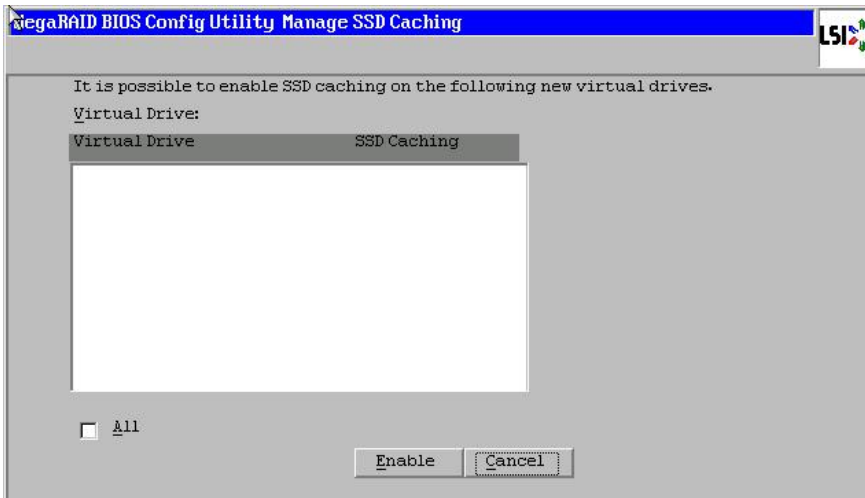
15. After you finish the virtual drive definitions, click **Next**. The **Configuration Preview** window appears showing VD0.
16. Check the virtual drive configuration in the **Configuration Preview** window and click **Accept** to save the configuration.

Figure 78 Completed Virtual Drive Definition



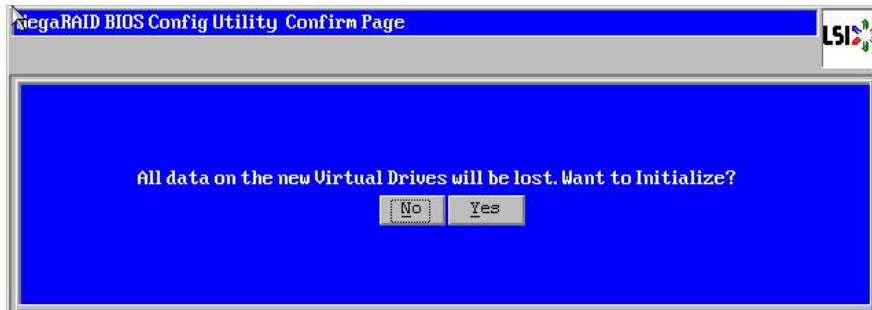
17. Click **Yes** to save the configuration.
18. In the **Managing SSD Caching** window, Click **Cancel**.

Figure 79 SSD Caching Window\_2



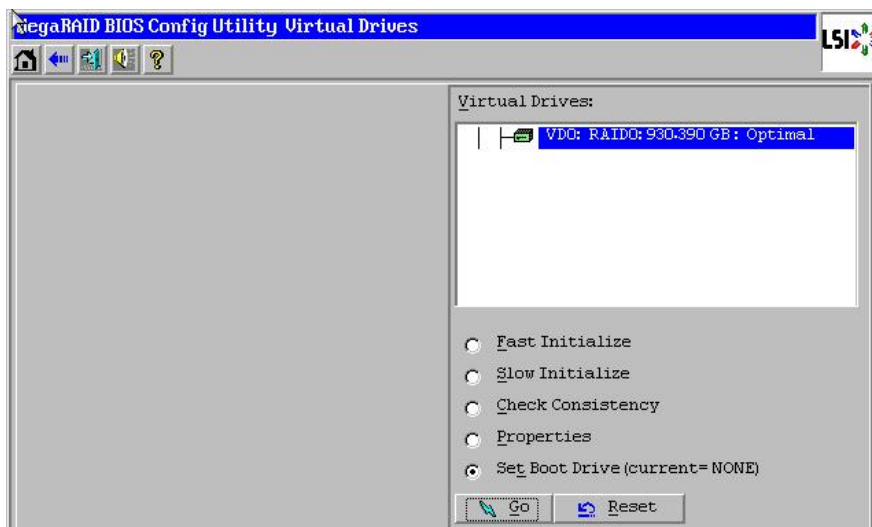
19. Click Yes. When asked to confirm the initialization.

Figure 80 Initializing Virtual Drive Window



20. Set VD0 as the Boot Drive and click Go.

Figure 81 Setting Virtual Drive as Boot Drive



21. Click Home.

22. Review the configuration and Click **Exit**.

The steps above can be repeated to configure disks 2-24 or using Linux based MegaCLI commands as described in Section on Configuring Data Drives later in this document.

## Installing Red Hat Linux 6.4

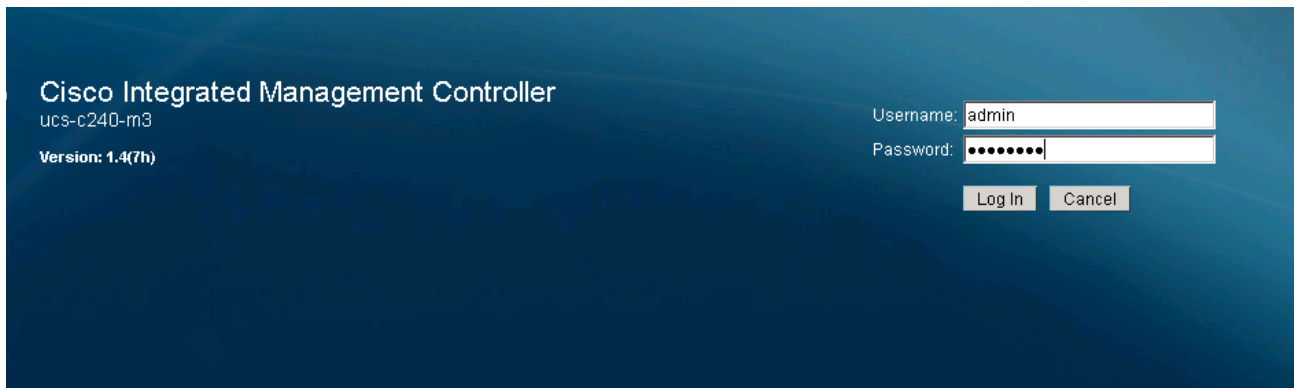
The following section provides detailed procedures for installing Red Hat Linux 6.4.

There are multiple methods to install Red Hat Linux operating system. The installation procedure described in this deployment guide uses KVM console and virtual media from Cisco Integrated Management Console.

## Installing the Operating system

1. Using a web browser connect to CIMC IP address of the nodes.

*Figure 82 CIMC Login Page*



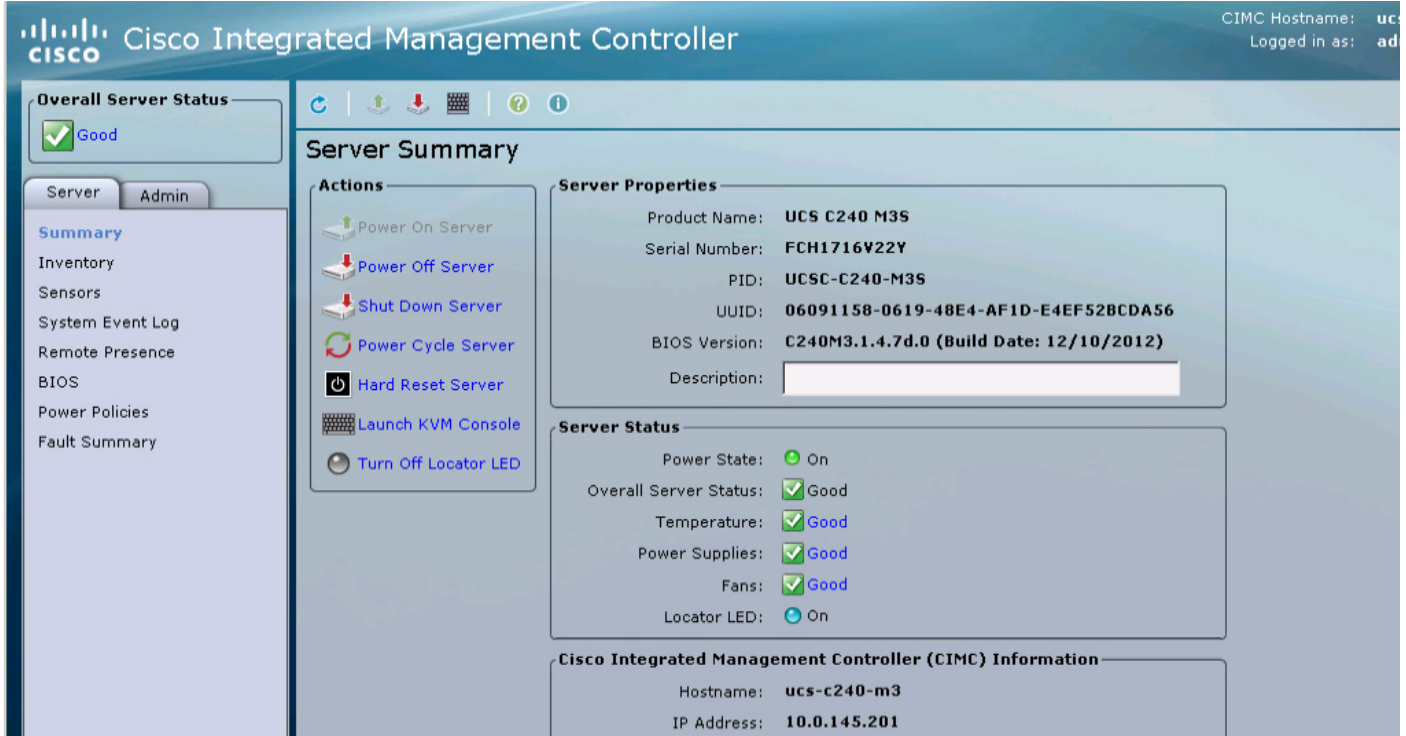
Cisco Integrated Management Controller  
ucs-c240-m3  
Version: 1.4(7h)

Username: admin  
Password: .....

Log In Cancel

2. Enter the default username “**admin**” and password “**password**”.

Figure 83 CIMC Server Summary Page

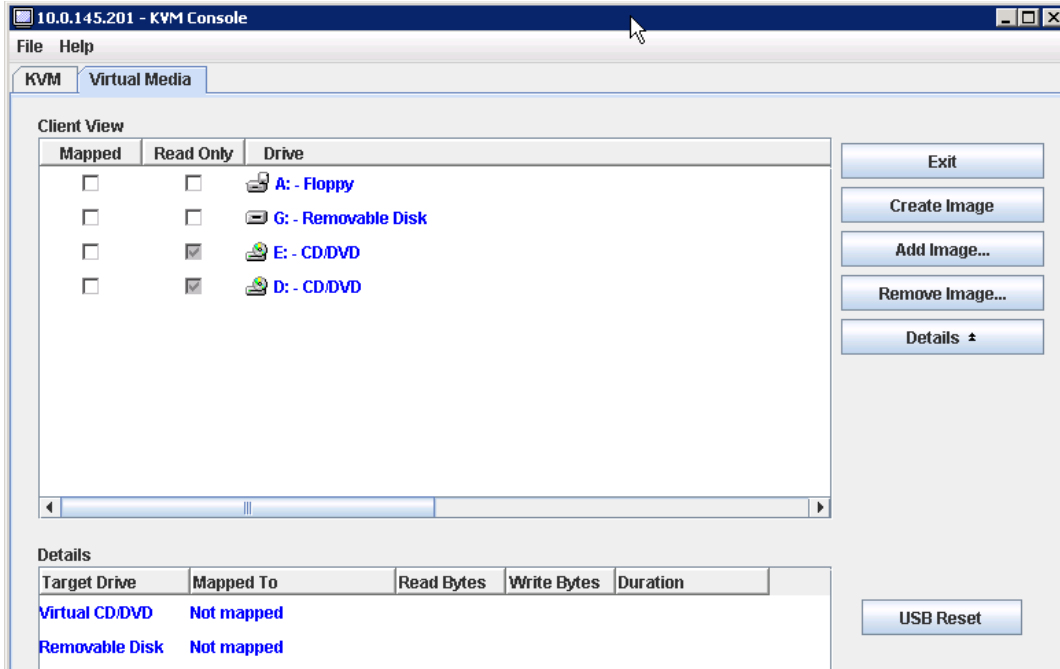


The screenshot displays the Cisco Integrated Management Controller (CIMC) interface. At the top, the Cisco logo and 'Cisco Integrated Management Controller' are visible, along with the CIMC Hostname and user login information. The main content area is titled 'Server Summary' and is divided into several sections:

- Overall Server Status:** Shows a green checkmark and the word 'Good'.
- Actions:** A list of server management options: Power On Server, Power Off Server, Shut Down Server, Power Cycle Server, Hard Reset Server, Launch KVM Console, and Turn Off Locator LED.
- Server Properties:** A box containing:
  - Product Name: UCS C240 M3S
  - Serial Number: FCH1716V22Y
  - PID: UCSC-C240-M3S
  - UUID: 06091158-0619-48E4-AF1D-E4EF52BCDA56
  - BIOS Version: C240M3.1.4.7d.0 (Build Date: 12/10/2012)
  - Description: (empty text box)
- Server Status:** A box showing various health indicators:
  - Power State: On (green dot)
  - Overall Server Status: Good (green checkmark)
  - Temperature: Good (green checkmark)
  - Power Supplies: Good (green checkmark)
  - Fans: Good (green checkmark)
  - Locator LED: On (blue dot)
- Cisco Integrated Management Controller (CIMC) Information:** A box showing:
  - Hostname: ucs-c240-m3
  - IP Address: 10.0.145.201

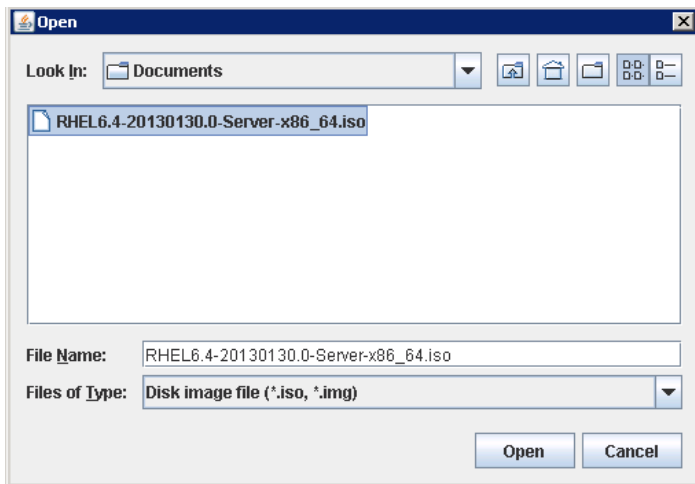
3. Click on **Launch KVM Console**.
4. On the security warning dialogue box check the “**Always trust content from this publisher**” checkbox and select **Yes**.
5. Once the KVM console is launched click on the **Virtual Media** tab.

Figure 84 Launching KVM Console



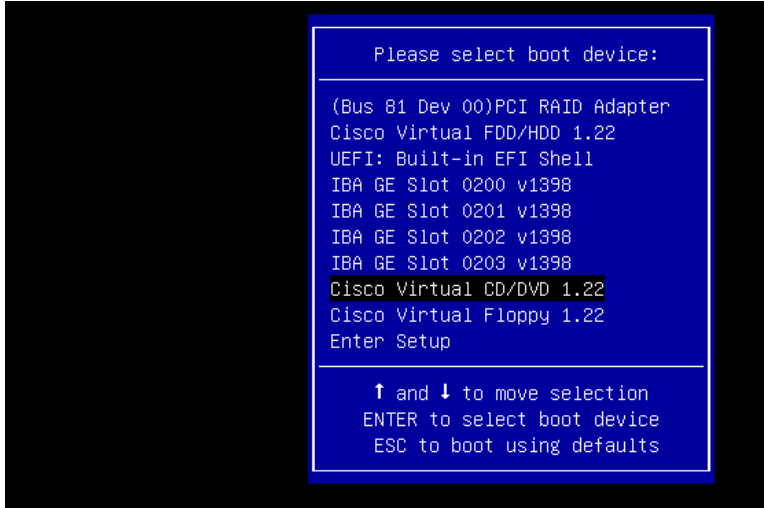
- Click on **Add Image...** box and load the appropriate Redhat Enterprise 6.4 Installer iso image and click **Open**.

Figure 85 Loading RedHat Enterprise 6.4 Installer



- On virtual media tab check the check box for the image that was loaded in earlier step.
- Click on the **KVM** tab in the window, click **macros > ctrl+alt+del**.
- Press **F6** to select the boot media and select Cisco Virtual CD/DVD 1.22 and press **enter**.

*Figure 86*      *Selecting Boot Media*



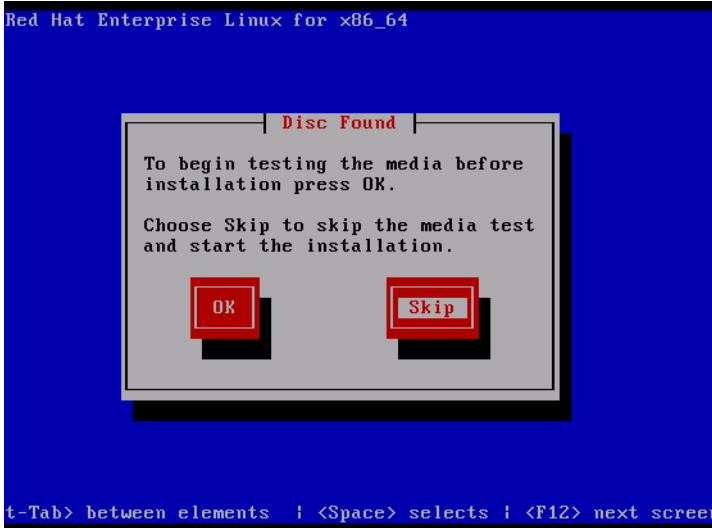
10. Select the **Install or upgrade an Existing System**.

*Figure 87*      *Selecting the Install or Upgrade an Existing System.*



11. Skip the Media test as we are installing from an ISO Image.

*Figure 88*      *Skipping Media Test*



12. Click Next

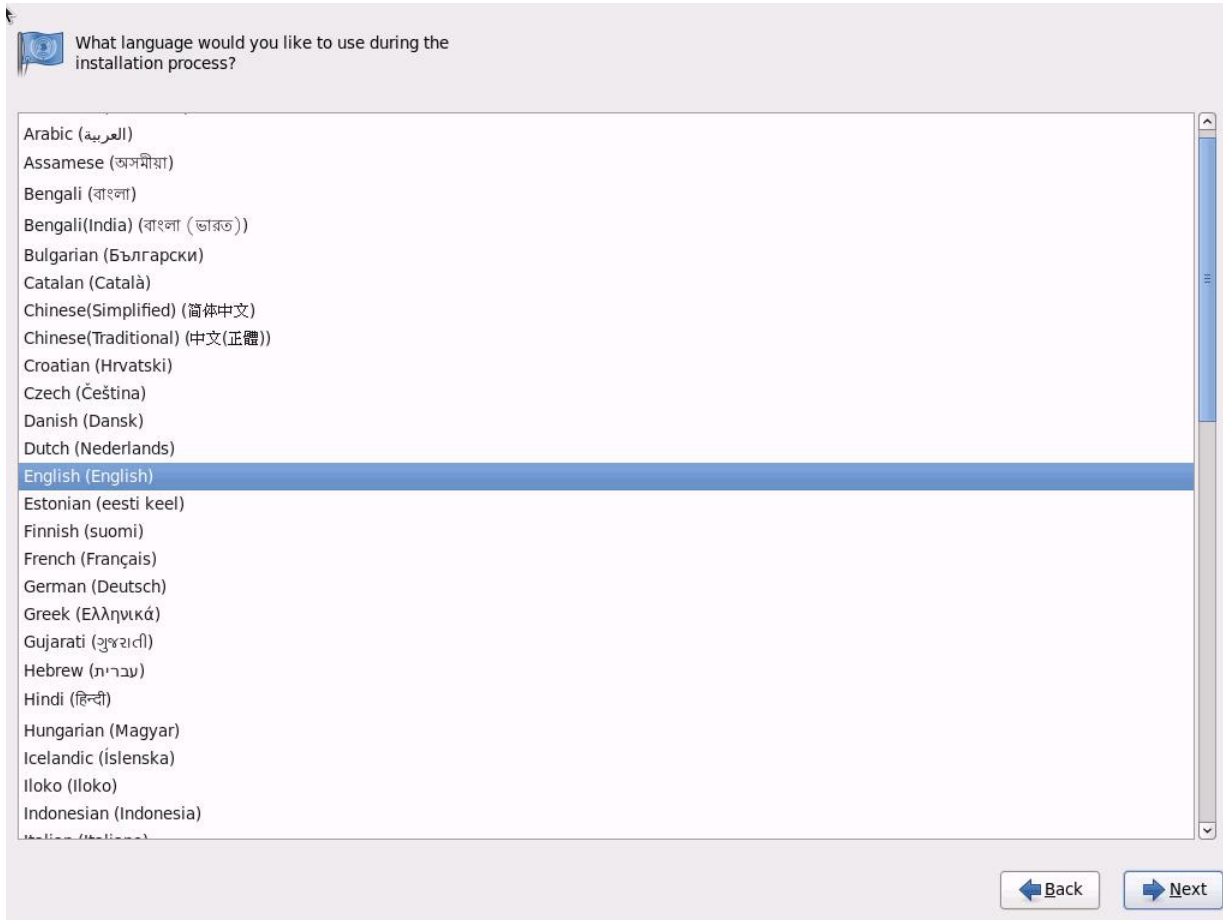
*Figure 89*      *Red Hat Linux Welcome Screen*



13. Select the Language for the Installation



**Figure 90**      *Selecting Language for the Installation*



**14. Select Basic Storage Devices.**

*Figure 91*      *Selecting Basic Storage Devices*

What type of devices will your installation involve?

**Basic Storage Devices**

- Installs or upgrades to typical types of storage devices. If you're not sure which option is right for you, this is probably it.

**Specialized Storage Devices**

- Installs or upgrades to enterprise devices such as Storage Area Networks (SANs). This option will allow you to add FCoE / iSCSI / zFCP disks and to filter out devices the installer should ignore.

15. Select **Fresh Installation**.

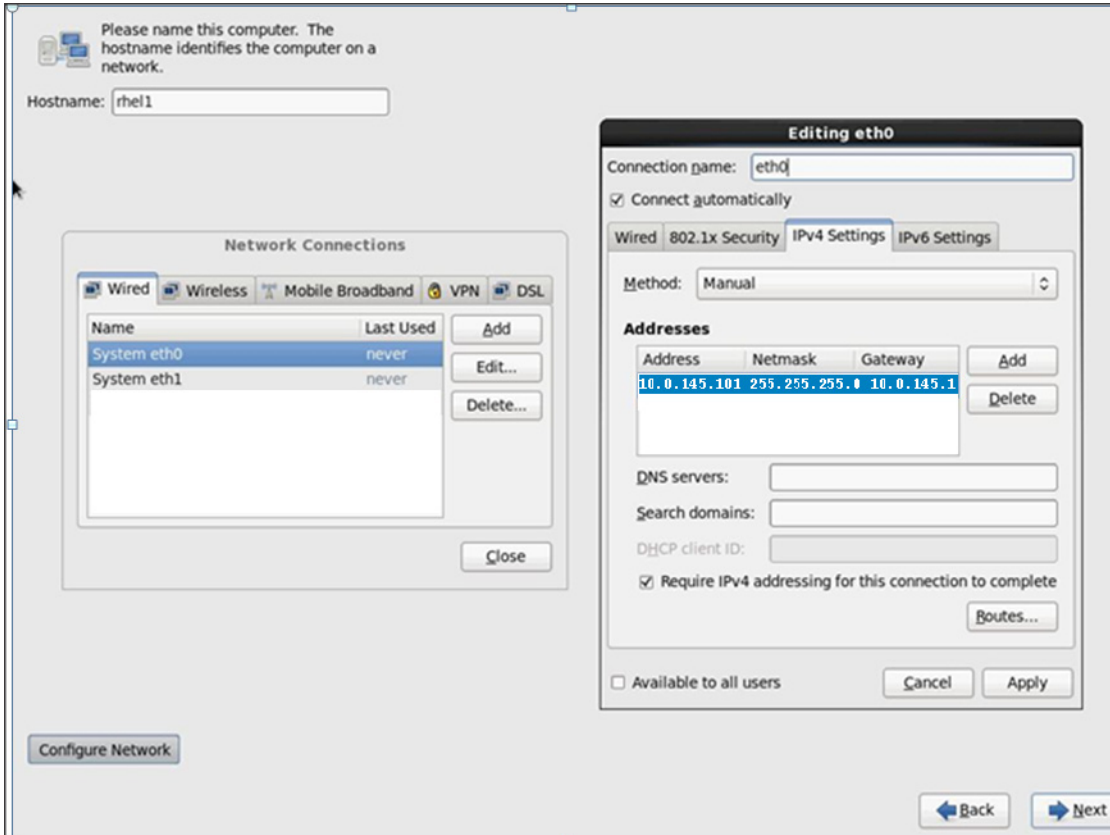
16. Enter the Host name of the server and Click **Next**.

**Figure 92**      *Selecting Fresh Installation*



17. Click **Configure Network**. The **Network Connections** window should then appear.
18. In the **Network Connections** window, select the **Wired** tab.
19. Select the interface System eth0 and click **Edit**.
20. Editing System eth0 appears
21. Check the check box **Connect** automatically.
22. In the drop down menu select **Manual Method**.
23. Click **Add** and enter IP Address, Netmask and the Gateway.
24. For this demonstration we use the following:
  - IP Address: 10 . 0 . 145 . 101
  - Netmask: 255 . 255 . 255 . 0
  - Gateway: 10 . 0 . 145 . 1
25. Add DNS servers (optional)
26. Click **Apply**

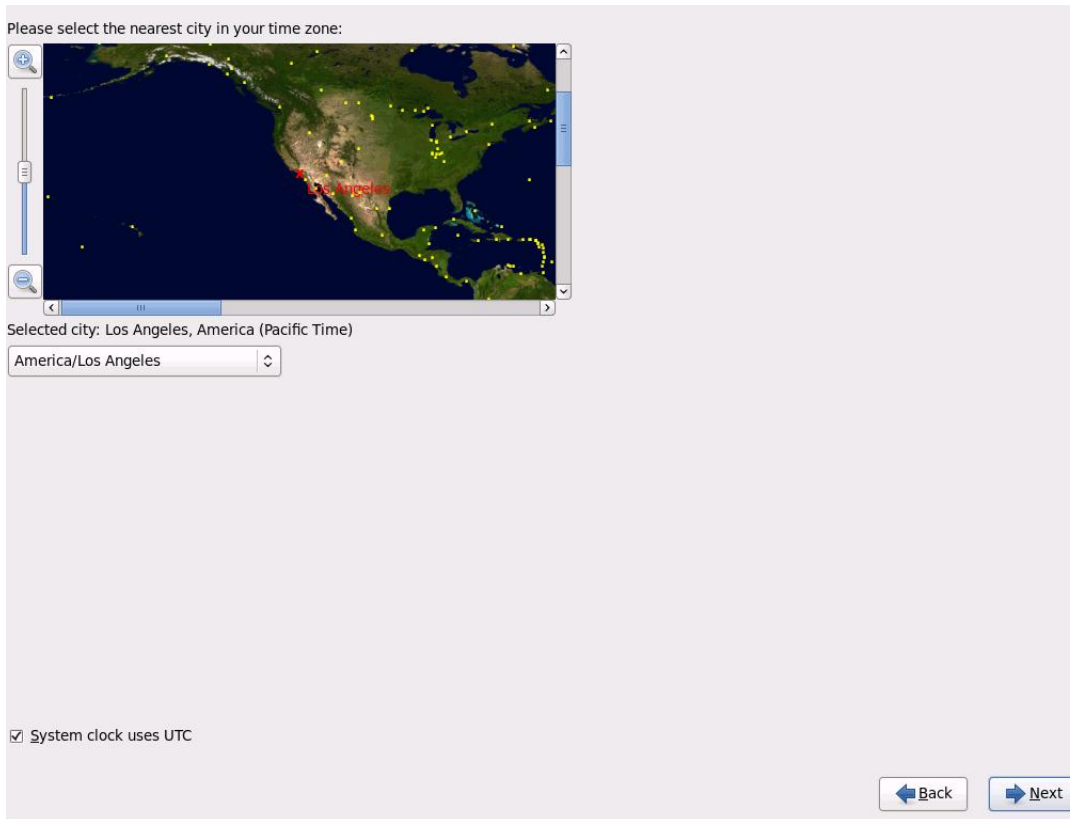
Figure 93 Configuring Network for eth0



**Note** Table 8 lists ip address of nodes in the cluster.

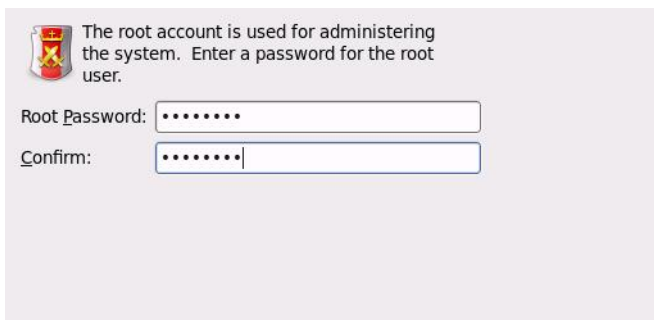
27. Select the appropriate Time Zone.

**Figure 94**      *Selecting Time Zone*



28. Enter the root Password and click **Next**.

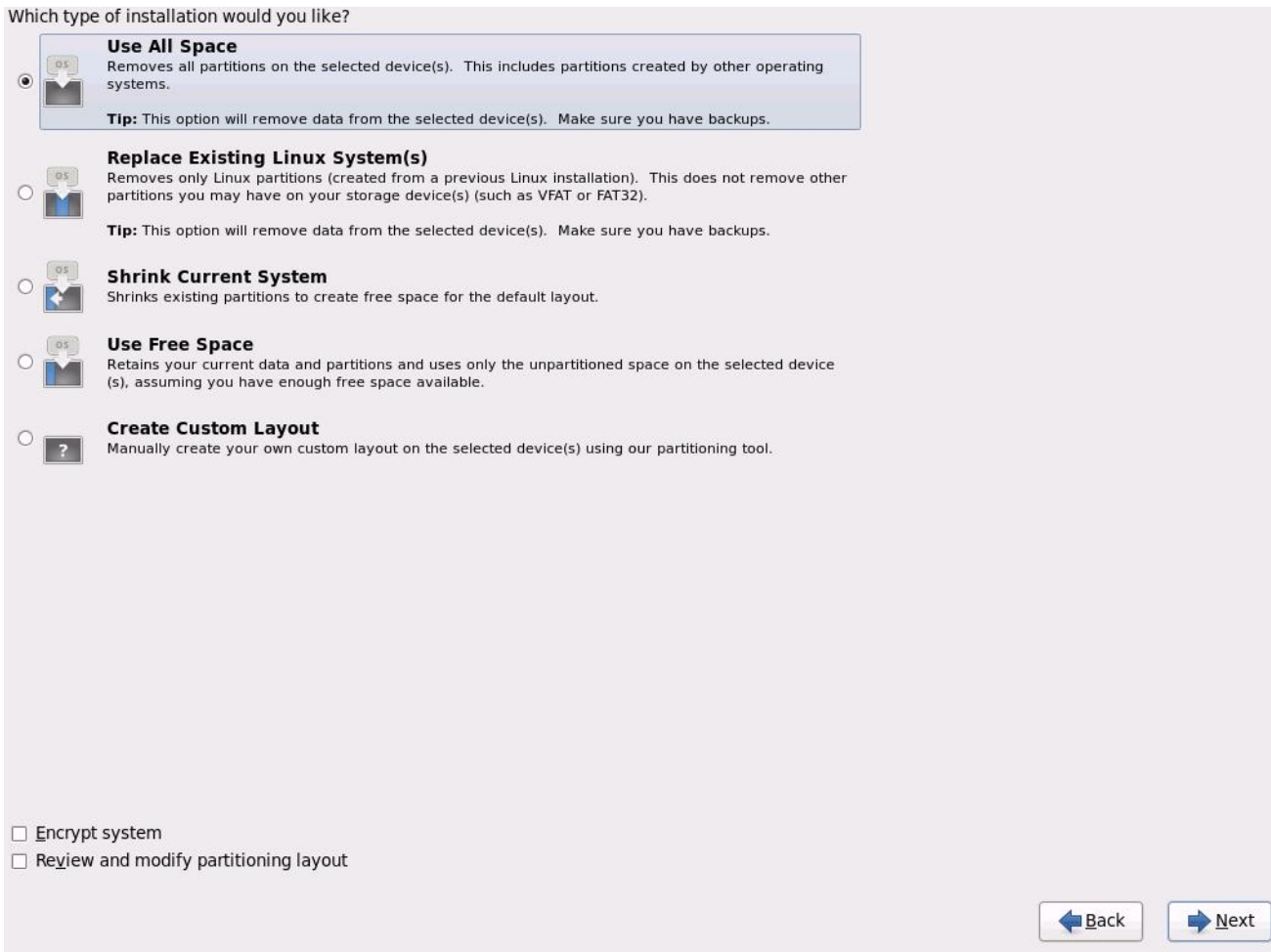
**Figure 95**      *Entering root Password*



29. Select **Use All Space** and click **Next**.

30. Choose an appropriate boot drive.

**Figure 96** *Selecting Install Options*



31. Click **Write Changes to disk** and click **Next**.

**Figure 97** *Confirming Formatting of Disk*



32. Select **Basic Server Installation** and click **Next**.

**Figure 98**      *Selecting type of Installation*

The default installation of Red Hat Enterprise Linux is a basic server install. You can optionally select a different set of software now.

- Basic Server
- Database Server
- Web Server
- Identity Management Server
- Virtualization Host
- Desktop
- Software Development Workstation
- Minimal

Please select any additional repositories that you want to use for software installation.

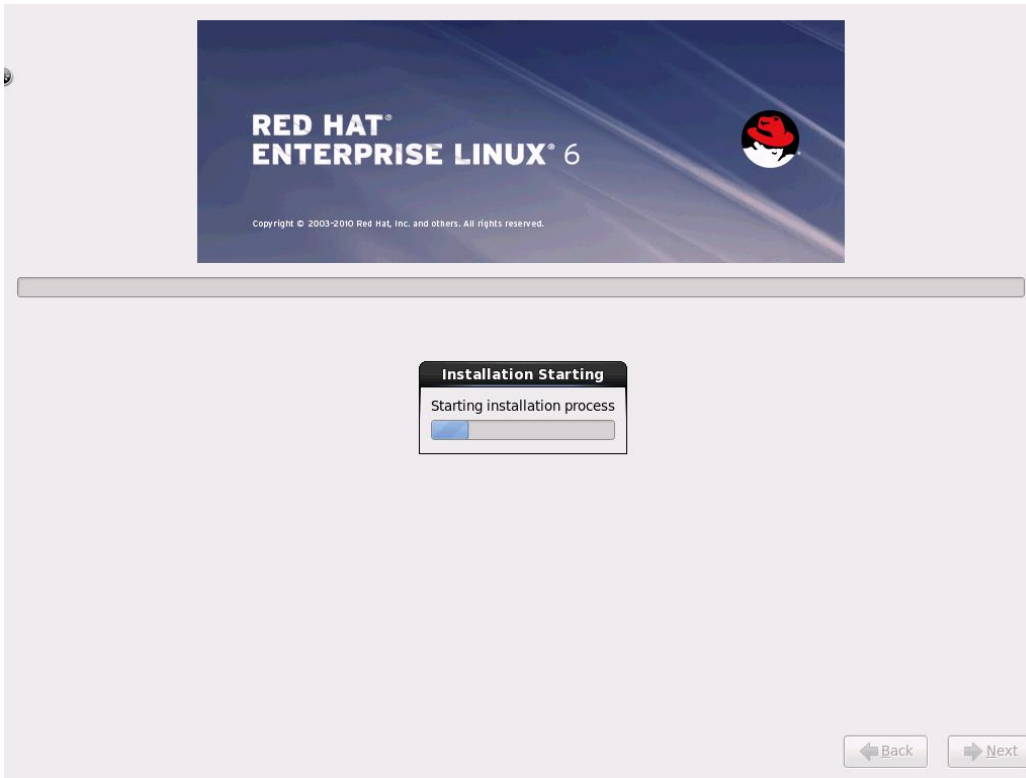
- High Availability
- Load Balancer
- Red Hat Enterprise Linux
- Resilient Storage

You can further customize the software selection now, or after install via the software management application.

Customize later     Customize now

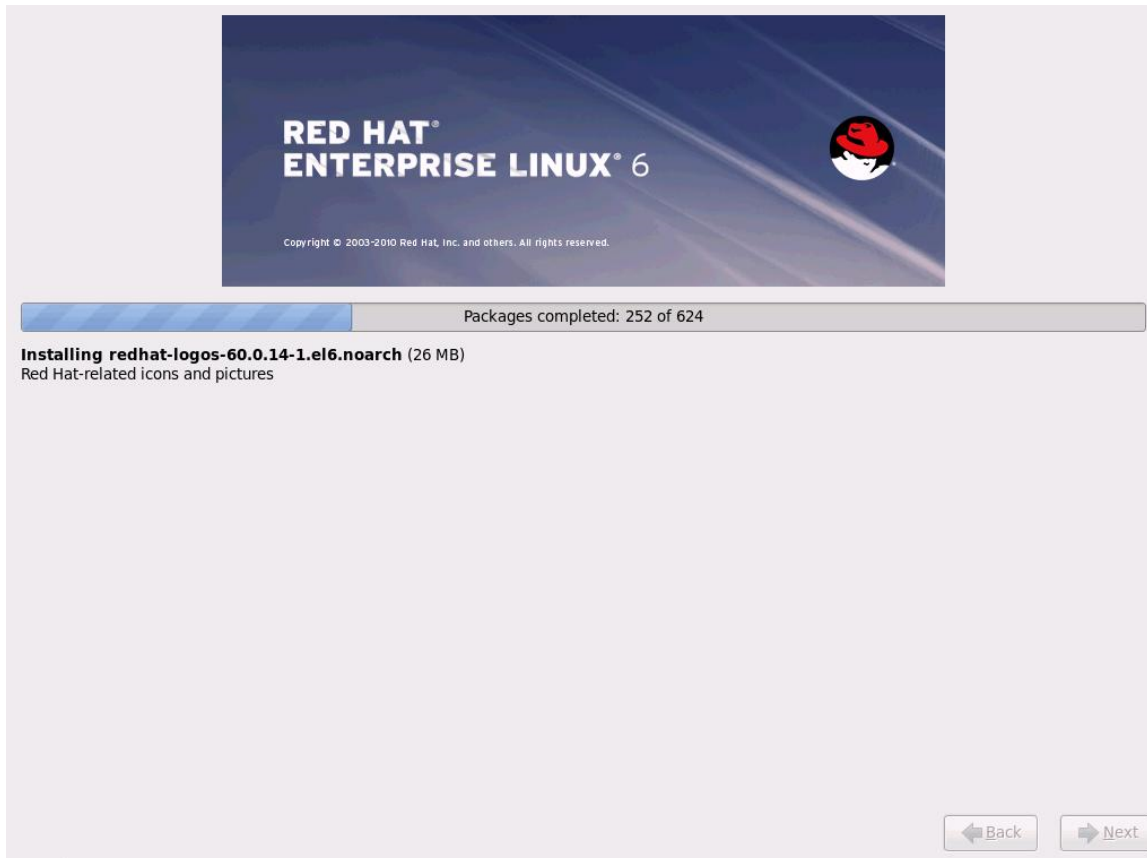
33. After the installer is finished loading, it will continue with the install as shown in the figures below:

*Figure 99 Starting Installation*





**Figure 100**      *Installation in Progress*



34. Once the installation is complete reboot the system.
35. Repeat steps (step 1 to 34) to install Red Hat Linux on Servers 2 through 45.



**Note**

---

The OS installation and configuration of the nodes that is mentioned above can be automated through PXE boot or third party tools.

---

## Server NIC Bonding

Perform the following steps for all the 45 server nodes to create the NIC Bonding

1. Use the following command to create a bonding configuration file and add the following content.

```
Vi /etc/modprobe.d/bonding.conf
```



**Figure 103**      *Configuring Policy Layer*

```

DEVICE=bond0
IPADDR="10.0.145.101"
NETMASK=255.255.255.0
BOOTPROTO=none
ONBOOT=yes
BONDING_OPTS="mode=4 miimon=1000 xmit_hash_policy=layer3+4"
~
~
~
~
~
~
~
~
~
~
~

```

3. Configure the connected Ethernet interface port that will be the part of this bond and add the following content.

```
vi /etc/sysconfig/network-scripts/ifcfg-eth0
```

content:

```

DEVICE= eth0
ONBOOT=yes
MASTER=bond0
SLAVE=yes
BOOTPROTO=none

```

**Figure 104**      *Configuring Ethernet Interface Port*

```

root@rhel1:~
e
~
e
~
e
~
e
~
e
~
~
~
~
~
~

```

```

DEVICE=eth0
ONBOOT=yes
MASTER=bond0
SLAVE=yes
BOOTPROTO=none
~
~
~
~
~
~
~
~

```

4. Repeat step 3 for other Ethernet interface

```
vi /etc/sysconfig/network-scripts/ifcfg-eth1
```

content:

```
DEVICE= eth1
```

```
ONBOOT=yes
MASTER=bond0
SLAVE=yes
BOOTPROTO=none
```

- Restart the network service

**Command:** Service network restart

**Table 0-1 List of Hosts and its Bond**

Host Name	Bond
rhel1	10.0.145.101
rhel2	10.0.145.102
rhel3	10.0.145.103
rhel4	10.0.145.104
rhel5	10.0.145.105
rhel6	10.0.145.106
rhel7	10.0.145.107
rhel8	10.0.145.108
rhel9	10.0.145.109
rhel10	10.0.145.110
rhel11	10.0.145.111
rhel12	10.0.145.112
rhel13	10.0.145.113
rhel14	10.0.145.114
rhel15	10.0.145.115
rhel16	10.0.145.116
.....	
Rhel145	10.0.145.145

## Post OS Install Configuration

Choose one of the nodes of the cluster as Admin Node for for management such as CDH installation, parallel shell, creating a local Red Hat repo and others. This CVD uses rhel1 for this purpose.

### Setting Up Password-less Login

To manage all of the clusters nodes from the admin node we need to setup password-less login. It assists in automating common tasks with Parallel-SSH (pssh) and shell-scripts without having to use passwords.

Once Red Hat Linux is installed across all the nodes in the cluster, follow the steps below in order to enable password less login across all the nodes.

- Login to the **Admin Node (rhel1)**

```
ssh 10.0.145.101
```

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

**Figure 105** *Creating Public and Private Keys*

```
[root@rhell ~]# ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/root/.ssh/id_rsa):
Created directory '/root/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /root/.ssh/id_rsa.
Your public key has been saved in /root/.ssh/id_rsa.pub.
The key fingerprint is:
ab:4e:78:10:54:81:4e:04:8d:af:4f:a4:b2:c4:bb:88 root@rhell
The key's randomart image is:
+--[ RSA 2048]-----+
|  .ooo. |
|  ..+   |
|  +.    |
|  +.    |
|  +.    |
|  +.    S |
```

3. Then 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_key`.

```
For IP in {101..145}; do echo -n "$IP -> "; ssh-copy-id -i ~/.ssh/id_rsa.pub
10.0.145.$IP; done
```

4. Enter yes for Are you sure you want to continue connecting (yes/no)?
5. Enter the password of the remote host.

## Installing and Configuring Parallel Shell

### Parallel-SSH

Parallel SSH is used to run commands on several hosts at the same time. It takes a file of hostnames and a bunch of common ssh parameters as parameters, executes the given command in parallel on the nodes specified.

1. From the system that is connected to the Internet, download `pssh`.

```
wget https://parallel-ssh.googlecode.com/files/pssh-2.3.1.tar.gz
```

**Figure 106** *Downloading pssh*

```
[root@redhat ~]# wget https://parallel-ssh.googlecode.com/files/pssh-2.3.1.tar.gz
--2013-04-24 05:39:42-- https://parallel-ssh.googlecode.com/files/pssh-2.3.1.tar.gz
Resolving parallel-ssh.googlecode.com... 74.125.129.82, 2607:f8b0:400e:c02::52
Connecting to parallel-ssh.googlecode.com|74.125.129.82|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 23427 (23K) [application/x-gzip]
Saving to: âpssh-2.3.1.tar.gz.1â

100%[=====]
2013-04-24 05:39:43 (240 KB/s) - âpssh-2.3.1.tar.gz.1â
```

```
scp pssh-2.3.1.tar.gz rhell:/root
```

2. Copy `pssh-2.3.1.tar.gz` to the **Admin Node**

```
ssh rhell
tar xzf pssh-2.3.1.tar.gz
cd pssh-2.3.1
python setup.py install
```

Figure 107 Configuring pssh-2.3.1

```

pssh-2.3.1.tar.gz
[root@redhat ~]# ssh rhell
root@rhell's password:
Last login: Wed Apr 24 09:06:38 2013 from 10.29.160.90
[root@rhell ~]# tar xzf pssh-2.3.1.tar.gz
[root@rhell ~]# cd pssh-2.3.1
[root@rhell pssh-2.3.1]# python setup.py install
running install
running build
running build_py
running build_scripts
running install_lib
running install_scripts
changing mode of /usr/bin/pslurp to 755
changing mode of /usr/bin/pnuke to 755
changing mode of /usr/bin/prsync to 755
changing mode of /usr/bin/pscp to 755
changing mode of /usr/bin/pssh-askpass to 755
changing mode of /usr/bin/pssh to 755
running install_data
running install_egg_info
Removing /usr/lib/python2.6/site-packages/pssh-2.3.1-py2.6.egg-info
Writing /usr/lib/python2.6/site-packages/pssh-2.3.1-py2.6.egg-info

```

3. Extract and Install pssh on the Admin node.
4. Create a host file containing the IP addresses of all the nodes and all the Datanodes in the cluster. This file is passed as a parameter to pssh to identify the nodes to run the commands on.

```

vi /root/allnodes
# This file contains ip address of all nodes of the cluster
#used by parallel-shell (pssh). For Details man pssh
10.0.145.101
10.0.145.102
10.0.145.103
10.0.145.104
10.0.145.105
10.0.145.106
10.0.145.107
10.0.145.108
10.0.145.109
10.0.145.110
10.0.145.111
10.0.145.112
10.0.145.113
10.0.145.114
10.0.145.115
10.0.145.116
...
10.0.145.145

vi /root/datanodes
10.0.145.103
10.0.145.104
10.0.145.105
10.0.145.106
10.0.145.107
10.0.145.108
10.0.145.109
10.0.145.110
10.0.145.111
10.0.145.112
10.0.145.113
10.0.145.114
10.0.145.115
10.0.145.116

```

```
...
10.0.145.145
```

## Cluster Shell

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

```
wget http://dl.fedoraproject.org/pub/epel//6/x86_64/clustershell-1.6-1.el6.noarch.rpm
scp clustershell-1.6-1.el6.noarch.rpm rhel1:/root/
Login to rhel1 and install cluster shell
```

```
yum install clustershell-1.6-1.el6.noarch.rpm
Edit /etc/clustershell/groups file to include hostnames for all the nodes of the cluster. For 45 node cluster
all: rhel[1-45]
```

## Configuring /etc/hosts

Follow the steps below to create the host file across all the nodes in the cluster:

1. Populate the host file with IP addresses and corresponding hostnames on the Admin node (rhel1).

```
vi /etc/hosts
127.0.0.1 localhost localhost.localdomain localhost4 localhost4.localdomain4
::1 localhost localhost.localdomain localhost6 localhost6.localdomain6
10.0.145.101rhel1
10.0.145.102rhel2
10.0.145.103rhel3
10.0.145.104rhel4
10.0.145.105rhel5
10.0.145.106rhel6
10.0.145.107rhel7
10.0.145.108rhel8
10.0.145.109rhel9
10.0.145.110rhel10
10.0.145.111rhel11
10.0.145.112rhel12
10.0.145.113rhel13
10.0.145.114rhel14
10.0.145.115rhel15
10.0.145.116rhel16
. . . . .
10.0.145.145rhel45
```

2. Deploy /etc/hosts from the admin node (rhel1) to all the nodes via the following pscp command:

```
pscp -h /root/allnodes /etc/hosts /etc/hosts.
```

Figure 108 Deploy ETC Hosts

```
[root@rhel1 ~]# pscp -h /root/allnodes /etc/hosts
/etc/hosts
[1] 17:26:13 [SUCCESS] 10.0.145.101
[2] 17:26:13 [SUCCESS] 10.0.145.104
[3] 17:26:13 [SUCCESS] 10.0.145.103
[4] 17:26:13 [SUCCESS] 10.0.145.109
[5] 17:26:13 [SUCCESS] 10.0.145.107
[6] 17:26:13 [SUCCESS] 10.0.145.111
[7] 17:26:13 [SUCCESS] 10.0.145.108
[8] 17:26:13 [SUCCESS] 10.0.145.102
[9] 17:26:13 [SUCCESS] 10.0.145.117
[10] 17:26:13 [SUCCESS] 10.0.145.112
[11] 17:26:13 [SUCCESS] 10.0.145.121
[12] 17:26:13 [SUCCESS] 10.0.145.110
[13] 17:26:13 [SUCCESS] 10.0.145.105
[14] 17:26:13 [SUCCESS] 10.0.145.106
[15] 17:26:13 [SUCCESS] 10.0.145.113
....
[45] 17:26:13 [SUCCESS] 10.0.145.131
```

## Creating Redhat Local Repo

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

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

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

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

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

```
scp rhel-server-6.4-x86_64-dvd.iso rhel1:/root
```

Here we assume you have the Red Hat ISO file located in your present working directory.

```
mkdir -p /mnt/rheliso
```

```
mount -t iso9660 -o loop /root/rhel-server-6.4-x86_64-dvd.iso /mnt/rheliso/
```

4. Next, copy the contents of the ISO to the /var/www/html/rhelrepo directory

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

5. Now on rhel1 create a .repo file to enable the use of the yum command.

```
vi /var/www/html/rhelrepo/rheliso.repo
[rhel6.4]
name=Red Hat Enterprise Linux 6.4
baseurl=http://10.0.145.101/rhelrepo
gpgcheck=0
enabled=1
```



### Note

Based on this repo file yum requires httpd to be running on rhel1 for other nodes to access the repository. Steps to install and configure httpd are in the following section.

6. Copy the rheliso.repo to all the nodes of the cluster.

```
pscp -h /root/allnodes /var/www/html/rhelrepo/rheliso.repo /etc/yum.repos.d/
```



Figure 109 Copying Repo Files

```
[root@rhell ~]# pscp -h /root/allnodes /var/www/html/rhelrepo/rheliso.repo
/etc/yum.repos.d/
[1] 20:01:24 [SUCCESS] 10.0.145.101
[2] 20:01:24 [SUCCESS] 10.0.145.103
[3] 20:01:24 [SUCCESS] 10.0.145.105
[4] 20:01:24 [SUCCESS] 10.0.145.104
[5] 20:01:24 [SUCCESS] 10.0.145.108
[6] 20:01:24 [SUCCESS] 10.0.145.102
[7] 20:01:24 [SUCCESS] 10.0.145.107
[8] 20:01:24 [SUCCESS] 10.0.145.111
[9] 20:01:24 [SUCCESS] 10.0.145.106
[10] 20:01:24 [SUCCESS] 10.0.145.113
[11] 20:01:24 [SUCCESS] 10.0.145.110
[12] 20:01:24 [SUCCESS] 10.0.145.116
[13] 20:01:24 [SUCCESS] 10.0.145.114
[14] 20:01:24 [SUCCESS] 10.0.145.121
[15] 20:01:24 [SUCCESS] 10.0.145.115
[16] 20:01:24 [SUCCESS] 10.0.145.119
...
[45] 20:01:25 [SUCCESS] 10.0.145.145
```

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

```
vi /etc/yum.repos.d/rheliso.repo
[rhel6.4]
name=Red Hat Enterprise Linux 6.4
baseurl=file:///var/www/html/rhelrepo
gpgcheck=0
enabled=1
```

- pssh -h /root/allnodes "yum clean all"

Figure 110 Creating repository of files

```
[root@rhell ~]# pssh -h /root/allnodes "yum clean all"
[1] 17:09:12 [SUCCESS] 10.0.145.102
[2] 17:09:12 [SUCCESS] 10.0.145.110
[3] 17:09:12 [SUCCESS] 10.0.145.107
[4] 17:09:12 [SUCCESS] 10.0.145.105
[5] 17:09:12 [SUCCESS] 10.0.145.101
[6] 17:09:12 [SUCCESS] 10.0.145.109
[7] 17:09:12 [SUCCESS] 10.0.145.111
[8] 17:09:12 [SUCCESS] 10.0.145.117
[9] 17:09:12 [SUCCESS] 10.0.145.104
[10] 17:09:12 [SUCCESS] 10.0.145.116
[11] 17:09:12 [SUCCESS] 10.0.145.129
[12] 17:09:12 [SUCCESS] 10.0.145.106
[13] 17:09:12 [SUCCESS] 10.0.145.103
[14] 17:09:12 [SUCCESS] 10.0.145.121
[15] 17:09:12 [SUCCESS] 10.0.145.115
[16] 17:09:12 [SUCCESS] 10.0.145.119
.....
[45] 17:09:12 [SUCCESS] 10.0.145.145
```

- Creating the Red Hat Repository Database.

Install the createrepo package. Use it to regenerate the repository database(s) for the local copy of the RHEL DVD contents. Then purge the yum caches.

```
yum -y install createrepo
cd /var/www/html/rhelrepo
createrepo .
yum clean all
```

**Figure 111** *Creating Red Hat Repository Database*

```

root@rhell1 rhelrepo]# createrepo .
368/3596 - Packages/pygobject2-doc-2.20.0-5.el6.x86_64.rpm
so-8859-1 encoding on Ville Skyttä <ville.skytta@iki.fi> - 2.8.2-2

596/3596 - Packages/lohit-bengali-fonts-2.4.3-6.el6.noarch.rpm
aving Primary metadata
aving file lists metadata
aving other metadata

```

## Upgrading LSI driver

The latest LSI driver is required for performance and bug fixes. The latest drivers can be downloaded from the link below:

<http://software.cisco.com/download/release.html?mdfid=284296254&flowid=31743&softwareid=283853158&release=1.5.1&relind=AVAILABLE&rellifecycle=&reltype=latest>

In the ISO image, the required driver `kmod-megaraid_sas-06.602.03.00_rhel6.4-2.x86_64.rpm` can be located at `ucs-cxxx-drivers.1.5.1\Linux\Storage\LSI\92xx\RHEL\RHEL6.4`

From a node connected to the Internet, download and transfer `kmod-megaraid_sas-06.602.03.00_rhel6.4-2.x86_64.rpm`

to `rhell` (admin node). Install the rpm on all nodes of the cluster using the following pssh commands. For this example the rpm is assumed to be in present working directory of `rhell`.

```
ssh rpscp -h /root/allnodes kmod-megaraid_sas-06.602.03.00_rhel6.4-2.x86_64.rpm /root/
```

**Figure 112** *Installing rpm on all Nodes*

```

[root@rhell1 ~]# pscp -h /root/allnodes kmod-megaraid_sas-06.602.03.00_rhel6
.4-2.x86_64.rpm /root/
[1] 17:26:13 [SUCCESS] 10.0.145.101
[2] 17:26:13 [SUCCESS] 10.0.145.104
[3] 17:26:13 [SUCCESS] 10.0.145.103
[4] 17:26:13 [SUCCESS] 10.0.145.109
[5] 17:26:13 [SUCCESS] 10.0.145.107
[6] 17:26:13 [SUCCESS] 10.0.145.111
[7] 17:26:13 [SUCCESS] 10.0.145.108
[8] 17:26:13 [SUCCESS] 10.0.145.102
[9] 17:26:13 [SUCCESS] 10.0.145.117
[10] 17:26:13 [SUCCESS] 10.0.145.112
[11] 17:26:13 [SUCCESS] 10.0.145.121
[12] 17:26:13 [SUCCESS] 10.0.145.110
[13] 17:26:13 [SUCCESS] 10.0.145.105
[14] 17:26:13 [SUCCESS] 10.0.145.106
[15] 17:26:13 [SUCCESS] 10.0.145.113
....
[45] 17:26:13 [SUCCESS] 10.0.145.145

```

```
pssh -h /root/allnodes "rpm -ivh kmod-megaraid_sas-06.602.03.00_rhel6.4-2.x86_64.rpm "
```

Figure 113 Command Output

```
[root@rhell1 ~]# pssh -h /root/allnodes "rpm
megaraid_sas-06.602.03.00_rhel6.4-2.x86_64.rp
[1] 17:26:13 [SUCCESS] 10.0.145.101
[2] 17:26:13 [SUCCESS] 10.0.145.104
[3] 17:26:13 [SUCCESS] 10.0.145.103
[4] 17:26:13 [SUCCESS] 10.0.145.109
[5] 17:26:13 [SUCCESS] 10.0.145.107
[6] 17:26:13 [SUCCESS] 10.0.145.111
[7] 17:26:13 [SUCCESS] 10.0.145.108
[8] 17:26:13 [SUCCESS] 10.0.145.102
[9] 17:26:13 [SUCCESS] 10.0.145.117
[10] 17:26:13 [SUCCESS] 10.0.145.112
[11] 17:26:13 [SUCCESS] 10.0.145.121
[12] 17:26:13 [SUCCESS] 10.0.145.110
[13] 17:26:13 [SUCCESS] 10.0.145.105
[14] 17:26:13 [SUCCESS] 10.0.145.106
[15] 17:26:13 [SUCCESS] 10.0.145.113
```

Ensure that the above installed version of `kmod-megaraid_sas` driver is being used on all nodes by running the command “`modinfo megaraid_sas`” on all nodes

```
pssh -h /root/allnodes "modinfo megaraid_sas | head -5"
```

Figure 114 Executing “`modinfo megaraid_sas`” Command

```
filename: /lib/modules/2.6.32-358.el6.x86_64/extra/megaraid_sas/megaraid_sas.ko
description: LSI MegaRAID SAS Driver
author: megaraidlinux@lsi.com
version: 06.602.03.00
license: GPL
```

## Installing httpd

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

The **Red Hat** repository is hosted using HTTP on the admin node, this machine is accessible by all the hosts in the cluster.

```
yum -y install httpd
```

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

```
/etc/httpd/conf/httpd.conf
```

*Figure 115 Adding ServerName 10.0.145.101:80*

```
#
# UseCanonicalName: Determines how Apache constructs self-referencing
# URLs and the SERVER_NAME and SERVER_PORT variables.
# When set "Off", Apache will use the Hostname and Port supplied
# by the client. When set "On", Apache will use the value of the
# ServerName directive.
#
UseCanonicalName Off
```

3. Ensure httpd is able to read the repofiles

```
chcon -R -t httpd_sys_content_t /var/www/html/rhelrepo
```

4. Start httpd

```
service httpd start
chkconfig httpd on
```

## Installing xfsprogs

Install `xfsprogs` on all the nodes for xfs filesystem.

```
pssh -h /root/allnodes "yum -y install xfsprogs"
```

*Figure 116 Installing Xfsprogs*

```
[root@rhell ~]# pssh -h /root/allnodes "yum -y
install xfsprogs"
[1] 17:26:13 [SUCCESS] 10.0.145.101
[2] 17:26:13 [SUCCESS] 10.0.145.104
[3] 17:26:13 [SUCCESS] 10.0.145.103
[4] 17:26:13 [SUCCESS] 10.0.145.109
[5] 17:26:13 [SUCCESS] 10.0.145.107
[6] 17:26:13 [SUCCESS] 10.0.145.111
[7] 17:26:13 [SUCCESS] 10.0.145.108
[8] 17:26:13 [SUCCESS] 10.0.145.102
[9] 17:26:13 [SUCCESS] 10.0.145.117
[10] 17:26:13 [SUCCESS] 10.0.145.112
[11] 17:26:13 [SUCCESS] 10.0.145.121
[12] 17:26:13 [SUCCESS] 10.0.145.110
[13] 17:26:13 [SUCCESS] 10.0.145.105
[14] 17:26:13 [SUCCESS] 10.0.145.106
[15] 17:26:13 [SUCCESS] 10.0.145.113
....
[45] 17:26:13 [SUCCESS] 10.0.145.145
```

## NTP Configuration

The Network Time Protocol (NTP) is used to synchronize the time of all the nodes within the cluster. The Network Time Protocol daemon (`ntpd`) sets and maintains the system time of day in synchronism with the timeserver located in the admin node (`rhell`). 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.

**Note**

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

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

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

Create `/root/ntp.conf` on the admin node (rhell) and copy it to all nodes

```
vi /root/ntp.conf
server 10.0.145.101
driftfile /var/lib/ntp/drift
restrict 127.0.0.1
restrict -6 ::1
includefile /etc/ntp/crypto/pw
keys /etc/ntp/keys
```

**Figure 117** Copy Configuration to all Nodes

```
[root@rhell ~]# for SERVER in {101..145}; do scp /root/ntp.conf 10.0.145.$SERVER:/etc/ntp.conf; done
ntp.conf 100% 142 0.1KB/s 00:00
ntp.conf 100% 142 0.1KB/s 00:00
ntp.conf 100% 142 0.1KB/s 00:00
ntp.conf 100% 142 0.1KB/s 00:00
ntp.conf 100% 142 0.1KB/s 00:00
ntp.conf 100% 142 0.1KB/s 00:00
ntp.conf 100% 142 0.1KB/s 00:00
ntp.conf 100% 142 0.1KB/s 00:00
ntp.conf 100% 142 0.1KB/s 00:00
ntp.conf 100% 142 0.1KB/s 00:00
ntp.conf 100% 142 0.1KB/s 00:00
ntp.conf 100% 142 0.1KB/s 00:00
ntp.conf 100% 142 0.1KB/s 00:00
ntp.conf 100% 142 0.1KB/s 00:00
ntp.conf 100% 142 0.1KB/s 00:00
ntp.conf 100% 142 0.1KB/s 00:00
```

Copy `ntp.conf` file from the admin node to `/etc` of all the nodes by executing the following command in the admin node (rhell)

```
for SERVER in {101..145}; do scp /root/ntp.conf 10.0.145.$SERVER:/etc/ntp.conf; done
```

**Note**

Do not use `pssh /root/allnodes` command without editing the host file `allnodes` as it overwrites `/etc/ntp.conf` from the admin node.

Synchronize the time and restart NTP daemon on all nodes

```
pssh -h /root/allnodes "service ntpd stop"
pssh -h /root/allnodes "ntpdate rhell"
pssh -h /root/allnodes "service ntpd start"
```

*Figure 118 Synchronize Time and Restart NTP Daemon*

```
[root@rhell ~]# pssh -h /root/allnodes "service
ntpd restart"
[1] 17:26:13 [SUCCESS] 10.0.145.101
[2] 17:26:13 [SUCCESS] 10.0.145.104
[3] 17:26:13 [SUCCESS] 10.0.145.103
[4] 17:26:13 [SUCCESS] 10.0.145.109
[5] 17:26:13 [SUCCESS] 10.0.145.107
[6] 17:26:13 [SUCCESS] 10.0.145.111
[7] 17:26:13 [SUCCESS] 10.0.145.108
[8] 17:26:13 [SUCCESS] 10.0.145.102
[9] 17:26:13 [SUCCESS] 10.0.145.117
[10] 17:26:13 [SUCCESS] 10.0.145.112
[11] 17:26:13 [SUCCESS] 10.0.145.121
[12] 17:26:13 [SUCCESS] 10.0.145.110
[13] 17:26:13 [SUCCESS] 10.0.145.105
[14] 17:26:13 [SUCCESS] 10.0.145.106
[15] 17:26:13 [SUCCESS] 10.0.145.113
....
[45] 17:26:13 [SUCCESS] 10.0.145.145
```

Ensure restart of NTP daemon across reboots

```
pssh -h /root/allnodes "chkconfig ntpd on"
```

*Figure 119 Restarting NTP Daemon*

```
[root@rhell ~]# pssh -h /root/allnodes "chkconfig ntpd on"
[1] 17:26:13 [SUCCESS] 10.0.145.101
[2] 17:26:13 [SUCCESS] 10.0.145.104
[3] 17:26:13 [SUCCESS] 10.0.145.103
[4] 17:26:13 [SUCCESS] 10.0.145.109
[5] 17:26:13 [SUCCESS] 10.0.145.107
[6] 17:26:13 [SUCCESS] 10.0.145.111
[7] 17:26:13 [SUCCESS] 10.0.145.108
[8] 17:26:13 [SUCCESS] 10.0.145.102
[9] 17:26:13 [SUCCESS] 10.0.145.117
[10] 17:26:13 [SUCCESS] 10.0.145.112
[11] 17:26:13 [SUCCESS] 10.0.145.121
[12] 17:26:13 [SUCCESS] 10.0.145.110
[13] 17:26:13 [SUCCESS] 10.0.145.105
[14] 17:26:13 [SUCCESS] 10.0.145.106
[15] 17:26:13 [SUCCESS] 10.0.145.113
....
[45] 17:26:13 [SUCCESS] 10.0.145.145
```

## Enabling Syslog

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

```
clush -B -a rsyslogd -v
clush -B -a service rsyslog status
```

## Setting ulimit

On each node, `ulimit -n` specifies the number of inodes that can be opened simultaneously. With the default value of 1024, the system appears to be out of disk space and shows no inodes available. This value should be set to 64000 on every node.

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

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

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

Verify the `ulimit` setting with the following steps:



### Note

`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

Run the following command at a command line. The command should report 64000.

```
clush -B -a ulimit -n
```

## Disabling SELinux

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

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

```
pssh -h /root/allnodes "sed -i 's/SELINUX=enforcing/SELINUX=disabled/g'
/etc/selinux/config "
```

*Figure 120 Disabling SELinux on all Nodes*

```
[root@rhell ~]# pssh -h /root/allnodes "sed -i
's/enforcing/disabled/g' /etc/selinux/config"
[1] 17:26:13 [SUCCESS] 10.0.145.101
[2] 17:26:13 [SUCCESS] 10.0.145.104
[3] 17:26:13 [SUCCESS] 10.0.145.103
[4] 17:26:13 [SUCCESS] 10.0.145.109
[5] 17:26:13 [SUCCESS] 10.0.145.107
[6] 17:26:13 [SUCCESS] 10.0.145.111
[7] 17:26:13 [SUCCESS] 10.0.145.108
[8] 17:26:13 [SUCCESS] 10.0.145.102
[9] 17:26:13 [SUCCESS] 10.0.145.117
[10] 17:26:13 [SUCCESS] 10.0.145.112
[11] 17:26:13 [SUCCESS] 10.0.145.121
[12] 17:26:13 [SUCCESS] 10.0.145.110
[13] 17:26:13 [SUCCESS] 10.0.145.105
[14] 17:26:13 [SUCCESS] 10.0.145.106
[15] 17:26:13 [SUCCESS] 10.0.145.113
....
[45] 17:26:13 [SUCCESS] 10.0.145.145
```

```
pssh -h /root/allnodes "setenforce 0"
```

**Note**


---

The above command may fail if SELinux is already disabled

---

## Set TCP Retries

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

1. Edit the file `/etc/sysctl.conf` and add the following line:

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

2. Save the file and run:

```
clush -B -a sysctl -p
```

## Disabling the Linux Firewall

The default Linux firewall settings are far too restrictive for any Hadoop deployment. Since the UCS Big Data deployment will be in its own isolated network, there's no need to leave the iptables service running.

```
pssh -h /root/allnodes "service iptables stop"
```

*Figure 121 Command Output*

```
[root@rhell ~]# pssh -h /root/allnodes "service iptables
stop"
[1] 17:26:13 [SUCCESS] 10.0.145.101
[2] 17:26:13 [SUCCESS] 10.0.145.104
[3] 17:26:13 [SUCCESS] 10.0.145.103
[4] 17:26:13 [SUCCESS] 10.0.145.109
[5] 17:26:13 [SUCCESS] 10.0.145.107
[6] 17:26:13 [SUCCESS] 10.0.145.111
[7] 17:26:13 [SUCCESS] 10.0.145.108
[8] 17:26:13 [SUCCESS] 10.0.145.102
[9] 17:26:13 [SUCCESS] 10.0.145.117
[10] 17:26:13 [SUCCESS] 10.0.145.112
[11] 17:26:13 [SUCCESS] 10.0.145.121
[12] 17:26:13 [SUCCESS] 10.0.145.110
[13] 17:26:13 [SUCCESS] 10.0.145.105
[14] 17:26:13 [SUCCESS] 10.0.145.106
[15] 17:26:13 [SUCCESS] 10.0.145.113
....
[45] 17:26:13 [SUCCESS] 10.0.145.145
```

```
pssh -h /root/allnodes "chkconfig iptables off"
```



Figure 122 Command Output

```
[root@rhell ~]# pssh -h /root/allnodes "chkconfig iptables
off"
[1] 17:26:13 [SUCCESS] 10.0.145.101
[2] 17:26:13 [SUCCESS] 10.0.145.104
[3] 17:26:13 [SUCCESS] 10.0.145.103
[4] 17:26:13 [SUCCESS] 10.0.145.109
[5] 17:26:13 [SUCCESS] 10.0.145.107
[6] 17:26:13 [SUCCESS] 10.0.145.111
[7] 17:26:13 [SUCCESS] 10.0.145.108
[8] 17:26:13 [SUCCESS] 10.0.145.102
[9] 17:26:13 [SUCCESS] 10.0.145.117
[10] 17:26:13 [SUCCESS] 10.0.145.112
[11] 17:26:13 [SUCCESS] 10.0.145.121
[12] 17:26:13 [SUCCESS] 10.0.145.110
[13] 17:26:13 [SUCCESS] 10.0.145.105
[14] 17:26:13 [SUCCESS] 10.0.145.106
[15] 17:26:13 [SUCCESS] 10.0.145.113
....
[45] 17:26:13 [SUCCESS] 10.0.145.145
```

## Configuring Data Drives on NameNode

In the section on Configuring Disk Drives for OS on Namenodes describes the steps to configure the first two disk drives for the operating system for nodes rhell and rhel2. Remaining disk drives can also be configured Raid 1 similarly or by using MegaCli as described below.

1. From the LSI website  
<http://www.lsi.com/support/Pages/Download-Results.aspx?keyword=9271-8i>
2. Download MegaCli and its dependencies and transfer to Admin node.
  - scp /root/MegaCli64 rhell:/root/
  - scp /root/Lib\_Utills-1.00-08.noarch.rpm rhell:/root/
  - scp /root/Lib\_Utills2-1.00-01.noarch.rpm rhell:/root/
3. Copy all three files to all the nodes using the following commands:
  - pscp -h /root/allnodes /root/MegaCli64 /root/

Figure 123 Copying Three Files to the Nodes

```
[root@rhell ~]# pscp -h /root/allnodes /root/MegaCli64
/root/
[1] 17:26:13 [SUCCESS] 10.0.145.101
[2] 17:26:13 [SUCCESS] 10.0.145.104
[3] 17:26:13 [SUCCESS] 10.0.145.103
[4] 17:26:13 [SUCCESS] 10.0.145.109
[5] 17:26:13 [SUCCESS] 10.0.145.107
[6] 17:26:13 [SUCCESS] 10.0.145.111
[7] 17:26:13 [SUCCESS] 10.0.145.108
[8] 17:26:13 [SUCCESS] 10.0.145.102
[9] 17:26:13 [SUCCESS] 10.0.145.117
[10] 17:26:13 [SUCCESS] 10.0.145.112
[11] 17:26:13 [SUCCESS] 10.0.145.121
[12] 17:26:13 [SUCCESS] 10.0.145.110
[13] 17:26:13 [SUCCESS] 10.0.145.105
[14] 17:26:13 [SUCCESS] 10.0.145.106
[15] 17:26:13 [SUCCESS] 10.0.145.113
....
[45] 17:26:13 [SUCCESS] 10.0.145.145
```

```
pscp -h /root/allnodes /root/Lib_Utills* /root/
```

Figure 124 Copying Files

```
[root@rhell ~]# pscp -h /root/allnodes /root/Lib_Utills*
/root/
[1] 17:26:13 [SUCCESS] 10.0.145.101
[2] 17:26:13 [SUCCESS] 10.0.145.104
[3] 17:26:13 [SUCCESS] 10.0.145.103
[4] 17:26:13 [SUCCESS] 10.0.145.109
[5] 17:26:13 [SUCCESS] 10.0.145.107
[6] 17:26:13 [SUCCESS] 10.0.145.111
[7] 17:26:13 [SUCCESS] 10.0.145.108
[8] 17:26:13 [SUCCESS] 10.0.145.102
[9] 17:26:13 [SUCCESS] 10.0.145.117
[10] 17:26:13 [SUCCESS] 10.0.145.112
[11] 17:26:13 [SUCCESS] 10.0.145.121
[12] 17:26:13 [SUCCESS] 10.0.145.110
[13] 17:26:13 [SUCCESS] 10.0.145.105
[14] 17:26:13 [SUCCESS] 10.0.145.106
[15] 17:26:13 [SUCCESS] 10.0.145.113
....
[45] 17:26:13 [SUCCESS] 10.0.145.145
```

4. Run the following command to install the rpms on all the nodes:

```
pssh -h /root/allnodes "rpm -ivh Lib_Utills*"
```

Figure 125 Installing RPMs on Nodes

```

Lib_Utils*"
[1] 16:03:19 [SUCCESS] 10.0.145.101
[2] 16:03:19 [SUCCESS] 10.0.145.104
[3] 16:03:19 [SUCCESS] 10.0.145.103
[4] 16:03:19 [SUCCESS] 10.0.145.109
[5] 16:03:19 [SUCCESS] 10.0.145.107
[6] 16:03:19 [SUCCESS] 10.0.145.111
[7] 16:03:19 [SUCCESS] 10.0.145.108
[8] 16:03:19 [SUCCESS] 10.0.145.102
[9] 16:03:19 [SUCCESS] 10.0.145.117
[10] 16:03:19 [SUCCESS] 10.0.145.112
[11] 16:03:19 [SUCCESS] 10.0.145.121
[12] 16:03:19 [SUCCESS] 10.0.145.110
[13] 16:03:19 [SUCCESS] 10.0.145.105
[14] 16:03:19 [SUCCESS] 10.0.145.106
[15] 16:03:19 [SUCCESS] 10.0.145.113
....
[45] 16:03:19 [SUCCESS] 10.0.145.145

```

5. Run the following script as root user on NameNode and Secondary NameNode to create the virtual drives.

```

vi /root/raid1.sh
./MegaCli64 -cfgldadd
r1[$1:3,$1:4,$1:5,$1:6,$1:7,$1:8,$1:9,$1:10,$1:11,$1:12,$1:13,$1:14,$1:15,$1:16,$1:17,
$1:18,$1:19,$1:20,$1:21,$1:22,$1:23,$1:24] wb ra nocachedbadbbu strpsz1024 -a0

```

The above script requires enclosure ID as a parameter. Run the following command to get enclosure id.

```

./MegaCli64 pdlist -a0 | grep Enc | grep -v 252 | awk '{print $4}' | sort | uniq -c |
awk '{print $2}'
chmod 755 raid1.sh

```

Run MegaCli script as follows

```

./raid1.sh <EnclosureID> <obtained by running the command above>

```

WB: Write back

RA: Read Ahead

NoCachedBadBBU: Do not write cache when the BBU is bad.

Strpsz1024: Strip Size of 1024K



**Note**

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

## Configuring the Filesystem for NameNodes

```

vi /root/driveconfnn.sh
#!/bin/bash
#disks_count=`lsblk -id | grep sd | wc -l`
#if [ $disks_count -eq 2 ]; then
#   echo "Found 2 disks"
#else
#   echo "Found $disks_count disks. Expecting 2. Exiting.."
#   exit 1
#fi
[[ "-x" == "${1}" ]] && set -x && set -v && shift 1
for X in /sys/class/scsi_host/host*/scan

```

```
do
echo '- - -' > ${X}
done
for X in /dev/sd?
do
echo $X
if [[ -b ${X} && `sbin/parted -s ${X} print quit|bin/grep -c boot` -ne 0 ]]
then
echo "$X bootable - skipping."
continue
else
Y=${X##*/}1
/sbin/parted -s ${X} mklabel gpt quit
/sbin/parted -s ${X} mkpart 1 6144s 100% quit
/sbin/mkfs.xfs -f -q -l size=65536b,lazy-count=1,su=256k -d sunit=1024,swidth=6144 -r
extsize=256k -L ${Y} ${X}1
(( $? )) && continue
/bin/mkdir -p /DATA/${Y}
(( $? )) && continue
/bin/mount -t xfs -o allocsize=128m,noatime,nobarrier,nodiratime ${X}1 /DATA/${Y}
(( $? )) && continue
echo "LABEL=${Y} /DATA/${Y} xfs allocsize=128m,noatime,nobarrier,nodiratime 0 0" >>
/etc/fstab
fi
done
```

## Configuring Data Drives on Data Nodes

In the section on Configuring Disk Drives for OS on Datanodes describes the steps to configure the first disk drive for the operating system for nodes rhel3 to rhel45. Remaining disk drives can also be configured similarly or using MegaCli as described below.

Issue the following command from the admin node to create the virtual drives with RAID 0 configurations on all the datanodes.

```
pssh -h /root/datanodes "./MegaCli64 -cfgeachdiskraid0 WB RA direct NoCachedBadBBU
strpsz1024 -a0"
WB: Write back
RA: Read Ahead
NoCachedBadBBU: Do not write cache when the BBU is bad.
Strpsz1024: Strip Size of 1024K
```



**Note**

- The command above will not override existing configurations. To clear and reconfigure existing configurations refer to Embedded MegaRAID Software Users Guide available at [www.lsi.com](http://www.lsi.com).
- Ensure all drives are up by running the following command.

```
./MegaCli64 -PDList -aAll |grep -i "Firmware state"
```

Figure 126 Configuring Data Drives

```
[root@rhell ~]# ./MegaCli64 -PDList -aAll |grep -i "Firmware state"
Firmware state: Online, Spun Up
Firmware state: Online, Spun Up
Firmware state: Online, Spun Up
Firmware state: Online, Spun Up
Firmware state: Online, Spun Up
Firmware state: Online, Spun Up
Firmware state: Online, Spun Up
Firmware state: Online, Spun Up
Firmware state: Online, Spun Up
Firmware state: Online, Spun Up
Firmware state: Online, Spun Up
Firmware state: Online, Spun Up
Firmware state: Online, Spun Up
Firmware state: Online, Spun Up
```

## Configuring the Filesystem for Data Nodes

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

To 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.

```
vi /root/driveconf.sh
#!/bin/bash
#disks_count=`lsblk -id | grep sd | wc -l`
#if [ $disks_count -eq 24 ]; then
#  echo "Found 24 disks"
#else
#  echo "Found $disks_count disks. Expecting 24. Exiting.."
#  exit 1
#fi
[[ "-x" == "${1}" ]] && set -x && set -v && shift 1
for X in /sys/class/scsi_host/host*/scan
do
echo '- - -' > ${X}
done
for X in /dev/sd?
do
echo $X
if [[ -b ${X} && `sbin/parted -s ${X} print quit|/bin/grep -c boot` -ne 0 ]]
then
echo "$X bootable - skipping."
continue
else
Y=${X##*/}1
/sbin/parted -s ${X} mklabel gpt quit
/sbin/parted -s ${X} mkpart 1 6144s 100% quit
/sbin/mkfs.xfs -f -q -l size=65536b,lazy-count=1,su=256k -d sunit=1024,swidth=6144 -r
extsize=256k -L ${Y} ${X}1
(( $? )) && continue
/bin/mkdir -p /DATA/${Y}
(( $? )) && continue
/bin/mount -t xfs -o allocsize=128m,noatime,nobarrier,nodiratime ${X}1 /DATA/${Y}
(( $? )) && continue
echo "LABEL=${Y} /DATA/${Y} xfs allocsize=128m,noatime,nobarrier,nodiratime 0 0" >>
/etc/fstab
fi
done
```

2. Run the following command to copy driveconf.sh to all the datanodes

```
pscp -h /root/datanodes /root/driveconf.sh /root/
```

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

```
pssh -h /root/datanodes "./driveconf.sh"
```

Figure 127 Command Output for Executing Script Across Data Nodes

```
[root@rhell ~]# pssh -h /root/datanodes `.`
[1] 17:26:13 [SUCCESS] 10.0.145.101
[2] 17:26:13 [SUCCESS] 10.0.145.104
[3] 17:26:13 [SUCCESS] 10.0.145.103
[4] 17:26:13 [SUCCESS] 10.0.145.109
[5] 17:26:13 [SUCCESS] 10.0.145.107
[6] 17:26:13 [SUCCESS] 10.0.145.111
[7] 17:26:13 [SUCCESS] 10.0.145.108
[8] 17:26:13 [SUCCESS] 10.0.145.102
[9] 17:26:13 [SUCCESS] 10.0.145.117
[10] 17:26:13 [SUCCESS] 10.0.145.112
[11] 17:26:13 [SUCCESS] 10.0.145.121
[12] 17:26:13 [SUCCESS] 10.0.145.110
[13] 17:26:13 [SUCCESS] 10.0.145.105
[14] 17:26:13 [SUCCESS] 10.0.145.106
[15] 17:26:13 [SUCCESS] 10.0.145.113
...
[45] 17:26:13 [SUCCESS] 10.0.145.145
```

## Installing Cloudera

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

## Prerequisite for CDH Installation

This section details the pre-requisites for CDH Installation such as setting up of CDH Repo.

### Cloudera Repo

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

```
mkdir -p /tmp/clouderarepo/
```

1. Download Cloudera Manager Repo

```
cd /tmp/clouderarepo/
```

```
wget http://archive.cloudera.com/cm5/redhat/6/x86_64/cm/cloudera-manager.repo
```

Figure 128 Downloading Cloudera Manager Repo

```
[root@redhat clouderarepo]# wget http://archive.cloudera.com/cm5/redhat/6/x86_64/cm5-2014-02-28 13:55:06-- http://archive.cloudera.com/cm5/redhat/6/x86_64/cm5
Resolving archive.cloudera.com... 184.73.217.71
Connecting to archive.cloudera.com|184.73.217.71|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 289 [text/plain]
Saving to: âcloudera-manager.repoâ

100%[=====]
2014-02-28 13:55:06 (24.4 MB/s) - âcloudera-manager.repoâ
reposync --config=./cloudera-manager.repo --repoid=cloudera-manager
```

Figure 129 Downloading Cloudera Manager

```
[root@redhat clouderarepo]# reposync --config=./cloudera-manager.repo --repoid=cloudera-manager
cloudera-manager | 951 B 00:00
cloudera-manager/primary | 4.0 kB 00:00
[cloudera-manager: 1 of 7 ] Downloading RPMS/x86_64/cloudera-manager-agent-5.0.0-0.cm5b2.p0.119.el6.x86_64.rpm
cloudera-manager-agent-5.0.0-0.cm5b2.p0.119.el6.x86_64.rpm | 3.7 MB 00:05
[cloudera-manager: 2 of 7 ] Downloading RPMS/x86_64/cloudera-manager-daemons-5.0.0-0.cm5b2.p0.119.el6.x86_64.rpm
cloudera-manager-daemons-5.0.0-0.cm5b2.p0.119.el6.x86_64.rpm | 324 MB 01:52
[cloudera-manager: 3 of 7 ] Downloading RPMS/x86_64/cloudera-manager-server-5.0.0-0.cm5b2.p0.119.el6.x86_64.rpm
cloudera-manager-server-5.0.0-0.cm5b2.p0.119.el6.x86_64.rpm | 7.9 kB 00:00
[cloudera-manager: 4 of 7 ] Downloading RPMS/x86_64/cloudera-manager-server-db-2-5.0.0-0.cm5b2.p0.119.el6.x86_64.rpm
cloudera-manager-server-db-2-5.0.0-0.cm5b2.p0.119.el6.x86_64.rpm | 9.7 kB 00:00
[cloudera-manager: 5 of 7 ] Downloading RPMS/x86_64/enterprise-debuginfo-5.0.0-0.cm5b2.p0.119.el6.x86_64.rpm
enterprise-debuginfo-5.0.0-0.cm5b2.p0.119.el6.x86_64.rpm | 668 kB 00:00
[cloudera-manager: 6 of 7 ] Downloading RPMS/x86_64/jdk-6u31-linux-amd64.rpm
jdk-6u31-linux-amd64.rpm | 68 MB 00:20
[cloudera-manager: 7 of 7 ] Downloading RPMS/x86_64/oracle-j2sdk1.7-1.7.0+update25-1.x86_64.rpm
oracle-j2sdk1.7-1.7.0+update25-1.x86_64.rpm | 91 MB 00:33
```

## 2. Download Cloudera Manager Installer.

```
cd /tmp/clouderarepo/
wget http://archive.cloudera.com/cm5/installer/latest/cloudera-manager-installer.bin
```

Figure 130 Downloading Cloudera Manager Installer

```
[root@redhat clouderarepo]# wget http://archive.cloudera.com/cm5/installer/latest/cloudera-manager-installer.bin
--2014-02-28 14:11:30-- http://archive.cloudera.com/cm5/installer/latest/cloudera-manager-installer.bin
Resolving archive.cloudera.com... 184.73.217.71
Connecting to archive.cloudera.com|184.73.217.71|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 510866 (499K) [application/octet-stream]
Saving to: âcloudera-manager-installer.binâ

100%[=====>] 510,866 565K/s in 0.9s
2014-02-28 14:11:31 (565 KB/s) - âcloudera-manager-installer.binâ
```

## 3. Download CDH5 Repo.

```
cd /tmp/clouderarepo/
wget http://archive.cloudera.com/cdh5/redhat/6/x86_64/cdh/cloudera-cdh5.repo
reposync --config=./cloudera-cdh5.repo --repoid=cloudera-cdh5
```

Figure 131 Downloading CDH5 Repo

```
[root@redhat clouderarepo]# reposync --config=./cloudera-cdh5.repo --repoid=cloudera-cdh5
cloudera-cdh5 | 951 B 00:00
cloudera-cdh5/primary | 40 kB 00:00
[cloudera-cdh5: 1 of 112 ] Downloading RPMS/noarch/avro-doc-1.7.5+cdh5.0.0+8-0.cdh5b2.p0.30.e16.noarch.rpm
avro-doc-1.7.5+cdh5.0.0+8-0.cdh5b2.p0.30.e16.noarch.rpm | 1.0 MB 00:01
[cloudera-cdh5: 2 of 112 ] Downloading RPMS/noarch/avro-libs-1.7.5+cdh5.0.0+8-0.cdh5b2.p0.30.e16.noarch.rpm
avro-libs-1.7.5+cdh5.0.0+8-0.cdh5b2.p0.30.e16.noarch.rpm | 12 MB 00:10
[cloudera-cdh5: 3 of 112 ] Downloading RPMS/noarch/avro-tools-1.7.5+cdh5.0.0+8-0.cdh5b2.p0.30.e16.noarch.rpm
avro-tools-1.7.5+cdh5.0.0+8-0.cdh5b2.p0.30.e16.noarch.rpm | 2.5 kB 00:00
[cloudera-cdh5: 4 of 112 ] Downloading RPMS/x86_64/bigtop-jsvc-0.6.0+cdh5.0.0+389-0.cdh5b2.p0.25.e16.x86_64.rpm
bigtop-jsvc-0.6.0+cdh5.0.0+389-0.cdh5b2.p0.25.e16.x86_64.rpm | 27 kB 00:00
[cloudera-cdh5: 5 of 112 ] Downloading RPMS/x86_64/bigtop-jsvc-debuginfo-0.6.0+cdh5.0.0+389-0.cdh5b2.p0.25.e16.x86_64.rpm
bigtop-jsvc-debuginfo-0.6.0+cdh5.0.0+389-0.cdh5b2.p0.25.e16.x86_64.rpm | 55 kB 00:00
[cloudera-cdh5: 6 of 112 ] Downloading RPMS/noarch/bigtop-tomcat-0.7.0+cdh5.0.0+0-0.cdh5b2.p0.25.e16.noarch.rpm
bigtop-tomcat-0.7.0+cdh5.0.0+0-0.cdh5b2.p0.25.e16.noarch.rpm | 7.2 MB 00:02
[cloudera-cdh5: 7 of 112 ] Downloading RPMS/noarch/bigtop-utils-0.7.0+cdh5.0.0+0-0.cdh5b2.p0.30.e16.noarch.rpm
bigtop-utils-0.7.0+cdh5.0.0+0-0.cdh5b2.p0.30.e16.noarch.rpm | 8.8 kB 00:00
```

#### 4. Download Impala Repo

```
cd /tmp/clouderarepo/
wget http://archive.cloudera.com/impala/redhat/6/x86_64/impala/cloudera-impala.repo
```

Figure 132 Download Impala Repo

```
[root@redhat clouderarepo]# wget http://archive.cloudera.com/impala/redhat/6/x86_64/impala/cloudera-impala.repo
--2014-02-28 14:22:42-- http://archive.cloudera.com/impala/redhat/6/x86_64/impala/cloudera-impala.repo
Resolving archive.cloudera.com... 184.73.217.71
Connecting to archive.cloudera.com|184.73.217.71|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 205 [text/plain]
Saving to: âcloudera-impala.repoâ

100%[=====>] 205 --.-K/s in 0s

2014-02-28 14:22:42 (38.3 MB/s) - âcloudera-impala.repoâ
```

```
reposync --config=./cloudera-impala.repo --repoid=cloudera-impala
```

Figure 133 Downloading Impala Repo

```
[root@redhat clouderarepo]# reposync --config=./cloudera-impala.repo --repoid=cloudera-impala
cloudera-impala | 951 B 00:00
cloudera-impala/primary | 2.3 kB 00:00
[cloudera-impala: 1 of 8 ] Downloading RPMS/noarch/bigtop-utils-0.4+300-1.cdh4.0.1.p0.1.e16.noarch.rpm
bigtop-utils-0.4+300-1.cdh4.0.1.p0.1.e16.noarch.rpm | 7.4 kB 00:00
[cloudera-impala: 2 of 8 ] Downloading RPMS/x86_64/impala-1.2.3-1.p0.352.e16.x86_64.rpm
impala-1.2.3-1.p0.352.e16.x86_64.rpm | 148 MB 01:46
[cloudera-impala: 3 of 8 ] Downloading RPMS/x86_64/impala-catalog-1.2.3-1.p0.352.e16.x86_64.rpm
impala-catalog-1.2.3-1.p0.352.e16.x86_64.rpm | 4.3 kB 00:00
[cloudera-impala: 4 of 8 ] Downloading RPMS/x86_64/impala-debuginfo-1.2.3-1.p0.352.e16.x86_64.rpm
```

#### 5. Copy the repository directory to the admin node



Figure 134 Copying Repository Directory

```
[root@redhat clouderarepo]# scp -r /tmp/clouderarepo/ rhell1:/var/www/html
cloudera-manager-installer.bin          100% 499KB 498.9KB/s 00:00
impala-server-1.2.3-1.p0.352.el6.x86_64.rpm 100% 4368 4.3KB/s 00:00
impala-udf-devel-1.2.3-1.p0.352.el6.x86_64.rpm 100% 33KB 33.4KB/s 00:00
impala-shell-1.2.3-1.p0.352.el6.x86_64.rpm 100% 695KB 695.5KB/s 00:00
impala-catalog-1.2.3-1.p0.352.el6.x86_64.rpm 100% 4396 4.3KB/s 00:00
impala-debuginfo-1.2.3-1.p0.352.el6.x86_64.rpm 100% 280MB 40.0MB/s 00:07
impala-1.2.3-1.p0.352.el6.x86_64.rpm 100% 148MB 49.4MB/s 00:03
impala-state-store-1.2.3-1.p0.352.el6.x86_64.rpm 100% 4456 4.4KB/s 00:00
bigtop-utils-0.4+300-1.cdh4.0.1.p0.1.el6.noarch.rpm 100% 7572 7.4KB/s 00:00
cloudera-impala.repo 100% 112 0.1KB/s 00:00
cloudera-manager.repo 100% 204 0.2KB/s 00:00
cloudera-manager-daemons-5.0.0-0.cm5b2.p0.119.el6.x86_64.rpm 100% 324MB 40.5MB/s 00:08
cloudera-manager-server-db-2-5.0.0-0.cm5b2.p0.119.el6.x86_64.rpm 100% 9884 9.7KB/s 00:00
oracle-j2sdk1.7-1.7.0+update25-1.x86_64.rpm 100% 91MB 45.7MB/s 00:02
jdk-6u31-linux-amd64.rpm 100% 68MB 34.0MB/s 00:02
enterprise-debuginfo-5.0.0-0.cm5b2.p0.119.el6.x86_64.rpm 100% 668KB 667.9KB/s 00:00
cloudera-manager-agent-5.0.0-0.cm5b2.p0.119.el6.x86_64.rpm 100% 3746KB 3.7MB/s 00:00
cloudera-manager-server-5.0.0-0.cm5b2.p0.119.el6.x86_64.rpm 100% 8128 7.9KB/s 00:00
hue-impala-3.5.0+cdh5.0.0+186-0.cdh5b2.p0.22.el6.x86_64.rpm 100% 22KB 21.7KB/s 00:00
hue-server-3.5.0+cdh5.0.0+186-0.cdh5b2.p0.22.el6.x86_64.rpm 100% 4544 4.4KB/s 00:00
hue-doc-3.5.0+cdh5.0.0+186-0.cdh5b2.p0.22.el6.x86_64.rpm 100% 1152KB 1.1MB/s 00:00
hue-zookeeper-3.5.0+cdh5.0.0+186-0.cdh5b2.p0.22.el6.x86_64.rpm 100% 34KB 34.0KB/s 00:00
hadoop-0.20-mapreduce-jobtracker-2.2.0+cdh5.0.0+1610-0.cdh5b2.p0 100% 5212 5.1KB/s 00:00
hadoop-client-2.2.0+cdh5.0.0+1610-0.cdh5b2.p0.51.el6.x86_64.rpm 100% 25KB 24.7KB/s 00:00
hadoop-httpfs-2.2.0+cdh5.0.0+1610-0.cdh5b2.p0.51.el6.x86_64.rpm 100% 20MB 19.9MB/s 00:01
hue-plugins-3.5.0+cdh5.0.0+186-0.cdh5b2.p0.22.el6.x86_64.rpm 100% 3168 3.1KB/s 00:00
hadoop-war-mrnodemanager-2.2.0+cdh5.0.0+1610-0.cdh5b2.p0.51.el6.x 100% 5064 5.0KB/s 00:00
```

6. On admin node (rhell) run create repo command.

```
cd /var/www/html/clouderarepo/
createrepo --baseurl http://10.0.145.101/clouderarepo/cloudera-manager/
/var/www/html/clouderarepo/cloudera-manager
```

Figure 135 Command Output for 'Create Repo'

```
[root@rhell1 clouderarepo]#createrepo --baseurl http://10.29.160.53/clouderarepo/cloudera-manager/ /var/w
/html/clouderarepo/cloudera-manager
Spawning worker 0 with 7 pkgs
Workers Finished
Gathering worker results

Saving Primary metadata
Saving file lists metadata
Saving other metadata
Generating sqlite DBs
Sqlite DBs complete
```

```
createrepo --base url http://10.0.145.101/clouderarepo/cloudera-cdh5
/var/www/html/clouderarepo/cloudera-cdh5
```

Figure 136 Command Output

```
[root@rhell1 clouderarepo]#createrepo --baseurl http://10.0.145.101/clouderarepo/cloudera-cdh5/
ml/clouderarepo/cloudera-cdh5
Spawning worker 0 with 112 pkgs
Workers Finished
Gathering worker results

Saving Primary metadata
Saving file lists metadata
Saving other metadata
Generating sqlite DBs
Sqlite DBs complete
```

```
createrepo --base url http://10.0.145.101/clouderarepo/cloudera-impala
/var/www/html/clouderarepo/cloudera-impala
```

Figure 137 Command Output

```
[root@rhell clouderarepo]#createrepo --baseurl http://10.0.145.101/clouderarepo
html/clouderarepo/cloudera-impala
Spawning worker 0 with 8 pkgs
Workers Finished
Gathering worker results

Saving Primary metadata
Saving file lists metadata
Saving other metadata
Generating sqlite DBs
Sqlite DBs complete
```

**Note**

Visit <http://10.0.145.101/clouderarepo> to verify the files.

7. Create the Cloudera Manager repo file with following contents:

```
vi /var/www/html/clouderarepo/cloudera-manager/cloudera-manager.repo
[cloudera-manager]
name=Cloudera Manager
baseurl=http://10.0.145.101/clouderarepo/cloudera-manager/
gpgcheck = 0
```

8. Create the CDH repo file with following contents:

```
vi /var/www/html/clouderarepo/cloudera-cdh5/cloudera-cdh5.repo
[cloudera-cdh5]
name=Cloudera's Distribution for Hadoop, Version 5
baseurl= http://10.0.145.101/clouderarepo/cloudera-cdh5
gpgcheck = 0
```

9. Create the Impala repo file with following contents:

```
vi /var/www/html/clouderarepo/cloudera-impala/cloudera-impala.repo
[cloudera-impala]
name=Impala
baseurl= http://10.0.145.101/clouderarepo/cloudera-impala
gpgcheck = 0
```

10. Copy the file `cloudera-manager.repo`, `cloudera-cdh5.repo`, `cloudera-impala.repo` into `/etc/yum.repos.d/` on the admin node to enable it to find the packages that are locally hosted.

```
cp /var/www/html/clouderarepo/cloudera-manager/cloudera-manager.repo /etc/yum.repos.d/
cp /var/www/html/clouderarepo/cloudera-cdh5/cloudera-cdh5.repo /etc/yum.repos.d/
cp /var/www/html/clouderarepo/cloudera-impala/cloudera-impala.repo /etc/yum.repos.d/
```

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

```
pscp -h /root/allnodes /etc/yum.repos.d/ cloudera-manager.repo /etc/yum.repos.d/
pscp -h /root/allnodes /etc/yum.repos.d/ cloudera-cdh5.repo /etc/yum.repos.d/
pscp -h /root/allnodes /etc/yum.repos.d/ cloudera-impala.repo /etc/yum.repos.d/
```

## Cloudera Installation

### Prerequisites

```
pssh -h /root/allnodes "sysctl -w vm.swappiness=0"
pssh -h /root/allnodes "echo never > /sys/kernel/mm/redhat_transparent_hugepage/enabled"
```

## Installing Cloudera Manager

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

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

Follow the steps below to install Cloudera Manager.

Update the repo files to point to local repository.

```
rm -f /var/www/html/clouderarepo/*.repo
cp /etc/yum.repos.d/c*.repo /var/www/html/clouderarepo/
```

1. Change the permission of Cloudera Manager Installer on the admin node.

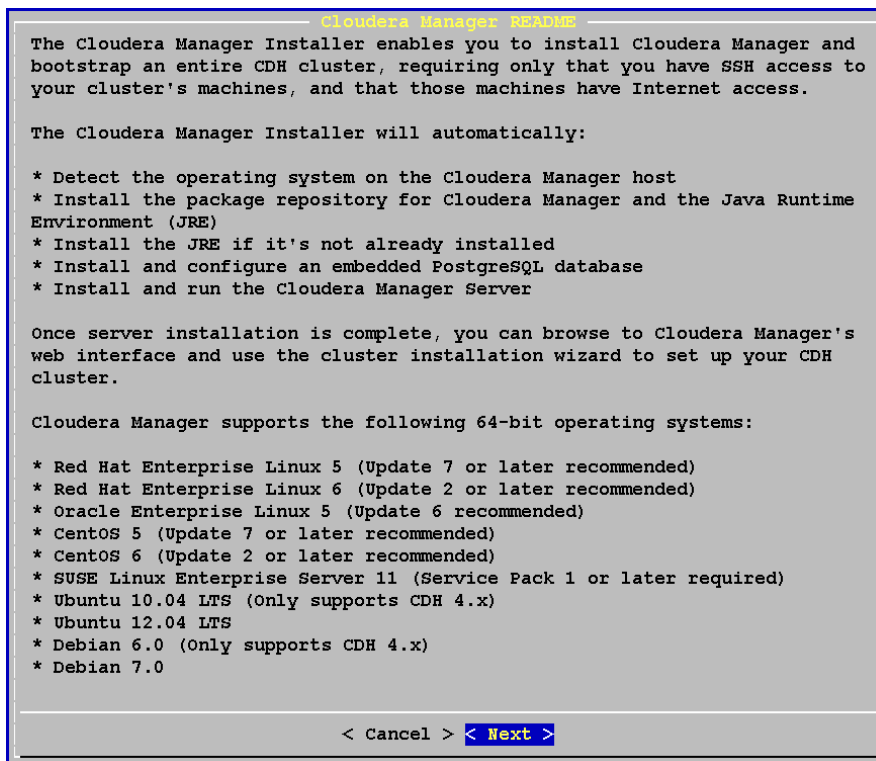
```
cd /var/www/html/clouderarepo
chmod +x cloudera-manager-installer.bin
```

2. Execute the following command in the admin node (rhell) to start **Cloudera Manager Installer**.

```
cd /var/www/html/clouderarepo/
./cloudera-manager-installer.bin
```

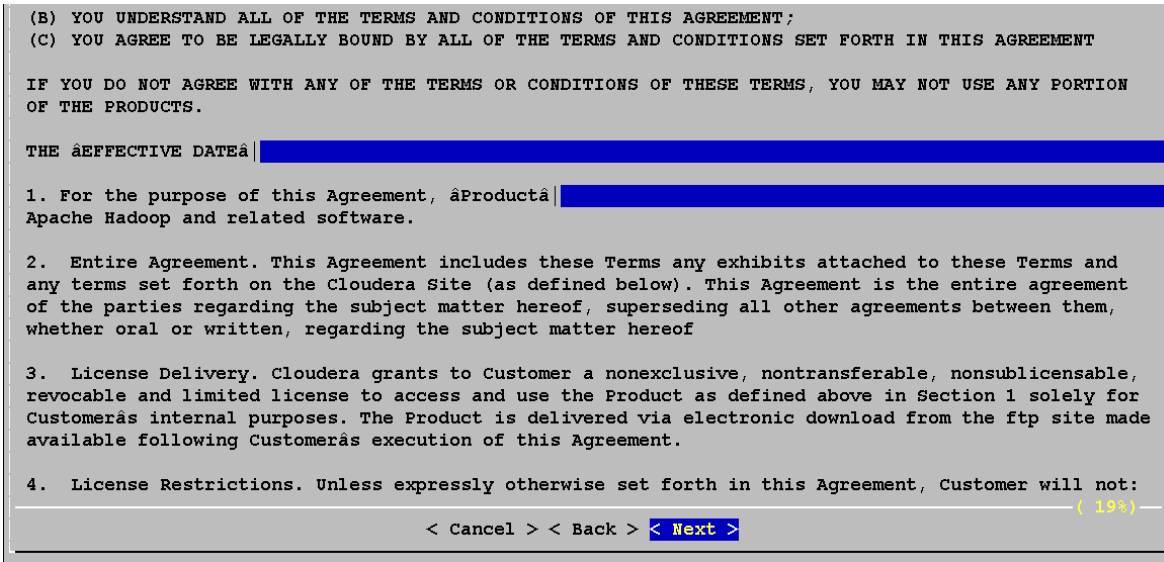
3. This displays the **Cloudera Manager Read Me** file. Click **Next**.

*Figure 138 Cloudera Read Me File*



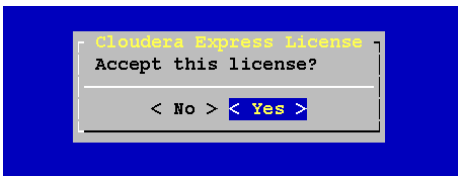
- Click **Next** in the **End User License agreement** page.

*Figure 139 Signing Terms and Conditions*



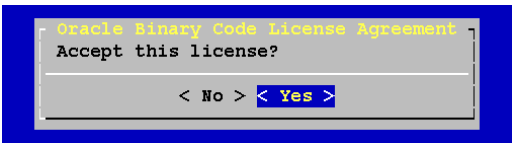
- Click **Yes** in the **License Agreement Confirmation** page.

*Figure 140 Accepting Cloudera Express License*



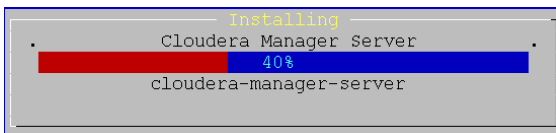
- Click **Next** in **Oracle Binary Code License Agreement** and **Yes** in the **Oracle Binary Code License Agreement for the Java SE Platform Products** page.

*Figure 141 Accepting Oracle Binary Code License Agreement*



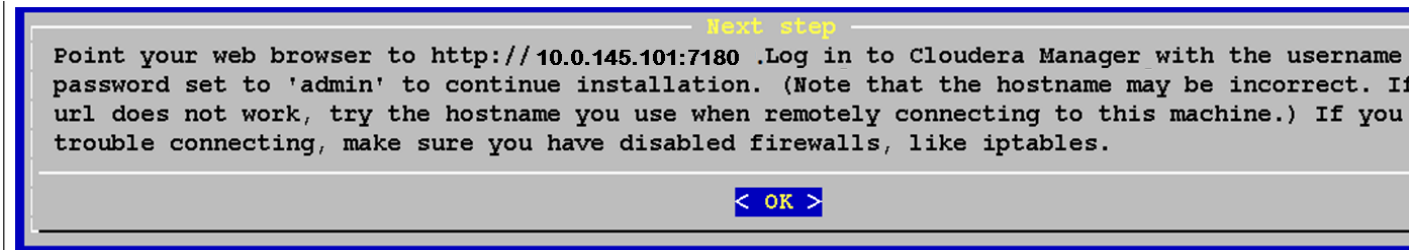
- Wait for the installer to install the packages needed for Cloudera Manager.

*Figure 142 Installing Cloudera Manager Server*



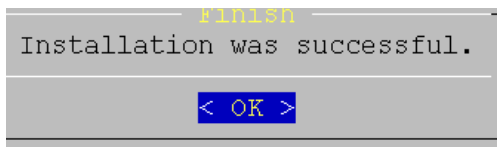
- Save the url displayed `http://10.0.145.101:7180`. You will need this url to access Cloudera Manager. If you are unable to connect to the server, check to see if iptables and SELinux are disabled.

Figure 143 Connecting to Server



9. Click OK.

Figure 144 Installation Completed



10. Once the installation of Cloudera Manager is complete. Install CDH5 using the Cloudera Manager web interface.

## Installing Cloudera Enterprise Data Hub (CDH5)

To install Cloudera Enterprise Data Hub, follow these steps:

1. Access the Cloudera Manager using the URL displayed by the Installer, <http:// 10.0.145.101:7180>.
2. Login to the Cloudera Manager. Enter "admin" for both the Username and Password fields.

Figure 145 Logging Into Cloudera Enterprise Data Hub

3. If you do not have a Cloudera license, click If you do have a Cloudera license, Click “Upload **Cloudera Enterprise Data Hub Trial Edition License**” and select your license.
4. Based on requirement Choose appropriate Cloudera Editions for Installation.

Figure 146 Installing Cloudera Enterprise Data Hub Trial

## Welcome to Cloudera Manager. Which edition do you want to deploy?

Upgrading to **Cloudera Enterprise Data Hub Edition** provides important features that help you manage and monitor your Hadoop clusters in mission-critical environments.

	Cloudera Express	Cloudera Enterprise Data Hub Edition Trial	Cloudera Enterprise
<b>License</b>	Free	<b>60 Days</b> After the trial period, the product will continue to function as <b>Cloudera Express</b> . Your cluster and your data will remain unaffected.	<b>Annual Subscription</b> Upload License Cloudera Enterprise is available in three editions: <ul style="list-style-type: none"> <li>• Basic Edition</li> <li>• Flex Edition</li> <li>• Data Hub Edition</li> </ul>
<b>Node Limit</b>	Unlimited	Unlimited	Unlimited
<b>CDH</b>	✓	✓	✓
<b>Core Cloudera Manager Features</b>	✓	✓	✓
<b>Advanced Cloudera</b>		✓	✓

5. Click **Continue** in the confirmation page.

Figure 147 Confirmation Page

## Thank you for choosing Cloudera Manager and CDH.

This installer will install **Cloudera Enterprise Data Hub Edition Trial 5.1.1** and enable you to later choose packages for the services below (there may be some license implications).

- Apache Hadoop (Common, HDFS, MapReduce, YARN)
- Apache HBase
- Apache ZooKeeper
- Apache Oozie
- Apache Hive
- Hue (Apache licensed)
- Apache Flume
- Cloudera Impala (Apache licensed)
- Apache Sqoop
- Cloudera Search (Apache licensed)
- Apache Spark

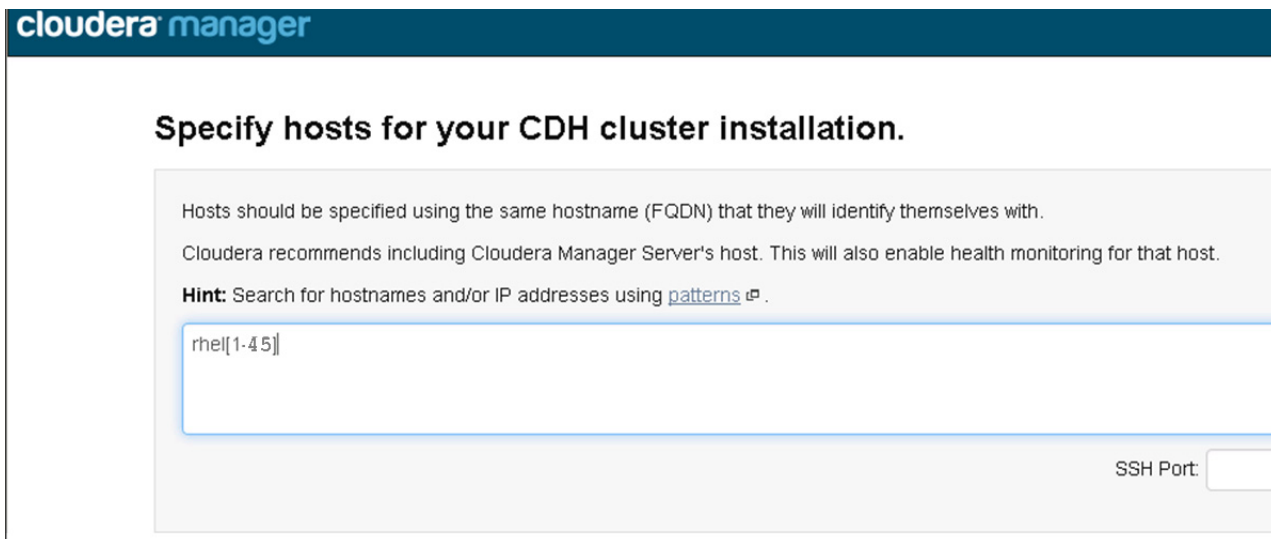
You are using Cloudera Manager to install and configure your system. You can learn more about Cloudera Manager by clicking on the **Support** menu above.

6. Specify the hosts that are part of the cluster using their IP addresses or hostname. The [Figure 148](#) shows use of a pattern to specify IP addresses range.

10.0.145.[101-145]

7. After the IP addresses are entered, click **Search**.

Figure 148 Searching for Cluster Nodes



The screenshot shows the Cloudera Manager interface with a dark blue header containing the text "cloudera manager". Below the header, the main heading reads "Specify hosts for your CDH cluster installation." A light gray box contains the following text: "Hosts should be specified using the same hostname (FQDN) that they will identify themselves with. Cloudera recommends including Cloudera Manager Server's host. This will also enable health monitoring for that host. **Hint:** Search for hostnames and/or IP addresses using [patterns](#) <sup>Ⓔ</sup>." Below this text is a large text input field containing the pattern "rhel[1-4.5]". To the right of the input field is a label "SSH Port:" followed by a small, empty input field.

8. Cloudera Manager will "discover" the nodes in the cluster. Verify that all desired nodes have been found and selected for installation.
9. Click Install **CDH On Selected Host**. CDH is Cloudera Distribution for Apache Hadoop.

Figure 149 Verifying and Selecting the Hosts

The screenshot shows the Cloudera Manager interface for specifying hosts. At the top, it says "Specify hosts for your CDH cluster installation." Below this, there is a text box explaining that hosts should be specified using the same hostname (FQDN) they will identify themselves with, and that Cloudera recommends including the Cloudera Manager Server's host. A hint suggests searching for hostnames and/or IP addresses using patterns. A status bar indicates "32 hosts scanned, 32 running SSH." Below this is a table of host verification results.

<input checked="" type="checkbox"/>	Expanded Query	Hostname (FQDN)	IP Address	Currently Managed	Result
<input checked="" type="checkbox"/>	rhel1	rhel1	10.0.145.101	No	✓ Host ready: 0 ms response time.
<input checked="" type="checkbox"/>	rhel10	rhel10	10.0.145.110	No	✓ Host ready: 2 ms response time.
<input checked="" type="checkbox"/>	rhel11	rhel11	10.0.145.111	No	✓ Host ready: 2 ms response time.
<input checked="" type="checkbox"/>	rhel12	rhel12	10.0.145.112	No	✓ Host ready: 2 ms response time.
<input checked="" type="checkbox"/>	rhel13	rhel13	10.0.145.113	No	✓ Host ready: 1 ms response time.
<input checked="" type="checkbox"/>	rhel14	rhel14	10.0.145.114	No	✓ Host ready: 1 ms response time.
<input checked="" type="checkbox"/>	rhel15	rhel15	10.0.145.115	No	✓ Host ready: 1 ms response time.
<input checked="" type="checkbox"/>	rhel16	rhel16	10.0.145.116	No	✓ Host ready: 2 ms response time.
<input checked="" type="checkbox"/>	rhel17	rhel17	10.0.145.117	No	✓ Host ready: 1 ms response time.
<input checked="" type="checkbox"/>	rhel18	rhel18	10.0.145.118	No	✓ Host ready: 1 ms response time.
<input checked="" type="checkbox"/>	rhel19	rhel19	10.0.145.119	No	✓ Host ready: 2 ms response time.
<input checked="" type="checkbox"/>	rhel2	rhel2	10.0.145.102	No	✓ Host ready: 1 ms response time.
<input checked="" type="checkbox"/>	rhel20	rhel20	10.0.145.120	No	✓ Host ready: 0 ms response time.
<input checked="" type="checkbox"/>	rhel21	rhel21	10.0.145.121	No	✓ Host ready: 1 ms response time.
<input checked="" type="checkbox"/>	rhel22	rhel22	10.0.145.122	No	✓ Host ready: 2 ms response time.

At the bottom of the interface, there are "Back" and "Continue" buttons.

10. For the method of installation, select the **Use Package** radio button.
11. For the CDH version, select the **CDH5** radio button.
12. For the specific release of CDH you want to install in your hosts, select **Custom Repository** radio button.
13. Enter the following URL for the repository within the admin node.  

```
http://10.0.145.101/clouderarepo/cloudera-cdh5
```
14. For the specific Impala release, select the Custom Repository radio button.
15. For the specific release of **Cloudera Manager**, select the **Custom Repository** radio button.
16. Enter the URL for the repository within the admin node.  

```
http://10.0.145.101/clouderarepo/cloudera-manager
```





Figure 150 Selecting the CDH Version

### Select Repository

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

**Choose Method**

Use Packages 

Use Parcels (Recommended)  More Options

**Select the version of CDH**

CDH 5

CDH 4

**Select the specific release of CDH you want to install on your hosts.**

Latest Release of CDH 5

CDH 5.1.0

CDH 5.0.2

CDH 5.0.1

CDH 5.0.0

Custom Repository

Example for SLES, Redhat or other RPM based distributions:

`http://archive.cloudera.com/cdh5/redhat/6/x86_64/cdh/5/`

Example for Ubuntu or other Debian based distributions:

`deb http://archive.cloudera.com/cdh5/ubuntu/lucid/amd64/cdh/ lucid-cdh5 contrib`

**Select the specific release of the Cloudera Manager Agent you want to install on your hosts.**

Matched release for this Cloudera Manager Server

Custom Repository

1 2 3 4 5 6

17. Provide SSH login credentials for the cluster and click **Start Installation**.

Figure 151 Login Credentials to Start CDH Installation

cloudera manager Support ▾

## Cluster Installation

**Provide SSH login credentials.**

Root access to your hosts is required to install the Cloudera packages. This installer will connect to your hosts via SSH and log in either directly as root or as another user with password-less sudo/pbrun privileges to become root.

Login To All Hosts As:  root  
 Another user

You may connect via password or public-key authentication for the user selected above.

Authentication Method:  All hosts accept same password  
 All hosts accept same private key

Enter Password:

Confirm Password:

SSH Port:

Number of Simultaneous Installations:   
(Running a large number of installations at once can consume large amounts of network bandwidth and other system resources)

[Back](#) 1 2 3 4 5 6 [Continue](#)

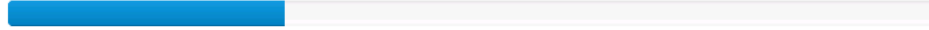
18. Make sure the installation across all the hosts is complete.

19. After the installation is complete, click **Continue**.

*Figure 152 Installation in Progress*

## Cluster Installation

### Installation in progress.



0 of 32 host(s) completed successfully. [Abort Installation](#)

Hostname	IP Address	Progress	Status
rhel1	10.0.145.101	<div style="width: 50%;"></div>	Installing sqoop2 package...
rhel10	10.0.145.110	<div style="width: 50%;"></div>	Installing impala package...
rhel11	10.0.145.111	<div style="width: 50%;"></div>	Installing impala package...
rhel12	10.0.145.112	<div style="width: 50%;"></div>	Installing impala package...

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

21. Review and verify the summary. Click Continue.

*Figure 153 Inspecting Hosts for Correctness*

**cloudera manager** Support ▾

## Cluster Installation

### Inspect hosts for correctness

Inspecting hosts... This could take a minute.

[Skip Host Inspector](#)

22. Select services that need to be started on the cluster.

Figure 154 Selecting CDH Version and Services

Select Repository

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

- Choose Method**
- Use Packages
  - Use Parcels (Recommended) More Options

- Select the version of CDH**
- CDH 5
  - CDH 4

Select the specific release of CDH you want to install on your hosts.

- Latest Release of CDH 5
- CDH 5.1.0
- CDH 5.0.2
- CDH 5.0.1
- CDH 5.0.0
- Custom Repository

Example for SLES, Redhat or other RPM based distributions:

`http://archive.cloudera.com/cdh5/redhat/6/x86_64/cdh/5/`

Example for Ubuntu or other Debian based distributions:

`deb http://archive.cloudera.com/cdh5/ubuntu/lucid/amd64/cdh/ lucid-cdh5 contrib`

Select the specific release of the Cloudera Manager Agent you want to install on your hosts.

- Matched release for this Cloudera Manager Server
- Custom Repository

Navigation bar with a "Back" button, a progress indicator (steps 1-6, with 1 highlighted), and a "Continue" button.

23. This is one of the critical steps in the installation. Inspect and customize the role assignments of all the nodes based on your requirements and click **Continue**.

24. Reconfigure the service assignment to match the [Table 8](#).

Table 8 Server Assignment

Service Name	Host
Service NameHost	rhel1
NameNoderhel1	rhel1
SNameNoderhel2	rhel1
HistoryServerrhel1	rhel1
ResouceManagerrhel1	rhel1
Hue Serverrhel1	rhel1
HiveMetastore Serverrhel1	rhel1

**Table 8** Server Assignment

Service Name	Host
HiveServer2	rhel1
HBase Master	rhel1
Oozie Server	rhel1
NodeManager	rhel3 to rhel45
RegionServer	rhel3 to rhel45
Sqoop Server	rhel1
Impala Catalog Server Daemon	rhel2
Solr Server	rhel1
Zookeeper	rhel1, rhel2, rhel3
Spark Master	rhel1
Spark Worker	rhel2
DataNode	rhel3 to rhel45

**Figure 155** SSH Login Credentials

## Cluster Installation

### Provide SSH login credentials.

Root access to your hosts is required to install the Cloudera packages. This installer will connect to your hosts via SSH and log in either directly as root or as another user with password-less sudo/pbrun privileges to become root.

Login To All Hosts As:  root  
 Another user

You may connect via password or public-key authentication for the user selected above.

Authentication Method:  All hosts accept same password  
 All hosts accept same private key

Enter Password:

Confirm Password:

## Scaling the Cluster

The role assignment recommendation above is for clusters of up to 45 servers. For clusters of 15 to 45 nodes the recommendation is to dedicate one server for name node and a second server for secondary name node and YARN Resource Manager. For larger clusters larger than 45 nodes the recommendation is to dedicate one server each for name node, secondary name node and YARN Resource Manager.

1. Select the **Use Embedded Database** radio button.
2. Click **Test Connection** and click **Continue**.

Figure 156 Database Setup

## Cluster Setup

### Database Setup

Configure and test database connections. If using custom databases, create the databases first according to the **Installing and Configuring an External Database** section of the [Installation Guide](#).

Use Custom Databases  
 Use Embedded Database

When using the embedded database, passwords are automatically generated. Please copy them down.

---

**Activity Monitor**

Currently assigned to run on **rhel1**.

Database Host Name:	Database Type:	Database Name :	Username:	Password:
rhel1:7432	PostgreSQL	amon	amon	IUmm2A12WH

---

**Reports Manager**

Currently assigned to run on **rhel1**.

Database Host Name:	Database Type:	Database Name :	Username:	Password:
rhel1:7432	PostgreSQL	rman	rman	I7E4UFAGiD

---

**Hive**

Database Host Name:	Database Type:	Database Name :	Username:	Password:
rhel1:7432	PostgreSQL	hive	hive	nZDVPI5nSG

[Test Connection](#)

3. Review and customize the configuration changes based on your requirements

Figure 157 Reviewing the Configuration Changes - Part1

### Review Changes

Set the following configuration values for your new role(s). Required values are marked with \*.

Parameter	Group ⓘ	Recommended Value	Description
<b>Service YARN (MR2 Included)</b>			
<b>NodeManager Local Directory List*</b> yarn.nodemanager.local-dirs	<b>NodeManager Default Group</b> <a href="#">Show Members</a> ⓘ	<input type="text" value="/DATA/sdb1/yarn/nm"/> + - <input type="text" value="/DATA/sdc1/yarn/nm"/> + - <input type="text" value="/DATA/sdd1/yarn/nm"/> + - <input type="text" value="/DATA/sde1/yarn/nm"/> + - <input type="text" value="/DATA/sdf1/yarn/nm"/> + - <input type="text" value="/DATA/sdg1/yarn/nm"/> + - <input type="text" value="/DATA/sdh1/yarn/nm"/> + -	List of directories on filesystem where a NodeManager stores intermediate data file

Figure 158 Reviewing the Configuration Changes - Part2

Service hbase			
HDFS Root Directory* hbase.rootdir	Service-Wide	/hbase default value	The HDFS directory shared by HBase RegionServers.
Enable Replication hbase.replication	Service-Wide	<input checked="" type="checkbox"/> <a href="#">Reset to the default value: false</a> ↩	Allow HBase tables to be replicated.
Enable Indexing	Service-Wide	<input checked="" type="checkbox"/> <a href="#">Reset to the default value: false</a> ↩	Allow indexing of tables in HBase by Lily HBase Indexer. <b>Note:</b> Replication must be enabled for indexing to work.
Service hdfs			
DataNode Data Directory* dfs.datanode.data.dir	DataNode Default Group <a href="#">Show Members</a> ⓘ	<input type="text" value="/DATA/sdb1/dfs/dn"/> + - <input type="text" value="/DATA/sdc1/dfs/dn"/> + - <input type="text" value="/DATA/sdd1/dfs/dn"/> + - <input type="text" value="/DATA/sde1/dfs/dn"/> + - <input type="text" value="/DATA/sdf1/dfs/dn"/> + - <input type="text" value="..."/> + -	Comma-delimited list of directories on the local file system where the DataNode stores HDFS block data. Typical values are /data/N/dfs/dn for N = 1, 2, 3... These directories should be mounted using the noatime option and the disks should be configured using JBOD. RAID is not

Figure 159 Reviewing the Configuration Changes - Part3

Service oozie			
Oozie Server Data Directory	Oozie Server Default Group <a href="#">Show Members</a> ⓘ	/var/lib/oozie/data default value	Directory where the Oozie Server will place its data. Only applicable when using Derby as the database type.
Service solr			
ZooKeeper Znode*	Service-Wide	/solr default value	ZooKeeper znode used to store information about this Search service.
HDFS Data Directory*	Service-Wide	/solr default value	HDFS directory used for storage by this Search service.
Service sqoop			
Sqoop Server Metastore Directory*	Sqoop Server Default Group <a href="#">Show Members</a> ⓘ	/var/lib/sqoop2 default value	Directory where the Sqoop Server will place its metastore data.
Service zookeeper			
Data Directory dataDir	Server Default Group <a href="#">Show Members</a> ⓘ	/var/lib/zookeeper default value	The disk location that ZooKeeper will use to store its database snapshots.
Transaction Log Directory dataLogDir	Server Default Group <a href="#">Show Members</a> ⓘ	/var/lib/zookeeper default value	The disk location that ZooKeeper will use to store its transaction logs.
<input type="button" value="⏪ Back"/>		<input type="button" value="1"/> <input type="button" value="2"/> <input type="button" value="3"/> <input checked="" type="button" value="4"/> <input type="button" value="5"/> <input type="button" value="6"/>	<input type="button" value="⏩ Continue"/>

4. Click **Continue** to start running the cluster services.

**Figure 160** Starting the Cluster Services

## Cluster Setup


### Progress

Command	Context	Status	Started at	Ended at
✓ <b>First Run</b>		Finished	Aug 28, 2014 5:35:03 PM EDT	Aug 28, 2014 5:44:08 PM EDT

Finished First Run of all services successfully.

### Command Progress

Completed 33 of 33 steps.



- ✓ Waiting for ZooKeeper Service to initialize  
Finished waiting  
[Details](#)
- ✓ Starting ZooKeeper Service  
Completed 3 steps successfully.  
[Details](#)
- ✓ Checking if the name directories of the NameNode are empty. Formatting HDFS only if empty.  
Successfully formatted NameNode.  
[Details](#)
- ✓ Starting HDFS Service  
Successfully started HDFS service  
[Details](#)
- ✓ Creating HDFS /tmp directory  
Successfully created HDFS directory /tmp.  
[Details](#)

1 2 3 4 5 6

⏪ Back Continue ⏩

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

**Figure 161** Installation Completion

## Cluster Setup

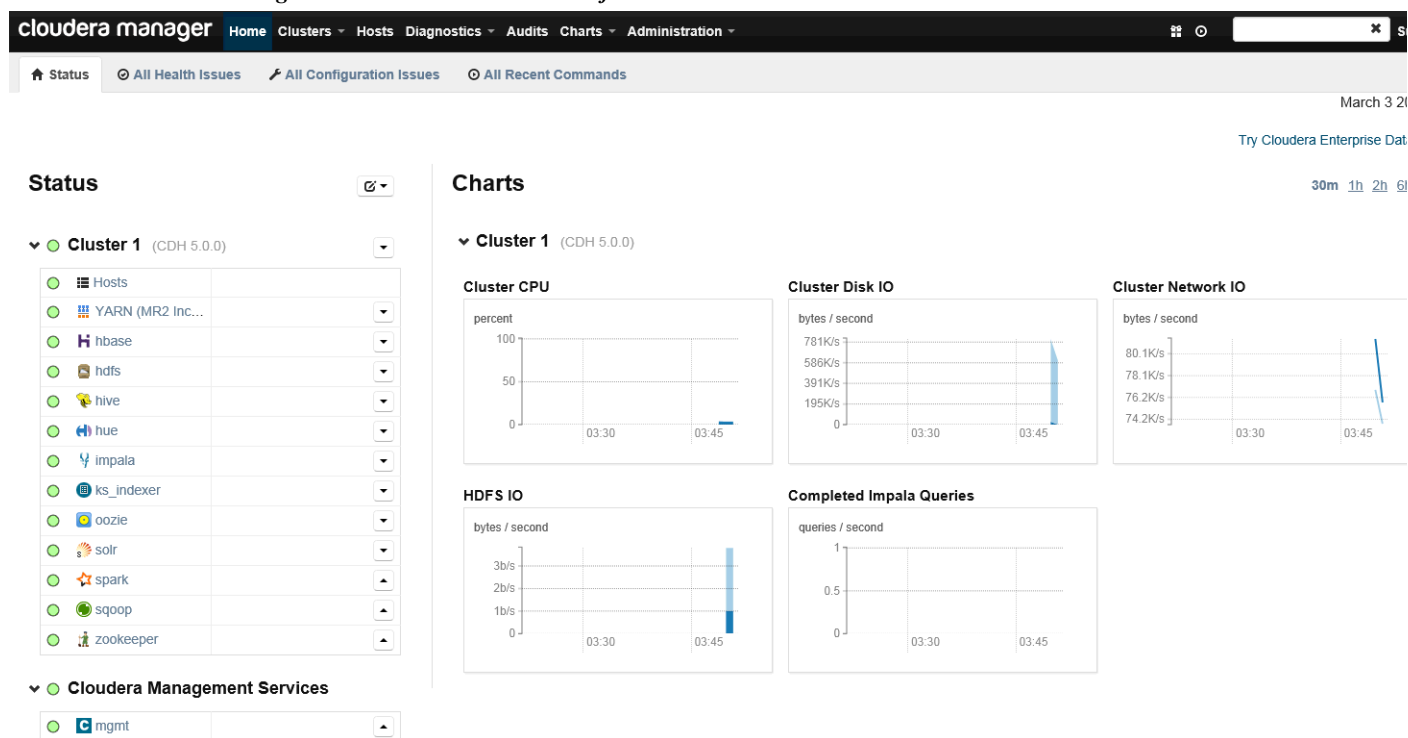
### Congratulations!

The services are installed, configured, and running on your cluster.

- Cloudera Manager will now show the status of all Hadoop services running on the cluster.



Figure 162 Service Status of the Cluster



## Conclusion

Hadoop has become a popular data management across all verticals. The Cisco Application Centric Infrastructure, along with Cloudera offers a dependable deployment model for enterprise Hadoop that offer a fast and predictable path for businesses to unlock value in big data.

The configuration detailed in the document can be extended to clusters of various sizes depending on what application demands as discussed in the Scalability section. Next generation Big Data Infrastructure needs to cater to the emerging trends in Big Data Applications to meet multiple Lines of Business (LOB) SLAs. The Cisco ACI brings numerous advantages to a big data cluster – single point of management for the network, enhanced performance, superior failure handling characteristics, unprecedented scalability. Further, ACI paves way to the next generation data center network accelerating innovation with its SDN capabilities in the big data space.

## Bill of Material

This section gives the BOM for the 45 node Performance and Capacity Balanced Cluster. See [Table 9](#) provides the BOM for Nexus 9k and APIC and [Table 10](#) provides the BOM for master rack servers. [Table 11](#) and [Table 12](#) provide the details of all the software components.

**Table 9** *Bill of Material for Nexus Device and APIC*

<b>Part Number</b>	<b>Description</b>	<b>Quantity</b>
N9K-C9508-B2	Nexus 9508 Chassis Bundle with 1 Sup, 3 PS, 2 SC, 6 FM, 3 FT	2
N9K-C93128TX	Nexus 9300 with 96p 1/10G-T and 1 uplink module slot.	1
N9K-C9396PX	Nexus 9300 with 48p 1/10G SFP+ and 1 uplink module slot	2
N9k-X9736PQ	Spine Line-Card	2
APIC-CLUSTER-L1	APIC Appliance	3
N9K POWERCABLES	Power Cables	3
CAB-C13-C14-AC	Power cord, C13 to C14 (recessed receptacle), 10A	4
QSFP-H40G-CU3M	40GBASE-CR4 Passive Copper Cable, 3m	13
SFP-H10GB-CU3M	10GBASE-CU SFP+ Cable 3 Meter	87
N9K-M12PQ	ACI Uplink Module for Nexus 9300, 12p 40G QSFP	3
N9K-C9500-RMK	Nexus 9500 Rack Mount Kit	2
CAB-C19-CBN	Cabinet Jumper Power Cord, 250 VAC 16A, C20-C19 Connectors	6
N9K-C9500-LC-CV	Nexus 9500 Linecard slot cover16	16
N9K-C9500-SUP-CV	Nexus 9500 Supervisor slot cover	2
N9K-PAC-3000W-B	Nexus 9500 3000W AC PS, Port-side Intake	6
N9K-SUP-A	Supervisor for Nexus 9500	2
N9K-SC-A	System Controller for Nexus 9500	4
N9K FABRIC	Fabric Module	2
N9300 RACK	Rack Mount Kit	3
N9K-C9300-RMK	Nexus 9300 Rack Mount Kit	3

**Table 10** *Bill of Material for Three Rack of Servers (45 Servers)*

<b>Part Number</b>	<b>Description</b>	<b>Quantity</b>
UCS-SL-CPA2-PC	Performance and Capacity Balanced Cluster	3
UCSC-C240-M3S	UCS C240 M3 SFF w/o CPU mem HD PCIe w/ rail kit expdr	42
UCS-RAID9271CV-8I	MegaRAID 9271CV with 8 internal SAS/SATA ports with Supercap	42
UCSC-PCIE-CSC-02	Cisco VIC 1225 Dual Port 10Gb SFP+ CNA	42

Part Number	Description	Quantity
CAB-9K12A-NA	Power Cord 125VAC 13A NEMA 5-15 Plug North America	96
UCSC-PSU2-1200	1200W 2u Power Supply For UCS	96
UCSC-RAIL-2U	2U Rail Kit for UCS C-Series servers	42
UCSC-HS-C240M3	Heat Sink for UCS C240 M3 Rack Server	96
UCSC-PCIF-01F	Full height PCIe filler for C-Series	144
UCS-CPU-E52660B	2.20 GHz E5-2660 v2/95W 10C/25MB Cache/DDR3 1866MHz	84
UCS-MR-1X162RZ-A	16GB DDR3-1866-MHz RDIMM/PC3-14900/dual rank/x4/1.5v	672
UCS-HD1T7KS2-E	1TB SAS 7.2K RPM 2.5 inch HDD/hot plug/drive sled mounted	1008
SFP-H10GB-CU3M=	10GBASE-CU SFP+ Cable 3 Meter	93
RACK-UCS2	Cisco R42610 standard rack w/side panels	1
RP208-30-1P-U-2=	Cisco RP208-30-U-2 Single Phase PDU 20x C13 4x C19 (Country Specific)	2
CON-UCW3-RPDUX	UC PLUS 24X7X4 Cisco RP208-30-U-X Single Phase PDU 2x (Country Specific)	6

*Table 11 Red Hat Enterprise Linux License*

#### Red Hat Enterprise Linux License

RHEL-2S-1G-3A	Red Hat Enterprise Linux	42
CON-ISV1-RH2S1G3A	3 year Support for Red Hat Enterprise Linux	42

*Table 12 Cloudera Software*

#### Cloudera

NA [Procured directly from Cloudera]	Cloudera	42
--------------------------------------	----------	----