Red Hat Openstack Architecture on Cisco UCS Platform

Last Updated: February 4, 2014

Cisco Validated Design

Building Architectures to Solve Business Problems

About the Authors

Mehul Bhatt

Mehul Bhatt, Virtualization Architect, Server Access Virtualization Business Unit, Cisco Systems

Mehul Bhatt has over 12 years of Experience in virtually all layers of computer networking. His focus area includes Unified Compute Systems, network and server virtualization design. Prior to joining Cisco Technical Marketing team, Mehul was Technical Lead at Cisco, Nuova systems and Bluecoat systems. Mehul holds a Masters degree in computer systems engineering and holds various Cisco career certifications.

Acknowledgements

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

- Ashok Rajagopalan-Cisco
- Mike Andren-Cisco
- Aniket Patankar-Cisco
- Sindhu Sudhir-Cisco
- Sankar Jayaram-Cisco
- Karthik Prabhakar-Red Hat
- Steve Reichard-Red Hat

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

Red Hat Enterprise Linux Openstack Architecture on Cisco UCS Platform

Executive Summary

OpenStack is a free and open source Infrastructure-as-a-Service (IaaS) cloud computing project released under the Apache License. It enables enterprises and service providers to offer on-demand computing resources by provisioning and managing large networks of virtual machines. Red Hat's OpenStack technology uses upstream OpenStack open source architecture and enhances it for Enterprise and service provider customers with better support structure. The Cisco Unified Computing System is a next-generation data center platform that unites computing, network, storage access, and virtualization into a single cohesive system. Cisco UCS is an ideal platform for the Openstack architecture. Combination of Cisco UCS platform and Red Hat OpenStack architecture accelerates your IT Transformation by enabling faster deployments, greater flexibility of choice, efficiency, and lower risk. This Cisco Validate Design document focuses on the OpenStack on Red Hat Enterprise Linux architecture on UCS platform for small to medium size business segments.

Introduction

OpenStack boasts a massively scalable architecture that can control compute, storage, and networking resources through a unified web interface. The OpenStack development community operates on a six-month release cycle with frequent milestones. Their code base is composed of many loosely coupled projects supporting storage, compute, image management, identity, and networking services. OpenStack's rapid development cycle and architectural complexity create unique challenges for enterprise customers adding OpenStack to their traditional IT portfolios.

Red Hat's OpenStack technology addresses these challenges. Red Hat Enterprise Linux OpenStack Platform (RHEL OSP) 3, Red Hat's third OpenStack release, delivers a stable code base for production deployments backed by Red Hat's open source software expertise. Red Hat Enterprise Linux OpenStack Platform 3 adopters enjoy immediate access to bug fixes and critical security patches, tight integration with Red Hat's enterprise security features including SELinux, and a steady release cadence between OpenStack versions. This allows Red Hat customers to adopt OpenStack with confidence, at their own pace, and on their own terms.

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

Virtualization is a key and critical strategic deployment model for reducing the Total Cost of Ownership (TCO) and achieving better utilization of the platform components like hardware, software, network and storage. However choosing the appropriate platform for virtualization can be a tricky task. Platform should be flexible, reliable and cost effective to facilitate the virtualization platform to deploy various enterprise applications. Also ability to slice and dice the underlying platform to size the application requirement is essential for a virtualization platform to utilize compute, network and storage resources effectively. In this regard, Cisco UCS solution implementing Red Hat OpenStack provide a very simplistic yet fully integrated and validated infrastructure for you to deploy VMs in various sizes to suite your application needs.

Target Audience

The reader of this document is expected to have the necessary training and background to install and configure Red Hat Enterprise Linux and Cisco Unified Computing System (UCS) and Unified Computing Systems Manager as well as high level understanding of OpenStack components. External references are provided where applicable and it is recommended that the reader be familiar with these documents.

Readers are also expected to be familiar with the infrastructure and network and security policies of the customer installation.

Purpose of this Document

This document describes the steps required to deploy and configure Red Hat OpenStack architecture on Cisco UCS platform to a level that will allow for confirmation that the basic components and connections are working correctly. The document addresses Small- to Medium-sized Businesses; however the architecture can be very easily expanded with predictable linear performance. While readers of this document are expected to have sufficient knowledge to install and configure the products used, configuration details that are important to this solution’s deployment s are specifically mentioned.

Solution Overview

Red Hat OpenStack architecture on Cisco UCS Platform

This solution provides an end-to-end architecture with Cisco, Red Hat, and OpenStack technologies that demonstrate high availability and server redundancy along with ease of deployment and use.

The following are the components used for the design and deployment:

- Cisco Unified Compute System (UCS) 2.1(2)
- Cisco C-Series Unified Computing System servers for compute and storage needs
- Cisco UCS VIC adapters
- Red Hat OpenStack 3.0 architecture

The solution is designed to host scalable, mixed application workloads. The scope of this CVD is limited to the infrastructure pieces of the solution, the CVD does not address the vast area of OpenStack components and multiple configuration choices available there.

Technology Overview

Cisco Unified Computing System

The Cisco Unified Computing System is a next-generation data center platform that unites compute, network, and storage access. The platform, optimized for virtual environments, is designed using open industry-standard technologies and aims to reduce total cost of ownership (TCO) and increase business agility. The system integrates a low-latency; lossless 10 Gigabit Ethernet unified network fabric with enterprise-class, x86-architecture servers. It is an integrated, scalable, multi chassis platform in which all resources participate in a unified management domain.

The main components of Cisco Unified Computing System are:

- **Computing**—The system is based on an entirely new class of computing system that incorporates blade servers based on Intel Xeon E5-2600 V2 Series Processors. The Cisco UCS servers offer the patented Cisco Extended Memory Technology to support applications with large datasets and allow more virtual machines per server.
- **Network**—The system is integrated onto a low-latency, lossless, 10-Gbps unified network fabric. This network foundation consolidates LANs, SANs, and high-performance computing networks which are separate networks today. The unified fabric lowers costs by reducing the number of network adapters, switches, and cables, and by decreasing the power and cooling requirements.
- **Virtualization**—The system unleashes the full potential of virtualization by enhancing the scalability, performance, and operational control of virtual environments. Cisco security, policy enforcement, and diagnostic features are now extended into virtualized environments to better support changing business and IT requirements.
- **Storage access—**Cisco C-Series servers can host large number of local SATA hard disks. The system provides consolidated access to both SAN storage and Network Attached Storage (NAS) over the unified fabric. By unifying the storage access the Cisco Unified Computing System can access storage over Ethernet, Fibre Channel, Fibre Channel over Ethernet (FCoE), and iSCSI. This provides customers with choice for storage access and investment protection. In addition, the server administrators can preassign storage access policies for system connectivity to storage resources, simplifying storage connectivity, and management for increased productivity.

The Cisco Unified Computing System is designed to deliver:

- A reduced Total Cost of Ownership (TCO) and increased business agility.
- Increased IT staff productivity through just-in-time provisioning and mobility support.
- A cohesive, integrated system which unifies the technology in the data center.
- Industry standards supported by a partner ecosystem of industry leaders.

Cisco UCS Manager

Cisco UCS Manager provides unified, embedded management of all software and hardware components of the Cisco Unified Computing System through an intuitive GUI, a command line interface (CLI), or an XML API. The Cisco UCS Manager provides unified management domain with centralized management capabilities and controls multiple chassis and thousands of virtual machines.

Cisco UCS Fabric Interconnect

The Cisco® UCS 6200 Series Fabric Interconnect is a core part of the Cisco Unified Computing System, providing both network connectivity and management capabilities for the system. The Cisco UCS 6200 Series offers line-rate, low-latency, lossless 10 Gigabit Ethernet, Fibre Channel over Ethernet (FCoE) and Fibre Channel functions.

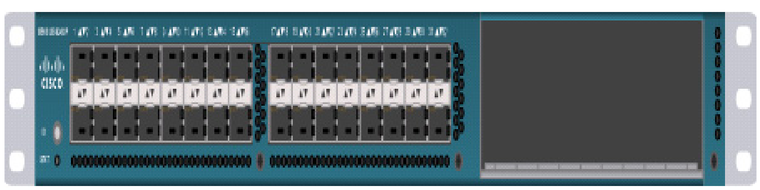
The Cisco UCS 6200 Series provides the management and communication backbone for the Cisco UCS B-Series Blade Servers and Cisco UCS 5100 Series Blade Server Chassis. All chassis, and therefore all blades, attached to the Cisco UCS 6200 Series Fabric Interconnects become part of a single, highly available management domain. In addition, by supporting unified fabric, the Cisco UCS 6200 Series provides both the LAN and SAN connectivity for all blades within its domain.

From a networking perspective, the Cisco UCS 6200 Series uses a cut-through architecture, supporting deterministic, low-latency, line-rate 10 Gigabit Ethernet on all ports, 1Tb switching capacity, 160 Gbps bandwidth per chassis, independent of packet size and enabled services. The product family supports Cisco low-latency, lossless 10 Gigabit Ethernet unified network fabric capabilities, which increase the reliability, efficiency, and scalability of Ethernet networks. The Fabric Interconnect supports multiple traffic classes over a lossless Ethernet fabric from a blade server through an interconnect. Significant TCO savings come from an FCoE-optimized server design in which network interface cards (NICs), host bus adapters (HBAs), cables, and switches can be consolidated.

Cisco UCS 6248UP Fabric Interconnect

The Cisco UCS 6248UP 48-Port Fabric Interconnect is a one-rack-unit (1RU) 10 Gigabit Ethernet, FCoE and Fiber Channel switch offering up to 960-Gbps throughput and up to 48 ports. The switch has 32 1/10-Gbps fixed Ethernet, FCoE and FC ports and one expansion slot.

Figure 1 Cisco UCS 6248UP Fabric Interconnect



Cisco UCS Fabric Extenders

Fabric Extenders are zero-management, low-cost, low-power consuming devices that distribute the system's connectivity and management planes into rack and blade chassis to scale the system without complexity. Designed never to lose a packet, Cisco fabric extenders eliminate the need for top-of-rack Ethernet and Fibre Channel switches and management modules, dramatically reducing infrastructure cost per server.

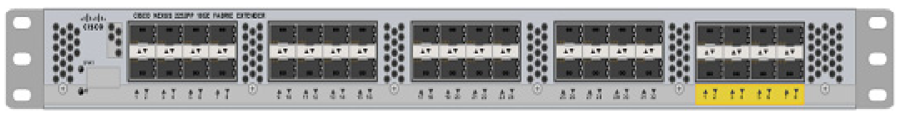
Cisco UCS 2232PP Fabric Extender

The Cisco Nexus® 2000 Series Fabric Extenders comprise a category of data center products designed to simplify data center access architecture and operations. The Cisco Nexus 2000 Series uses the Cisco® Fabric Extender architecture to provide a highly scalable unified server-access platform across a range of 100 Megabit Ethernet, Gigabit Ethernet, 10 Gigabit Ethernet, unified fabric, copper and fiber connectivity, rack, and blade server environments. The platform is ideal to support today's traditional Gigabit Ethernet while allowing transparent migration to 10 Gigabit Ethernet, virtual machine-aware unified fabric technologies.

The Cisco Nexus 2000 Series Fabric Extenders behave as remote line cards for a parent Cisco Nexus switch or Fabric Interconnect. The fabric extenders are essentially extensions of the parent Cisco UCS Fabric Interconnect switch fabric, with the fabric extenders and the parent Cisco Nexus switch together forming a distributed modular system. This architecture enables physical topologies with the flexibility and benefits of both top-of-rack (ToR) and end-of-row (EoR) deployments.

Today's data centers must have massive scalability to manage the combination of an increasing number of servers and a higher demand for bandwidth from each server. The Cisco Nexus 2000 Series increases the scalability of the access layer to accommodate both sets of demands without increasing management points within the network.

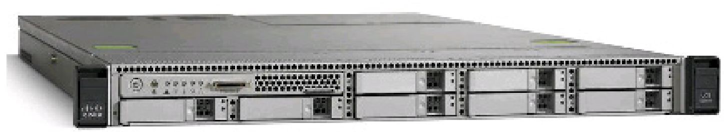
Figure 2 Cisco UCS 2232PP Fabric Extender



Cisco C220 M3 Rack Mount Servers

Building on the success of the Cisco UCS C220 M3 Rack Servers, the enterprise-class Cisco UCS C220 M3 server further extends the capabilities of the Cisco Unified Computing System portfolio in a 1-rack-unit (1RU) form factor. And with the addition of the Intel® Xeon® processor E5-2600 product family, it delivers significant performance and efficiency gains.

Figure 3 Cisco UCS C220 M3 Rack Mount Server



The Cisco UCS C220 M3 also offers up to 256 GB of RAM, eight drives or SSDs, and two 1GE LAN interfaces built into the motherboard, delivering outstanding levels of density and performance in a compact package.

Cisco C240 M3 Rack Mount Servers

The UCS C240 M3 High Density Small Form Factory Disk Drive Model rack server is designed for both performance and expandability over a wide range of storage-intensive infrastructure workloads from big data to collaboration. The enterprise-class UCS C240 M3 server extends the capabilities of Cisco's Unified Computing System portfolio in a 2U form factor with the addition of the Intel® Xeon E5-2600 v2 and E5-2600 series processor family CPUs that deliver the best combination of performance, flexibility and efficiency gains. In addition, the UCS C240 M3 server provides 24 DIMM slots, up to 24 drives and 4 x 1 GbE LOM ports to provide outstanding levels of internal memory and storage expandability along with exceptional performance.

Figure 4 Cisco UCS C240 M3 Rack Mount Server

Cisco I/O Adapters

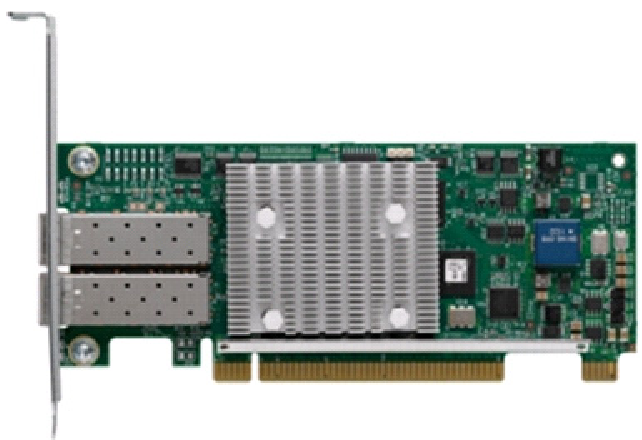
The Cisco UCS rack mount server has various Converged Network Adapters (CNA) options. The UCS 1225 Virtual Interface Card (VIC) option is used in this Cisco Validated Design.

A Cisco® innovation, the Cisco UCS Virtual Interface Card (VIC) 1225 is a dual-port Enhanced Small Form-Factor Pluggable (SFP+) 10 Gigabit Ethernet and Fibre Channel over Ethernet (FCoE)-capable PCI Express (PCIe) card designed exclusively for Cisco UCS C-Series Rack Servers.

UCS 1225 VIC provides the capability to create multiple vNICs (up to 128) on the CNA. This allows complete I/O configurations to be provisioned in virtualized or non-virtualized environments using just-in-time provisioning, providing tremendous system flexibility and allowing consolidation of multiple physical adapters.

System security and manageability is improved by providing visibility and portability of network policies and security all the way to the virtual machines. Additional 1225 features like VM-FEX technology and pass-through switching, minimize implementation overhead and complexity.

Figure 5 Cisco UCS 1225 VIC



UCS 2.1 Singe Wire Management

Cisco UCS Manager 2.1 supports an additional option to integrate the C-Series Rack Mount Server with Cisco UCS Manager called "single-wire management". This option enables Cisco UCS Manager to manage the C-Series Rack-Mount Servers using a single 10 GE link for both management traffic and data traffic. When you use the single-wire management mode, one host facing port on the FEX is sufficient to manage one rack-mount server, instead of the two ports you will use in the Shared-LOM mode. Cisco VIC 1225, Cisco UCS 2232PP FEX and Single-Wire management feature of UCS 2.1

tremendously increases the scale of C-Series server manageability. By consuming as little as one port on the UCS Fabric Interconnect, you can manage up to 32 C-Series server using single-wire management feature.

UCS Differentiators

Cisco's Unified Compute System is revolutionizing the way servers are managed in data-center. Following are the unique differentiators of UCS and UCS Manager.

1. **Embedded management**—In UCS, the servers are managed by the embedded firmware in the Fabric Interconnects, eliminating need for any external physical or virtual devices to manage the servers. Also, a pair of FIs can manage up to 40 chassis, each containing 8 blade servers. This gives enormous scaling on the management plane.
2. **Unified fabric**—In UCS, from blade server chassis or rack server fabric-extender to FI, there is a single Ethernet cable used for LAN, SAN and management traffic. This converged I/O results in reduced cables, SFPs and adapters – reducing capital and operational expenses of overall solution.
3. **Auto Discovery**—By simply inserting the blade server in the chassis or connecting rack server to the fabric extender, discovery and inventory of compute resource occurs automatically without any management intervention. The combination of unified fabric and auto-discovery enables the wire-once architecture of UCS, where compute capability of UCS can be extended easily while keeping the existing external connectivity to LAN, SAN and management networks.
4. **Policy based resource classification**—Once a compute resource is discovered by UCS Manager, it can be automatically classified to a given resource pool based on policies defined. This capability is useful in multi-tenant cloud computing. This CVD showcases the policy based resource classification of UCS Manager.
5. **Combined Rack and Blade server management**—UCS Manager can manage B-series blade servers and C-series rack server under the same UCS domain. This feature, along with stateless computing makes compute resources truly hardware form factor agnostic. In this CVD, we are showcasing combinations of B and C series servers to demonstrate stateless and form-factor independent computing work load.
6. **Model based management architecture**—UCS Manager architecture and management database is model based and data driven. An open, standard based XML API is provided to operate on the management model. This enables easy and scalable integration of UCS Manager with other management system, such as VMware vCloud director, Microsoft System Center, and Citrix Cloud Platform.
7. **Policies, Pools, Templates**—The management approach in UCS Manager is based on defining policies, pools and templates, instead of cluttered configuration, which enables a simple, loosely coupled, data driven approach in managing compute, network and storage resources.
8. **Loose referential integrity**—In UCS Manager, a service profile, port profile or policies can refer to other policies or logical resources with loose referential integrity. A referred policy cannot exist at the time of authoring the referring policy or a referred policy can be deleted even though other policies are referring to it. This provides different subject matter experts to work independently from each-other. This provides great flexibility where different experts from different domains, such as network, storage, security, server and virtualization work together to accomplish a complex task.
9. **Policy resolution**—In UCS Manager, a tree structure of organizational unit hierarchy can be created that mimics the real life tenants and/or organization relationships. Various policies, pools and templates can be defined at different levels of organization hierarchy. A policy referring to another policy by name is resolved in the organization hierarchy with closest policy match. If no policy with

specific name is found in the hierarchy of the root organization, then special policy named "default" is searched. This policy resolution practice enables automation friendly management APIs and provides great flexibility to owners of different organizations.

10. **Service profiles and stateless computing**—A service profile is a logical representation of a server, carrying its various identities and policies. This logical server can be assigned to any physical compute resource as far as it meets the resource requirements. Stateless computing enables procurement of a server within minutes, which used to take days in legacy server management systems.
11. **Built-in multi-tenancy support**—The combination of policies, pools and templates, loose referential integrity, policy resolution in organization hierarchy and a service profiles based approach to compute resources makes UCS Manager inherently friendly to multi-tenant environment typically observed in private and public clouds.
12. **Extended Memory**—The extended memory architecture of UCS servers allows up to 760 GB RAM per server – allowing huge VM to physical server ratio required in many deployments, or allowing large memory operations required by certain architectures like Big-Data.
13. **Virtualization aware network**—VM-FEX technology makes access layer of network aware about host virtualization. This prevents domain pollution of compute and network domains with virtualization when virtual network is managed by port-profiles defined by the network administrators' team. VM-FEX also off loads hypervisor CPU by performing switching in the hardware, thus allowing hypervisor CPU to do more virtualization related tasks. VM-FEX technology is well integrated with VMware vCenter, Linux KVM and Hyper-V SR-IOV to simplify cloud management.
14. **Simplified QoS**—Even though Fibre Channel and Ethernet are converged in UCS fabric, built-in support for QoS and lossless Ethernet makes it seamless. Network Quality of Service (QoS) is simplified in UCS Manager by representing all system classes in one GUI panel.

Red Hat Enterprise Linux OpenStack Architecture

Red Hat Enterprise Linux OpenStack Platform provides the foundation to build private or public Infrastructure-as-a-Service (IaaS) for cloud-enabled workloads. It allows organizations to leverage OpenStack, the largest and fastest growing open source cloud infrastructure project, while maintaining the security, stability, and enterprise readiness of a platform built on Red Hat Enterprise Linux.

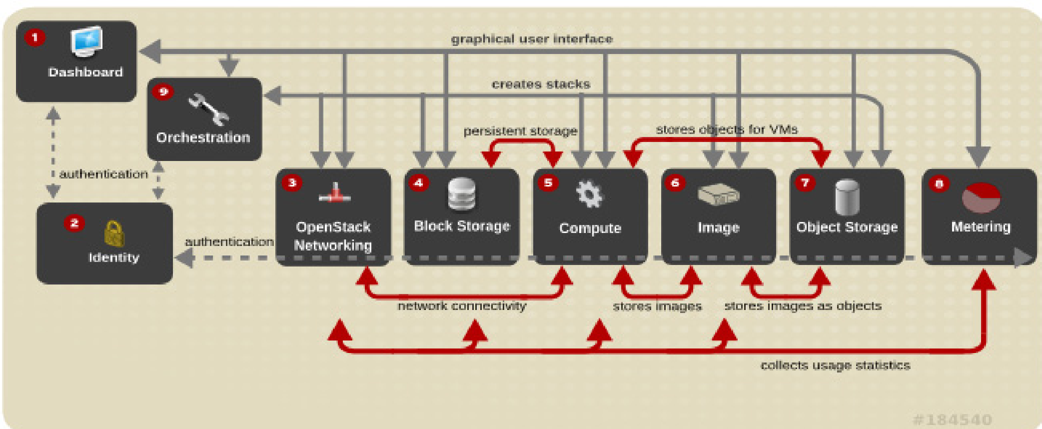
Red Hat Enterprise Linux OpenStack Platform gives organizations a truly open framework for hosting cloud workloads, delivered by Red Hat subscription for maximum flexibility and cost effectiveness. In conjunction with other Red Hat technologies, Red Hat Enterprise Linux OpenStack Platform allows organizations to move from traditional workloads to cloud-enabled workloads on their own terms and time lines, as their applications require. Red Hat frees organizations from proprietary lock-in, and allows them to move to open technologies while maintaining their existing infrastructure investments.

Unlike other OpenStack distributions, Red Hat Enterprise Linux OpenStack Platform provides a certified ecosystem of hardware, software, and services, an enterprise life cycle that extends the community OpenStack release cycle, and award-winning Red Hat support on both the OpenStack modules and their underlying Linux dependencies. Red Hat delivers long-term commitment and value from a proven enterprise software partner so organizations can take advantage of the fast pace of OpenStack development without risking the stability and supportability of their production environments.

Red Hat Enterprise Linux OpenStack Platform 3 ("Grizzly") Services

Red Hat Enterprise Linux OpenStack Platform 3 is based on the upstream "Grizzly" OpenStack release. Red Hat Enterprise Linux OpenStack Platform 3 is Red Hat third release. The first release was based on the "Essex" OpenStack release. The second release was based on the "Folsom" OpenStack release. It was the first release to include extensible block and volume storage services. Grizzly includes all of Folsom's features along with a more robust network automation platform and support for metering and orchestration.

Figure 6 OpenStack Platform 3 Services



Identity Service ("Keystone")

This is a central authentication and authorization mechanism for all OpenStack users and services. It supports multiple forms of authentication including standard username and password credentials, token-based systems and AWS-style logins that use public/private key pairs. It can also integrate with existing directory services such as LDAP.

The Identity service catalog lists all of the services deployed in an OpenStack cloud and manages authentication for them through endpoints. An endpoint is a network address where a service listens for requests. The Identity service provides each OpenStack service – such as Image, Compute, or Block Storage -- with one or more endpoints.

The Identity service uses tenants to group or isolate resources. By default users in one tenant can't access resources in another even if they reside within the same OpenStack cloud deployment or physical host. The Identity service issues tokens to authenticated users. The endpoints validate the token before allowing user access. User accounts are associated with roles that define their access credentials. Multiple users can share the same role within a tenant.

The Identity Service is comprised of the keystone service, which responds to service requests, places messages in queue, grants access tokens, and updates the state database.

Image Service (“Glance”)

This service discovers, registers, and delivers virtual machine images. They can be copied via snapshot and immediately stored as the basis for new instance deployments. Stored images allow OpenStack users and administrators to provision multiple servers quickly and consistently. The Image Service API provides a standard RESTful interface for querying information about the images.

By default the Image Service stores images in the /var/lib/glance/images directory of the local server’s file system where Glance is installed. The Glance API can also be configured to cache images in order to reduce image staging time. The Image Service supports multiple back end storage technologies including Swift (the OpenStack Object Storage service), Amazon S3, and Red Hat Storage Server.

The Image service is composed of the openstack-glance-api that delivers image information from the registry service, and the openstack-glance-registry which manages the metadata associated with each image.

Compute Service (“Nova”)

OpenStack Compute provisions and manages large networks of virtual machines. It is the backbone of OpenStack’s IaaS functionality. OpenStack Compute scales horizontally on standard hardware enabling the favorable economics of cloud computing. Users and administrators interact with the compute fabric via a web interface and command line tools.

Key features of OpenStack Compute include:

- Distributed and asynchronous architecture, allowing scale out fault tolerance for virtual machine instance management
- Management of commoditized virtual server resources, where predefined virtual hardware profiles for guests can be assigned to new instances at launch
- Tenants to separate and control access to compute resources
- VNC access to instances via web browsers

OpenStack Compute is composed of many services that work together to provide the full functionality. The openstack-nova-cert and openstack-nova-consoleauth services handle authorization. The openstack-nova-api responds to service requests and the openstack-nova-scheduler dispatches the requests to the message queue. The openstack-nova-conductor service updates the state database which limits direct access to the state database by compute nodes for increased security. The openstacknova-compute service creates and terminates virtual machine instances on the compute nodes. Finally, openstack-nova-novncproxy provides a VNC proxy for console access to virtual machines via a standard web browser.

Block Storage (“Cinder”)

While the OpenStack Compute service provisions ephemeral storage for deployed instances based on their hardware profiles, the OpenStack Block Storage service provides compute instances with persistent block storage. Block storage is appropriate for performance sensitive scenarios such as databases or frequently accessed file systems. Persistent block storage can survive instance termination. It can also be moved between instances like any external storage device. This service can be backed by a variety of enterprise storage platforms or simple NFS servers. This service’s features include:

- Persistent block storage devices for compute instances
- Self-service user creation, attachment, and deletion

- A unified interface for numerous storage platforms
- Volume snapshots

The Block Storage service is comprised of openstack-cinder-api which responds to service requests and openstack-cinder-scheduler which assigns tasks to the queue. The openstack-cinder-volume service interacts with various storage providers to allocate block storage for virtual machines. By default the Block Storage server shares local storage via the ISCSI tgtd daemon.

Network Service ("Neutron")

OpenStack Networking is a scalable API-driven service for managing networks and IP addresses. OpenStack Networking gives users self-service control over their network configurations. Users can define, separate, and join networks on demand. This allows for flexible network models that can be adapted to fit the requirements of different applications.

OpenStack Networking has a pluggable architecture that supports numerous physical networking technologies as well as native Linux networking mechanisms including openvswitch and linuxbridge.

OpenStack Networking is composed of several services. The quantum-server exposes the API and responds to user requests. The quantum-l3-agent provides L3 functionality, such as routing, through interaction with the other networking plug-ins and agents. The quantum-dhcp-agent provides DHCP to tenant networks. There are also a series of network agents that perform local networking configuration for the node's virtual machines.

Note In previous OpenStack versions the Network Service was named Quantum. In the Grizzly release Quantum was renamed to Neutron. However, many of the command line utilities in RHOS 3.0 retain the legacy name.

Dashboard ("Horizon")

The OpenStack Dashboard is an extensible web-based application that allows cloud administrators and users to control and provision compute, storage, and networking resources. Administrators can use the Dashboard to view the state of the cloud, create users, assign them to tenants, and set resource limits. The OpenStack Dashboard runs as an Apache HTTP server via the httpd service.

Note Both the Dashboard and command line tools can be used to manage an OpenStack environment. This document focuses on the command line tools because they offer more granular control and insight into OpenStack's functionality.

Object Store Service ("Swift")

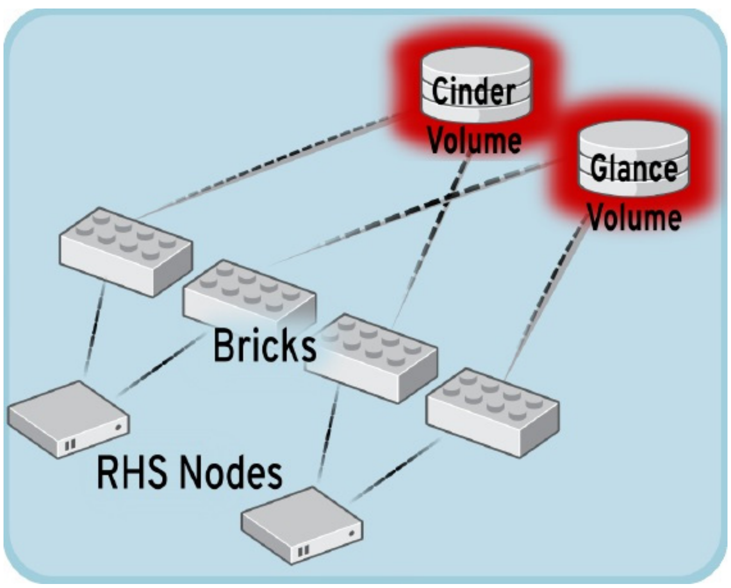
The OpenStack Object Storage service provides a fully distributed, API-accessible storage platform that can be integrated directly into applications or used for backup, archiving and data retention. It provides redundant, scalable object storage using clusters of standardized servers capable of storing petabytes of data. Object Storage is not a traditional file system, but rather a distributed storage system for static data. Objects and files are written to multiple disks spread throughout the data center. Storage clusters scale horizontally simply by adding new servers. The OpenStack Object Storage service is not discussed in this reference architecture. Red Hat Storage Server offers many of the core functionalities of this service.

Red Hat Storage for Server

Red Hat Storage Server (RHSS) is an enterprise storage solution that enables enterprise-wide storage sharing with a single access point across data storage locations. It has a scaleout, network-attach architecture to accommodate exponential data growth. Red Hat Enterprise Linux OpenStack Platform 3 does not depend on Red Hat Storage Server, but in this reference architecture RHSS is the back end storage for both the Block and Image Services. The Red Hat Storage client driver enables block storage support. Gluster volumes are used to store virtual images.

The RHS cluster is composed of two servers. Each server contains two local XFS file systems called bricks. One brick from each RHS Server is combined with a corresponding brick on the other RHS Server to make a replicated volume. Therefore, the RHS Servers present two replicated volumes – one for the Image Service and one for Block Storage Service – composed of four bricks. Both volumes are synchronously replicated. If either RHS Server becomes unavailable, all data is still available via the remaining node.

Figure 7 Red Hat Storage Server Architecture Overview



Red Hat Enterprise Linux

Red Hat Enterprise Linux 6, the latest release of Red Hat trusted data center platform, delivers advances in application performance, scalability, and security. With Red Hat Enterprise Linux 6, physical, virtual, and cloud computing resources can be deployed within the data center.

Note This reference architecture is based on Red Hat Enterprise Linux 6.4. However, Red Hat Enterprise Linux OpenStack Platform 3 uses a non-standard kernel version 2.6.32-358.114.1.openstack in order to support NETWORK NAMESPACES. Many of the robust features of OpenStack networking such as duplicate IP address ranges across tenants require network namespaces.

Supporting Technologies

This section describes the supporting technologies used to develop this reference architecture beyond the OpenStack services and core operating system. Supporting technologies include:

- MySQL

A state database resides at the heart of an OpenStack deployment. This SQL database stores most of the build-time and run-time state information for the cloud infrastructure including available instance types, networks, and the state of running instances in the compute fabric. Although OpenStack theoretically supports any SQL-Alchemy compliant database, Red Hat Enterprise Linux OpenStack Platform 3 uses MySQL, a widely used open source database packaged with Red Hat Enterprise Linux 6.

- Qpid

OpenStack services use enterprise messaging to communicate tasks and state changes between clients, service endpoints, service scheduler, and instances. Red Hat Enterprise Linux OpenStack Platform 3 uses Qpid for open source enterprise messaging. Qpid is an Advanced Message Queuing Protocol (AMQP) compliant, cross-platform enterprise messaging system developed for low latency based on an open standard for enterprise messaging. Qpid is released under the Apache open source license.

- KVM

Kernel-based Virtual Machine (KVM) is a full virtualization solution for Linux on x86 and x86_64 hardware containing virtualization extensions for both Intel and AMD processors. It consists of a loadable kernel module that provides the core virtualization infrastructure. Red Hat Enterprise Linux OpenStack Platform Compute uses KVM as its underlying hypervisor to launch and control virtual machine instances.

- Packstack

Packstack is a Red Hat Enterprise Linux OpenStack Platform 3 installer. Packstack uses Puppet modules to install parts of OpenStack via SSH. Puppet modules ensure OpenStack can be installed and expanded in a consistent and repeatable manner. This reference architecture uses Packstack for a multi-server deployment. Through the course of this reference architecture, the initial Packstack installation is modified with OpenStack Network and Storage service enhancements.

Architectural overview

This CVD focuses on the architecture for Red Hat OpenStack 3 on UCS platform using Cisco UCS C-series servers for storage. Cisco UCS C220 M3 servers are used as compute nodes and UCS C240 M3 servers are used as storage nodes. Storage high availability and redundancy are achieved using Red Hat Storage Server on OpenStack. UCS C-series servers are managed by UCSM, which provides ease of infrastructure management and built-in network high availability.

Table 1 lists the various hardware and software components which occupies different tiers of the architecture under test:

Table 1 Hardware and Software Components of the Architecture

Vendor	Name	Version	Description
Cisco	Cisco UCS Manager	2.1(3a)	Cisco UCS Manager software
Cisco	Cisco VIC 1225	2.1(3a)	Cisco Virtual Interface Card (adapter) firmware
Cisco	Cisco UCS 6248UP Fabric Interconnect	5.0(3)N2(2.11)	Cisco UCS fabric inter-connect firmware
Cisco	Cisco 2232PP Fabric Extender	5.0(3)N2(2.11.2)	Cisco UCS Fabric Extender
Cisco	Cisco UCS C220M3 Servers	1.5(2) or later – CIMC C220M3.1.5.2.23 - BIOS	Cisco UCS C220M3 Rack Servers
Cisco	Cisco UCS C240M3 Servers	1.5(2) or later – CIMC C220M3.1.5.2.23 - BIOS	Cisco UCS C240M3 Rack Servers
Red Hat	Red Hat Enterprise Linux	2.6.32-358.118.1.open-stack.el6.x86_64	Red Hat Enterprise Linux 6.4 release

Table 2 outlines the C220M3 server configuration, used as compute nodes in this architecture (per server basis).

Table 2 Server Configuration Details

Component	Capacity
Memory (RAM)	128 GB (16 X 8 GB DIMM)
Processor	2 x Intel® Xenon ® E5-2600 V2, CPUs 2.0 GHz, 8cores, 16 threads
Local storage	Cisco UCS RAID SAS 2008M-8i Mezzanine Card, With 6 x 300 GB disks for RAID6 configuration

Table 3 outlines the C240M3 server configuration, used as storage nodes in this architecture (per server basis).

Table 3 Server Configuration Details

Component	Capacity
Memory (RAM)	128 GB (16 X 8 GB DIMM)

Table 3 Server Configuration Details

Component	Capacity
Processor	2 x Intel® Xenon ® E5-2600 V2, CPUs 2.0 GHz, 8cores, 16 threads
Local storage	LSI 6G MegaRAID SAS 9266-8i, With 24 x 1 TB disks, with RAID1 and RAID0 configuration

Figure 8 show a high level architecture.

Figure 8 Reference Architecture

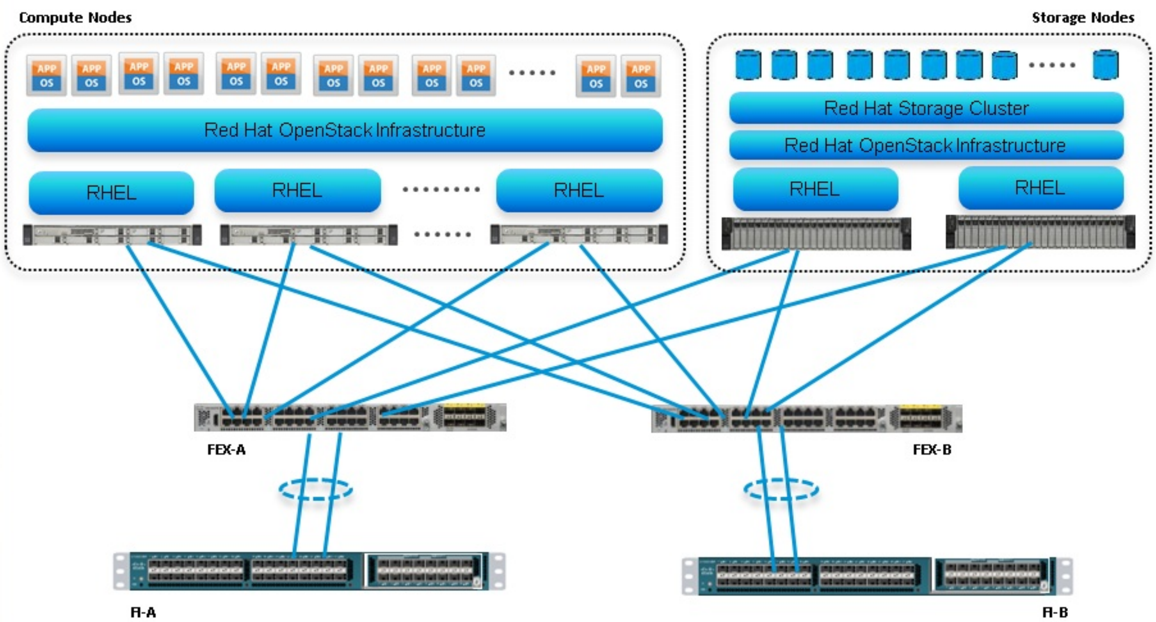


Figure 8 highlights the high level design points of Red Hat OpenStack architecture on UCS Platform:

- Redundant UCS FIs, Fabric Extenders and multiple cables provide network high availability
- Multiple hard disks per storage node combined with multiple storage nodes provide storage high availability through Red Hat Storage Cluster module.
- Infrastructure network is on a separate 1GE network. Out of band UCS management and other legacy infrastructure components, such as Syslog server, are connected to infrastructure network.

This design does not dictate or require any specific layout of infrastructure network. The Out Of Band UCS Manager access, hosting of supporting infrastructure such as Syslog server are hosted on infrastructure network. However, design does require accessibility of certain VLANs from the infrastructure network to reach the servers.

Virtual Networking

This architecture demonstrates use and benefits of Adapter-FEX technology using Cisco UCS VIC adapter. Each C220 M3 and C240 M3 server has one Cisco VIC 1225 physical adapter with two 10 GE links going to fabric A and fabric B for high availability. Cisco UCS VIC 1225 presents two virtual Network Interface Cards (vNICs) to the hypervisor with two virtual interfaces (one on each fabric) in active/passive mode. These vNICs are capable to do fabric failover, so if the Fabric Extender of Fabric Interconnect reboots or all the uplinks on the FI are lost, the vNIC would move traffic from fabric A to fabric B (or vice-a-versa) transparently. The MAC addresses to these vNICs are assigned using MAC address pool defined on the UCSM.

In the hypervisor layer, this architecture is using Neutron (Quantum) networking layer, with Open-vSwitch for virtual networking. Different VLANs are used for different tenants for logical separation of domains. Within a given tenant's realm, different VLANs can be used on per tier basis too in case of multi-tier applications. In other words, architecture does not dictate one VLAN per tenant.

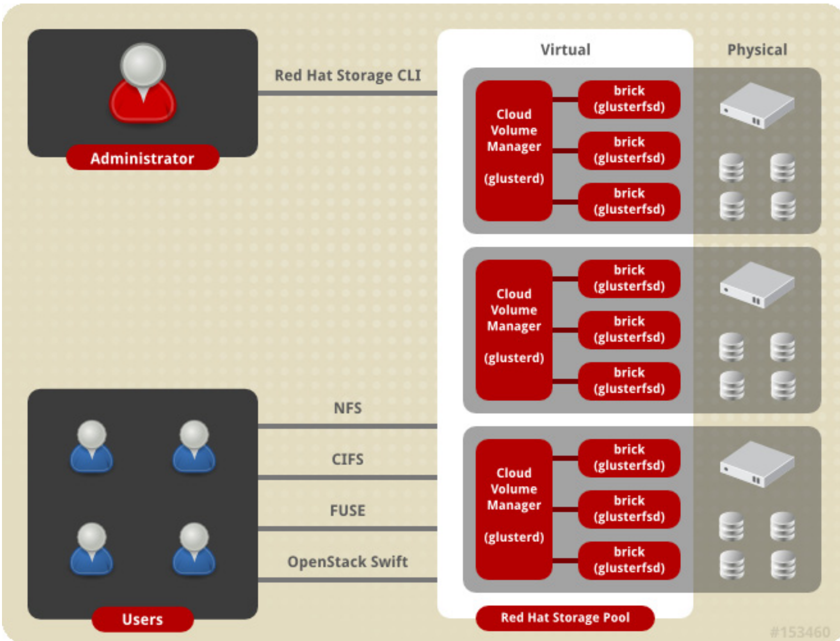
Storage Virtualization

There are 24 x 1TB SAS disks per C240 M3 server. First two disks are put in RAID 1 configuration and is the bootable device. RHEL 6.4 is installed on this RAID 1 volume. All remaining 22 disks are configured as individual disks in RAID0 configuration. In Linux terminology, /dev/sda is where OS is installed and the disks /dev/sdb to /dev/sdw are available to Cinder for Red Hat Storage Cluster as storage devices.

At the heart of the Red Hat Storage design is a completely new view of how storage should be architected. The result is a system that has immense scalability, is highly resilient, and offers extraordinary performance.

In a scale-out system, one of the biggest challenges is to keep track of the logical and physical location of data (and metadata). Most distributed systems solve this problem by creating a metadata server which keeps track of data and location of metadata. This creates both a central point of failure and a huge performance bottleneck. As traditional systems add more files, more servers, or more disks, the central metadata server becomes a performance bottleneck. Unlike other traditional solutions, Red Hat Storage does not need a metadata server and locates files algorithmically using the elastic hashing algorithm. This no-metadata server architecture ensures better performance, linear scalability, and reliability.

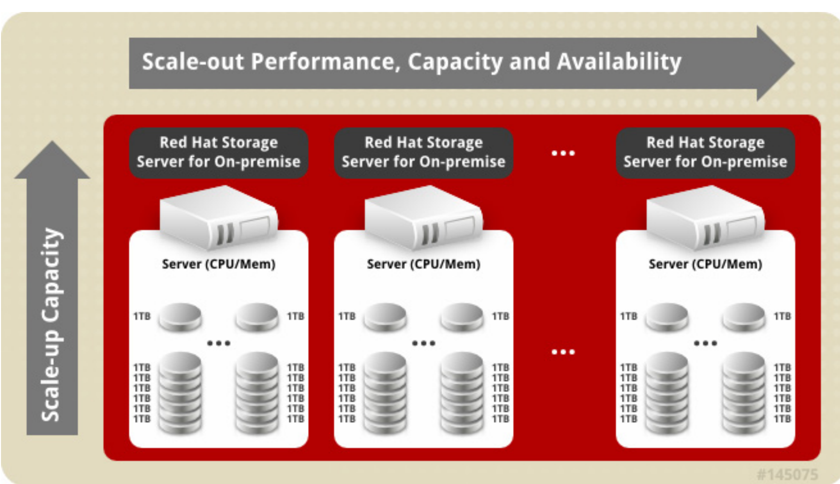
Figure 9 Red Hat Storage Architecture



The Red Hat Storage Server enables enterprises to treat physical storage as a virtualized, scalable, and centrally managed pool of storage by using local hard drives on the servers. It supports multi-tenancy by partitioning users or groups into logical volumes on shared storage. It enables users to eliminate, manage and improve their dependence on high cost, monolithic, and difficult deployment storage arrays.

You can add capacity in a matter of minutes across a wide variety of workloads without affecting performance. Storage can also be centrally managed across a variety of workloads thus increasing storage efficiency.

Figure 10 Red Hat Storage Server Scaling



Red Hat Storage Server for On-Premise is based on glusterFS, an open source distributed file system with a modular, stackable design, and a unique no-metadata server architecture. This no-metadata server architecture ensures better performance, linear scalability, and reliability.

Service Profile Design

This architecture implements following design steps to truly achieve stateless computing on the servers:

- Service profiles are derived from service profile template for consistency.
- The RHEL host uses following identities in this architecture:
  - Host UUID
  - Mac Addresses: one per each vNIC on the server

  All of these identifiers are defined in their respective identifier pools and the pool names are referred in the service profile template.
- Server pools are defined with automatic qualification policy and criteria. Rack servers are automatically put in the pool as and when they are fully discovered by UCS Manager. This eliminates the need to manually assign servers to server pool.
- Service profile template is associated to the server pool. This eliminates the need to individually associating service profiles to physical servers.

Given this design and capabilities of UCS and UCS Manager, a new server can be procured within minutes if the scale needs to be increased or if a server needs to be replaced by different hardware. In case, if a server has physical fault (faulty memory, or PSU or fan, for example), using following steps, a new server can be procured within minutes:

- Put the faulty server in maintenance mode. This would move VMs running on fault server to other healthy servers on the cluster.
- Disassociate the service profile from the faulty server and physically remove the server for replacement of faulty hardware (or to completely remove the faulty server).
- Physically install the new server and connect it to the Fabric Extenders. Let the new server be discovered by UCS Manager.
- Associate the service profile to the newly deployed rack server and install RHEL on the local disk.
- The new server would assume the role of the old server with all the identifiers intact.

Given that this architecture assumes deployment of OpenStack from scratch, there is no external image repository available. Once, storage nodes are up and running, you can even host the images. Thus, the architecture achieves the true statelessness of the computing in the data-center. If there are enough identifiers in all the id-pools, and if more servers are attached to UCS system in future, more service profiles can be derived from the service profile template and the private cloud infrastructure can be easily expanded.

Network High Availability Design

Following are the key aspects of this solution:

- Cisco adapter-FEX technology to introduce virtual NICs to host OS
- Fabric failover feature of adapter-FEX is exploited to provide high availability
- Two 10GE links between FI and FEX provides enough bandwidth over subscription for the given size of cloud. The over subscription can be reduced by adding more 10GE links between FI and FEX if needed by the VMs running on the hosts.
- Two vNICs per host – one for private network within the OpenStack environment and one for the public access of the Linux hosts.

- All the hosts are divided in two groups – one having their active data network on fabric A and one having their active data network on fabric B. This achieves fair load balancing on two fabrics in addition to the redundancy.
- All key OpenStack services are running on more than one host to make it highly available. See the following section for more details on OpenStack services placement.

OpenStack Services Placement

Table 4 shows the final service placement for all OpenStack services. The API-listener services (including quantum-server) run on the cloud controller in order to field client requests. The Network node runs all other Network services except for those necessary for Nova client operations, which also run on the Compute nodes. The Dashboard runs on the client system to prevent self-service users from accessing the cloud controller directly.

Table 4 Service Placement

Host Name	Role	Services
rhos-node1	Compute, Controller	openstack-nova-compute, quantum-openvswitch-agent, *-api,
rhos-node2	Compute	openstack-nova-compute, quantum-openvswitch-agent, openstack-keystone
rhos-node3	Compute, Controller	openstack-nova-compute, quantum-openvswitch-agent, *-api
rhos-node4	Compute	openstack-nova-compute, quantum-openvswitch-agent,
rhos-node5	Compute	openstack-nova-compute, quantum-openvswitch-agent,
rhos-node6	Compute	openstack-nova-compute, quantum-openvswitch-agent,
rhos-storage-node1	Storage	openstack-cinder-volume, quantum-openvswitch-agent, openstack-glance-registry, openstack-glance-scrubber
rhos-storage-node2	Storage	openstack-cinder-volume, quantum-openvswitch-agent, openstack-glance-registry, openstack-glance-scrubber

Sizing Guidelines

In any discussion about virtual infrastructures, it is important to first define a reference workload. Not all servers perform the same tasks, and it is impractical to build a reference that takes into account every possible combination of workload characteristics.

Defining the Reference Workload

To simplify the discussion, we have defined a representative customer reference workload. By comparing your actual customer usage to this reference workload, you can extrapolate which reference architecture to choose.

OpenStack defines various reference VMs as shown in Table 5.

Table 5 Virtual Machine Characteristics

Instance Flavor	Parameters
Tiny	512 MB RAM, No disk, 1 vCPU
Small	2 GB RAM, 20 GB disk, 1 vCPU
Medium	4 GB RAM, 40 GB disk, 2 vCPU
Large	8 GB RAM, 80 GB disk, 4 vCPU
Extra Large	16 GB RAM, 160 GB disk, 8 vCPU

This specification for a virtual machine is not intended to represent any specific application. Rather, it represents a single common point of reference to measure other virtual machines.

You must design your cloud to provide N + 1 hosts high availability. In order to do so, consider the largest resource required by all the VMs, divide it by the single physical server resources and round it up. This would give you required number of hosts. Add one more host to provide N+1 HA.

For example, all the instances required to run on your cloud would require combined 620 GB of RAM. With 128 GB RAM per server, this would require 5 servers. To provide N + 1 HA, you would need 6 compute nodes and divide the load across all the hosts. In this case, if one of the hosts has to go down for maintenance, remaining servers can still carry the load of all instances. This example assumes that RAM requirements is the highest across all instances.

Configuration Guidelines

The configuration for Red Hat OpenStack architecture on UCS Platform is divided in to following steps:

1. Connecting network cables
2. Preparing UCS FIs and configure UCSM
3. Configuring local disks of the storage nodes
4. Installing RHEL servers
5. Installing OpenStack packages on the servers
6. Running PackStack to configure OpenStack
7. Testing the installation

Connecting Network Cables

See the Cisco UCS FI, FEX, and C-series server configuration guide for detailed information about how to mount the hardware on the rack. Following diagrams show connectivity details for the architecture covered in this document.

As shown in the following figure, there are four major cabling sections in this architecture:

1. Upstream Connectivity (shown in purple)
2. FIs to Fabric Extenders links (shown in blue)
3. Fabric Extenders to C220M3 server links (shown in green)
4. Infrastructure connectivity (not shown)

Figure 11 Detailed Connectivity Diagram of the Architecture

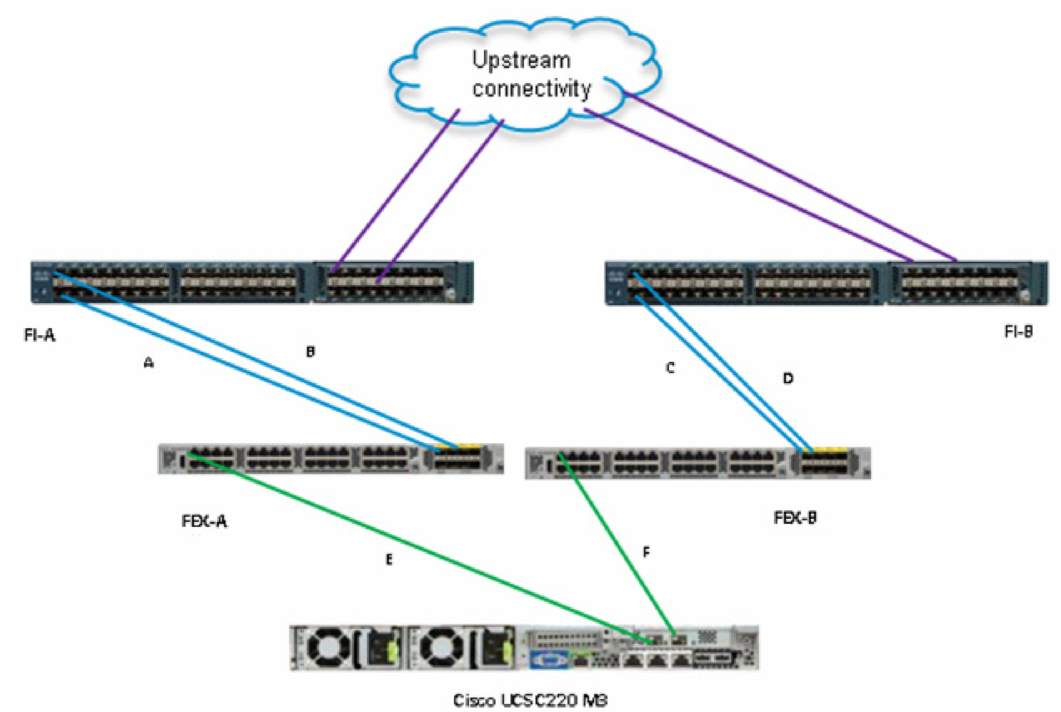


Figure 12 elaborates the detailed cable connectivity for the architecture.

Figure 12 Connectivity Details of the Architecture

Cable ID	Peer 1	Peer 2	VLAN	Mode	Description
A, B	FI-A, Eth 1/1, 1/2	FEX-A uplinks	N/A	Server	FI/FEX 20GE port-channel connectivity
C, D	FI-A, Eth 1/1, 1/2	FEX-B uplinks	N/A	Server	FI/FEX 20GE port-channel connectivity
E	FEX-A, port 1	C220-M3 VIC port 1	N/A	VNTag (internal)	Server to fabric A. VLANs are allowed on per vNIC basis
F	FEX-B, port 1	C220-M3 VIC port 2	N/A	VNTag (internal)	Server to fabric B. VLANs are allowed on per vNIC basis
(not marked)	Eth 2/1, 2/2 on FI-A and FI-B	Uplink switch	All	Uplink	Uplink to Infrastructure network

The cable connectivity diagram shows only one example C220M3 server, but all the rack servers (compute as well as storage nodes) connect in the similar manner.

Upstream connectivity is not shown in detail, but a pair of Nexus 5000 series switches is recommended. In that case, multiple UCS domains can connect to a pair of Nexus 5000 switches to provide highly available, scalable network. Virtual Port-Channel is recommended between Nexus 5000 series switches and FIs to reduce network instability during reboot of any of the switches or FIs.

Connect all the cables as outlined above, and you would be ready to configure UCS Manager.

Preparing UCS FIs and configure UCS Manager

Configure UCS FIs and UCS Manager can be subdivided in to following segments:

1. Initial Configuration of Cisco UCS FIs, page 27
2. Configuration for Server Discovery, page 29
3. Upstream/ Global Network Configuration, page 32
4. Configure Identifier Pools, page 35
5. Configure Server Pool and Qualifying Policy, page 41
6. Configure Service Profile Template, page 49
7. Instantiate Service Profiles from the Service Profile Template, page 63

Following subsections provided details on each of the steps mentioned above.

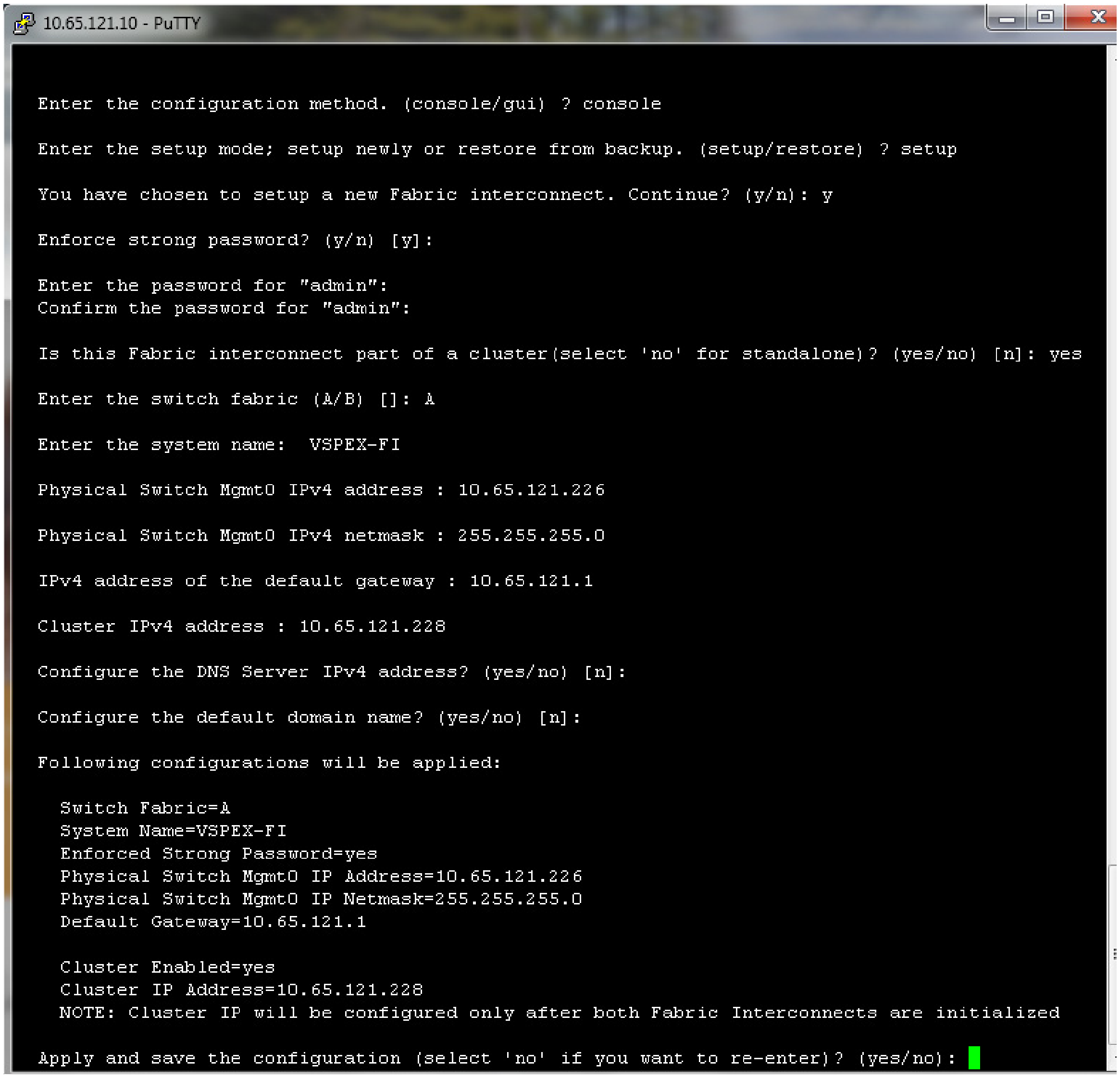
Initial Configuration of Cisco UCS FIs

At this point of time, the Cisco UCS FIs, FEX, and Blade Servers or Rack Servers must be mounted on the rack and appropriate cables must be connected. Two 100 Mbps Ethernet cables must be connected between two FIs for management pairing. Two redundant power supplies are provided per FI, it is highly recommended that both the power supplies are plugged in, ideally drawing power from two different power strips. Connect mgmt0 interfaces of each FI to the infrastructure network, and put the switch port connected to FI in access mode with access VLAN as management VLAN.

To perform initial FI configuration, follow these steps:

1. Attach RJ-45 serial console cable to the first FI, and connect the other end to the serial port of laptop. Configure password for the "admin" account, fabric ID "A", UCS system name, management IP address, subnet mask and default gateway and cluster IP address (or UCS Manager Virtual IP address), as the initial configuration script walks you through the configuration. Save the configuration, which will take you to UCS Manager CLI login prompt.

Figure 13 Initial Configurations of Cisco UCS Fabric Interconnect



2. Now disconnect the RJ-45 serial console from the FI that you just configured and attach it to the other FI. Other FI would detect that its peer has been configured, and will prompt to just join the cluster. Only information you need to provide is the FI specific management IP address, subnet mask and default gateway. Save the configuration.

Figure 14 Configuring Peer a Fabric Interconnect

```
  Enter the configuration method. (console/gui) ? console

  Installer has detected the presence of a peer Fabric interconnect. This Fabric interconnect wi
ll be added to the cluster. Continue (y/n) ? y

  Enter the admin password of the peer Fabric interconnect:
    Connecting to peer Fabric interconnect... done
    Retrieving config from peer Fabric interconnect... done
    Peer Fabric interconnect Mgmt0 IP Address: 10.65.121.226
    Peer Fabric interconnect Mgmt0 IP Netmask: 255.255.255.0
    Cluster IP address          : 10.65.121.228

  Physical Switch Mgmt0 IPv4 address : 10.65.121.227


  Apply and save the configuration (select 'no' if you want to re-enter)? (yes/no):
```

3. Once initial configurations on both FIs are completed, you can disconnect the serial console cable. Now, UCS Manager will be accessible through web interface (https://<ucsm-virtual-ip>/) or SSH. Connect to UCS Manager using SSH, and see HA status. As there is common device connected between two FIs (a rack server or blade server chassis), the status shows as "HA NOT READY", but you must see both FI A and FI B in "Up" state as shown Figure 15.

Figure 15 Cisco UCS Fabric Interconnect - Cluster State

```
VSPEX-FI-A# show cluster state
Cluster Id: 0xec91409a491011e2-0xb7a4547feeaa1564

A: UP, PRIMARY
B: UP, SUBORDINATE

HA NOT READY
No device connected to this Fabric Interconnect
VSPEX-FI-A#
```

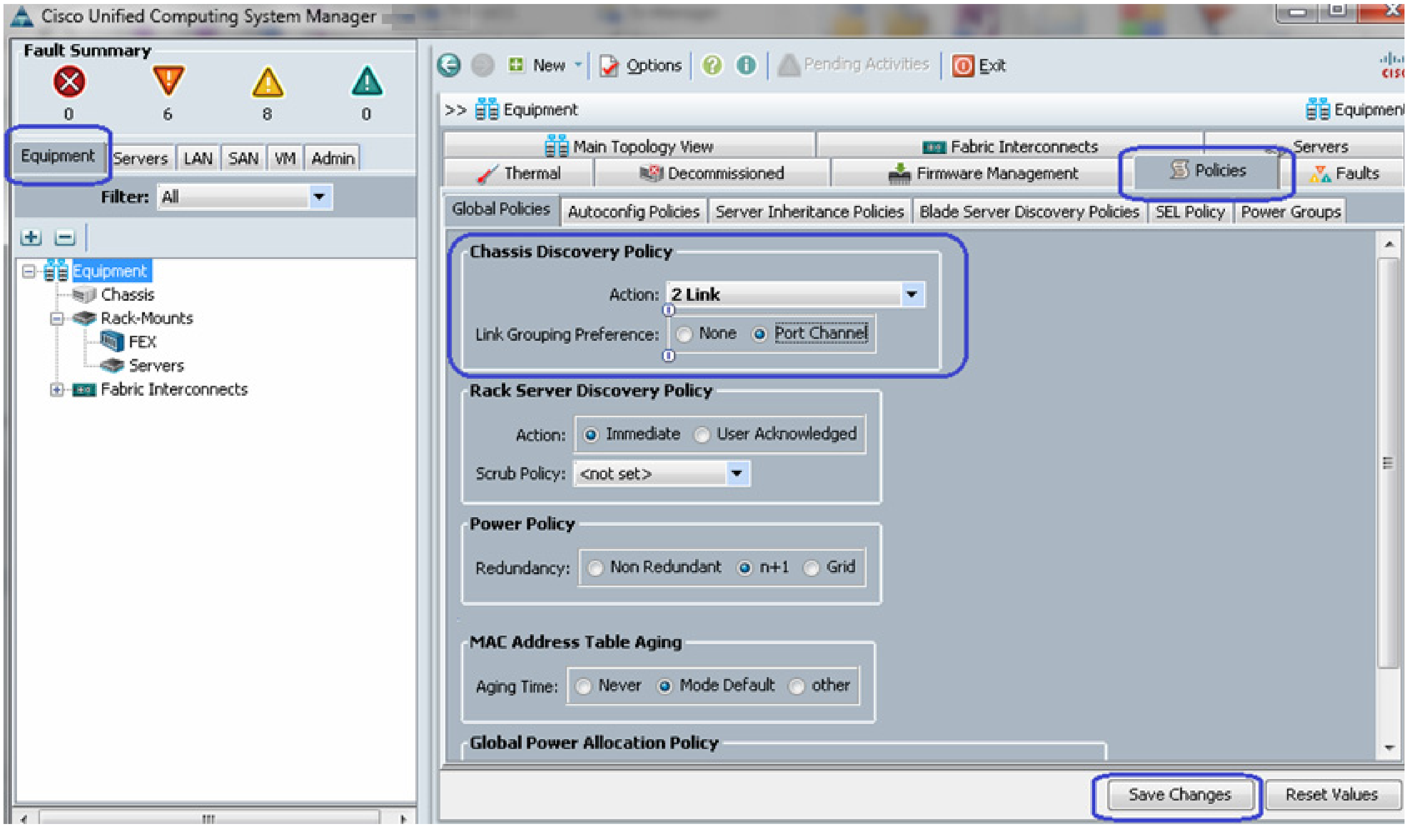
Configuration for Server Discovery

All the Ethernet ports of FIs are unconfigured and shutdown by default. You need to classify these ports as server facing ports, and uplink ports.

To configure the ports for proper server auto-discovery, follow these steps:

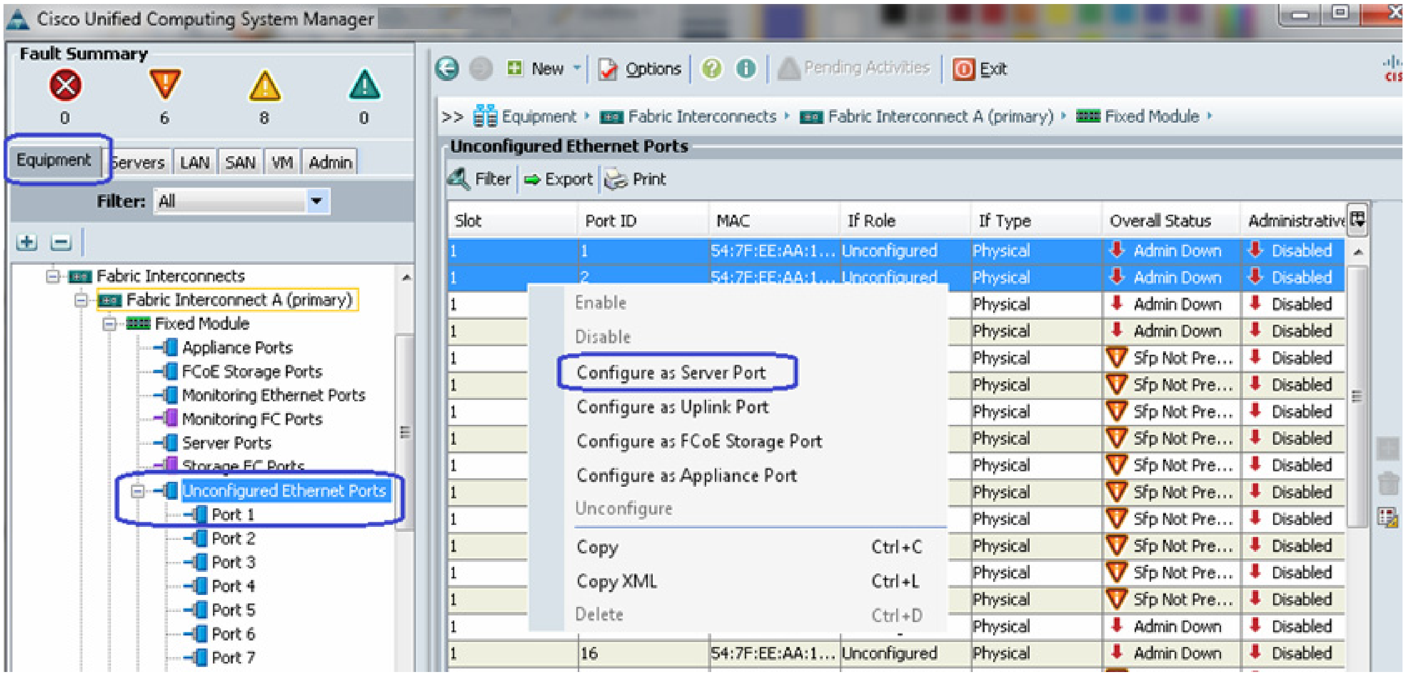
1. To configure chassis discovery policy that specifies server side connectivity, using a web browser, access the UCS Manager from the management virtual IP address and download the Java applet to launch UCS Manager GUI. Click **Equipment** tab in the left pane, and then **Policies** tab in the right pane. In Chassis Discovery Policy, For Actions field choose **2 Link.** Two links represent the two 10 GE links that are connected between FI and FEX per fabric. Also, change Link Grouping Preference to **Port Channel** for better bandwidth utilization and link level high-availability as shown in Figure 16. Save the changes.

Figure 16 Configuring Chassis Discovery Policy



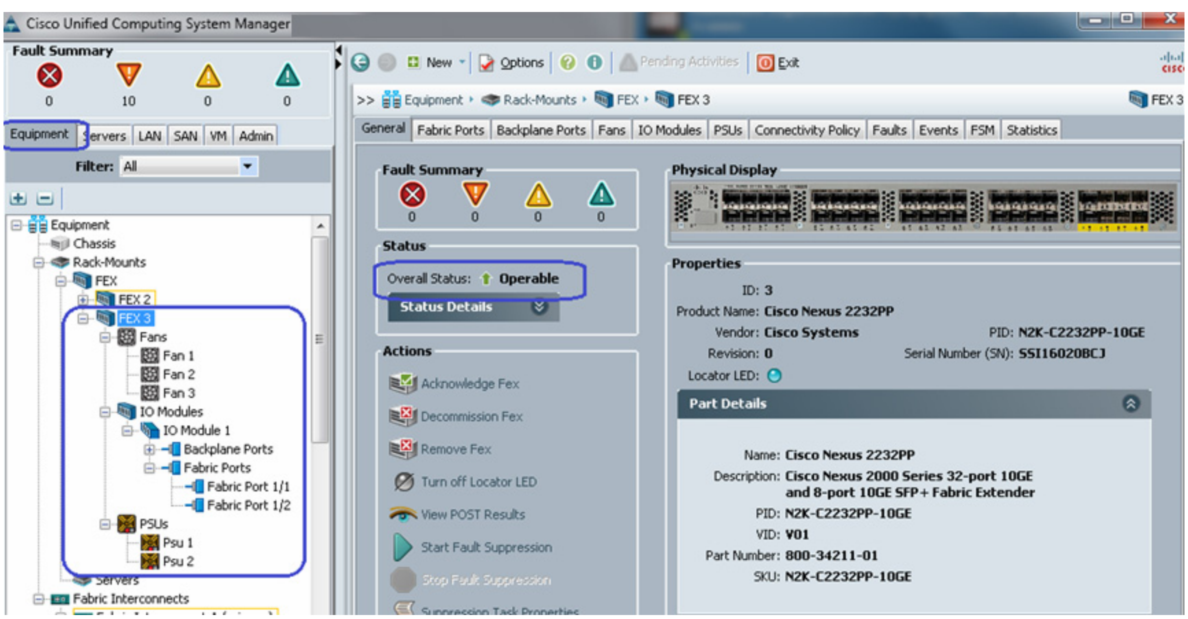
2. Next, identify ports connected to the Chassis or FEX per FI basis. Click the **Equipment** tab, expand **Fabric Interconnects**, choose an FI, for example, Fabric Interconnect A, click **Unconfigured Ethernet Ports**, and select the two ports connected to the FEX-A. Right-click, and choose **Configure as Server Port**. Click **Yes** on the confirmation pop-up window.

Figure 17 Configuring Ethernet Ports as Server Ports



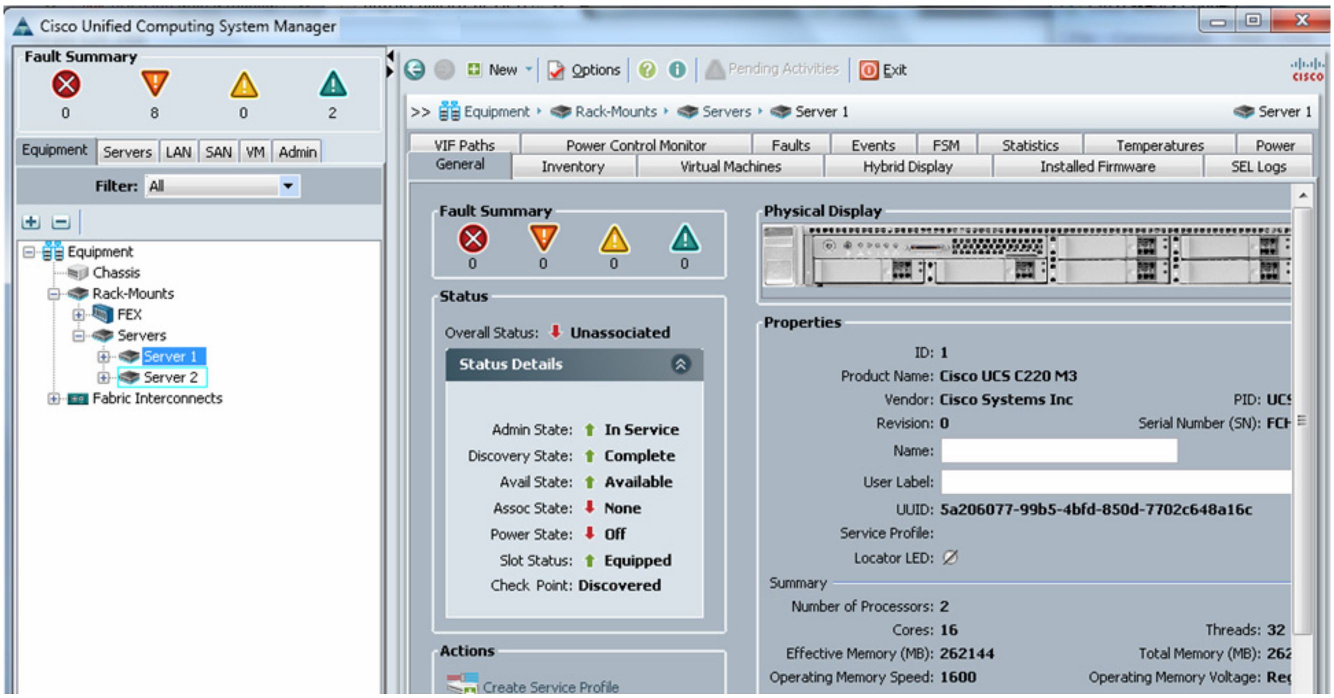
3. Repeat step 2 for the other FI as well.
4. Once server ports are configured on both FIs, the Chassis or FEX auto-discovery gets started. In case of FEX, after the deep discovery of FEX is complete, you will see two Fabric Extenders in the **Equipment** tab with overall status shown as Operable.

Figure 18 Overall Status of FEX After Auto-Discovery



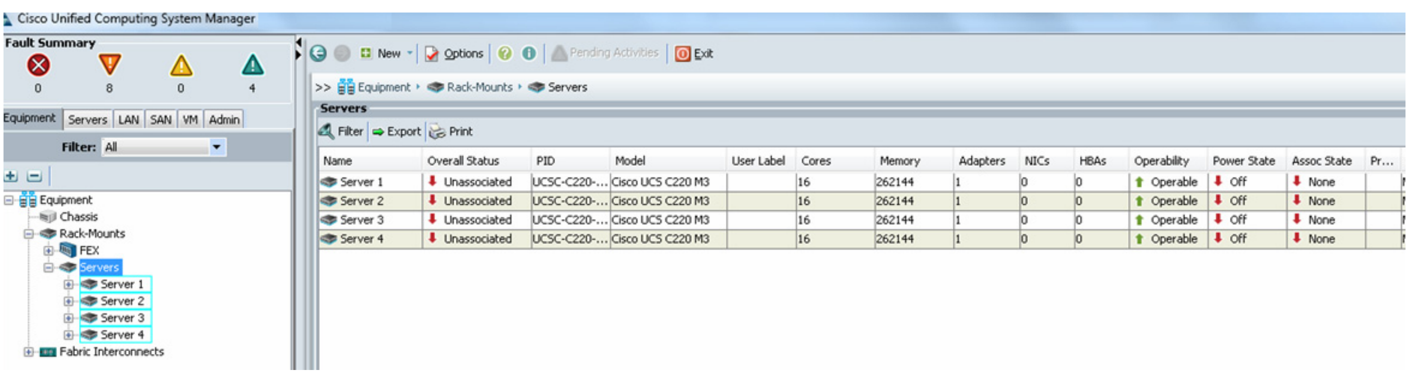
5. After the Chassis and FEX auto-discovery, the Blade Server and Rack Server auto-discovery will get started respectively. As and when the servers are discovered, you will see them getting added in the **Equipment** tab with overall status shown as Unassociated and availability state as Available, and discovery state as Complete.

Figure 19 Overall Status of Rack Servers After Discovery



6. Once all the servers are discovered, you can see the summary of all of them by choosing **Equipment** tab > **Rack-Mounts** > **Servers** as shown below.

Figure 20 Summary of Rack Servers After the Discovery



Upstream/ Global Network Configuration

This subsection lists a few upstream/ global network configuration:

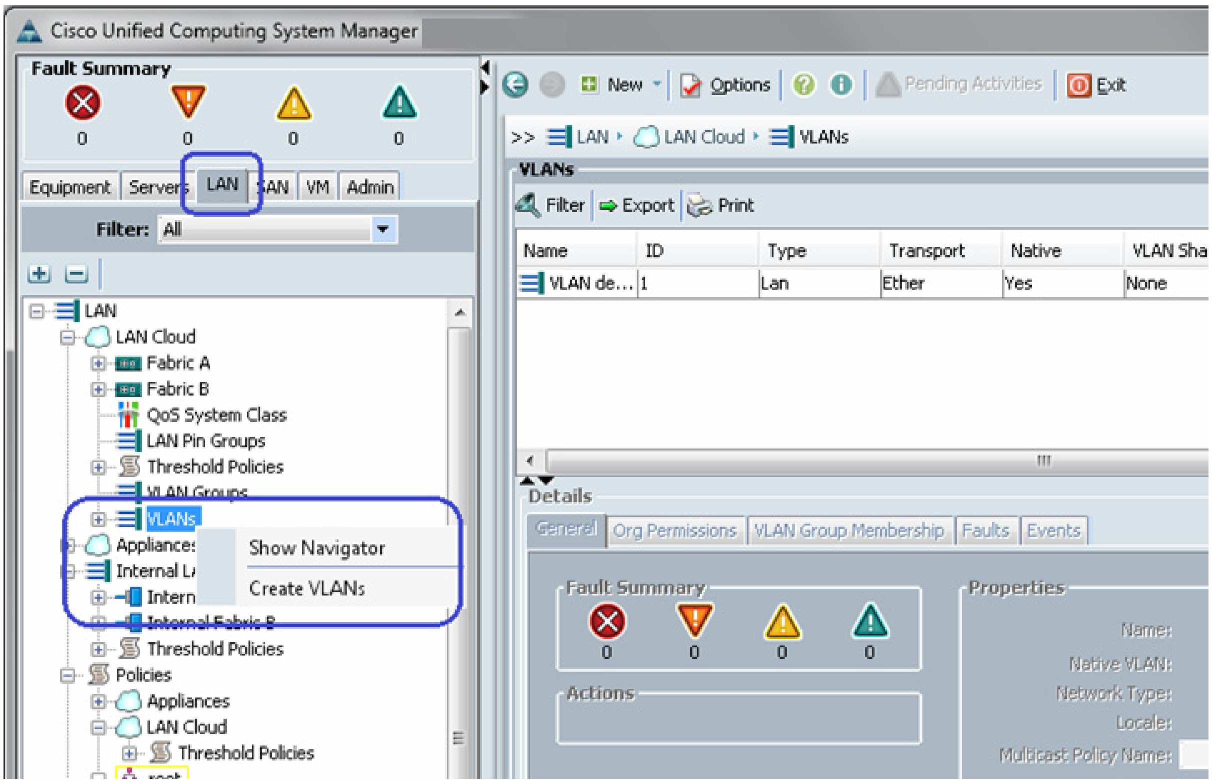
1. Uplink VLAN configuration

2. Configure Uplink ports

To configure upstream/ global network, follow these steps:

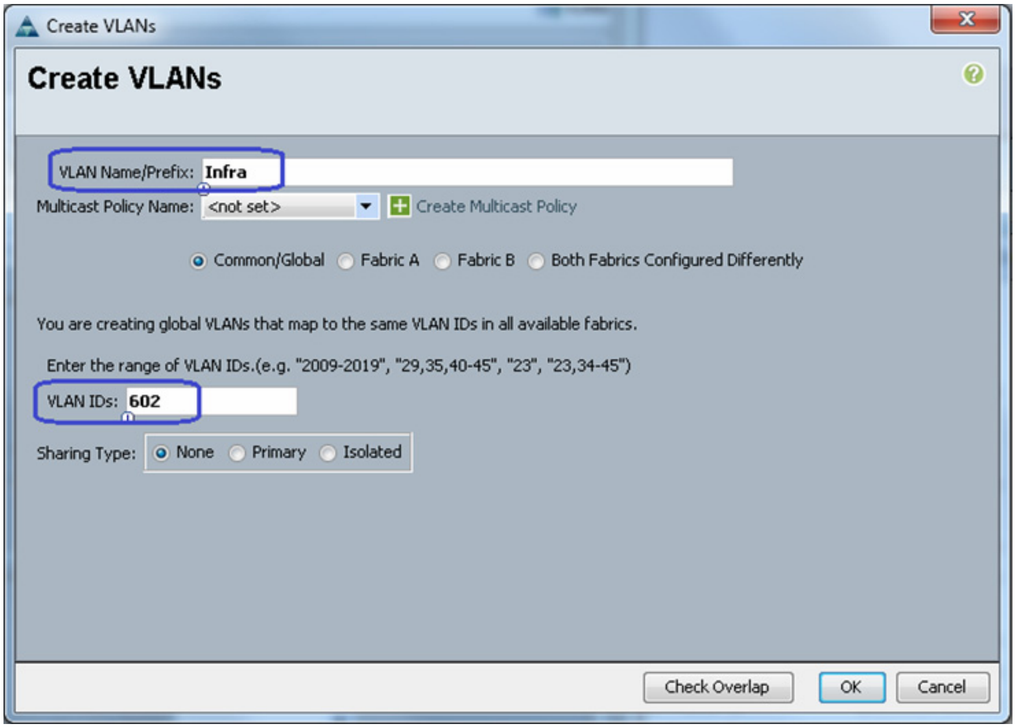
1. Click the **LAN** tab, expand **LAN Cloud** and right-click on VLANs and Click **Create VLANs.**

Figure 21 Creating VLANs



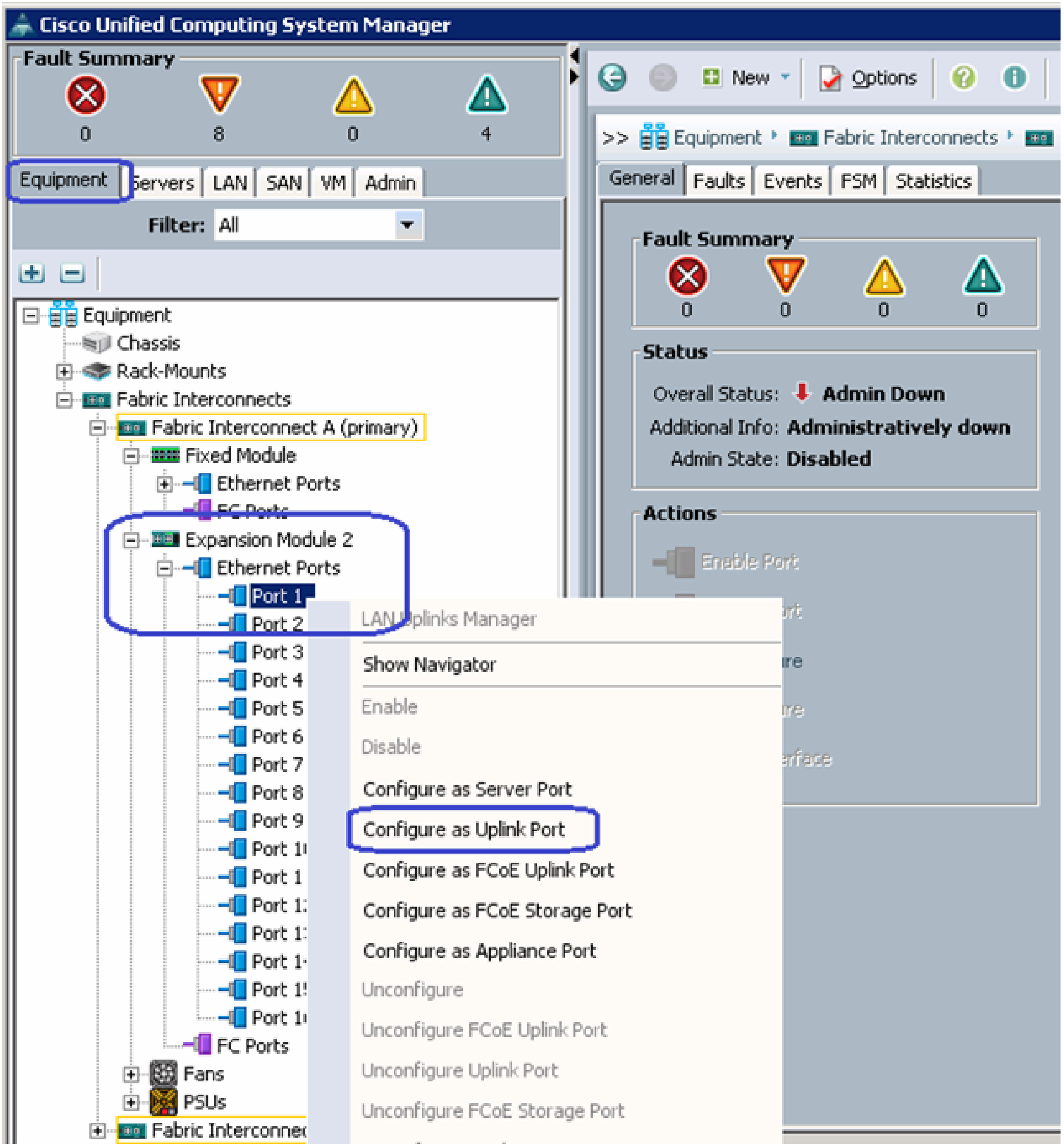
2. Enter the name of the VLAN and assign a VLAN ID. Make sure the default option **Common/Global** radio button is selected. Click **OK** to deploy the VLAN.

Figure 22 Entering Details of VLAN



3. Repeat the steps for “RHOS-Data” and various tenant VLANs.
4. To configure Uplink ports connected to the infrastructure network, click the **Equipment** tab, expand **Fabric Interconnects**, choose a particular FI, expand **Expansion Module 2** (this may vary depending on which port you have chosen as uplink port), right-click on the Ethernet port, and choose **Configure as Uplink Port**. Repeat this step for all the uplink ports on each FI.

Figure 23 Configuring Ethernet Ports as Uplink Ports



Configure Identifier Pools

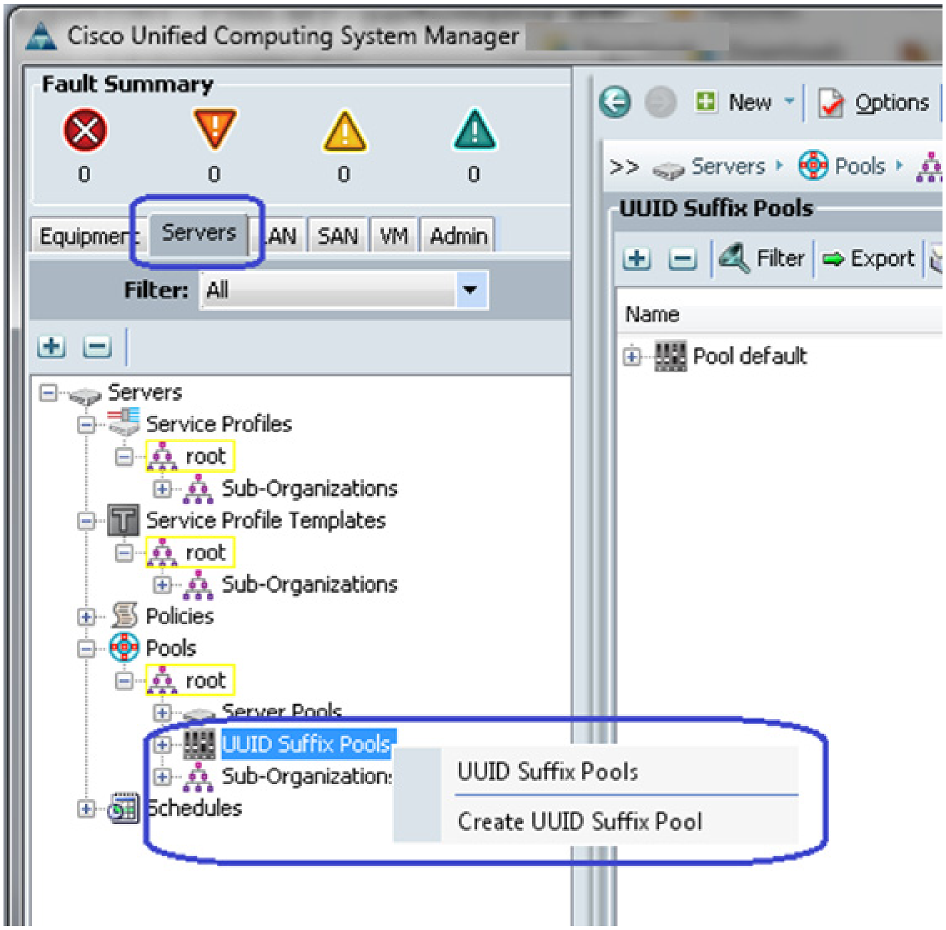
In this section, we would configure following identifier pools used by service profile:

1. Server UUID pool
2. MAC address pool
3. Management IP address pool

To configure pools mentioned above, follow these steps:

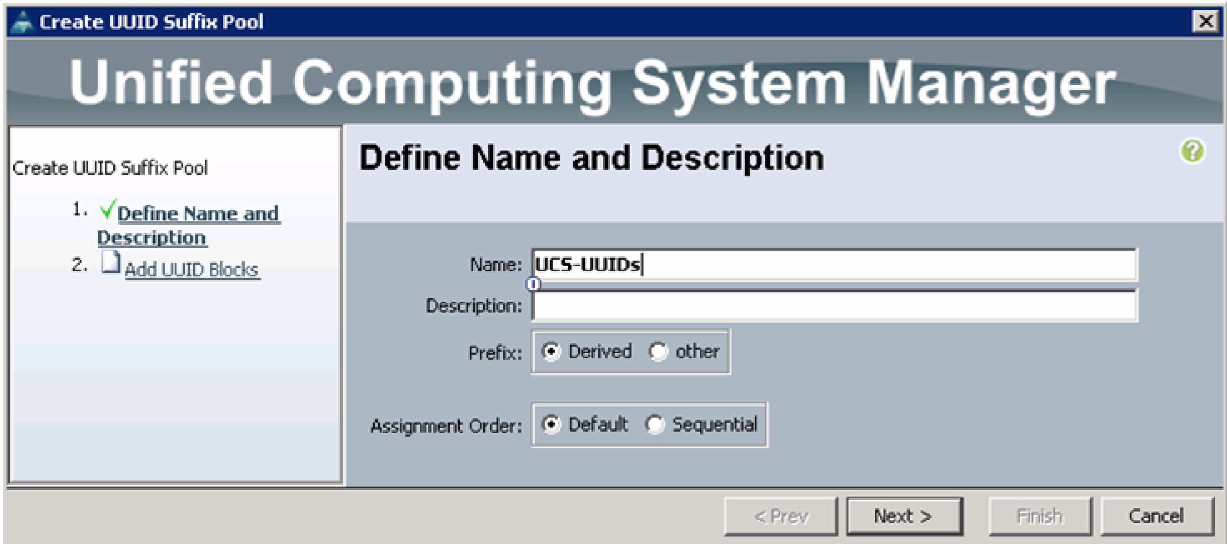
1. From the **Servers** tab, expand **Servers** > **Pools** > **root**, and right-click on UUID Suffix pools and click **Create UUID Suffix Pool**.

Figure 24 Creating UUID Suffix Pool



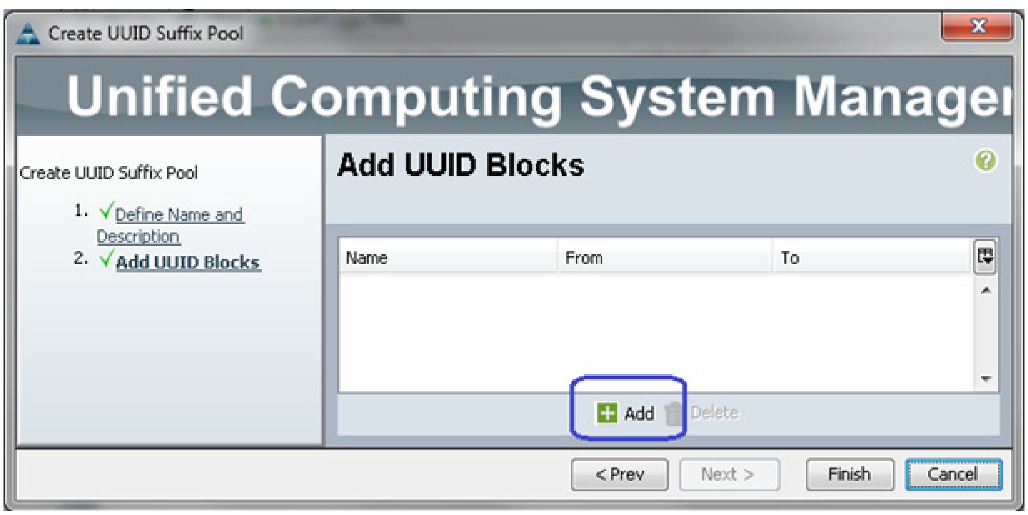
2. Enter the name and description to the UUID suffix pool. Keep other configuration as default.

Figure 25 Details for Creating UUID Suffix Pool



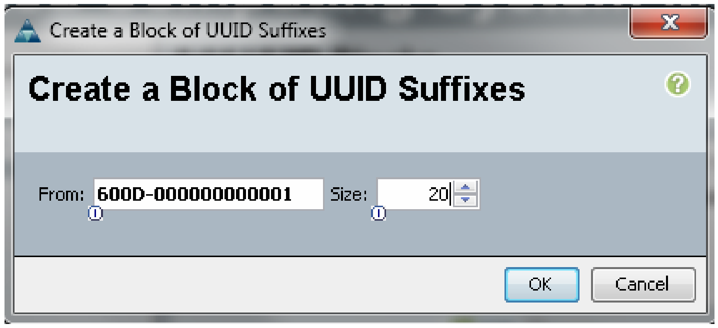
3. Click Add to add UUID block.

Figure 26 Adding UUID Block



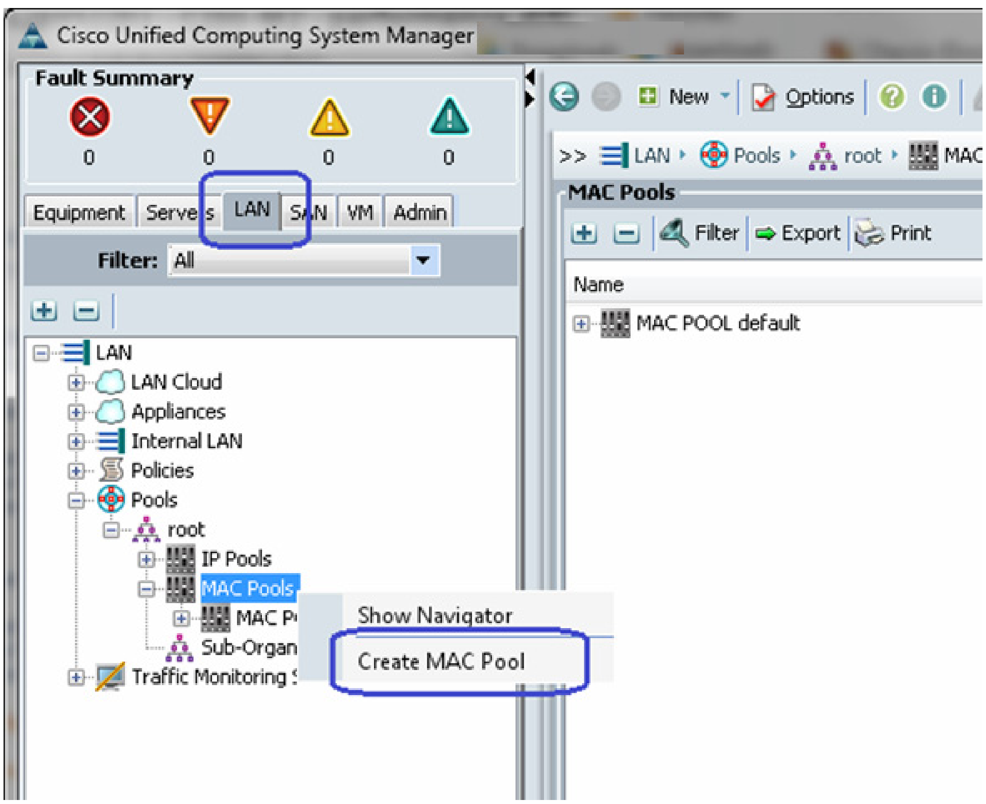
4. Specify the beginning of the UUIDs, and have a large size of UUID block to accommodate future expansion.

Figure 27 Specifying Block Size



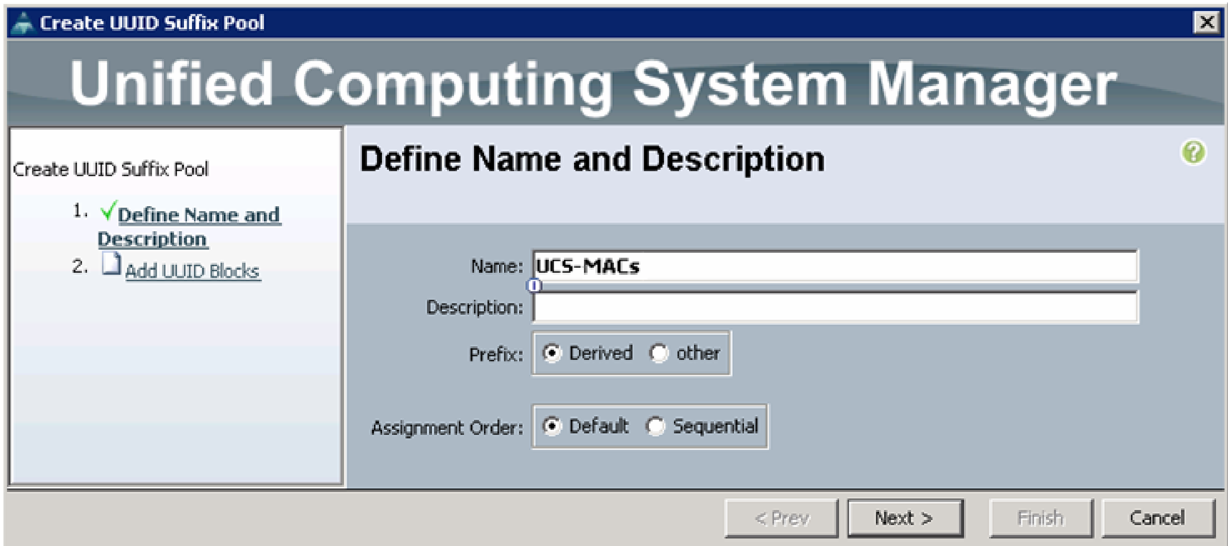
5. Click **OK** and then **Finish** to deploy UUID pool.
6. Click the **LAN** tab, expand **LAN** > **Pools** > **root**, right-click on MAC Pools and select **Create MAC Pool**.

Figure 28 Creating MAC Pool



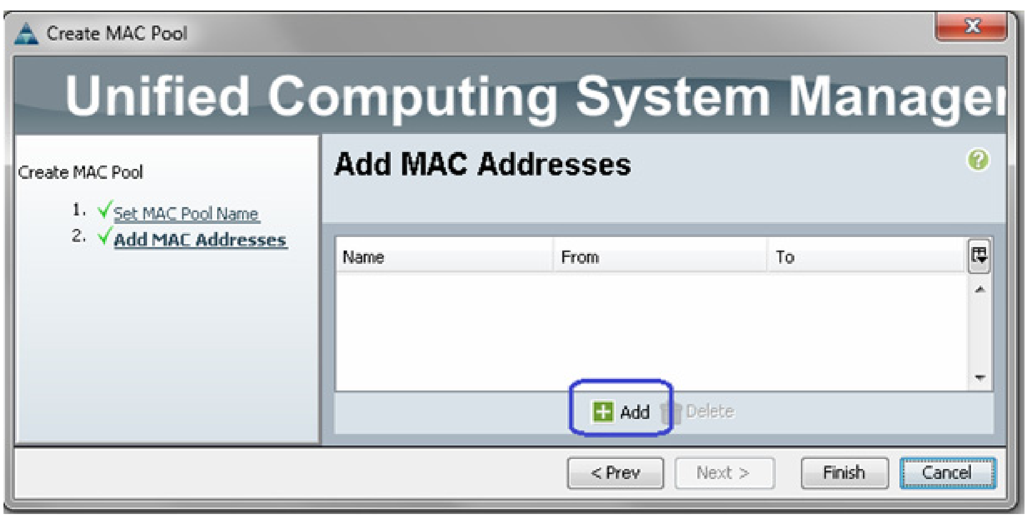
7. Enter the name and description for MAC pool and click **Next**.

Figure 29 Details for Creating MAC Pool



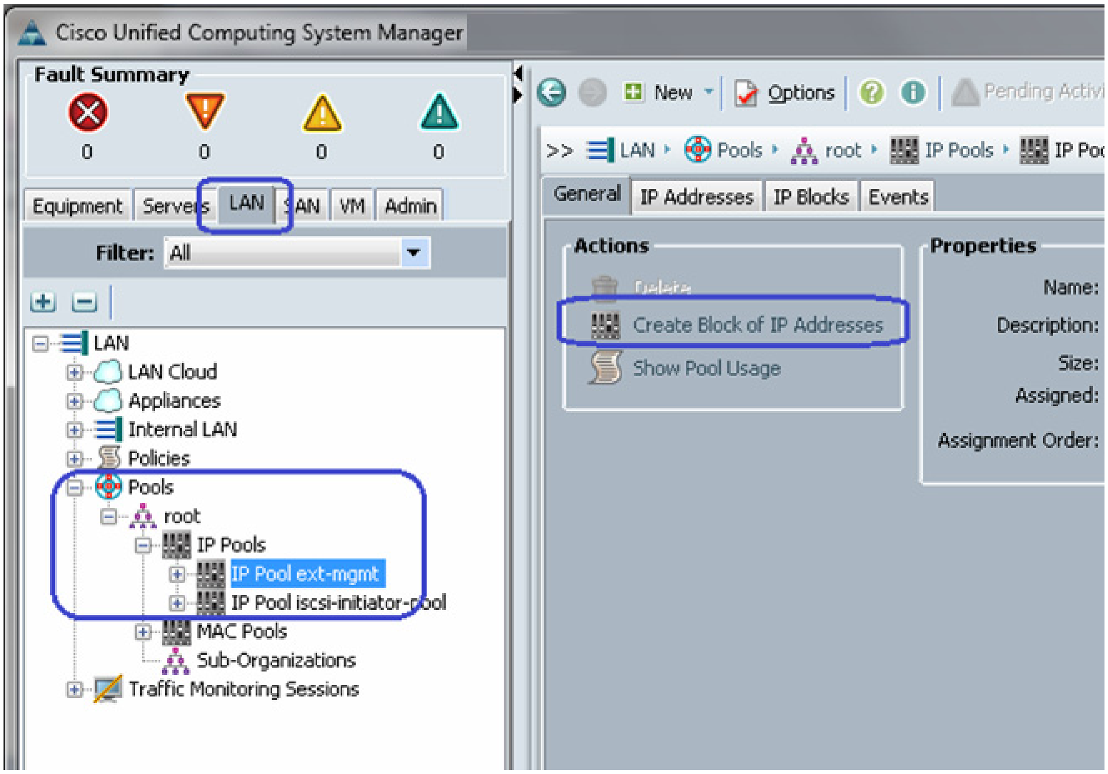
8. Click  to add MAC pool block.

Figure 30 Adding MAC Address



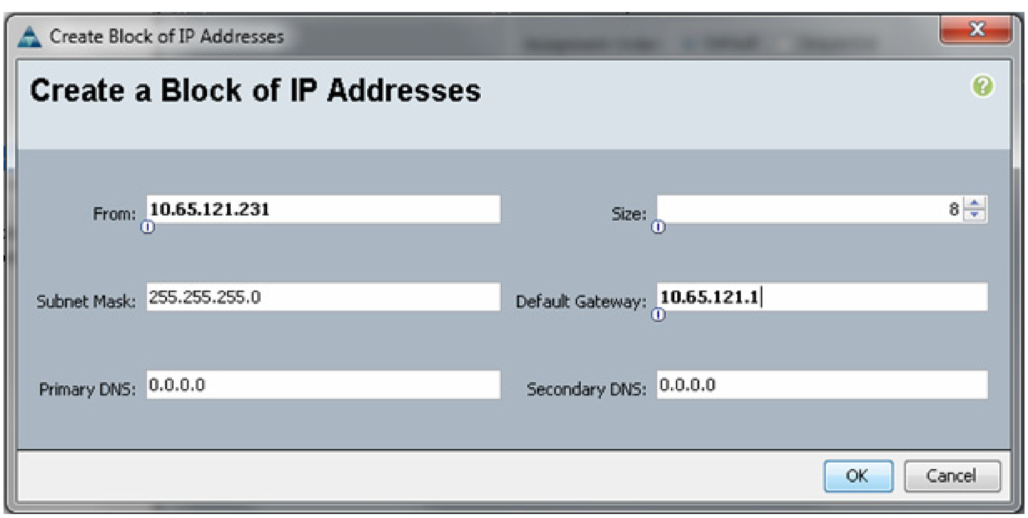
9. Enter the initial MAC address and size of the block. As always, provide large number of MAC addresses to accommodate future expansion. We will require 6 MAC addresses per server.
10. Next is creation of the management IP address block for KVM access of the servers. The default pool for server CIMC management IP addresses are created with the name **ext-mgmt**. From the **LAN** tab, expand **LAN** > **Pools** > **root** > **IP Pools** > **IP Pool ext-mgmt**, and click the **Create Block of IP addresses** link in the right pane.

Figure 31 Creating IP Address Block



11. Enter the initial IP address, size of the pool, default gateway and subnet mask. Click **OK** to deploy the configuration. IP addresses will be assigned to various Rack-Mount server CIMC management access from this block.

Figure 32 Specifying the IP address Block Size



Configure Server Pool and Qualifying Policy

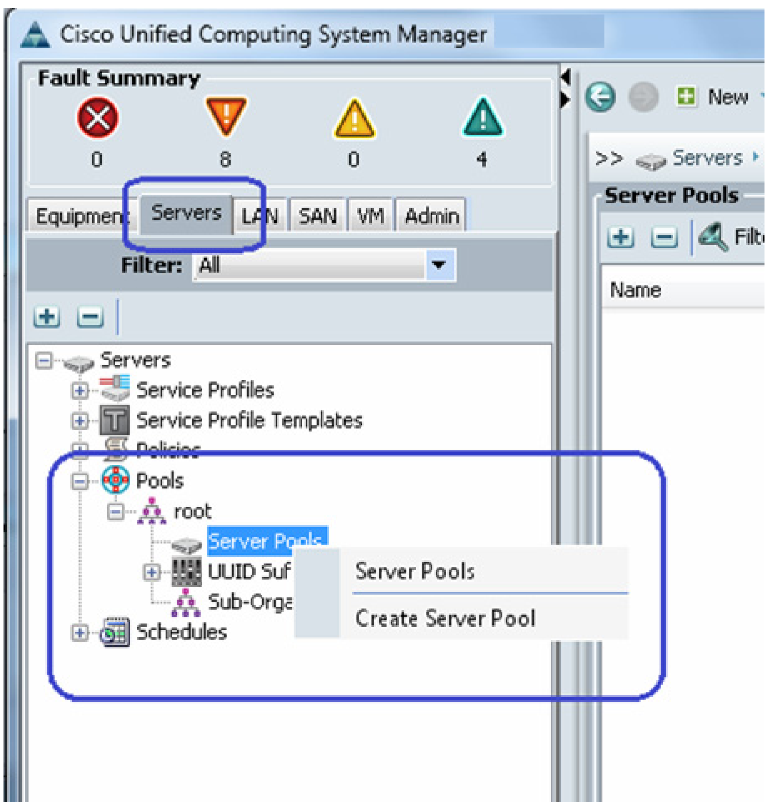
Creation and policy based auto-population of server pool can be sub-divided into the following tasks:

1. Creation of server pool
2. Creation of server pool policy qualification
3. Creation of server pool policy

Follow these steps to complete the three tasks mentioned above:

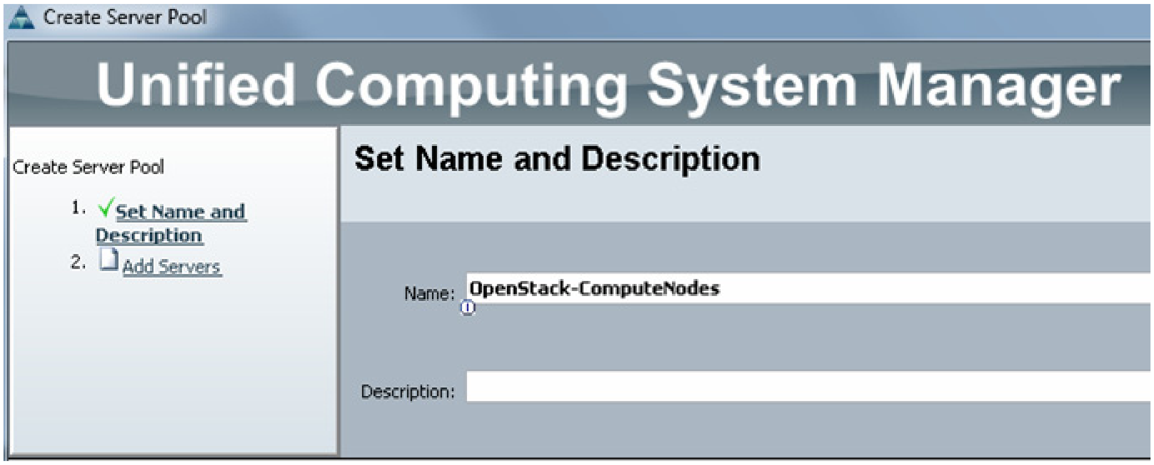
1. From the **Servers** tab, expand **Servers** > **Pools** > **root**, right-click on Server Pools and choose **Create Server Pool**.

Figure 33 Creating Server Pools



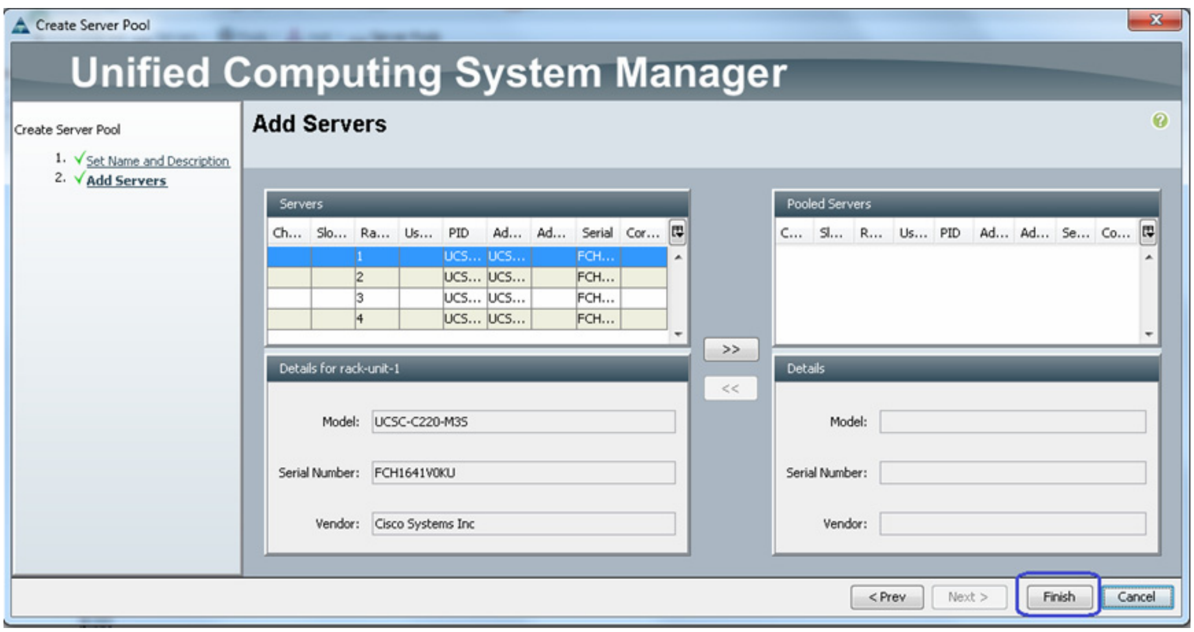
2. Enter the name of the server pool in the Name field, and click **Next**.

Figure 34 Entering Details in the Create Server Pool Wizard



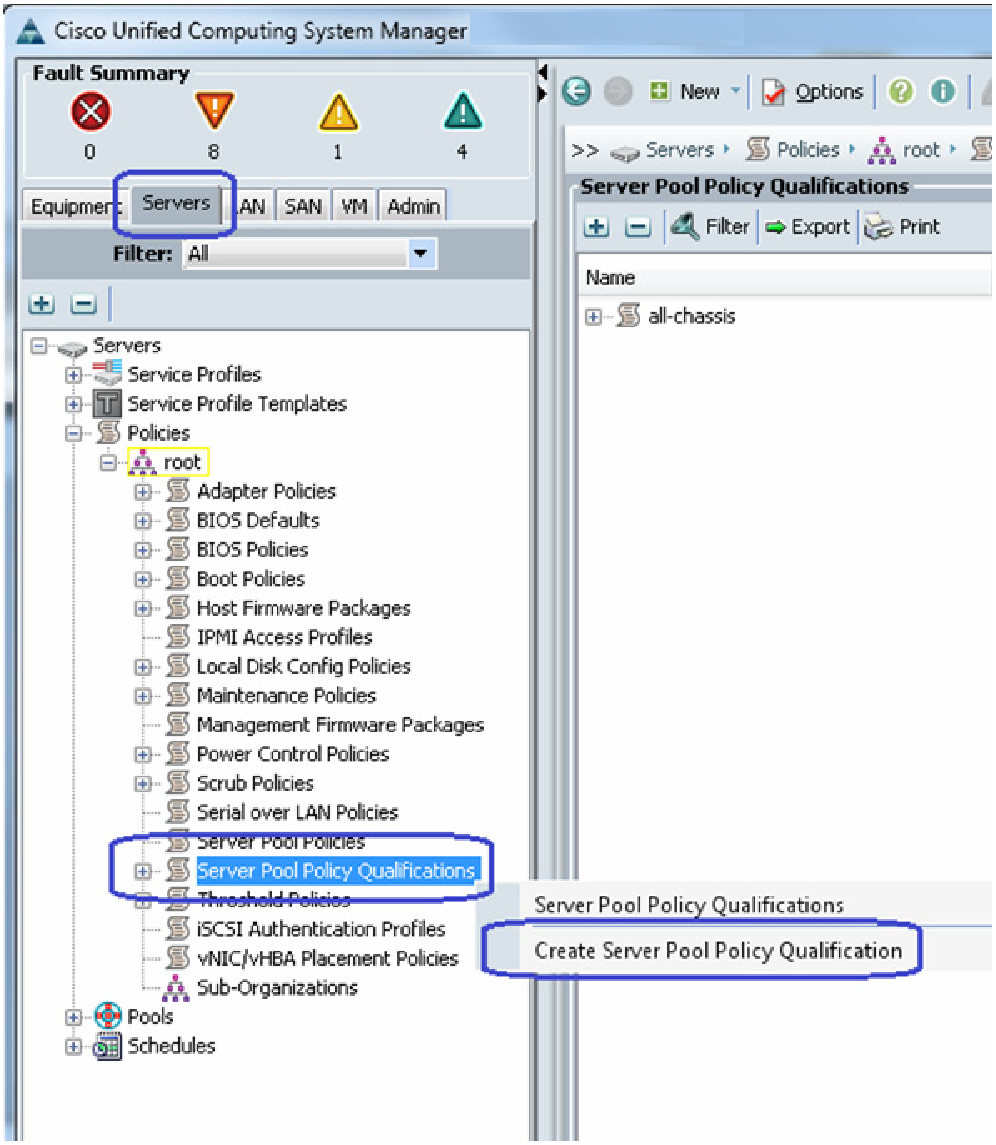
3. Click **Finish** to create the empty server pool. We would add the compute resources to this pool dynamically, based on policy.

Figure 35 Adding Servers in the Create Server Pool Wizard



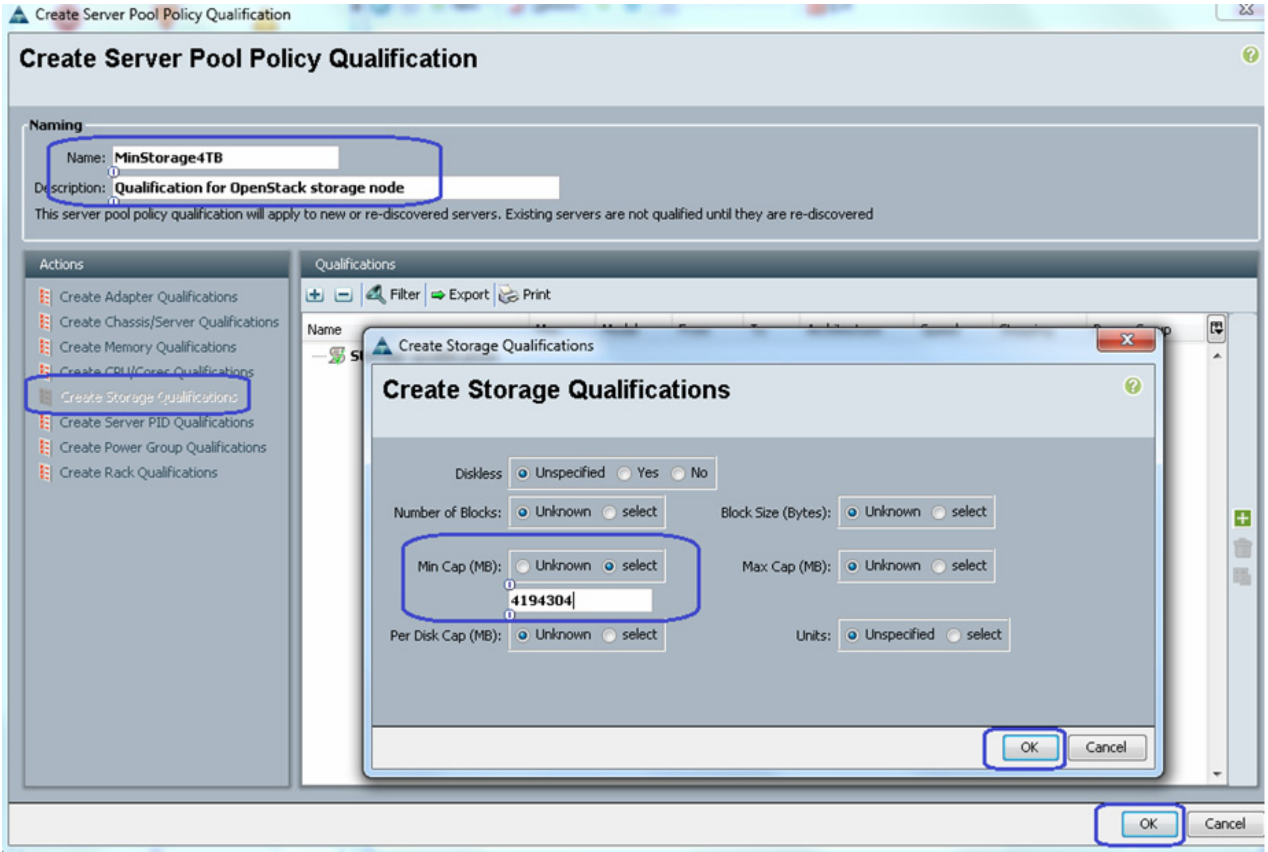
4. From the **Servers** tab, expand **Servers** > **Policies** > **root**, right-click on **Server Pool Policy Qualifications** and choose **Create Server Pool Policy Qualification**.

Figure 36 Creating Server Pool Policy Qualification



5. Enter the name for the server policy qualification criterion as MinStorage4TB in the Name field. In the left pane under Actions choose **Create Memory Qualifications** to server policy qualification criterion. Choose storage qualification criterion and provide minimum storage capacity as 4194304 MB (for 4 TB storage) as shown in Figure 37. Click **OK** twice to save the storage qualification.

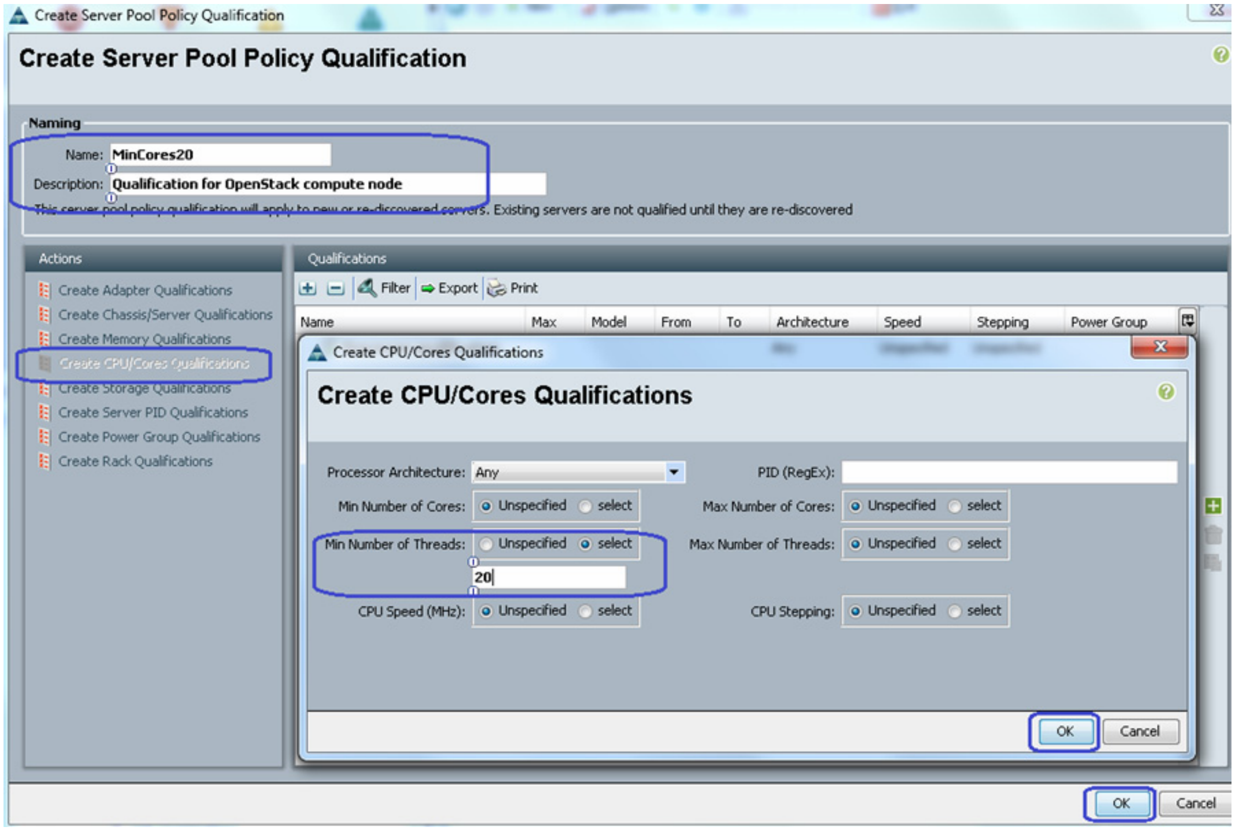
Figure 37 Creating Memory Qualification for Storage Nodes



6. Similarly, to create qualification for compute nodes, enter the name as MinCore20 in the server policy qualification criterion. Choose CPU/Cores qualification criterion and provide minimum cores as 20 as shown in Figure 38. Click **OK** twice to save the compute node qualification.

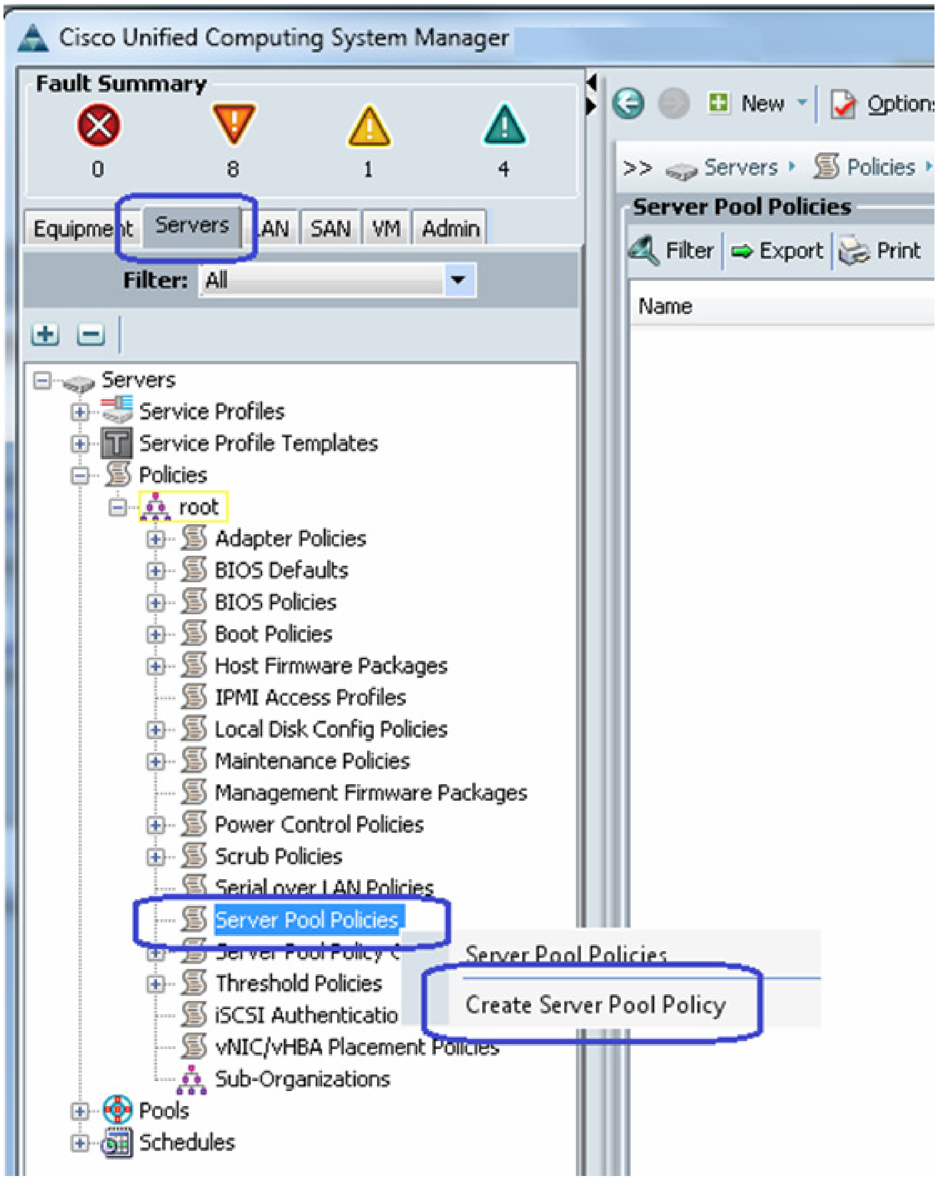
Note This is just an example criterion, you can choose a criterion that suites your requirement.

Figure 38 Creating Memory Qualification for Compute Nodes



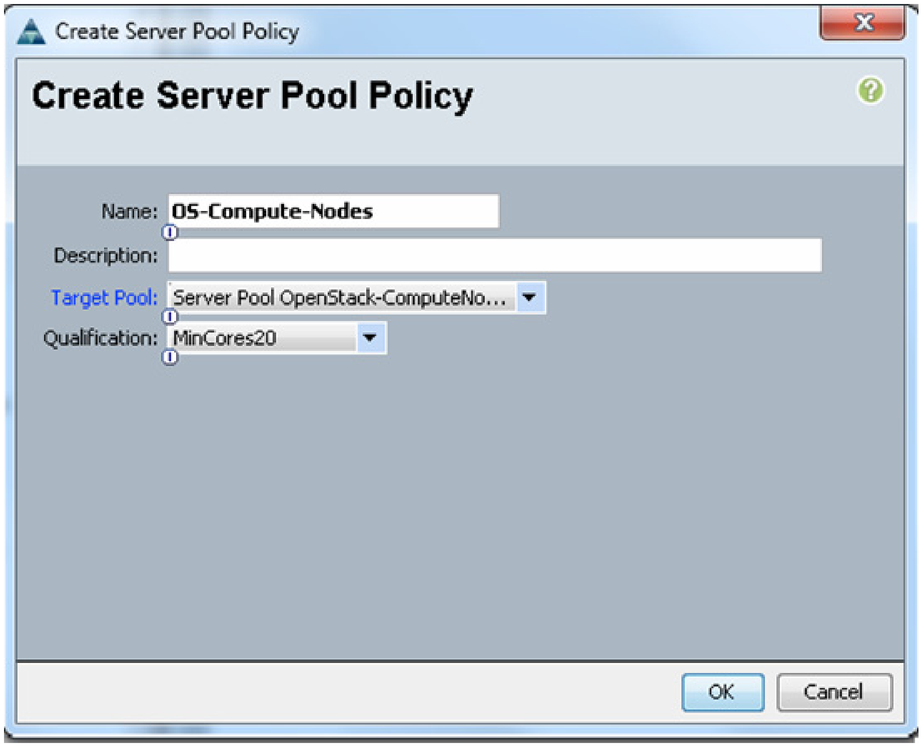
7. From the **Servers** tab, expand **Servers** > **Policies** > **root**, right-click on Server Pool Policies and choose **Create Server Pool Policy**.

Figure 39 Creating Server Pool Policy



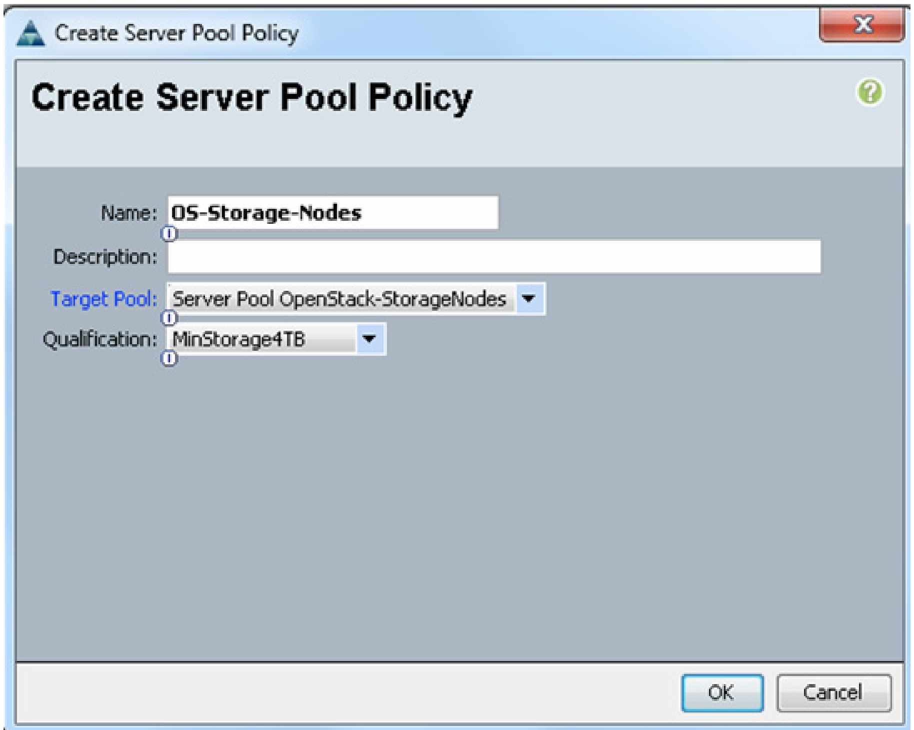
8. Enter the name as OS-Compute-Nodes in the server pool policy. Choose recently created Target Pool and Qualification for compute nodes. Click **OK** to deploy the configuration.

Figure 40 Details for Creating Server Pool Policy - Compute



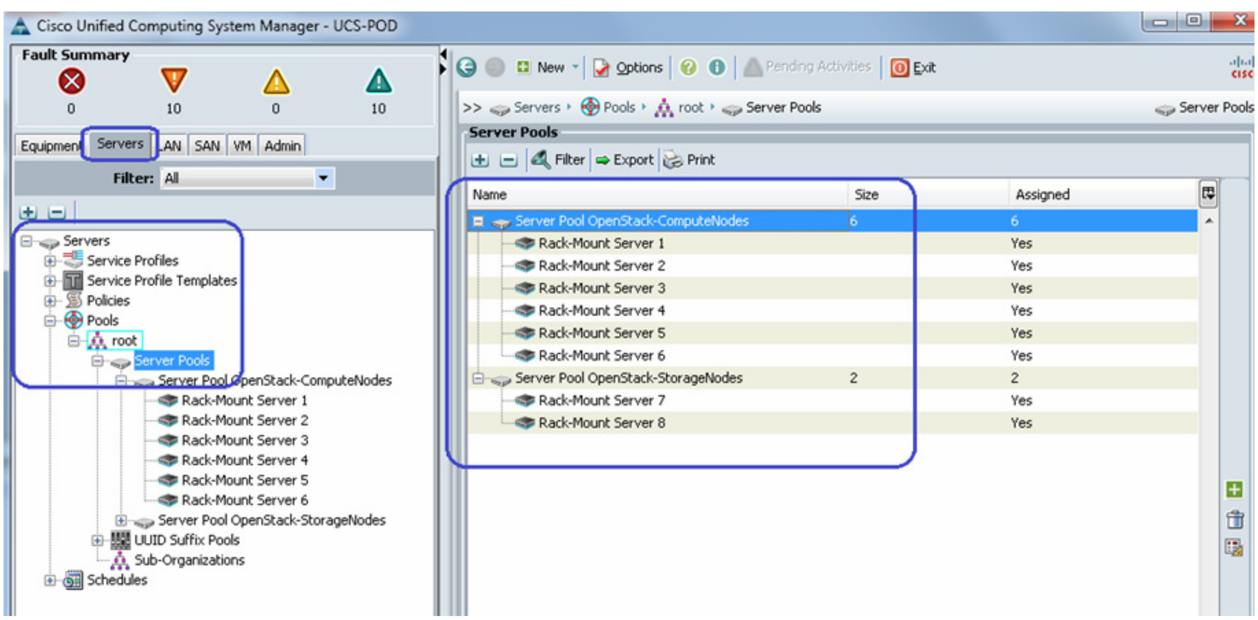
9. Similarly, create an other Server Pool Policy for storage nodes. Enter the name as OS-Storage-Nodes. Choose recently created Target Pool and Qualification for storage nodes. Click **OK** to deploy the configuration.

Figure 41 Details for Creating Server Pool Policy - Storage



10. If you go back to the server pool created in step 1 above and click the **Servers** tab on right pane, you will see that all the compute resources that meet the qualification criteria are dynamically added to the server pool. Figure 42 shows all the dynamically added resources in the server pool.

Figure 42 Qualified Compute Resources Automatically Added to the Server Pool



Configure Service Profile Template

At this point, we are ready to create service profile template, from which we can instantiate individual service profiles later.

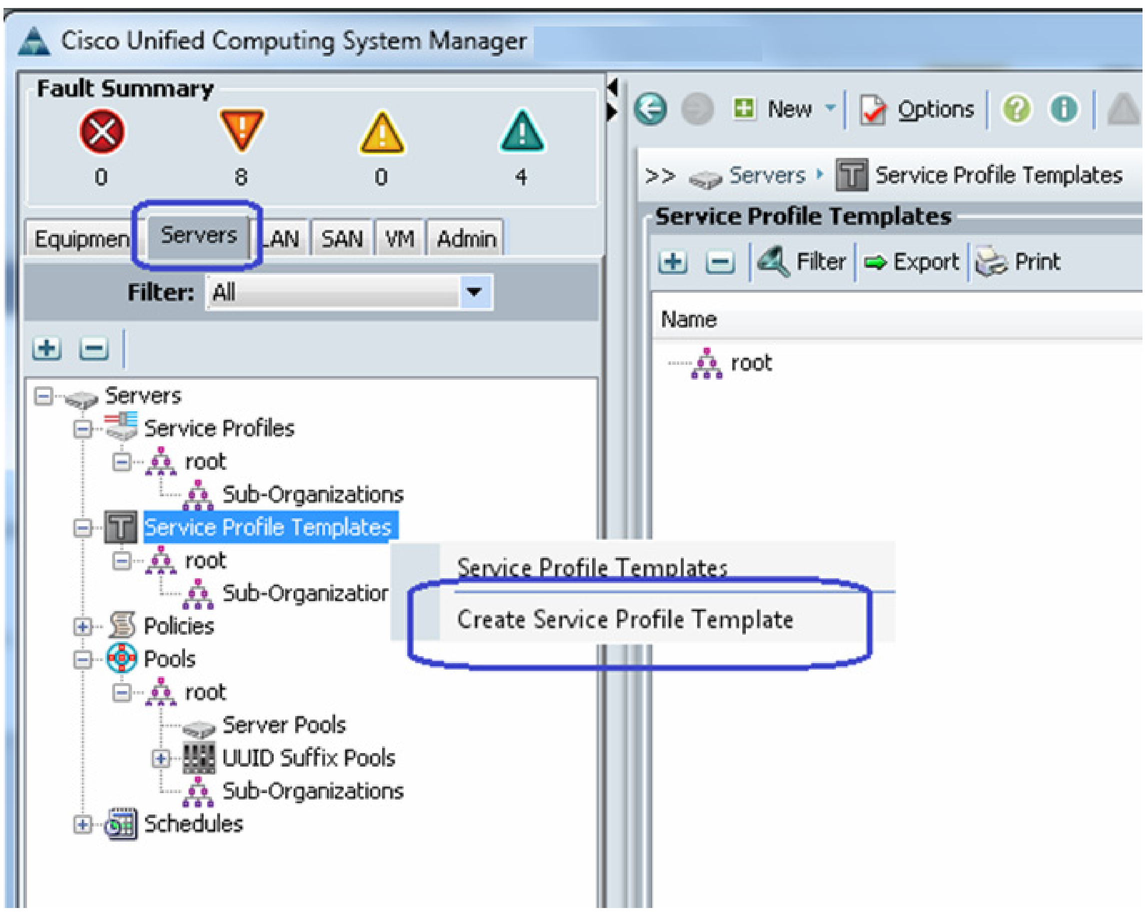
We need to create three service profile templates:

1. RHOS-A: For compute nodes with system VNICs on fabric A
2. RHOS-B: For compute nodes with system VNICs on fabric B
3. RHOS-Storage: For storage nodes

To create service profile template, follow these steps:

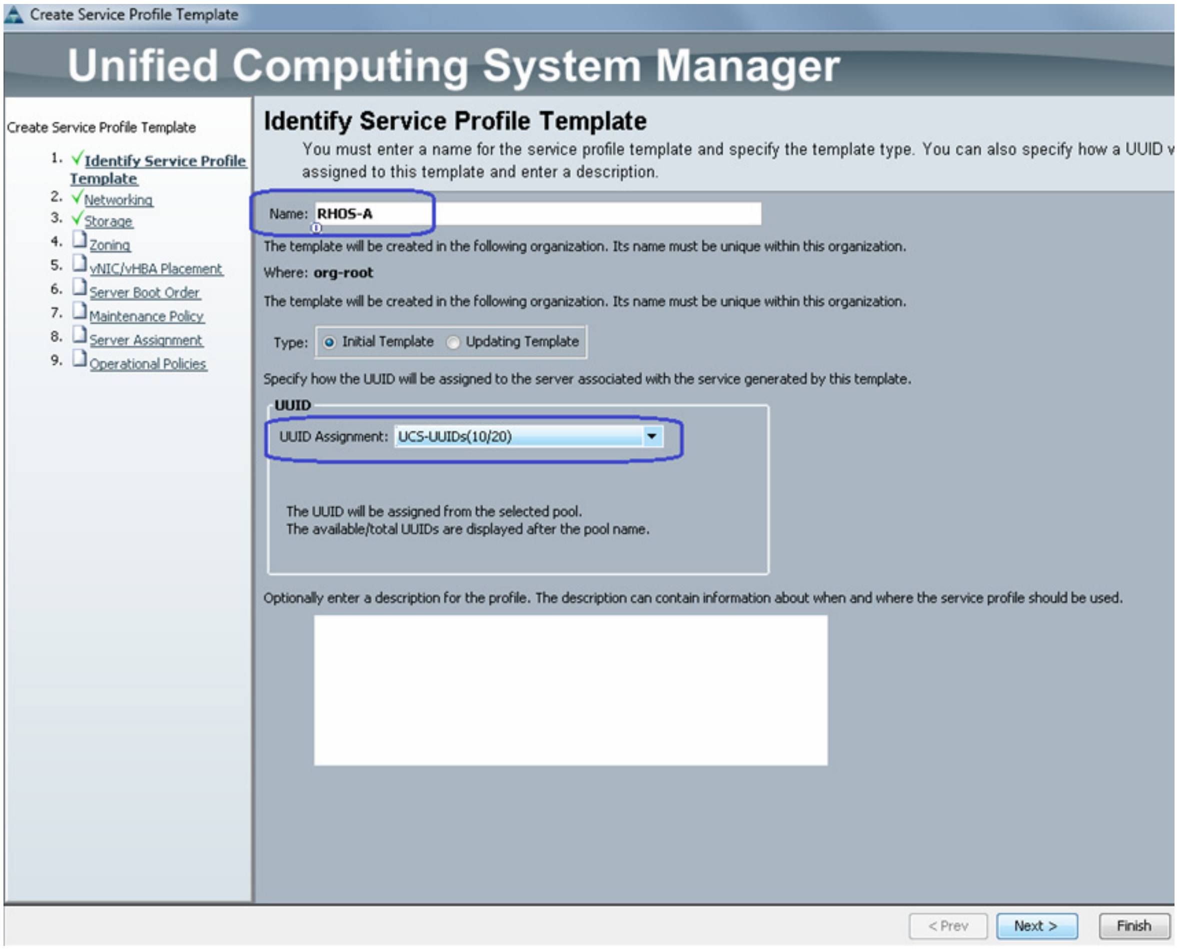
1. From the **Servers** tab. Expand **Servers** > **Service Profile Templates**, right-click on service profile templates and choose **Create Service Profile Template**.

Figure 43 Creating Service Profile Template



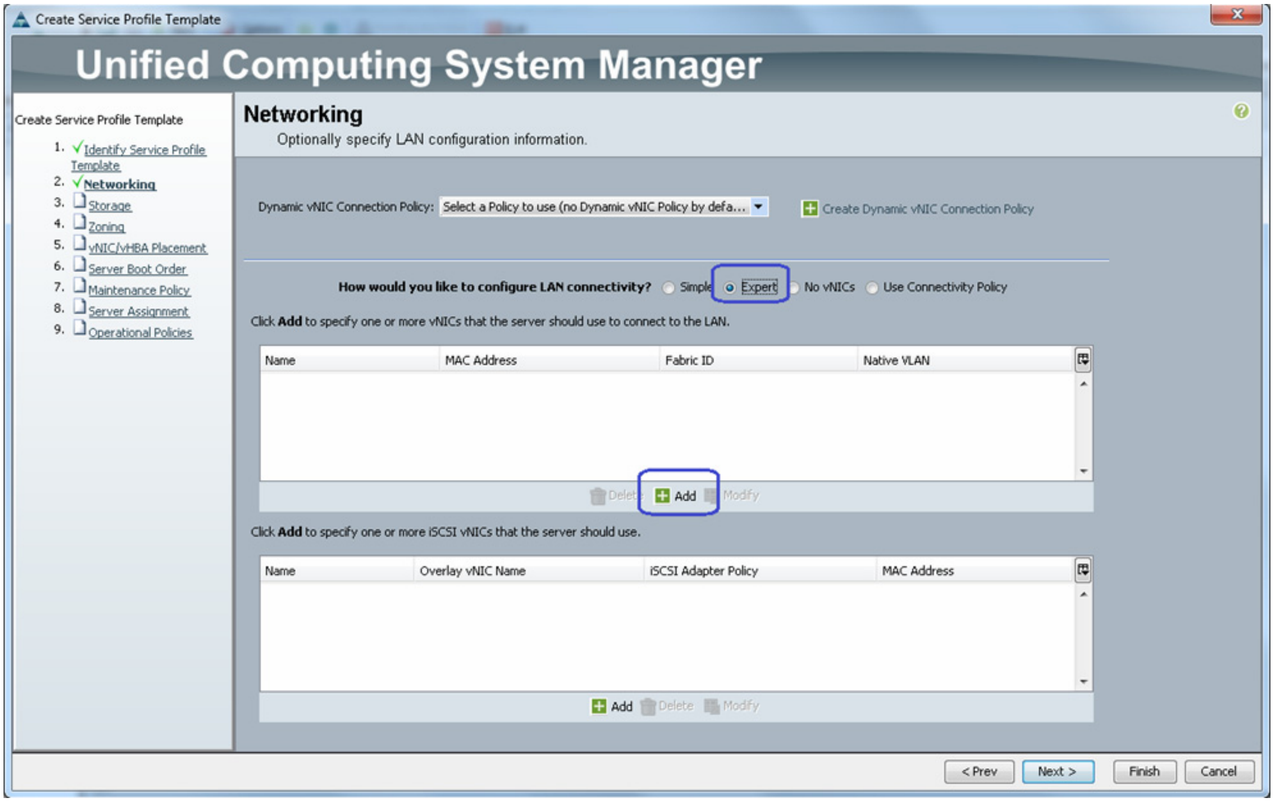
2. Enter the service profile template name in the name field, keep the type as **Initial Template**, and choose **UUID pool** for UUID assignment.

Figure 44 Creating Service Profile Template - Entering Details



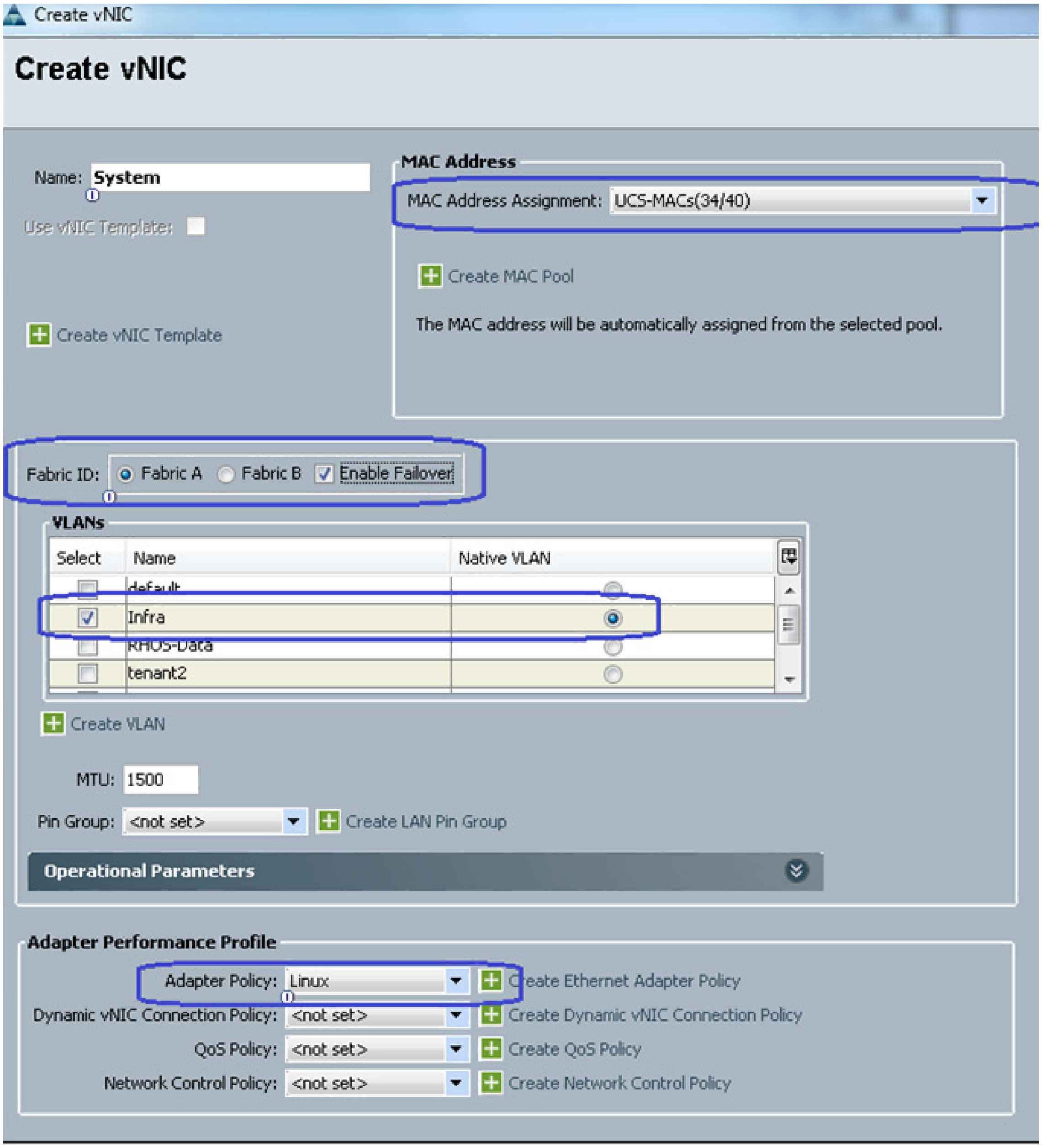
3. Click the **Expert** radio button for configure LAN connectivity. Click Add to create a vNIC.

Figure 45 Creating Service Profile Template - LAN Configuration Details



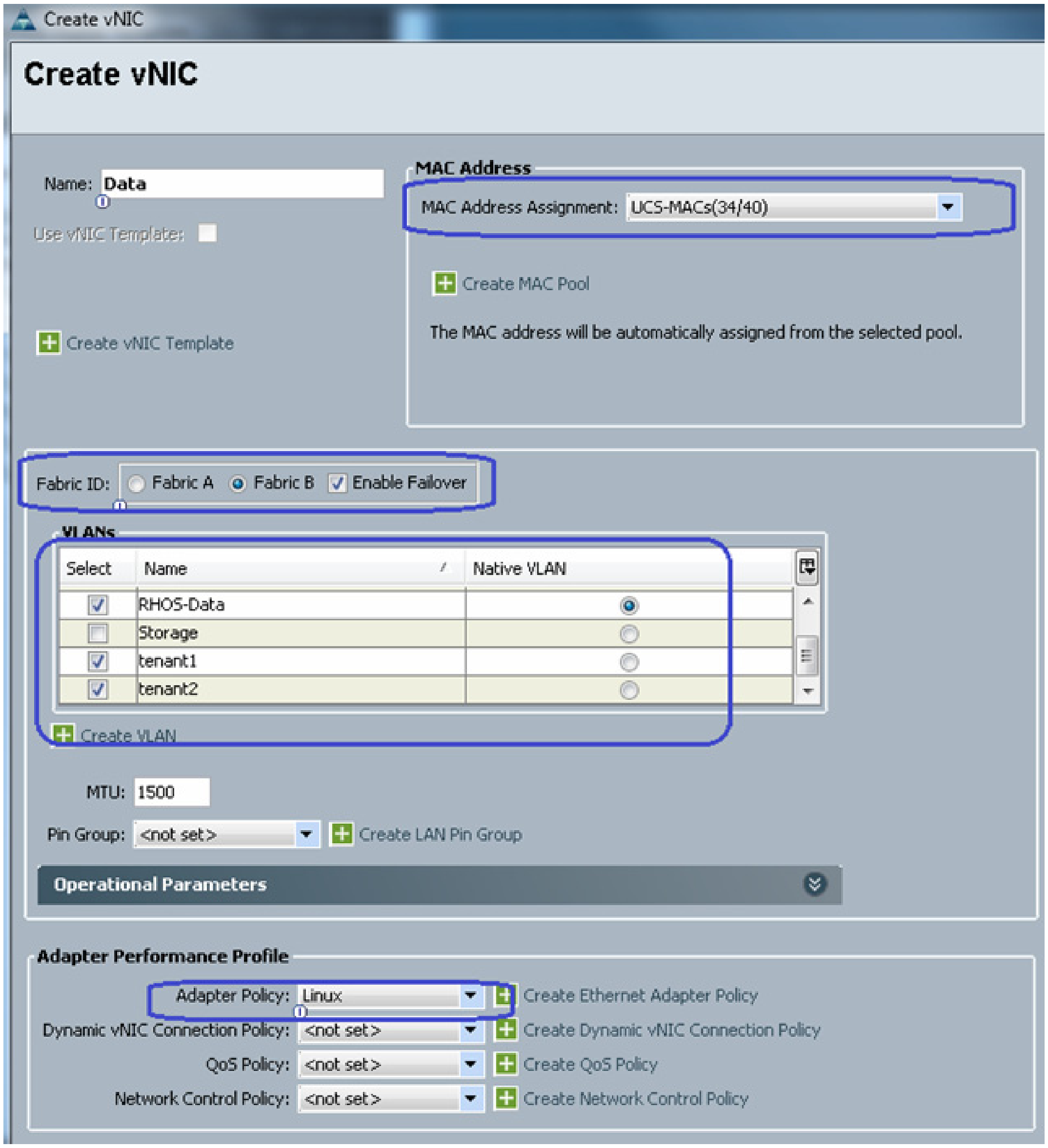
4. Create a system vNIC for fabric A. Enter System as the vNIC name, choose the MAC pool created in section D, click the radio button **fabric A** for fabric ID, check the check box **Infra** for VLANs and click the native VLAN radio button. For Adapter Policy field, choose **Linux.**

Figure 46 Creating a System vNIC



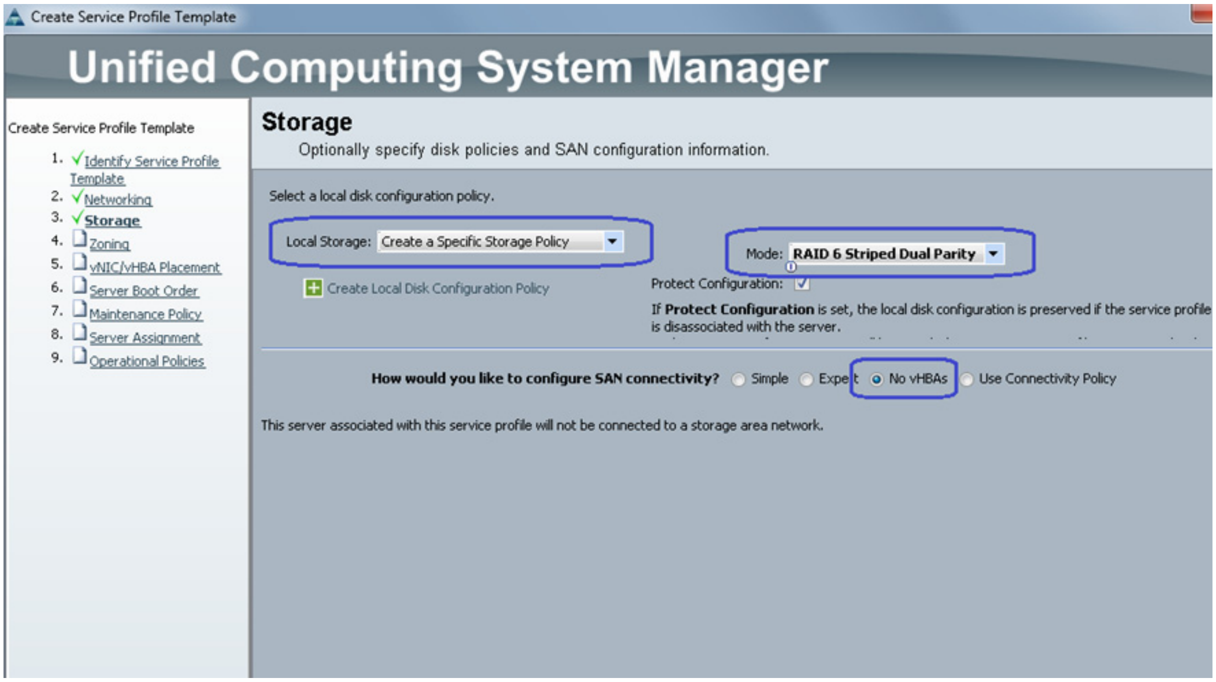
5. Similarly, create an other vNIC for VM data traffic. Enter Data as the vNIC name, choose the MAC pool created earlier, click the radio button **fabric B** for fabric ID, check the Enable Failover check box. check the check boxes RHOS-Data and various tenant VLANs with RHOS-Data as the native VLAN. For Adapter Policy field, choose **Linux**.

Figure 47 Creating vNIC for VM Data Traffic



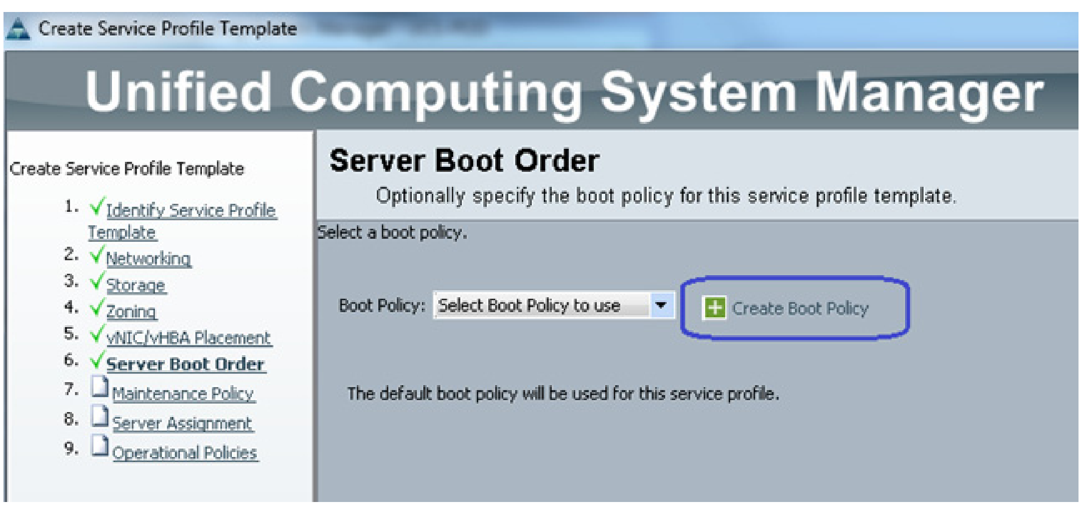
6. In the Storage window, for Local Storage, choose **Create a Specific Storage Policy** option from the drop-down list. For mode choose, **RAID 6 Stripped Dual Parity** option from the drop-down list. click the **No vHBA** radio button for SAN connectivity.

Figure 48 Creating Service Profile Template - Storage Configuration Details



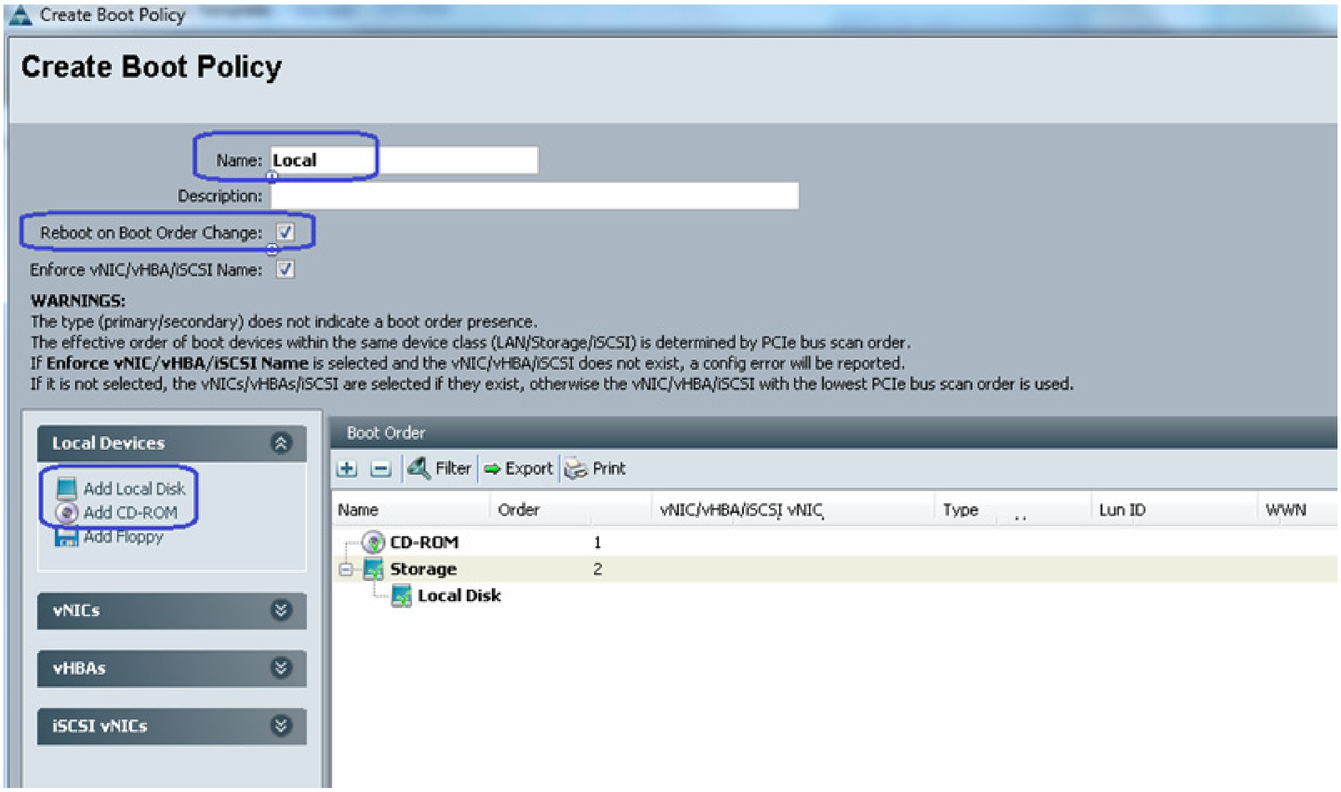
7. Keep default configurations in Zoning and vNIC/vHBA Placement windows by simply clicking **Next**.
8. In the Server Boot Order window, click **Create Boot Policy**.

Figure 49 Creating Service Profile Template - Configuring Boot Order



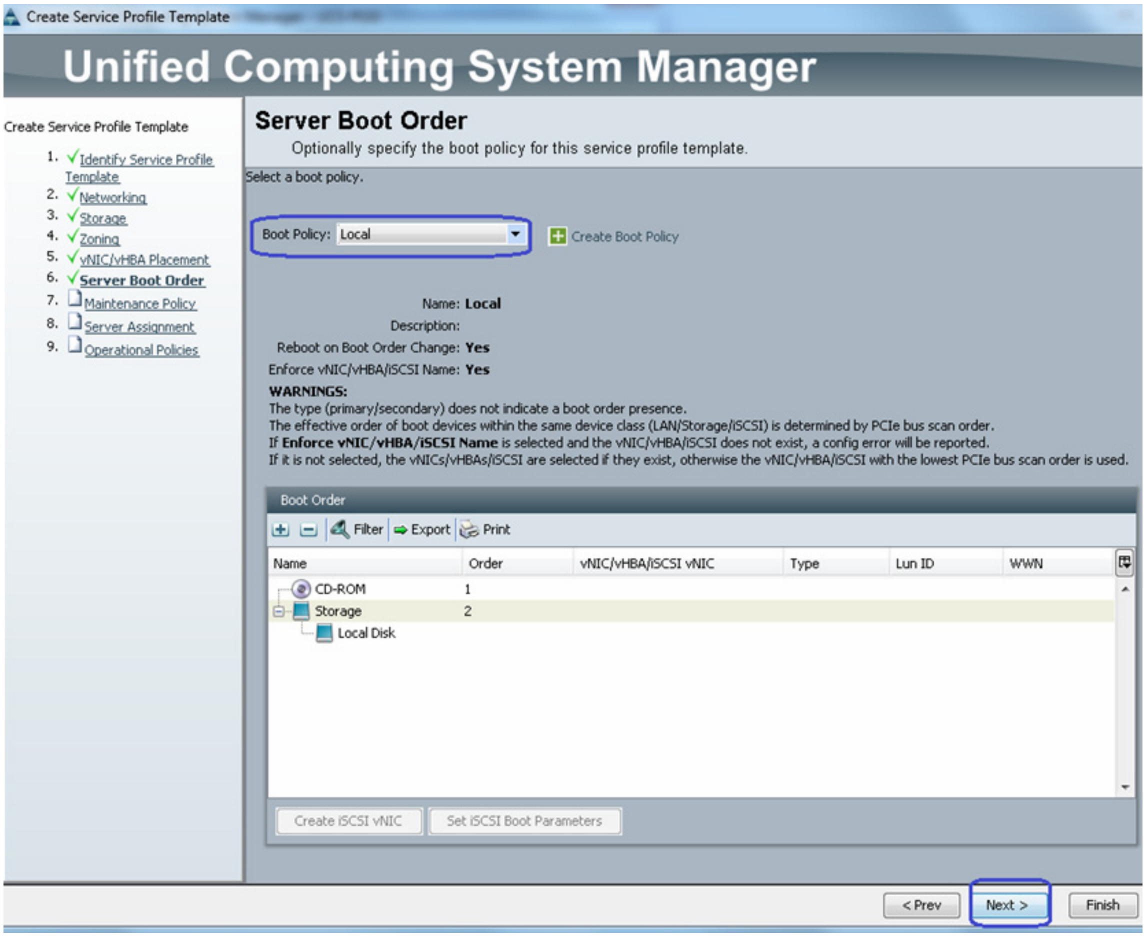
9. In the Create Boot Policy window, enter the name as Local in the Name Field, check the **Reboot on Boot Order Change** checkbox, firstly, click **Add CD-ROM** and then click **Add Local Disk** under Local Devices on left pane of the window. Click **OK** to create the boot policy.

Figure 50 Creating Boot Order Policy



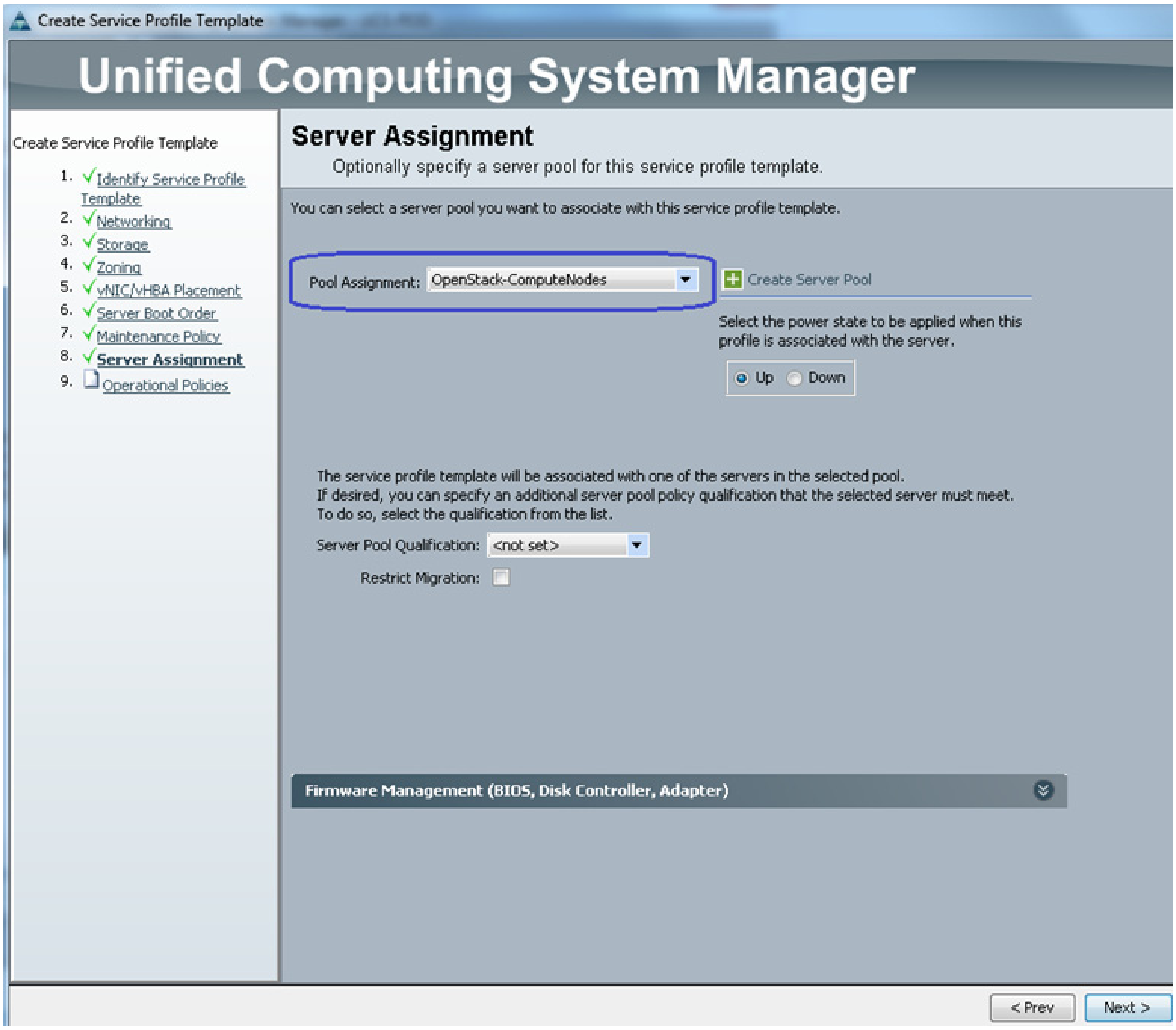
10. Now in the Server Boot order window, for Boot Policy, choose **Local** from the drop-down list. Click **Next**.

Figure 51 Configuring the Server Boot Order



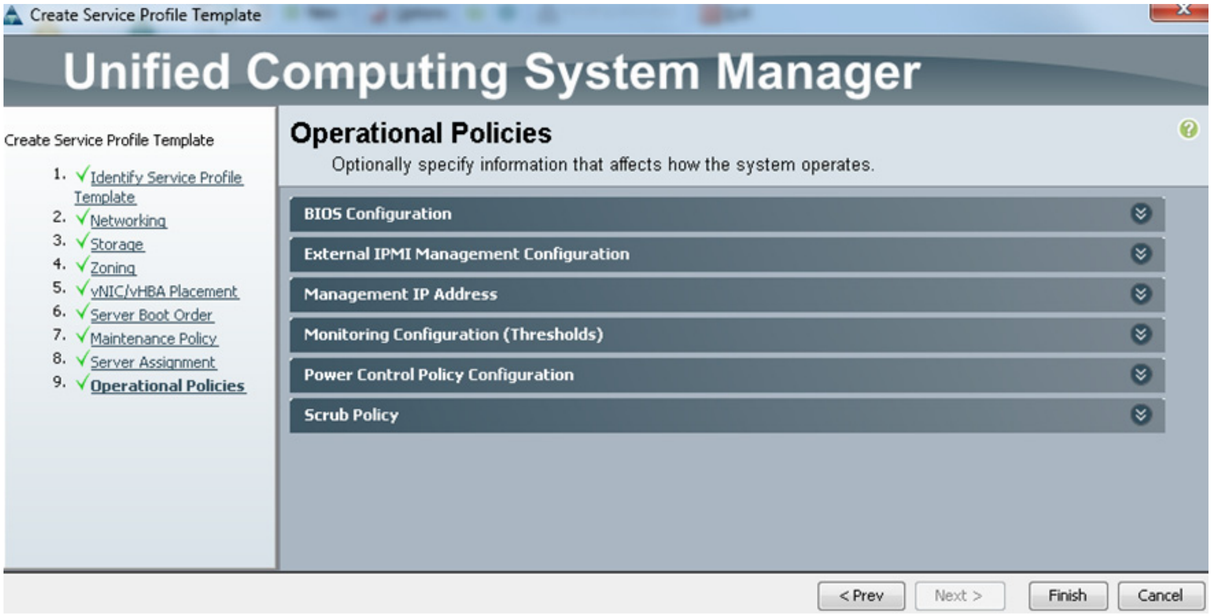
11. Click **Next** to go to the Maintenance Policy window. Keep all the fields at default and click **Next** to continue to Server Assignment window. For Pool Assignment, choose the **OpenStack-ComputeNodes** created earlier. Click **Next**.

Figure 52 Creating Service Profile Template - Configuring Server Assignment



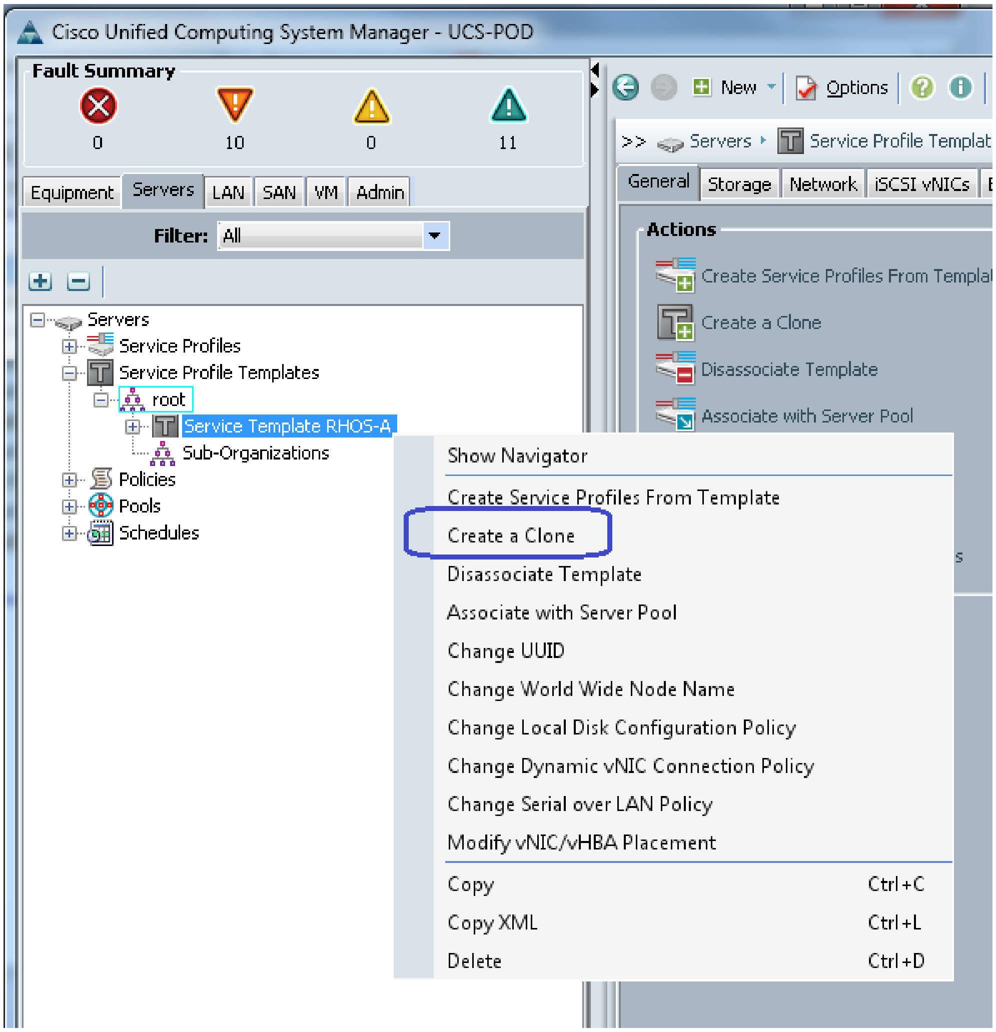
12. In the Operation Policies window, keep all the fields at default, and click **Finish** to deploy the Service Profile Template.

Figure 53 Creating Service Profile Template - Restore Default Settings for Operational Policy



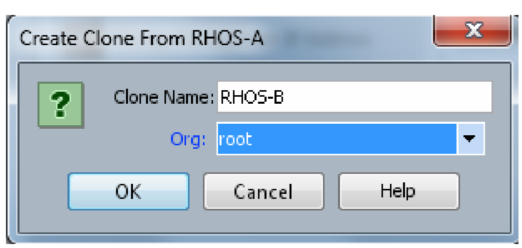
13. We can leverage the RHOS-A service profile template to create templates for RHOS-B and RHOS-Storage. Select the recently created Service Template RHOS-A by expanding Service Profile Template in the the **Servers** tab. **Servers** > **Service Profile Templates** > **root**, right-click on Service Template RHOS-A and click **Create a Clone**.

Figure 54 Cloning a Service Profile Template



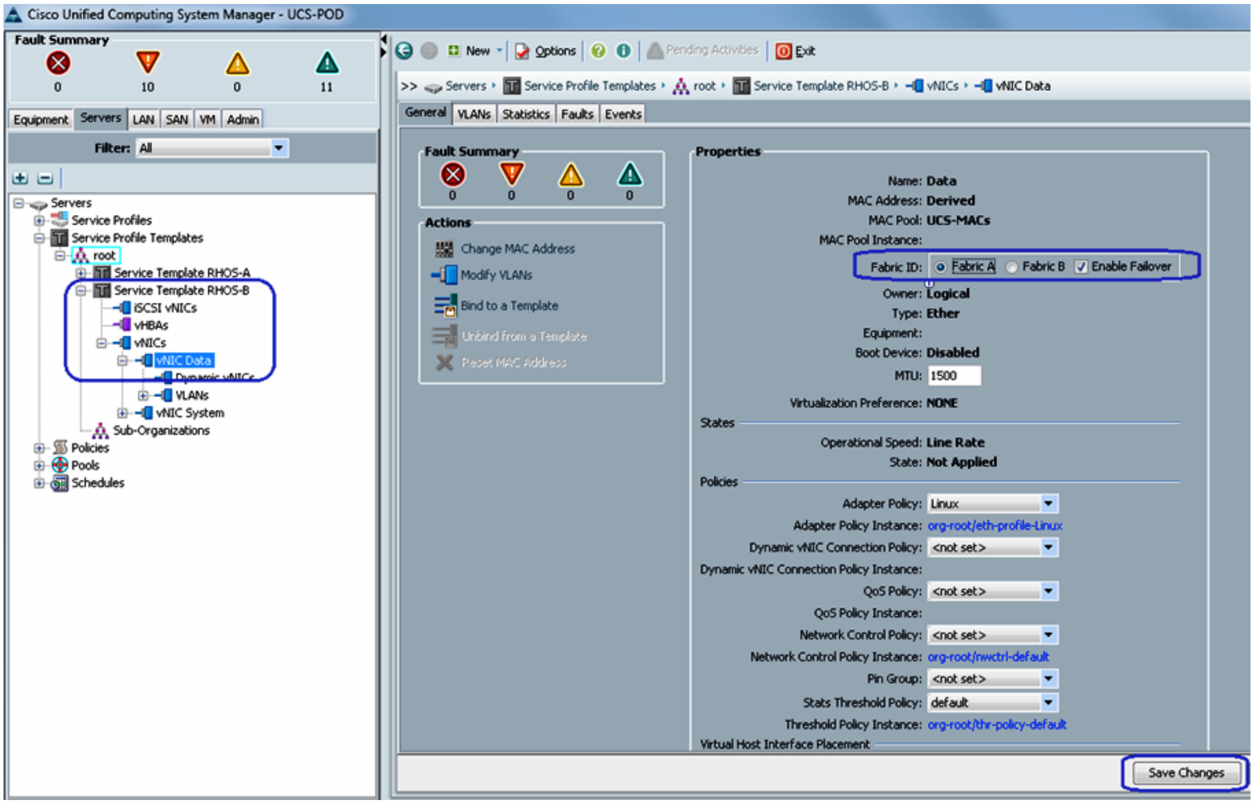
14. Enter the template name RHOS-B and for Org field, choose **root** from the drop-down list and click **OK**. This will create an identical service profile template, with the name RHOS-B.

Figure 55 Cloning RHOS-B from RHOS-A



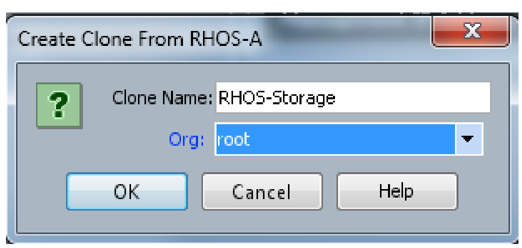
15. The only change that we want to make in RHOS-B is to swap primary fabric IDs of the System and Data VNICs. Expand Service Template RHOS-B, expand vNICs and select **vNIC Data** and change the Fabric ID to **Fabric A**. Click **Save Changes**.

Figure 56 Details of Service Template RHOS-B



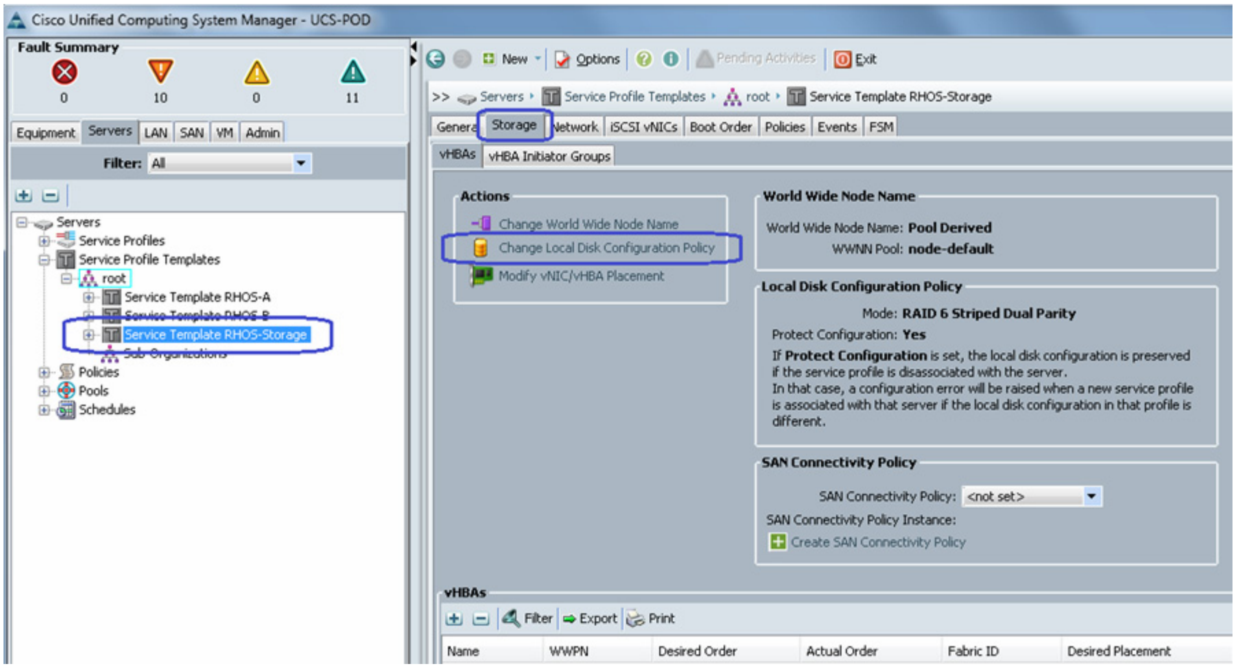
16. Similarly, go to System vNIC, and change its Fabric ID to **Fabric B** and click **Save Changes**.
17. Now repeat step 13 to clone Service Template RHOS-Storage from RHOS-A. Enter the name as RHOS-Storage and For Org, choose the option **root** from the drop-down list.

Figure 57 Cloning RHOS-Storage form RHOS-A



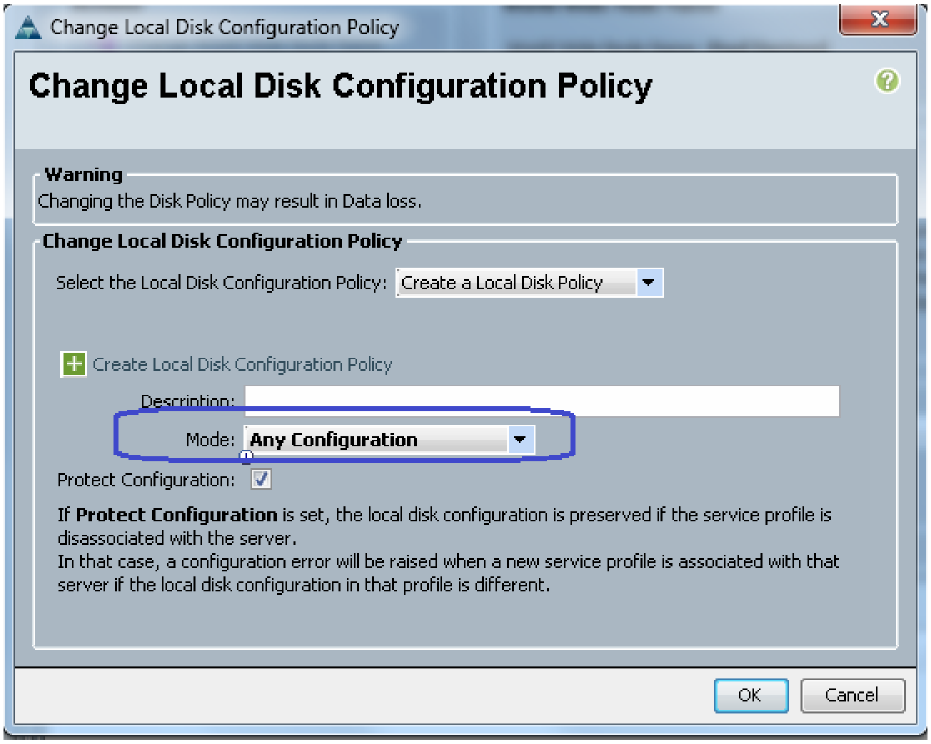
18. We need to edit the created Service Template RHOS-Storage. Select RHOS-Storage, click the **Storage** tab in the right pane, and click **Change Local Disk Configuration Policy**.

Figure 58 Changing Local Disk Configuration Policy for RHOS-Storage



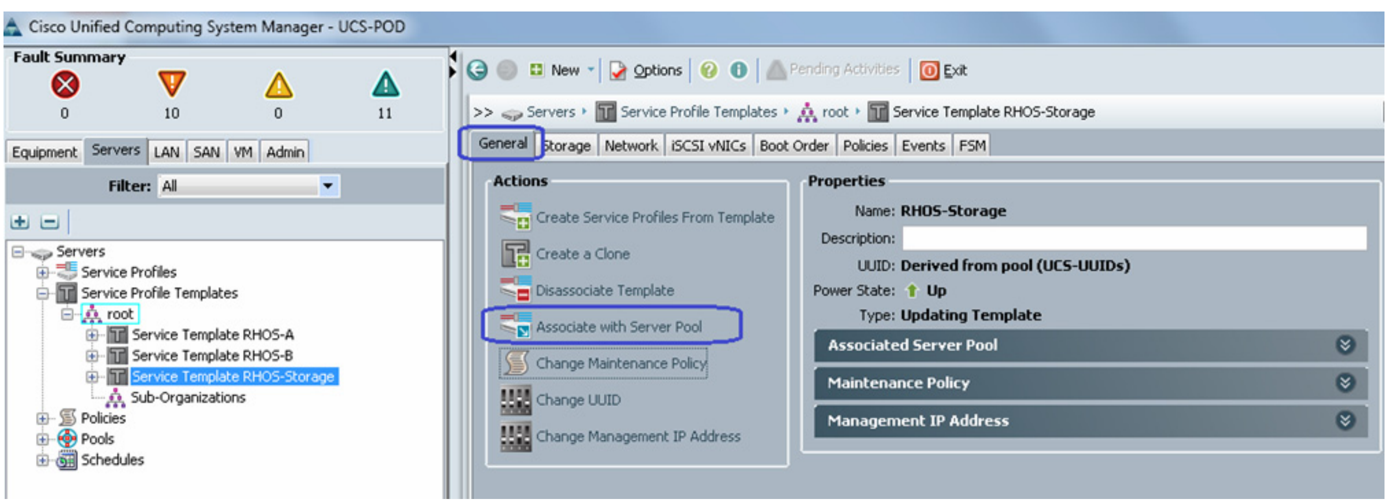
19. In the Change Local Disk Configuration Policy window, choose the option **Any Configuration** from the drop-down list for Mode. By selecting this option, UCS Manager will not alter any local disk configurations that were made off-line. We will expose individual disks as RAID0 configuration later. Click **OK** to save the changes.

Figure 59 Changing Local Disk Configuration Policy



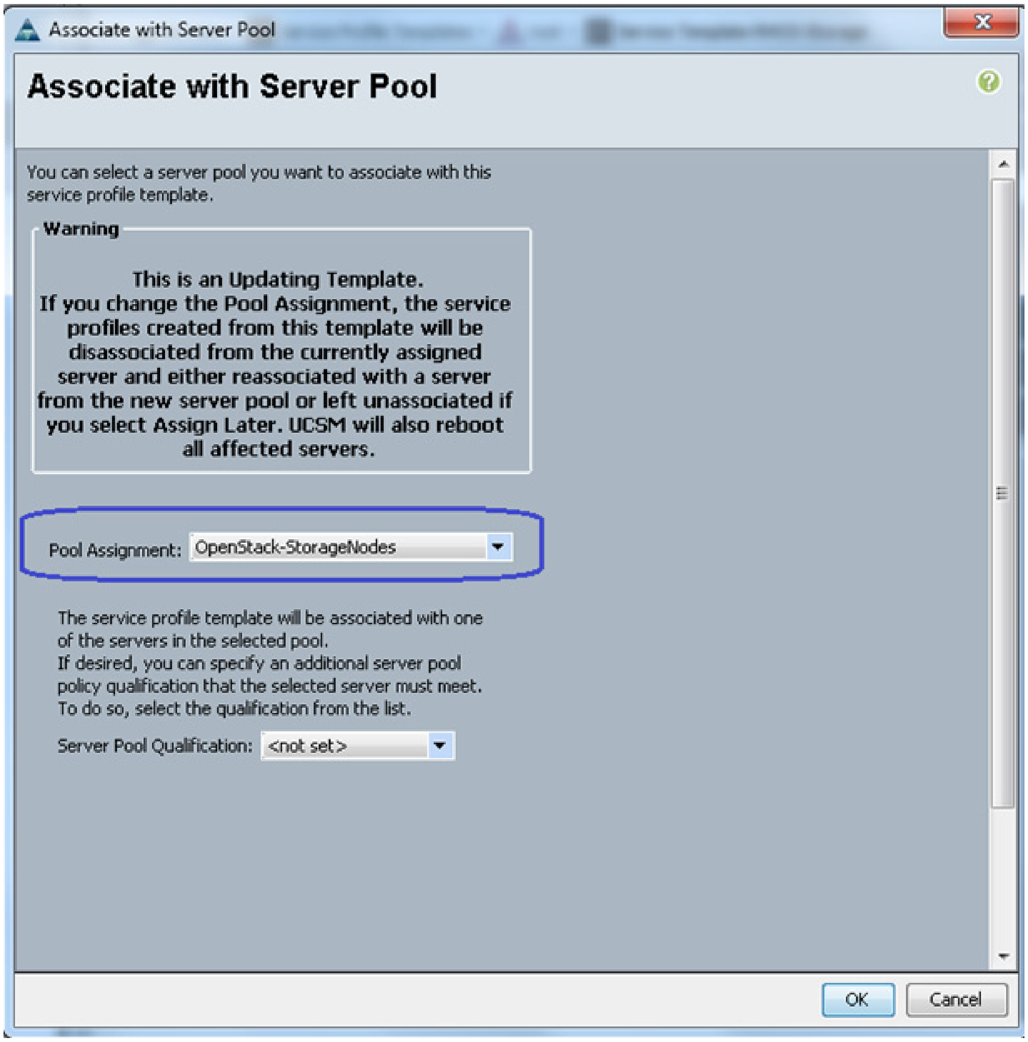
20. From the **Servers** tab, expand **root** and select **Service Template RHOS-Storage**. Click the **General** tab on the right pane of the window, and click **Associate with Server Pool**.

Figure 60 Associating the Template with the Server Pool



21. For Pool Assignment, choose the option **OpenStack-StorageNodes** from the drop-down list. Click **OK** to save the changes.

Figure 61 Associating Service Profile Template with the Server Pool

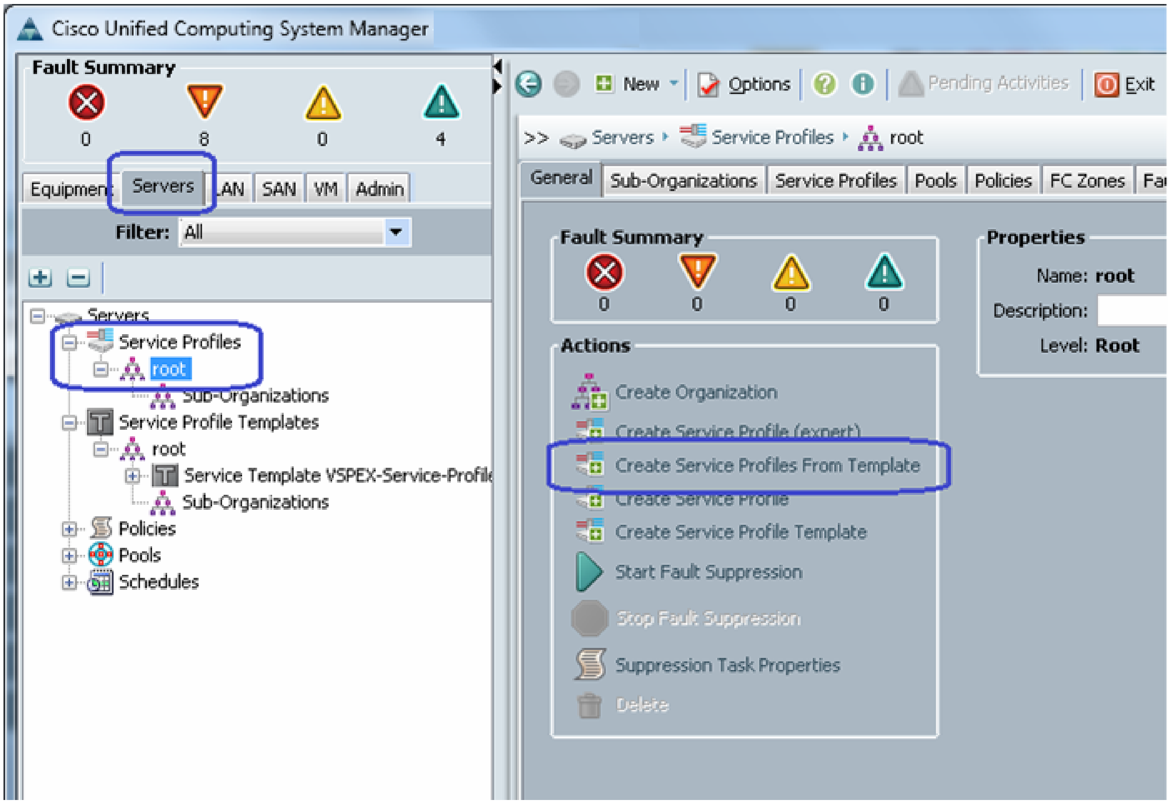


Instantiate Service Profiles from the Service Profile Template

As a final step to configure UCS Manager, we need to instantiate service profiles from the service profile template created in "Configure Service Profile Template" section on page 49. Follow these steps to instantiate service profiles from the service profile template:

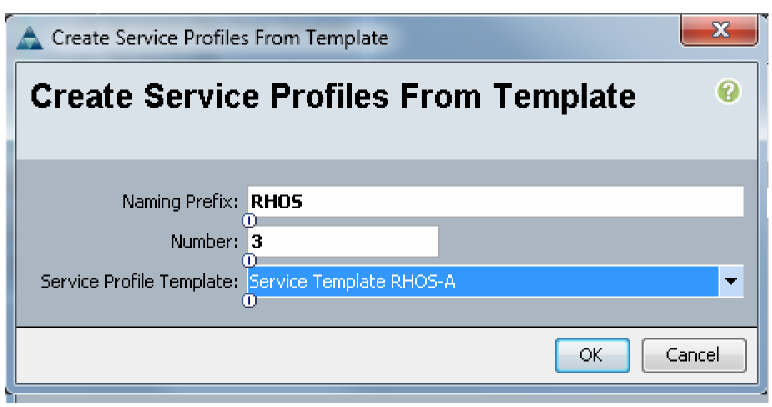
1. From the **Servers** tab, expand **Servers** > **Service profiles** > **root**, and click the **Create Service Profile from Template link** in the right pane.

Figure 62 Creating Service Profile from Template



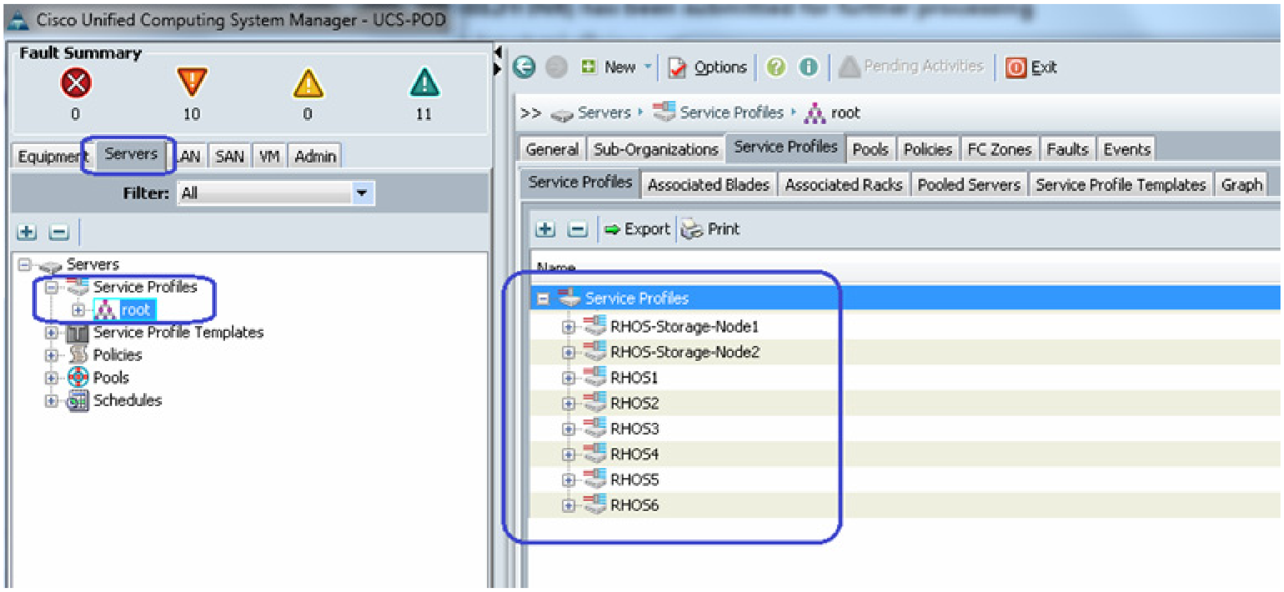
2. Enter the name as RHOS-A and for number of service profiles to be instantiated, enter 3 and choose the service profile template from the drop-down list.

Figure 63 Details for Creating Service Profiles



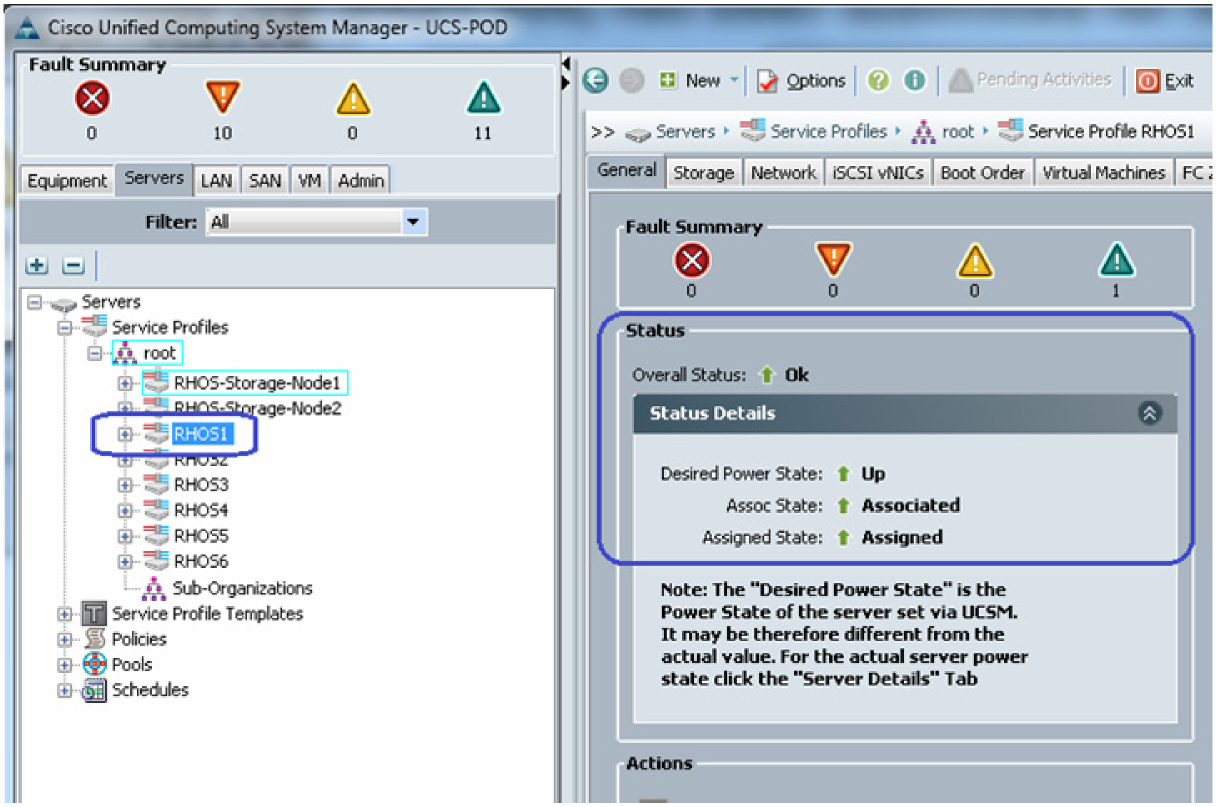
3. Repeat steps 1 and 2, for Service Template RHOS-B, with the same Name RHOS and same number of servers. Again, repeat steps 1 and 2 for Service Template RHOS-Storage. Enter the name as RHOS-Storage-Node, enter 2 for Number, and choose **RHOS-Storage** as service profile template from the drop-down list. Three service profiles are created in this example.
4. Six service profiles for compute nodes and two service profiles for storage nodes are created in this example as shown in Figure 64.

Figure 64 Window Showing All the Service Profiles Created from the Template



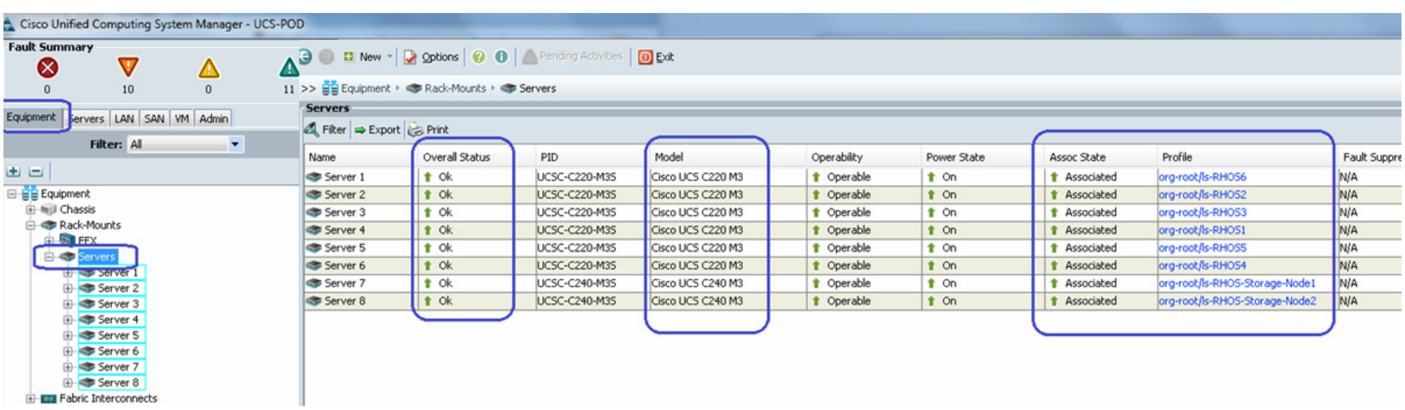
5. As the service profile template is assigned to a server pool, the service profiles instantiated from the template would be assigned to individual server resource from the server pool as far as they are available. You can select a given service profile to see its association state, and with which server it is associated.

Figure 65 Status Details Of Service Profiles



6. Eventually, all the four servers will be associated – you can see the summary by clicking **Servers** in the **Equipment** tab.

Figure 66 Summary of Service Profiles Showing Assigned State as Associated



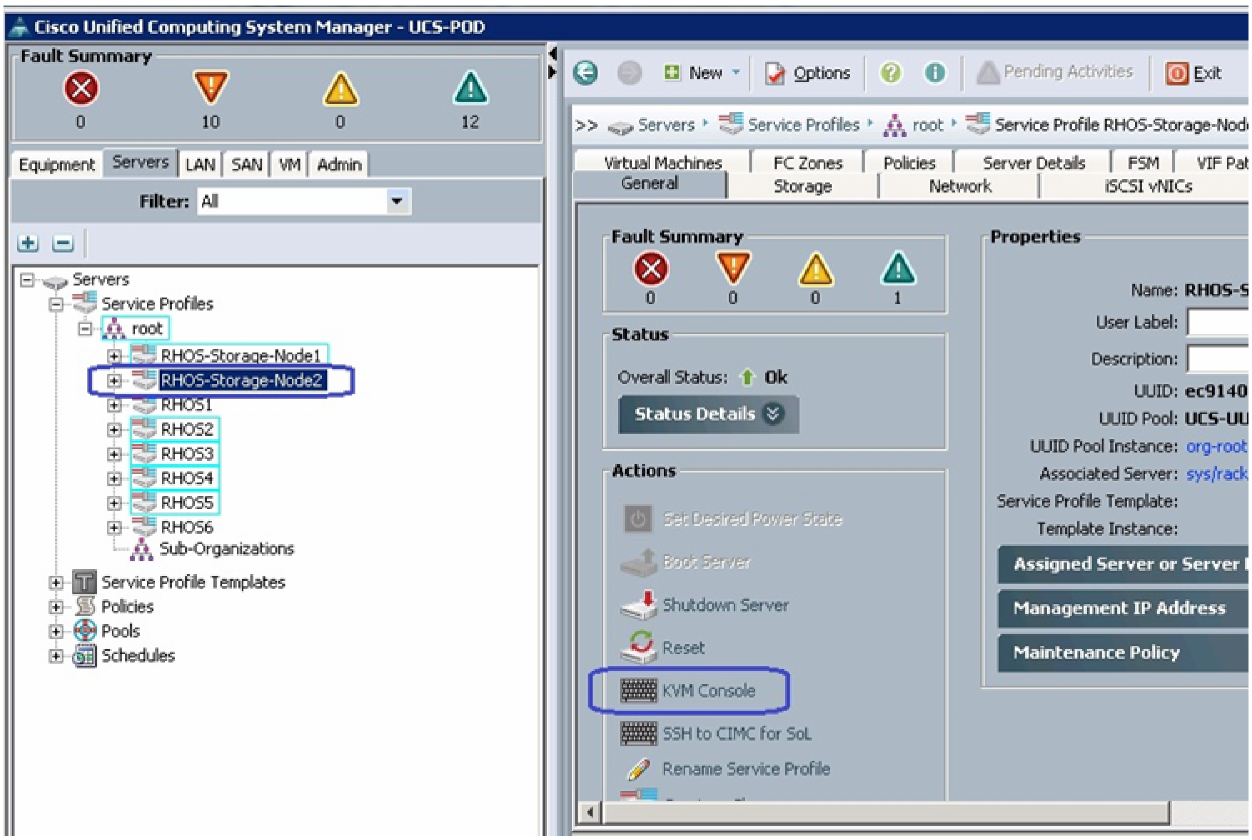
Configure Storage Node Local Disk

For storage nodes, we have 24 local hard drives. We use first two disks to install RHEL 6.4 and remaining 22 disks are exposed to OpenStack Cinder module to provide highly available block and object storage for the VMs instantiated on the compute nodes. Follow these steps to configure each storage node:

Note Automated scripts will soon be provided on Cisco Developers Network for this configuration.

1. From UCS Manager GUI, click the **Servers** tab, expand **Servers** > **Service Profiles** > **root**, and select a particular service profile. Click KVM Console in the right pane of the window.

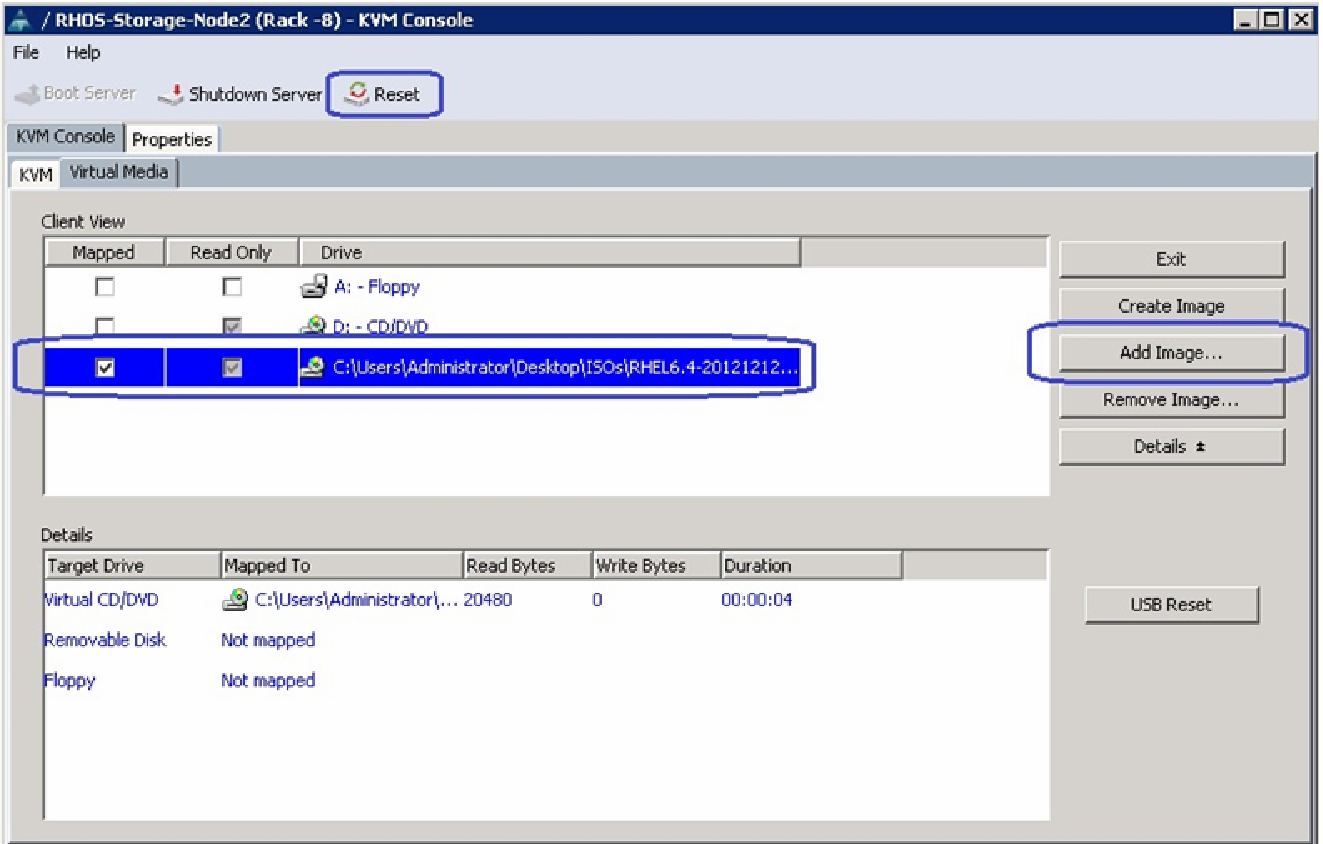
Figure 67 Launching KVM Console



2. Once the Java pallet of KVM is launched, click the **Virtual Media** tab and click **Add Image**. A window appears to select an ISO image. Browse through the local directory structure and select ISO image of the Red Hat Enterprise Linux 6.4 installer media.

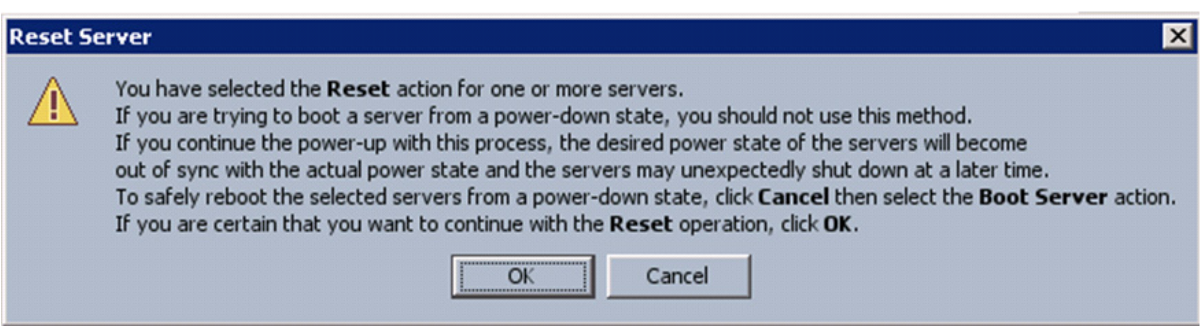
3. When the ISO image shows up in the list, check the Mapped check box and click **Reset** to reset the server.

Figure 68 Adding RHEL ISO Image



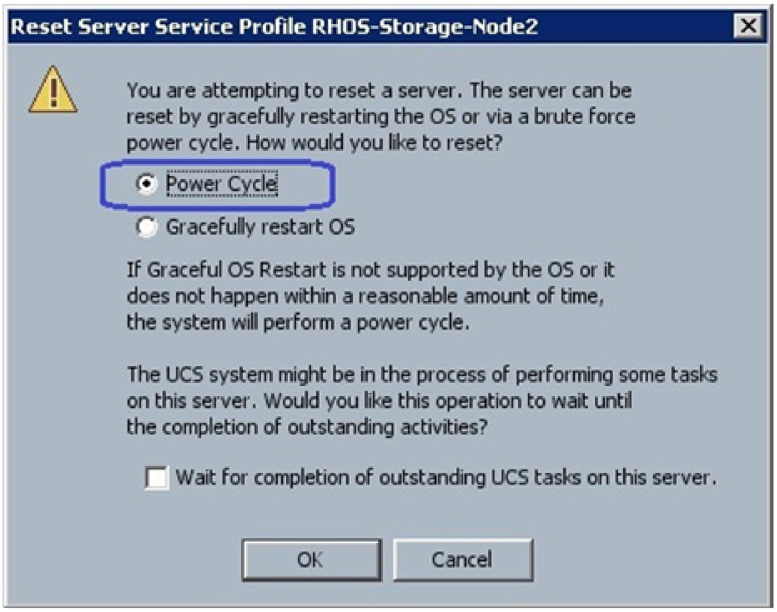
4. Click **OK** in the Reset Server warning message window.

Figure 69 Warning Message for Resetting the Server



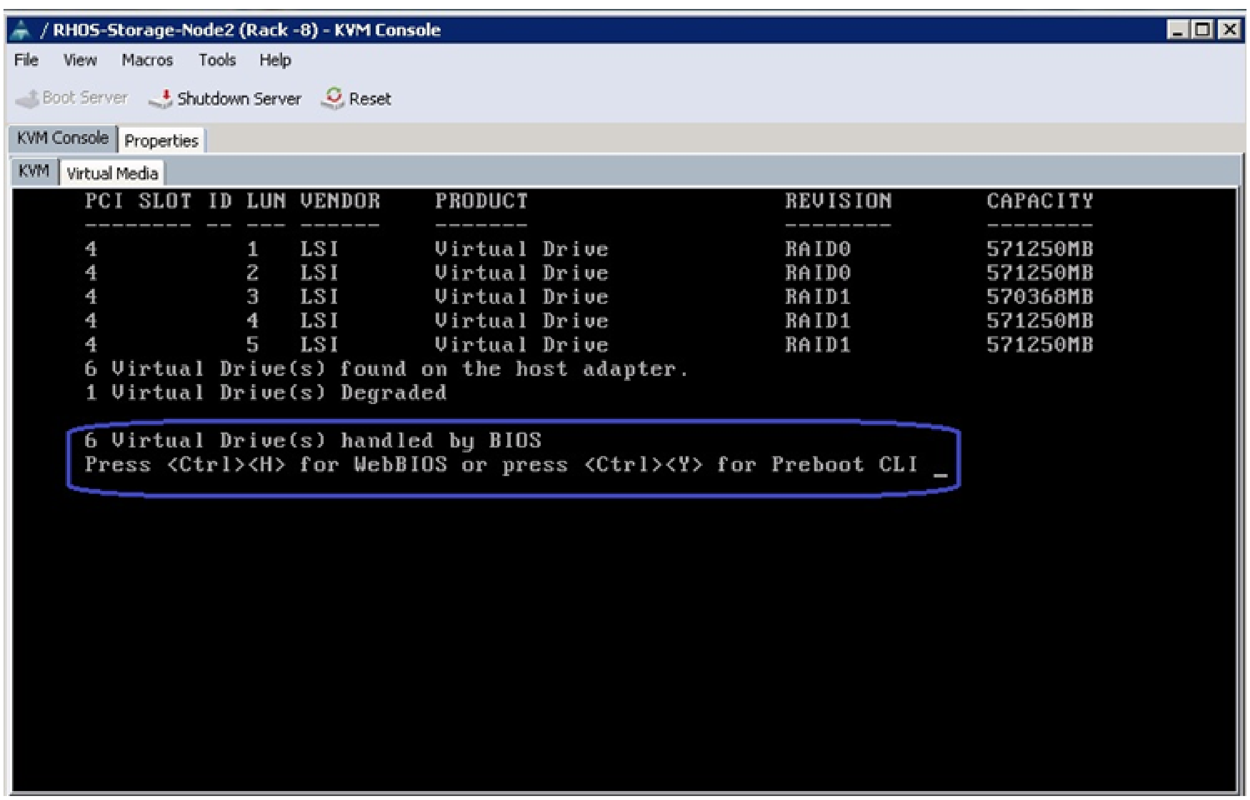
5. Click the **Power Cycle** radio button and click **OK**.

Figure 70 Selecting Resetting Option



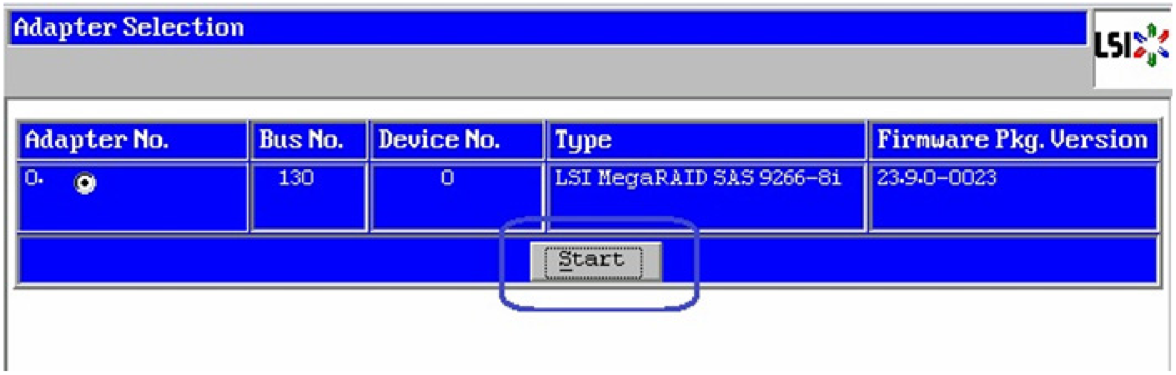
6. Click the **KVM** tab to see the console. In the console press <Ctrl><H> when prompted for entering WebBIOS.

Figure 71 Entering WebBIOS from KVM Console



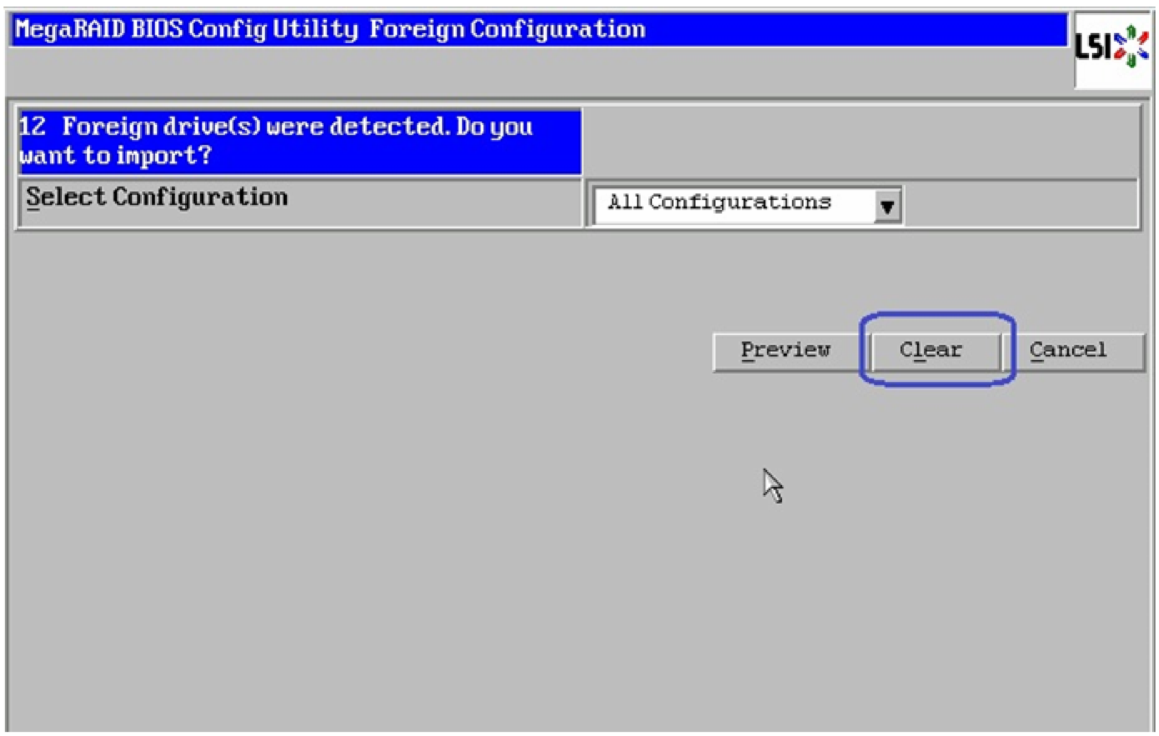
7. Click **Start** to begin the WebBIOS configuration wizard.

Figure 72 Entering WebBIOS Configuration Wizard



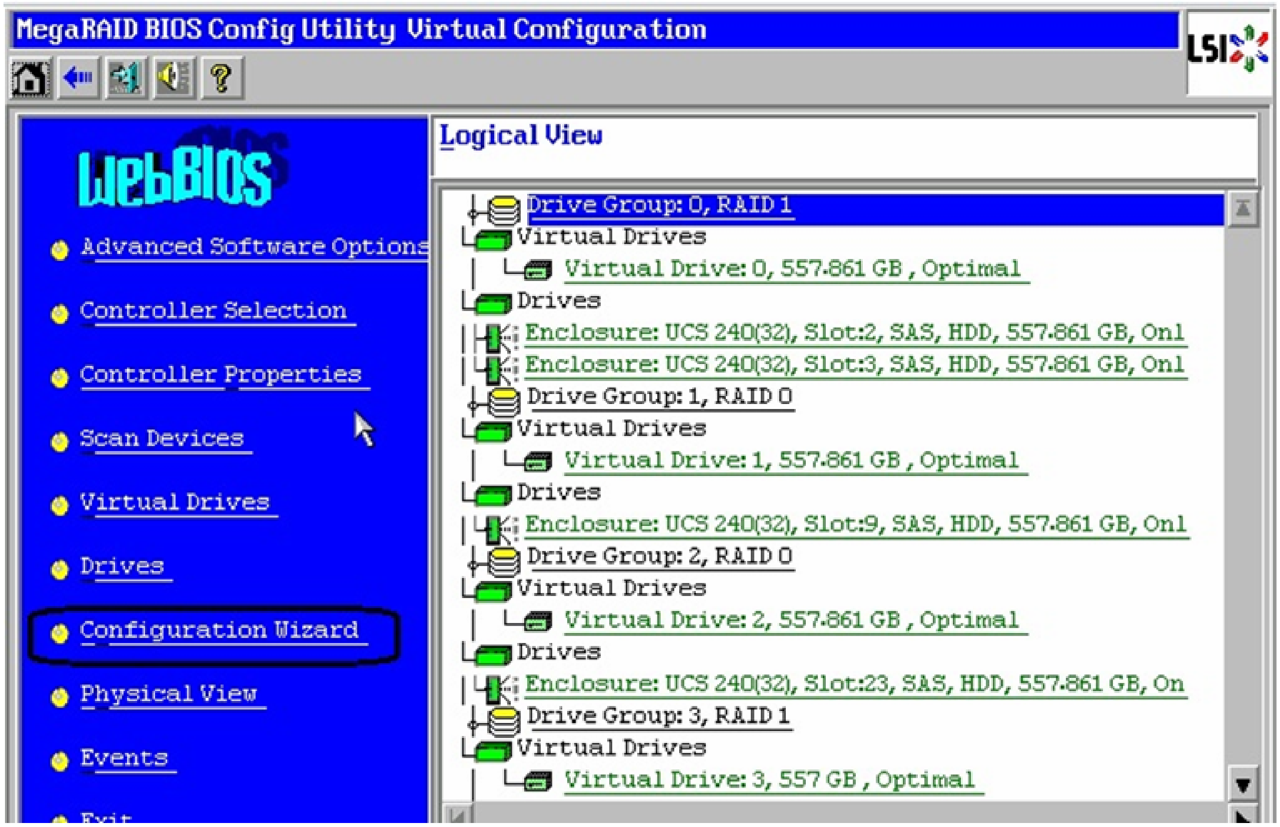
8. Click **Clear** to delete all the existing configurations.

Figure 73 Clearing All Existing Configurations



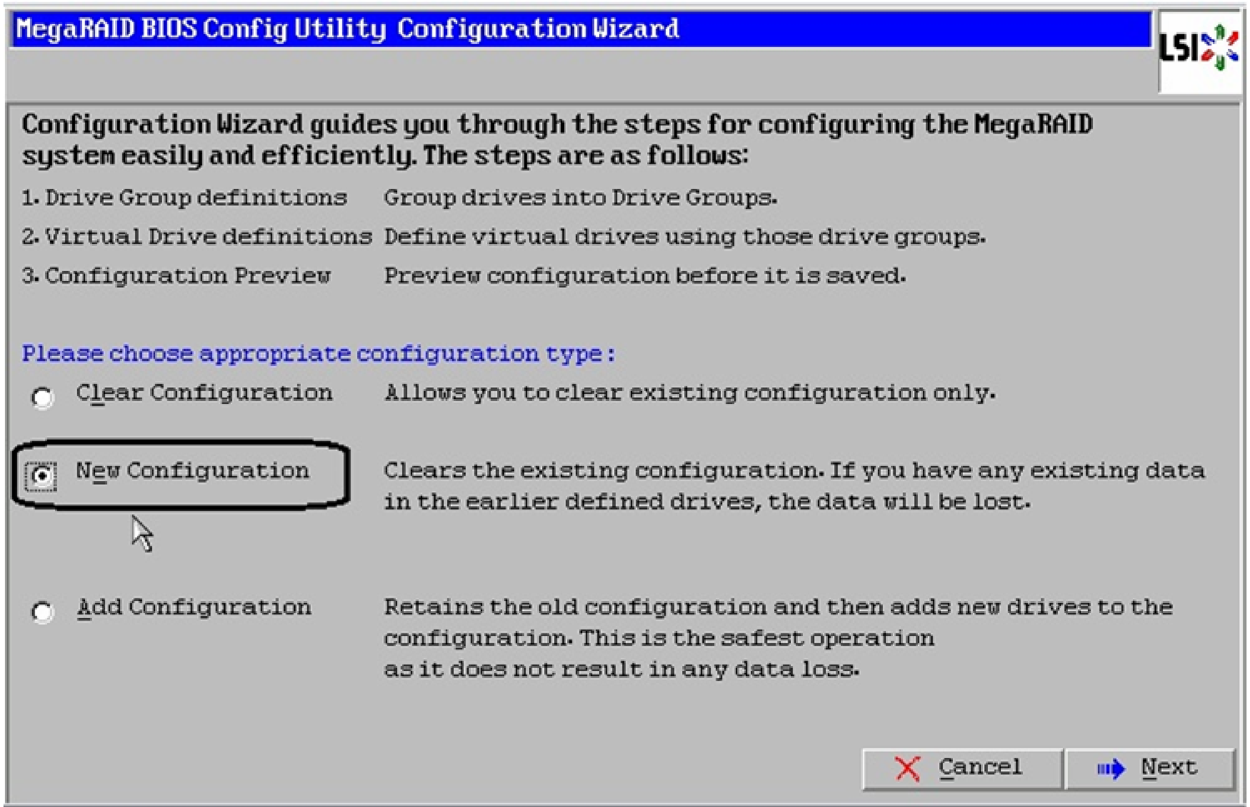
9. Click **Yes** on the confirmation window.
10. Click the option **Configuration Wizard** in the left pane of MegaRAID BIOS Config Utility window to configure all the disks.

Figure 74 Entering the Configuration Wizard



11. Click **New Configuration** and click **Next**.

Figure 75 Choosing the Configuration Type

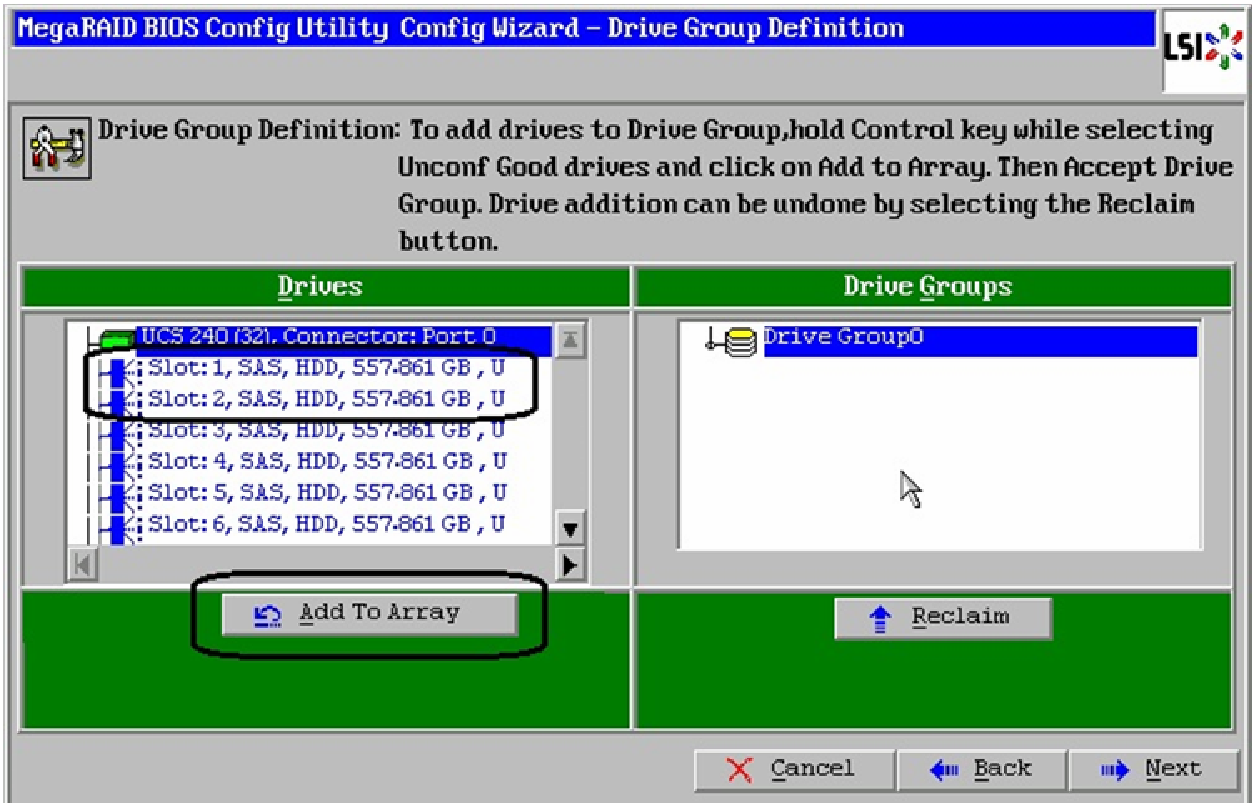


12. Click **Yes** in the confirmation window.
13. Choose the option **Manual Configuration** and click **Next**.

Figure 76 Choosing Configuration Method

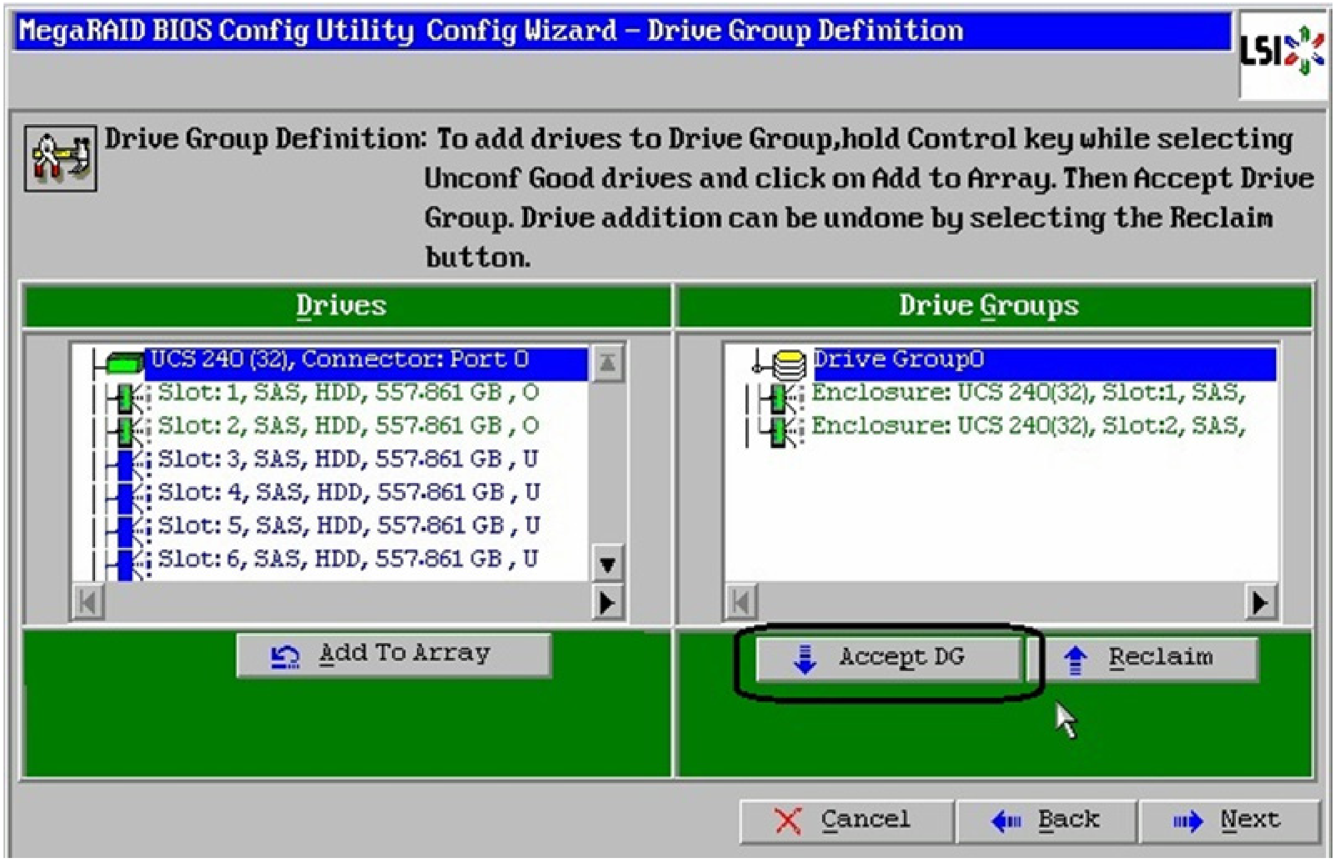
14. Select the first two SAS disks and click **Add to Array** to add them to the Drive Group0.

Figure 77 Adding the SAS Disks to Drive Group



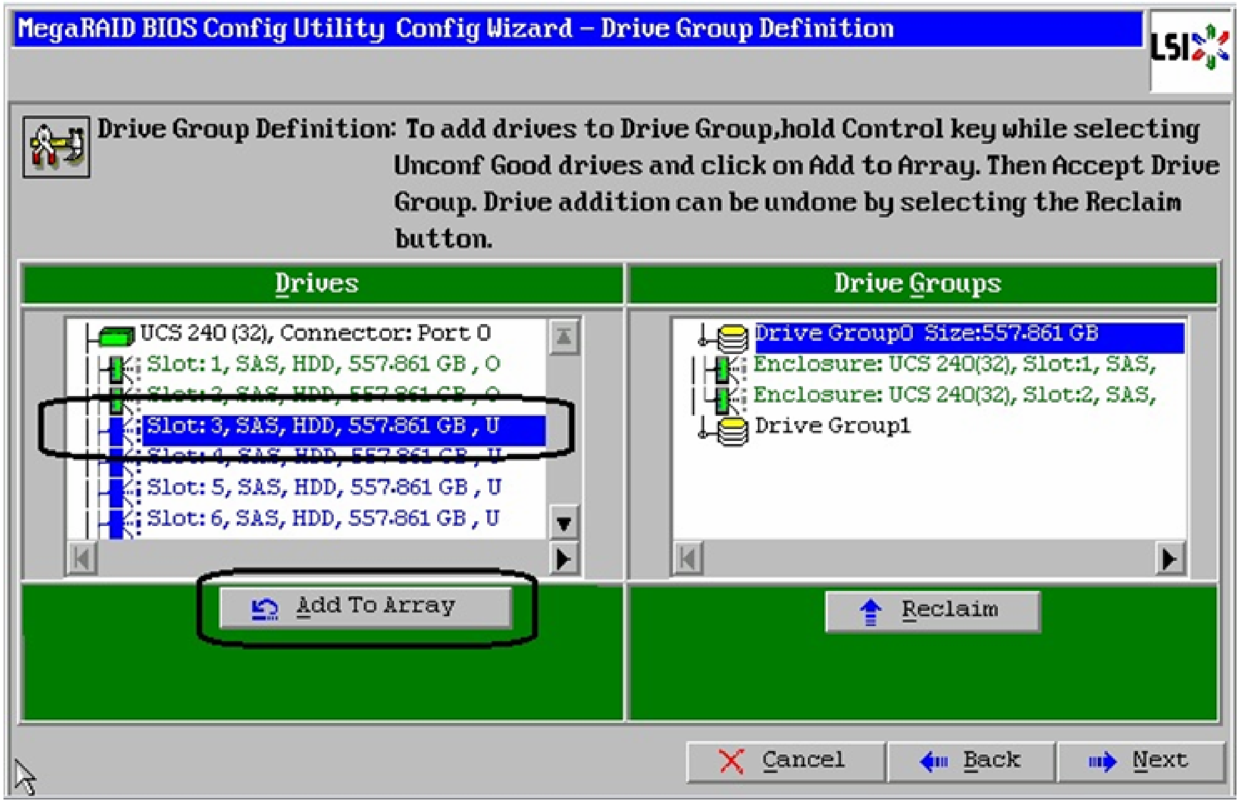
15. Once the disks are added to the drive group, click **Accept DG**.

Figure 78 Accepting the Drive Group



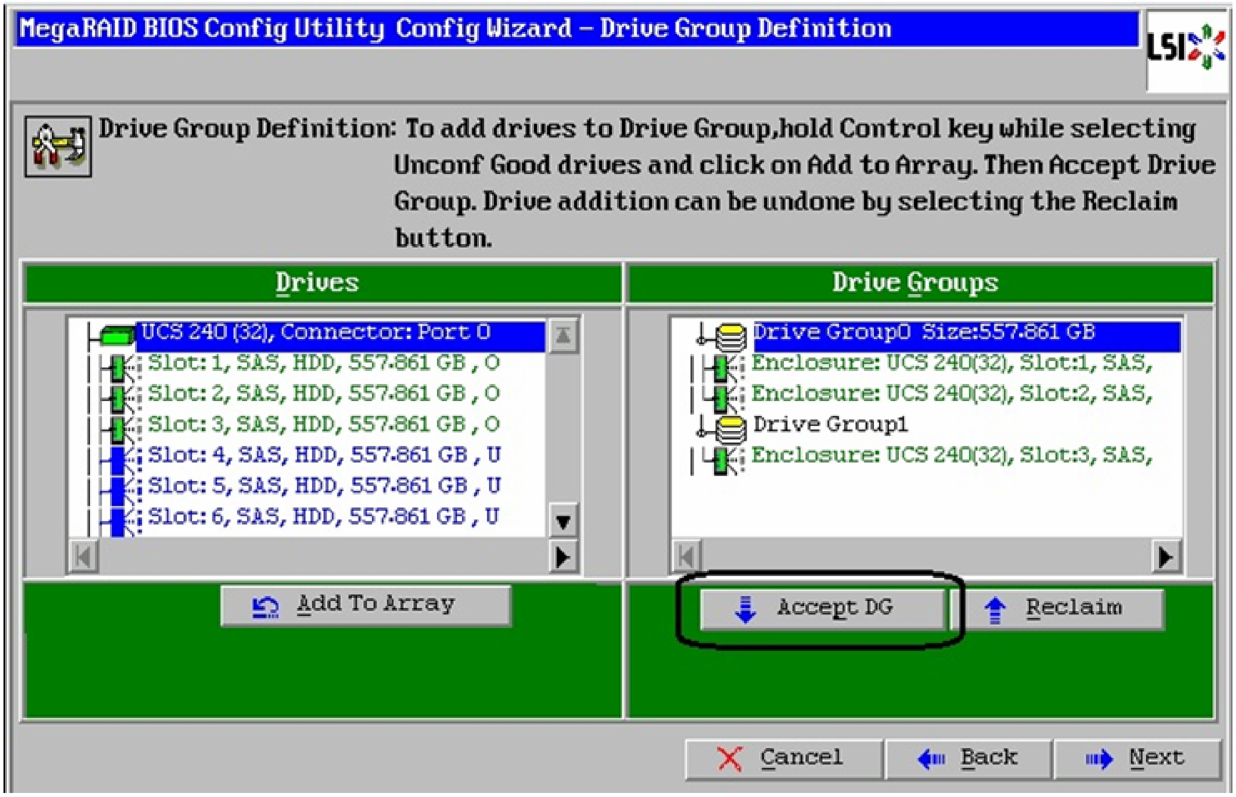
16. Now, select one disk at a time from the list of available drives, and add to the drive group by clicking **Add to Array**.

Figure 79 Adding Drives to Drive Groups



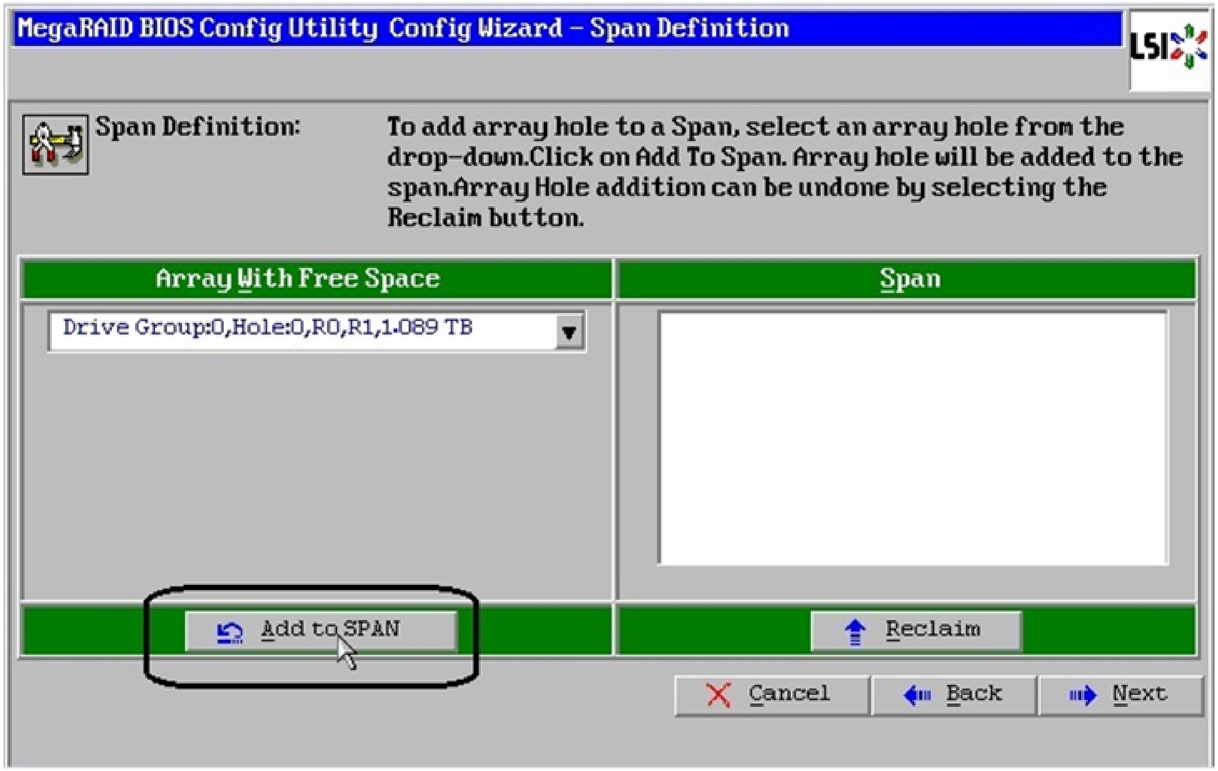
17. Click **Accept DG** to accept the drive group, and repeat step 14 for the all the Unassigned drives on the host.

Figure 80 Accepting Drive Groups



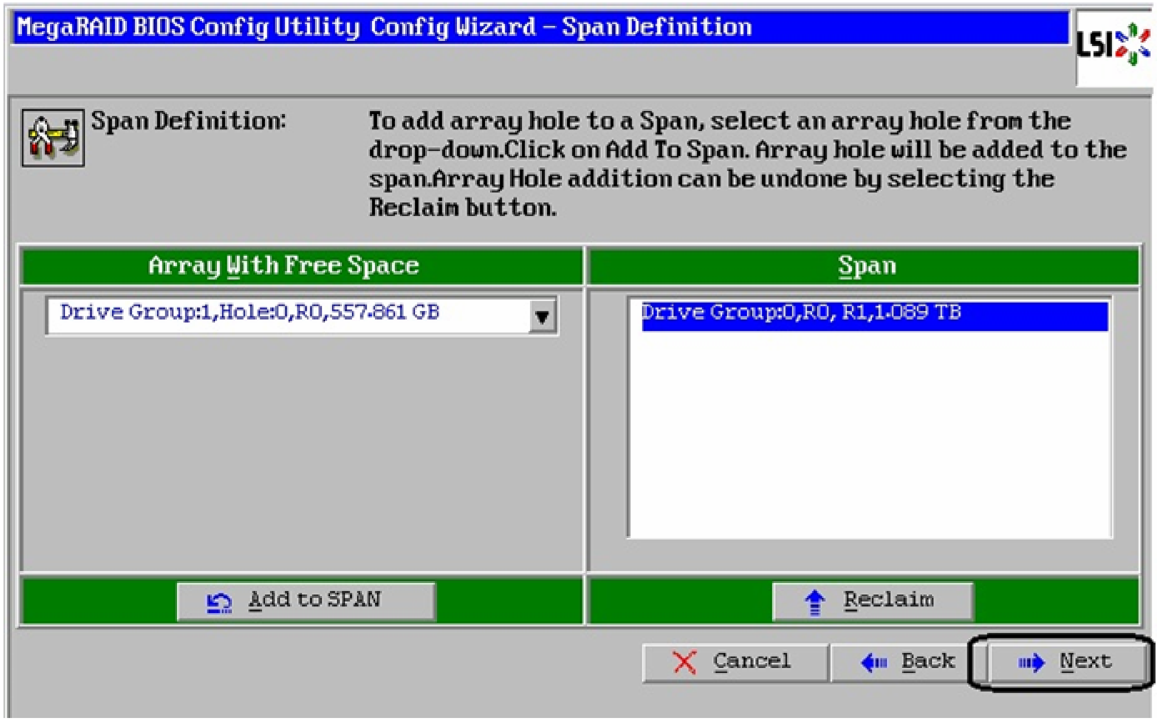
18. Once all the drives are added to the drive groups, click **Next**.
19. From the list of Array with free space, click **Add to SPAN**.

Figure 81 Adding Arrays with Free Space to Span



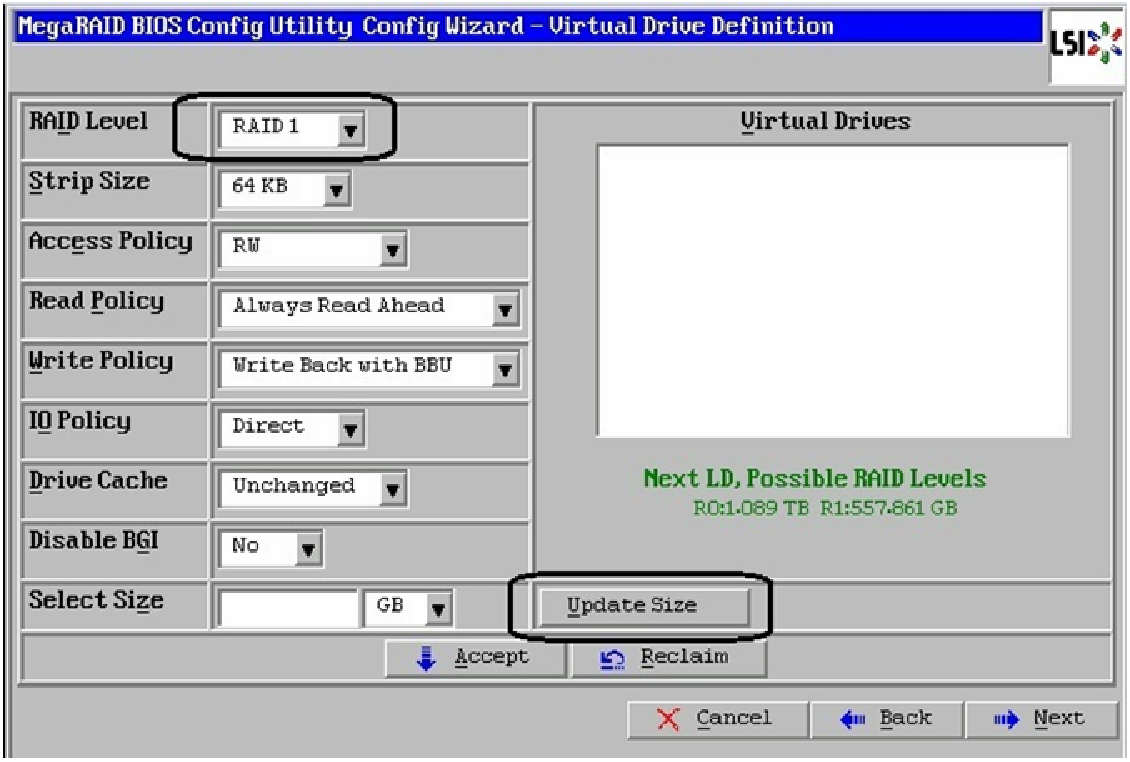
20. Click **Next** once the Drive Group is added to the Span list.

Figure 82 Array Added to Span



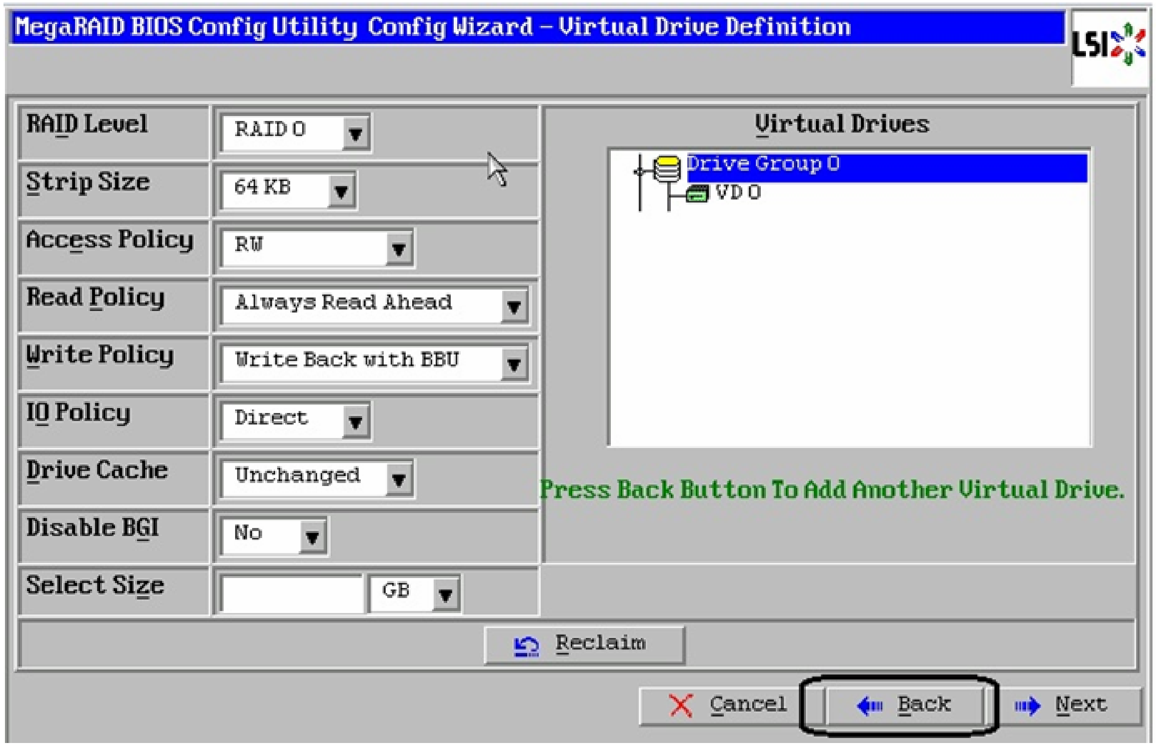
21. For the first two disks, use RAID 1 and click **Update Size**.

Figure 83 RAID 1 Configuration for Operating System



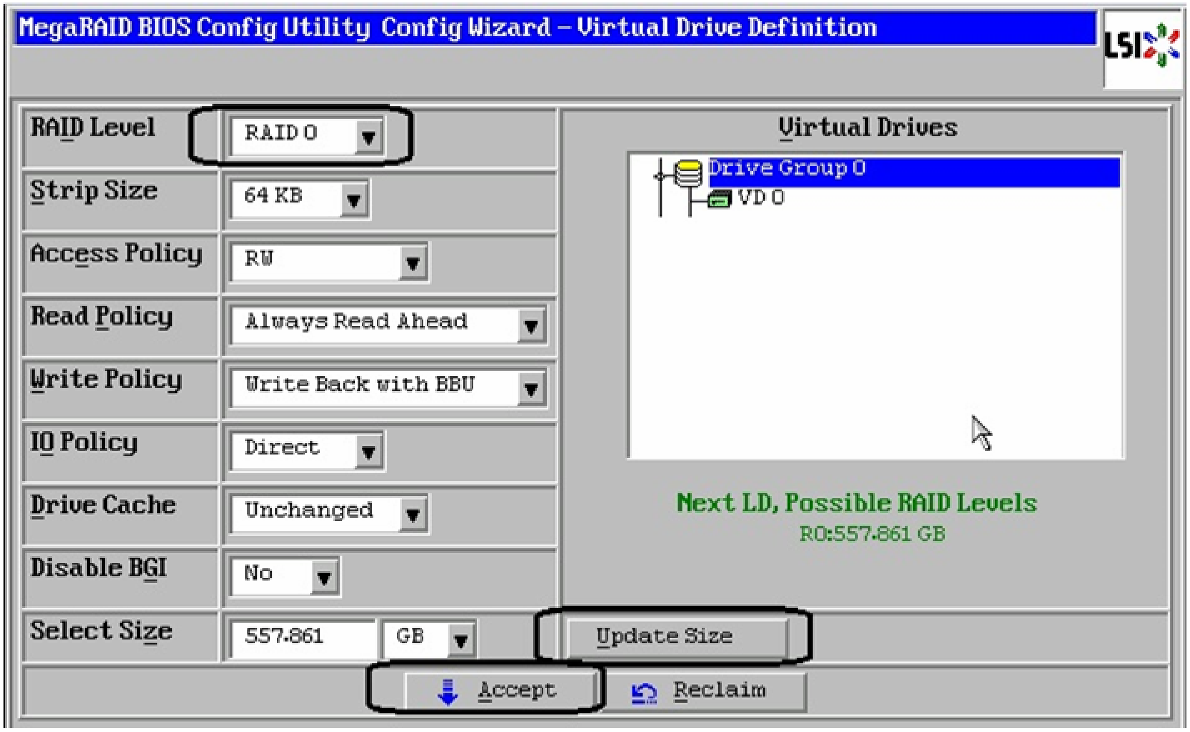
22. Click **Accept**, once the size is updated and click **Yes** in the confirmation window.
23. Click **Back** to select more drive groups from the previous window.

Figure 84 RAID 0 Configuration



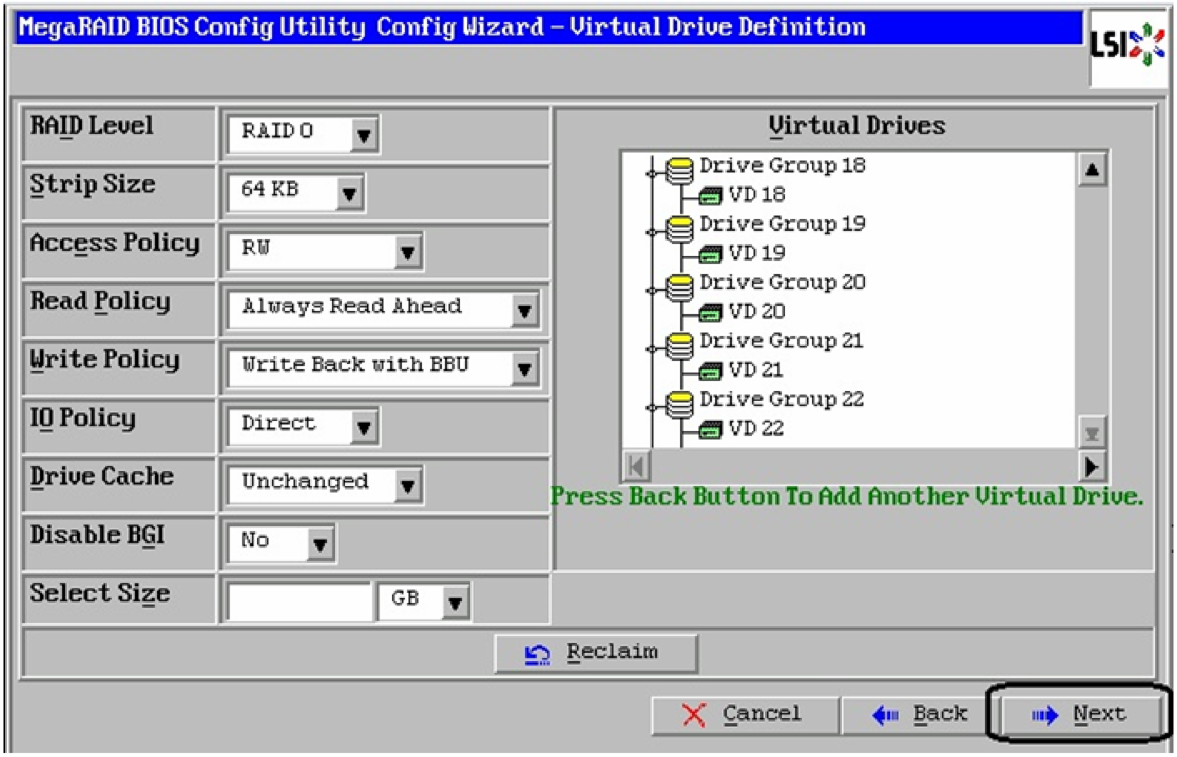
24. From the previous window, repeat steps 18 and 19 to select next drive group and add to Span and click **Next**. For all the drive groups with single drive, select RADI 0, click **Update Size** and once the size is updated click **Accept**. Click **Yes** on the confirmation window. Repeat steps 23 and 24 for all the remaining drives on the system.

Figure 85 RAID 0 Configuration for All the Remaining Disks



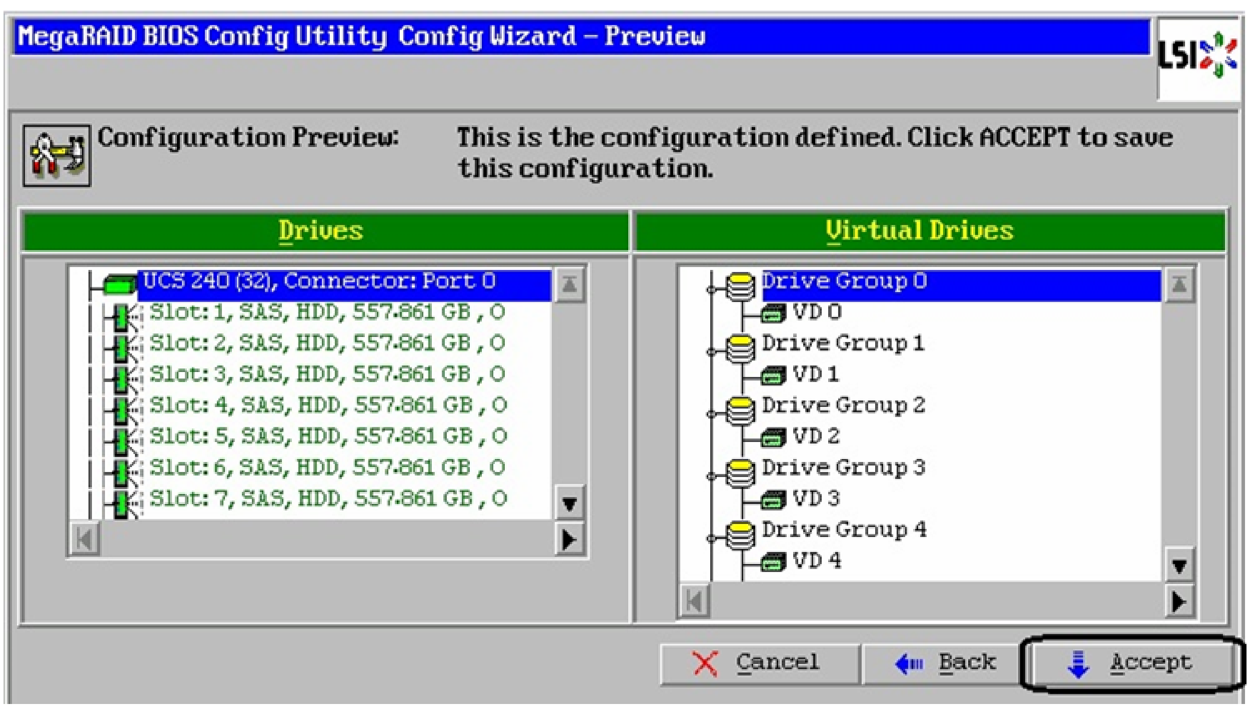
25. Once all the disks are configured in RAID0 drive group (except the first two disks, which are in RAID1), click **Next**.

Figure 86 Window Showing All the Configured Drive Groups



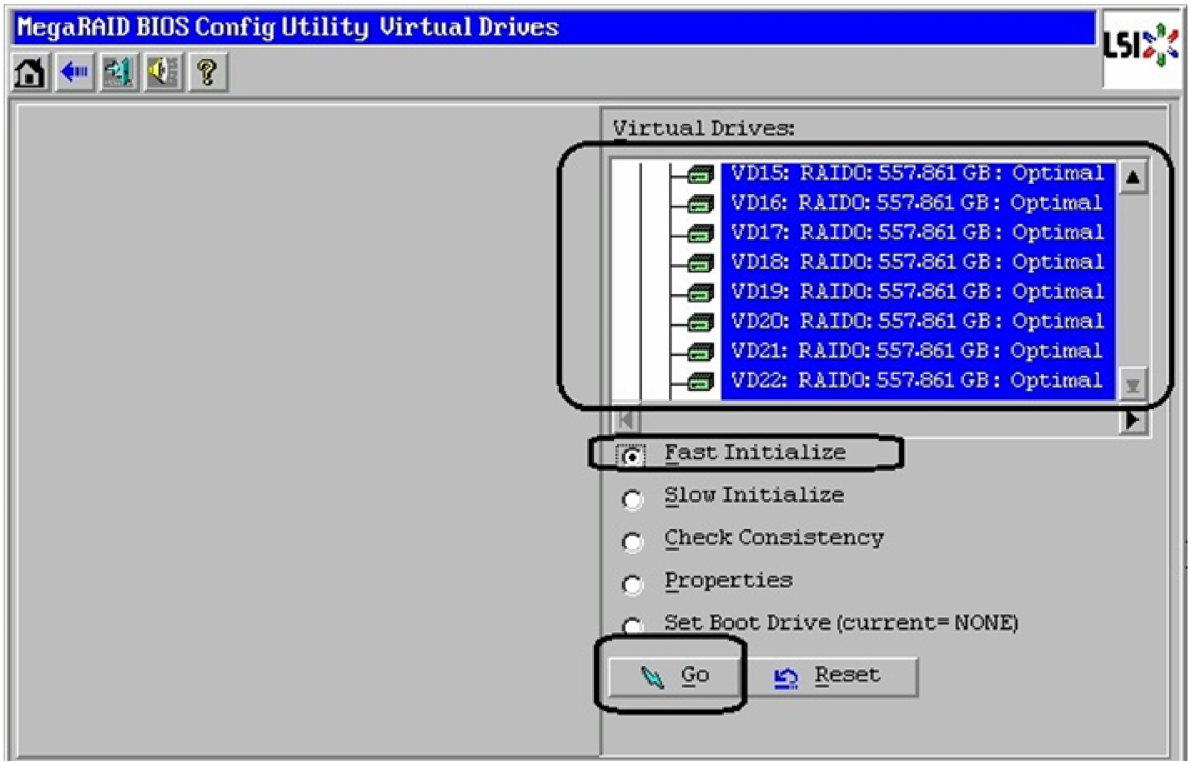
26. Click **Accept** to save the newly defined configuration.

Figure 87 Accept All the Configurations



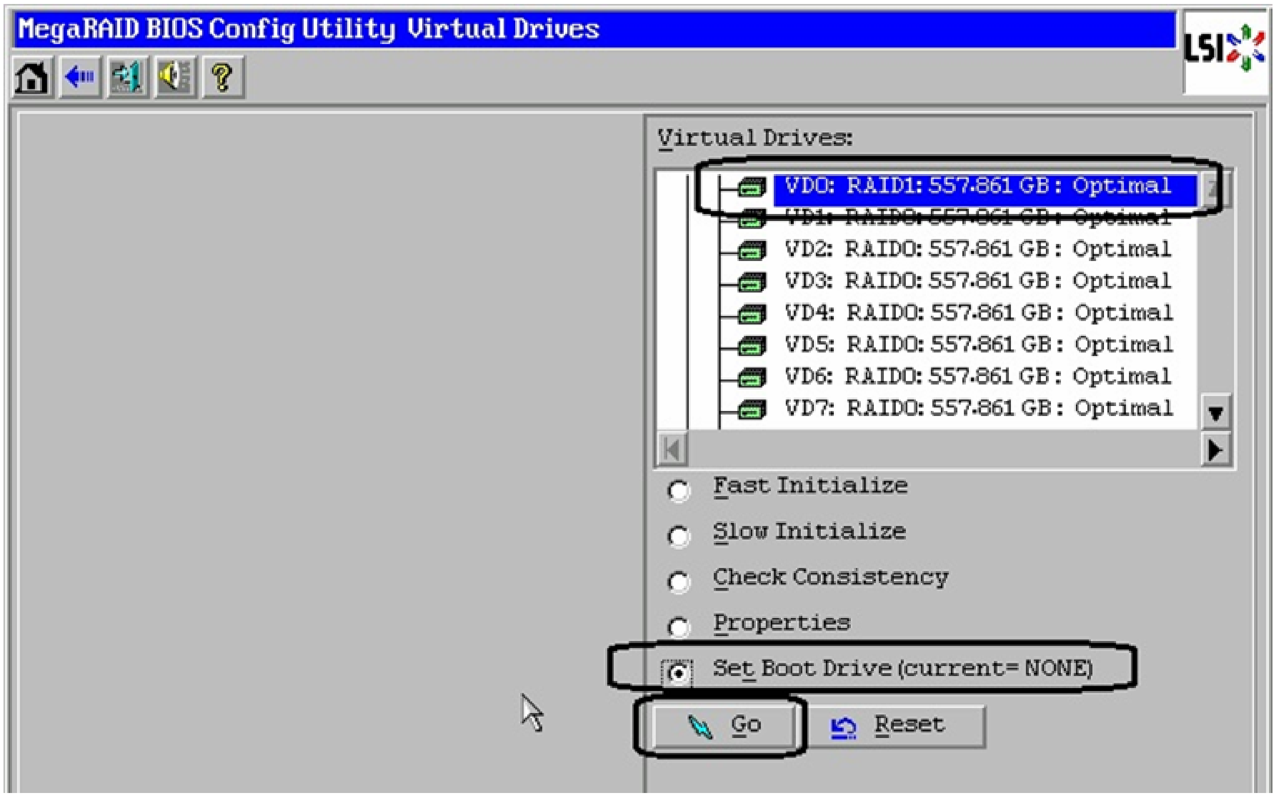
27. Click **Yes** twice to accept all the warning messages.
28. Select all the virtual disks, click the **Fast Initialize** radio button and click **Go**.

Figure 88 Settings of the Virtual Disks



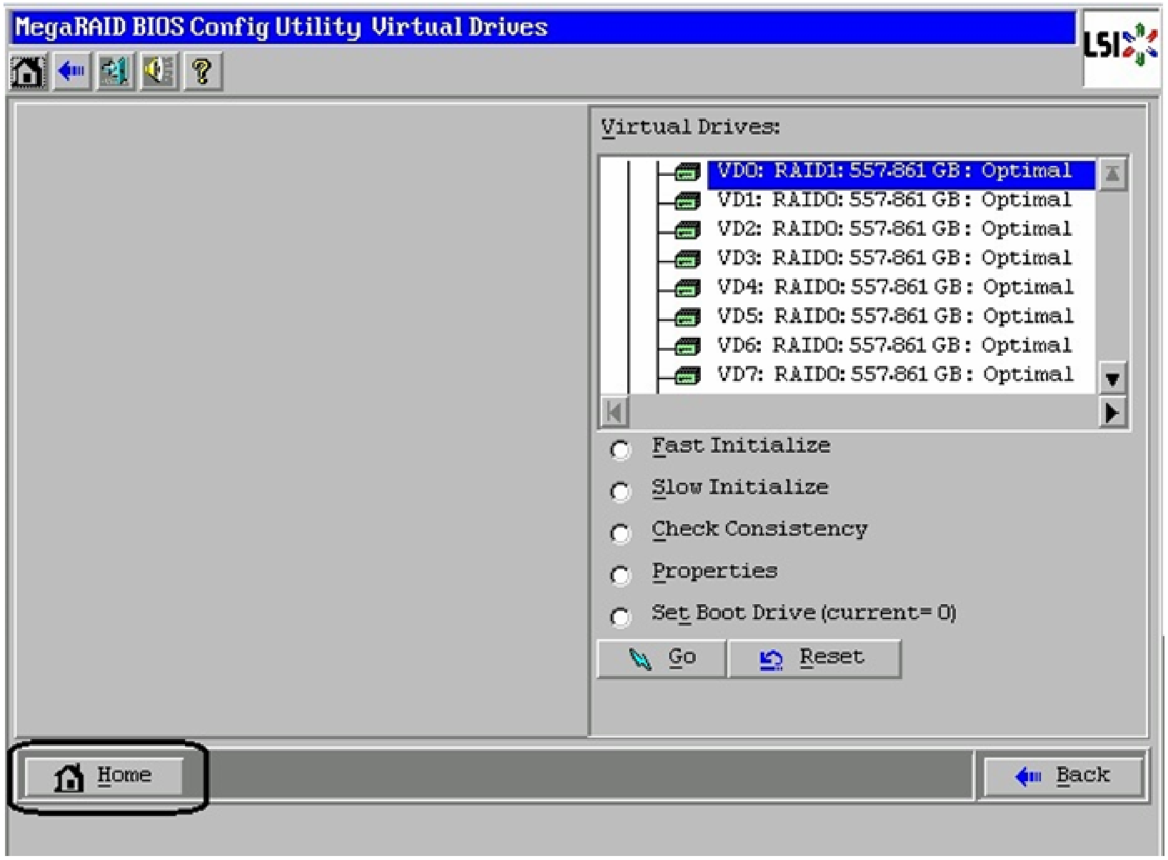
29. Click **Yes** on the warning message.
30. For the first virtual disk (VD0), click the **Set Boot Drive** radio button and click Go.

Figure 89 Settings of Virtual Drive VD0



31. Click **Home** and exit the configuration. Click **Yes** in the confirmation window.

Figure 90 Exiting the Virtual Drives Config Utility



32. Repeat these steps for the second storage node too. At this point, all the storage and compute nodes are ready for OS installation.

Install RHEL servers

Follow the following steps to install RHEL 6.4 image on the UCS servers:

1. From UCS Manager GUI, choose **Servers** tab, expand **Servers** > **Service Profiles** > **root**, and select a particular service profile. Repeat steps 1, 2, 3 and 4 of previous section to launch the Virtual Media, and boot from the CD-ROM drive mapped to the RHEL 6.4 ISO image.
2. Click **KVM** tab, RHEL 6.4 installation media will boot from the virtual disk mounted on the vMedia. Follow the steps to install RHEL on the local hard drive. You can keep all the settings at default or change them as per your requirements.
3. Once the OS is installed, reboot the machine and configure basic networking from KVM console:
   a. Edit the file **/etc/sysconfig/network-scripts/ifcfg-eth0** and add/ edit following lines:

```
IPADDR=<dotted-decimal-IP-addr>
NETMASK=<subnet-mask>
NM_CONTROLLED=no
ONBOOT=yes
```

   b. Configure default gateway by editing/ creating file **/etc/sysconfig/network-scripts/route-eth0** and adding line default via **<default-gw-ip> dev eth0**

c. Configure DNS server by editing/creating file /etc/resolv.conf and adding lines **nameserver <dns-server-1>**, **nameserver <dns-server-2>** etc. It is recommended that you add at least two DNS servers for redundancy.

d. After configuring basic networking, run **service network restart** to make those changes effective. Ping to default gateway and some external server (for example, www.cisco.com) to make sure that server is able to reach the external network.

e. On all the systems, edit /etc/hosts file and add hostname / IP address information to resolve the hostnames of all other nodes in the cluster.

f. For all the compute and storage nodes in the cluster, it is handy if passwordless SSH is configured. Choose one of the two nodes as the monitoring node / admin node of the cluster. Establish password less SSH connectivity from the monitoring node to the other node, as well as self:

```
touch /root/.ssh/authorized_keys
ssh-keygen -t dsa -f /root/.ssh/id_dsa -N ""
ssh-copy-id -i /root/.ssh/id_dsa.pub <other-node's-hostname>
ssh-copy-id -i /root/.ssh/id_dsa.pub <local-node's-hostname>
```

During the execution of "ssh-copy-id", you would be prompted for the password. After configuring the password less SSH, initiating ssh from any node to any other node should not require password.

g. If you require HTTP/ HTTPS proxy to reach external network, it is necessary to configure proxy for the Red Hat subscription manager. Edit /etc/rhsm/rhsm.conf file and provide HTTP proxy name at **proxy_hostname = <hostname>** line and TCP port number to **proxy-port = <port>** line.

Repeat these installation steps for all the servers. We are now ready to install OpenStack on the servers.

Install OpenStack Packages on the servers

First step towards installing the OpenStack components on RHEL servers is to obtain a license for the same from Red Hat, and attach all the nodes to the license using subscription-manager. Follow these steps to achieve that goal:

1. Run the following command to register a node to subscription manager. That will prompt for "Username" and "Password". Provide your Red Hat Network username and password.

Note Your RHN account must have Red Hat OpenStack entitlements to download OpenStack RPMs later.

```
# subscription-manager register
```

2. When the registration is successfully completed, you will see the following message:

   The system has been registered with id: **<IDENTIFIER>**

3. Use the **subscription-manager list –available** command to locate the pool identifier of the Red Hat Enterprise Linux subscription.

```
# subscription-manager list --available
+-------------------------------------------+
Available Subscriptions
+-------------------------------------------+
Product Name: Red Hat Enterprise Linux Server
Product Id: 69
```

```
Pool Id: <POOLID>
Quantity: 1
Service Level: None
Service Type: None
Multi-Entitlem ent: No
Expires: 01/01/2015
Machine Type: physical
```

4. The pool identifier is indicated in the Pool Id field associated with the Red Hat Enterprise Linux Server product. The identifier will be unique to your subscription. Take note of this identifier as it will be required to perform the next step. If you have multiple available subscriptions, make sure that you choose the one with OpenStack entitlement.
5. Use the **subscription-manager attach** command to the pool identifier identified in the previous step.

```
# subscription-manager attach --pool=<POOLID>
```

   Successfully attached a subscription for Red Hat Enterprise Linux Server.

6. Run the **yum repolist** command. This command ensures that the repository configuration file **/etc/yum .repos.d/redhat.repo** exists and is up to date.

```
# yum repolist
```

   Once repository metadata has been downloaded and examined, the list of repositories enabled will be displayed, along with the number of available packages, similar to following output:

```
repo id                        repo name                                   status
rhel-6-server-rpms             Red Hat Enterprise Linux 6 Server (RPM 10,623+327
...
...
repolist: 11,663
```

7. Install the **yum-utils** package. The **yum-utils** package is provided by the Red Hat Enterprise Linux subscription but provides the **yum-config-manager** utility required to complete configuration of the Red Hat OpenStack software repositories.

```
#yum install –y yum-utils
```

Note Depending on the options selected during Red Hat Enterprise Linux installation the **yum-utils** package may already be installed.

8. Use the **yum-config-manager** command to ensure that the correct software repositories are enabled. Each successful invocation of the command will display the updated repository configuration. Ensure that the repository for Red Hat OpenStack 1.0 (Essex) has been disabled.

```
# yum-config-manager --disable rhel-server-ost-6-preview-rpms

Loaded plugins: product-id
==== repo: rhel-server-ost-6-preview-rpms ====
[rhel-server-ost-6-preview-rpms]
bandwidth = 0
base_persistdir = /var/lib/yum/repos/x86_64/6Server
baseurl =
https://cdn.redhat.com /content/beta/rhel/server/6/6Server/x86_64/opensta
ck/essex/os
cache = 0
cachedir = /var/cache/yum/x86_64/6Server/rhel-server-ost-6-preview-rpms
cost = 1000
enabled = False
```

9. Ensure that the repository for Red Hat OpenStack 2.1 (Folsom) is disabled.

```
# yum-config-manager --disable rhel-server-ost-6-folsom-rpms

Loaded plugins: product-id
==== repo: rhel-server-ost-6-folsom-rpms ====
[rhel-server-ost-6-folsom-rpm s]
bandwidth = 0
base_persistdir = /var/lib/yum /repos/x86_64/6Server
baseurl =
https://cdn.redhat.com/content/beta/rhel/server/6/6Server/x86_64/opensta
ck/folsom/os
cache = 0
cachedir = /var/cache/yum/x86_64/6Server/rhel-server-ost-6-folsom-rpms
cost = 1000
enabled = False
```

10. Ensure that the repository for Red Hat OpenStack 3.0 (Grizzly) has been enabled.

```
# yum-config-manager --enable rhel-server-ost-6-3-rpms

Loaded plugins: product-id
==== repo: rhel-server-ost-6-3-rpms ====
[rhel-server-ost-6-3-rpms]
bandwidth = 0
base_persistdir = /var/lib/yum/repos/x86_64/6Server
baseurl =
https://cdn.redhat.com/content/dist/rhel/server/6/6Server/x86_64/openstack/3/os
cache = 0
cachedir = /var/cache/yum/x86_64/6Server/rhel-server-ost-6-3-rpms
cost = 1000
enabled = True
```

11. Run the **yum repolist** command. This command ensures that the repository configuration file **/etc/yum .repos.d/redhat.repo** exists and is up to date.

```
# yum repolist
```

Once repository metadata has been downloaded and examined, the list of repositories enabled will be displayed, along with the number of available packages, similar to following output:

```
repo id                         repo name                                 status
....
rhel-6-server-rpms              Red Hat Enterprise Linux 6 Server (RPM 10,623+327
rhel-server-ost-6-3-rpms        Red Hat OpenStack 3.0 (RPMs)                    690
rhel-server-ost-6-folsom-rpms   Red Hat OpenStack Folsom Preview (RPMs     10+458
repolist: 11,663
```

Note RHOS Folsom RPMs are now added to the list.

12. Install the **yum-plugin-priorities** package. The **yum-plugin-priorities** package provides a yum plug-in allowing configuration of per repository priorities.

```
# yum install -y yum-plugin-priorities
```

13. Use the **yum -config-manager** command to set the priority of the Red Hat OpenStack software repository to 1. This is the highest priority value supported by the **yum-plugin-priorities** plug-in.

```
# yum-config-manager --enable rhel -server-o st-6-3-rpms \
--setopt= "rhel-server-ost-6-3-rpms.priority=1"
Loaded plugins: product-id
==== repo: rhel-server-ost-6-3-rpms ====
```

```
[rhel-server-ost-6-3-rpms]
bandwidth = 0
base_persistdir = /var/lib/yum/repos/x86_64/6Server
baseurl =
https://cdn.redhat.com/content/dist/rhel/server/6/6Server/x86_64/openstack/3/os
cache = 0
cachedir = /var/cache/yum/x86_64/6Server/rhel-server-ost-6-3-rpms
cost = 1000
enabled = True
...
priority = 1
```

14. Run the **yum update** command and reboot to ensure that the most up to date packages, including the kernel, are installed and running.

```
# yum update -y
Loaded plugins: priorities, product-id, security, subscription-manager
This system is receiving updates from Red Hat Subscription Management.
rhel-6-server-cf-tools-1-rpms                          | 2.8 kB     00:00
rhel-6-server-rhev-agent-rpms                          | 3.1 kB     00:00
rhel-6-server-rpms                                     | 3.7 kB     00:00
...
...
(output omitted for brevity)

# reboot
```

At this point, the system is up-to-date with all the necessary OpenStack packages. In the next section, we would configure OpenStack using PackStack utility.

Run PackStack to Configure OpenStack

From the given set of Compute Nodes, identify one server as the controller node. We would install and run PackStack utility from the controller node itself. Follow these steps for the same:

1. Use the **yum** command to install the openstack-packstack package.

```
# yum install -y openstack-packstack
```

   Once installed, use “which” command to verify that the utility is now available at /usr/bin.

2. Use packstack interactively as shown below.

```
# packstack
```

3. For configuring the Public Key, each server involved in the OpenStack deployment is configured for key-based authentication. If you already have a public key that you wish to use for this, enter the path to it. If you do not, then press Enter and the utility will generate one for you and save it to **~/.ssh/id_rsa.pub**.

```
Enter the path to your ssh Public key to install on servers:
```

4. The PackStack script will prompt you to select the OpenStack services that you want to install and configure. At each prompt enter y to install the service, enter n to skip the service, or press Enter to select the default option listed in square brackets ([,]). Accept the defaults as of now.

```
Should Packstack install Glance im age service [y|n] [y] :
Should Packstack install Cinder volum e service [y|n] [y] :
Should Packstack install Nova com pute service [y|n] [y] :
Should Packstack install Quantum com pute service [y|n] [y] :
Should Packstack install Horizon dashboard [y|n] [y] :
Should Packstack install Swift object storage [y|n] [n] :
```

5. List **ntpd** servers.

```
Enter a comma separated list of NTP server(s). Leave plain if Packstack
Should not install ntpd on instances.: 10.65.255.2,10.65.255.3
```

6. Define service placement. The first few services are installed on the cloud controller.

```
Should Packstack install Nagios to monitor openstack hosts [y|n] [n] :
Enter the IP address of the MySQL server [10.65.121.207] :
Enter the password for the MySQL admin user : ********
Enter the IP address of the Qpid service [10.65.121.207] :
Enter the IP address of the Keystone server [10.65.121.207] :
Enter the IP address of the Glance server [10.65.121.207] :
Enter the IP address of the Cinder server [10.65.121.207] :
```

7. Create a 1G Cinder volume group to provide a default when Cinder is installed. This is deleted in subsequent steps when Red Hat Storage Server is used in place of the volume group.

```
Should Cinder's volumes group be created (for proof-of-concept installation)?
[y|n] [y] :
Enter Cinder's volumes group size [20G] : 1G
```

8. Place the nova-compute service on the compute nodes and all other Compute services on the cloud controller.

```
Enter the IP address of the Nova API service [10.65.121.207] :
Enter the IP address of the Nova Cert service [10.65.121.207] :
Enter the IP address of the Nova VNC proxy [10.65.121.207] :
Enter a comma separated list of IP addresses on which to install the Nova
Compute services [10.65.121.207] :
10.65.121.205, 10.65.121.206, 10.65.121.207, 10.65.121.208, 10.65.121.209,
10.65.121.210
Enter the IP address of the Nova Conductor service [10.65.121.207] :
Enter the IP address of the Nova Scheduler service [10.65.121.207] :
```

9. Accept the default CPU and RAM over-commitment ratios. The KVM hypervisor supports over committing CPUs and memory. Over committing is the process of allocating more virtualized CPUs or memory than there are physical resources on the system. CPU over commit allows under-utilized virtualized servers to run on fewer servers. You may need to adjust these values depending on your workload.

```
Enter the CPU overcommitment ratio. Set to 1.0 to disable CPU overcommitment
[16.0] :
Enter the RAM overcommitment ratio. Set to 1.0 to disable RAM overcommitment
 [1.5] :
```

10. Install the Quantum Server on the cloud controller. It should be co-resident with the other API listeners.

```
Enter the IP address of the Quantum server [10.65.121.207] : 10.65.121.207
Should Quantum use network namespaces? [y|n] [y] : y
```

11. Install the L3 and DHCP agents on the network server.

```
Enter a comma separated list of IP addresses on which to install the Quantum
L3 agent [10.65.121.207] : 10.65.121.205, 10.65.121.207
```

12. The Quantum L3 agent should use a provider network for external traffic. A provider networks maps an external network directly to a physical network. This gives tenants direct access to a public network.

```
Enter the bridge the Quantum L3 agent will use for external traffic, or
'provider' if using provider networks [br-ex] : provider
```

13. The DHCP agent should also be on the network server. This agent assigns IP addresses to instances via DHCP.

```
Enter a comma separated list of IP addresses on which to install Quantum DHCP
agent [10.65.121.207] : 10.65.121.208
```

14. Accept the default Open vSwitch plugin. Open vSwitch runs as a software–defined switch within KVM on the Compute nodes. It provides robust networking capability to the instances.

```
Enter the name of the L2 plugin to be used with Quantum [linuxbridge|
openvswitch] [openvswitch]:
```

15. Install the metadata agent on the network server. The metadata agent listens for customization requests and forwards them to "openstack-nova-api".

```
Enter a comma separated list of IP addresses on which to install the Quantum
metadata agent [10.65.121.207]: 10.65.121.206
```

16. Allocate VLAN networks for tenant networks. Tenant flows are separated internally by an internally assigned VLAN ID. GRE is not supported in this version of Red Hat Enterprise Linux OpenStack Platform.

```
Enter the type of network to allocate for tenant networks [local|vlan|gre] [local]
: vlan
```

17. Assign a VLAN range for the openvswitch plug-in. This range of tagged VLANs must be enabled on the physical switches that carry the OpenStack Network service traffic. Tenant traffic is converted to a physical VLAN ID as it traverses external bridge interface. For example, the VLAN range must be configured on the switch that carries communication between instances in the same tenant that reside on different Compute nodes.

```
Enter a comma separated list of VLAN ranges for the Quantum openvswitch
plugin: ucs-fabric:60:65
```

18. Enter the bridge mapping for the openvswitch plug-in. An openvswitch bridge acts like a virtual switch. Network interface devices connect to openvswitch bridge's ports. The ports can be configured like a physical switch's ports including VLAN configurations.

```
Enter a comma separated list of bridge mappings for the Quantum openvswitch
plugin: ucs-fabric:br-instances
```

19. Add the eth1 interface to the br-instances bridge. This maps the openvswitch bridge to the physical interface on the server.

```
Enter a comma separated list of OVS bridge:interface pairs for the Quantum
openvswitch plugin: br-instances:eth1
```

20. Install the client tools and Dashboard web interface on the controller node. HTTPS is not required for Horizon communication. This reference architecture assumes OpenStack is deployed as a private cloud. Enable HTTPS for a public cloud.

```
Enter the IP address of the client server [10.65.121.207] :
Enter the IP address of the Horizon server [10.65.121.207] :
Would you like to set up Horizon communication over https [y|n] [n] :
```

21. Enter RHN account information if the servers are not already registered. This account information is propagated to the servers by packstack. In this reference architecture the servers were registered to subscription manager after installation.

```
To subscribe each server to EPEL enter "y" [y|n] [n] :
Enter a comma separated list of URLs to any additional yum repositories to
install:
```

```
To subscribe each server to Red Hat enter a username here:
To subscribe each server to Red Hat enter your password here :
To subscribe each server to Red Hat Enterprise Linux 6 Server Beta channel
(only needed for Preview versions of RHOS) enter "y" [y|n] [n] :
To subscribe each server with RHN Satellite enter RHN Satellite server URL:
```

22. All necessary input has been provided at this point. PackStack provides you opportunity to verify and alter any information at this point. Type “yes” and enter if everything looks good.

```
Installer will be installed using the following configuration:
==============================================================
...
...
...
Proceed with the configuration listed above? (yes|no): yes
```

23. PackStack would require root authentication for each server. Provide root password for each server when prompted. PackStack would go through series of configuration to install OpenStack components on various nodes as specified in the iterative mode. At the end, you would see message similar to following output:

```
**** Installation completed successfully ******

Additional information:
* To use the command line tools you need to source the file /root/keystonerc_admin
created on 10.65.121.207
 * To use the console, browse to http://10.65.121.207/dashboard
 * Kernel package with netns support has been installed on host 10.65.121.207.
Because of the kernel update the host mentioned above requires reboot.
 * Kernel package with netns support has been installed on host 10.65.121.205.
Because of the kernel update the host mentioned above requires reboot.
 ...
<similar messages for all the hosts omitted for brevity>
...
 * The installation log file is available at:
/var/tmp/packstack/20130911-212232-u9q_AS/openstack-setup.log
```

24. Reboot all the hosts to make sure that kernel updates are effective.
25. By default, OpenStack would create an admin account, and would auto-generate a password. This would be visible in the answer file generated by PackStack with **CONFIG_KEYSTONE_ADMIN_PW** key. Copy this password.
26. Go to http://<controller-ip>/dashboard URL, and use “admin” as username and password copied from step 25 to login to Horizon Dashboard.
27. From the **Admin** tab, and in “System Panel” on left pane, click Users. This will list all existing users in the system, including “admin”. For “admin” user, click **Edit**.
28. On the pop-up window, provide a new password, confirm the password and click **Update User**. You would need to logout and log back in.

At this point of time, basic OpenStack services are up and running. Next, we would install and configure Red Hat Storage Server for storage high-availability.

Install and Configure Red Hat Storage Cluster

You can use the alternative way to obtain necessary OpenStack packages using RHN instead of using subscription-manager. By choosing appropriate client channel on RHN, you can issue “yum install glusterfs-fuse” command to install client side packages.

This section outlines high level configuration for Red Hat Storage Cluster on compute and storage nodes. For detailed description on configuring Red Hat Storage Server, see the appendices provided in this document: https://access.redhat.com/site/sites/default/files/attachments/rhelosp3_final_13_08_17.pdf

Configuration of Red Hat Storage Server architecture is divided in two parts:

1. Red Hat Storage Client configuration – this is done on all the compute nodes.
2. Red Hat Storage Server configuration – this is done on all the storage nodes.

Follow these steps for Red Hat Client configuration:

1. Red Hat Storage client components are required on all the compute nodes. Install Red Hat Storage driver and tuned using subscriber-manager and yum on all the compute nodes.

```
#yum install -y tuned
```

   You also need to install glusterfs and glusterfs-fuse, however, for Red Hat Enterprise Linux 6.4, it is not part of the subscription manager. You can manually download them from the following URLs and install it using "yum localinstall" as shown below. For Red Hat account to access these RPMs, go to:

   glusterfs-3.4.0.33rhs-1.el6_4.x86_64

   glusterfs-fuse-3.4.0.33rhs-1.el6_4.x86_64

```
#yum localinstall glusterfs*.rpm
```

   For more information on an alternate way to obtain necessary OpenStack packages using RHN instead of using subscription-manager. See,

   https://access.redhat.com/site/sites/default/files/attachments/rhelosp3_final_13_08_17.pdf

2. Create an Image Storage server mount point on all the compute nodes:

```
# mkdir -parents --verbose/var/lib/glance; chown 161.161 /var/lib/glance
```

3. Add the Image Storage server mount point to /etc/fstab on the compute nodes so it is mounted automatically at boot.

```
# cp /etc/fstab{,.orig}<TODO - ADD SECTION>
```

4. Mount /var/lib/glance:

```
# mount -a
```

5. Apply the virtual host tuning on the compute nodes with tuned-adm:

```
# tuned-adm profile virtual-host
# tuned-adm active
```

Follow these steps for Red Hat Storage Server configuration the storage nodes:

1. Configure the Image Storage server to use the Identity Service for authentication and allow direct URL access of images.

```
# openstack-config --set /etc/glance/glance-api.conf DEFAULT
# openstack-config --set /etc/nova/nova.conf DEFAULT
```

2. Configure the Block Storage server to use the Red Hat Storage share. Run these commands on the cloud controller.

```
# openstack-config --set /etc/cinder/cinder.conf DEFAULT volume_driver
cinder.volume.drivers.glusterfs.GlusterfsDriver
```

3. Configure Cinder to use the /etc/cinder/glusterfs.shares file.

```
# openstack-config --set /etc/cinder/cinder.conf DEFAULT glusterfs_shares_config
/etc/cinder/glusterfs.shares
```

4. Add the Red Hat Storage share to the file. The backupvolfile option adds the second server as a backup for the first to obtain volume information. Volume access is detailed in the volume information. Therefore access is not controlled by the server specified in the mount.

```
# echo -e "<storage-node1>:/RHOScinder -o backupvolfileserver=
<storage-node2>,selinux" > /etc/cinder/glusterfs.shares
```

5. Create the mount point.

```
# mkdir --parents --verbose /var/lib/cinder/mnt
# chown --verbose cinder.cinder /var/lib/cinder/mnt
```

6. Restart the Block Storage services on the cloud controller node.

```
# for i in $(chkconfig --list | awk ' /cinder/ { print $1 } '); do service $i
restart; done
# for i in $(chkconfig --list | awk ' /cinder/ { print $1 } '); do service $i
status; done
```

The last command must show that openstack-cinder-api, openstack-cinder-scheduler, and openstack-cinder-volume service are up and running.

7. Remove the volume group created by packstack. It is no longer necessary.

```
# vgremove cinder-volumes
```

8. Configure SELinux to allow virtual machines to use “fusefs” on all the compute nodes.

```
# setsebool -P virt_use_fusefs 1
```

9. Restart the compute nodes in order to relabel the file system with SELinux changes.

```
# touch /.autorelabel; shutdown -r now
```

At this point, Red Hat Storage Server volumes should be available for VM instances in the Horizon dashboard using cinder-api and glance-api internally.

Validating Red Hat OpenStack on UCS Platform

This section provides a list of items that should be reviewed once the solution has been configured. The goal of this section is to verify the configuration and functionality of specific aspects of the solution, and ensure that the configuration supports core availability requirements.

Post Install Checklist

The following configuration items are critical to functionality of the solution, and should be verified prior to deployment into production.

- Use Horizon Dashboard to create test Tenant, user, virtual machine image(s), network, subnet and volumes. Create virtual machine instances for the Tenant project using one of the preconfigured flavors and bootstrap the instances. You can create a router to connect all the virtual machines in different subnets and for the external network connectivity.
- Create a test virtual machine that accesses the datastore and is able to do read/write operations. Perform the virtual machine migration to a different compute node.

- During the live migration of the virtual machine, have a continuous ping to default gateway and make sure that network connectivity is maintained during and after the migration.

Verify the redundancy of the solution components

Following redundancy checks were performed at the Cisco lab to verify solution robustness. A continuous ping from VM to VM, and VM to outside network should not show significant failures (one or two ping drops might be observed at times, such as FI reboot). Also, all the data-stores must be visible and accessible from all the hosts at all the time.

1. Administratively shutdown one of the two server ports connected to the Fabric Extender A. Make sure that the connectivity is not affected. The traffic can be rebalanced by administratively enabling the shutdown port. This can be validated by clearing interface counters and showing the counters after forwarding some data from virtual machines.
2. Administratively shutdown both server ports connected to Fabric Extender A. Cisco UCS VIC fabric failover should kick-in, and compute nodes should be able to use fabric B in this case.
3. Repeat steps 1 and 2. Make sure that storage is still available from all the compute nodes. The traffic can be rebalanced by administratively enabling the shutdown port.
4. Reboot one of the two Fabric Interconnects while storage and network access from the compute nodes and VMs are going on. The switch reboot should not affect the operations of storage and network access from the VMs. On rebooting the FI, the network access load should be rebalanced across the two fabrics.
5. Fully load all the virtual machines of the solution following the N+1 HA guidelines mentioned before. Choose one of the compute nodes and migrate all the VMs running on that node to other active nodes. No VM should lose any network or storage accessibility during or after the migration. Shutdown the compute node where no VMs are running.
6. Reboot one of the two storage nodes. All the VMs must be able to have storage access during storage node reboot.

Bill of Material

Table 6 gives the list of the components used in the CVD for 250 virtual machines configuration

Table 6 List of Hardware Components Used in the CVD

Description	Part #
6 x Cisco UCS C220M3 Rack Servers	UCSC-C220-M3S
2 x Cisco UCS C240M3 Rack Servers	
CPU for C220M3 Rack Servers (2 per server)	UCS-CPU-E5-2650
CPU for C240M3 Rack Servers (2 per server)	
Memory for C220M3/ C240M3 Rack Servers (8 per server)	UCS-MR-1X162RY-A
Cisco UCS 1225 VIC Adapter (1 per server)	UCSC-PCIE-CSC-02
UCS 2232PP Fabric Extenders (2)	N2K-C2232PP-10GE
UCS 6248UP Fabric Interconnects (2)	UCS-FI-6248UP
10 Gbps SFP+ multifiber mode	SFP-10G-SR

For more information on details of the hardware components, see:

http://www.cisco.com/en/US/prod/collateral/ps10265/ps10493/C220M3_SFF_SpecSheet.pdf

Customer Configuration Data Sheet

Before you start the configuration, gather the customer-specific network and host configuration information. Table 7, Table 8, Table 9, Table 10 provide information on assembling the required network, host address, numbering, and naming information. This worksheet can also be used as a “leave behind” document for future reference.

Table 7 Common Server Information

Server Name	Purpose	Primary IP
	DNS Primary	
	DNS Secondary	
	DHCP	
	NTP	
	SMTP	
	SNMP	

Table 8 RHOS Node Information

Server Name	Purpose	Management IP	Private IP	OpenStack Role
	Compute node 1			
	Compute node 2			
	Compute node 6			
	Storage node 1			
	Storage node 2			

Table 9 Network Infrastructure Information

Description	IP	Subnet Mask	Default Gateway
UCS Manager Virtual IP address			
UCS Fabric Interconnect A address			
UCS Fabric Interconnect B address			

Table 10 VLAN Information

Name	Network Purpose	VLAN ID	Allowed Subnets
Infra	Virtual Machine Networking Management		
RHOS-Data	Data VLAN of private network		
Tenant1	VLAN for tenant1		
Tenant2	VLAN for tenant2		

References

- Cisco UCS:

  http://www.cisco.com/en/US/solutions/ns340/ns517/ns224/ns944/unified_computing.html
- Cisco UCSM 2.1 Configuration Guides:
  - CLI: http://www.cisco.com/en/US/docs/unified_computing/ucs/sw/cli/config/guide/2.1/b_UCSM_CLI_Configuration_Guide_2_1.html
  - GUI:

    http://www.cisco.com/en/US/docs/unified_computing/ucs/sw/gui/config/guide/2.1/b_UCSM_GUI_Configuration_Guide_2_1.html
- Red Hat OpenStack 3 Reference Architecture:

  https://access.redhat.com/site/sites/default/files/attachments/rhelosp3_final_13_08_17.pdf