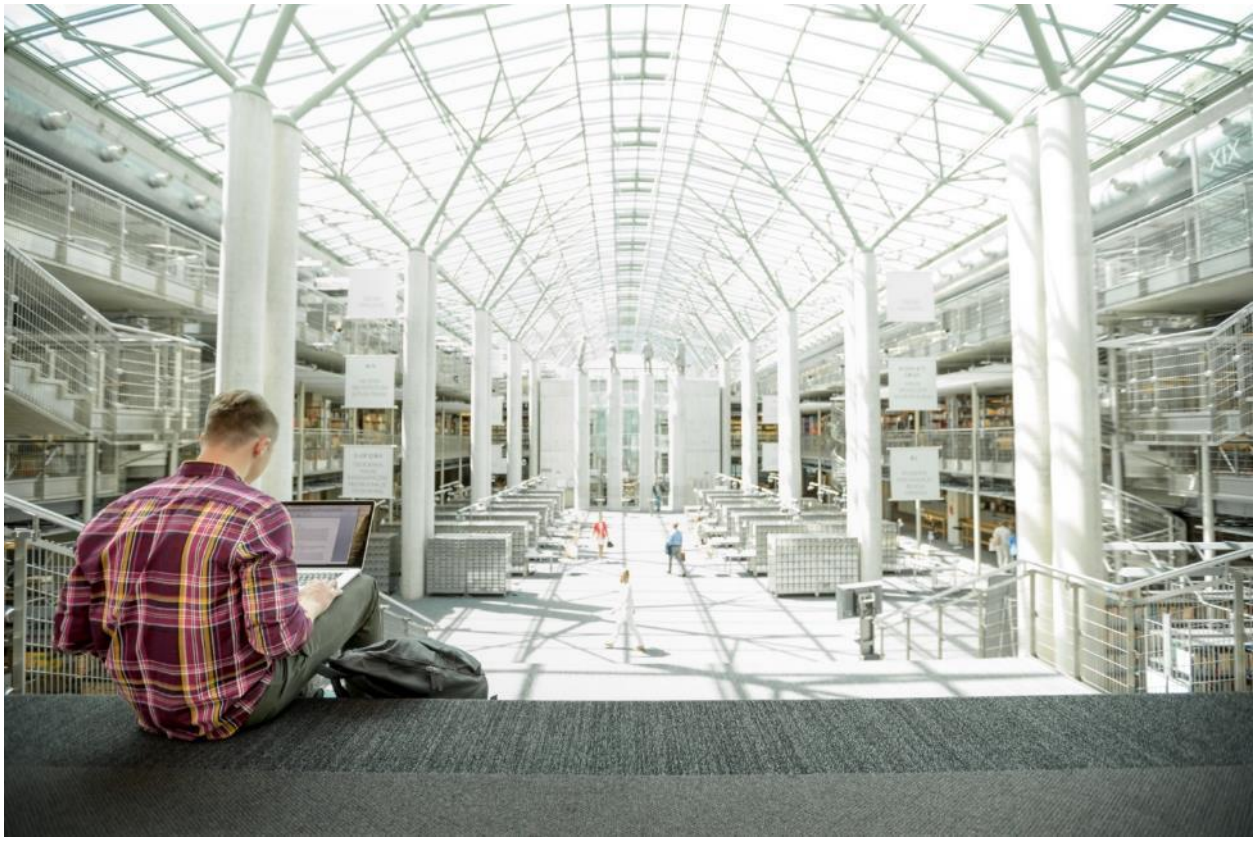


Cisco UCS C240 M5 Server with Cloudian HyperStore Object Storage

Deployment Guide for Cloudian HyperStore Object Storage Software with Cisco UCS C-Series Rack Servers

Published: January 22, 2020

CISCO VALIDATED DESIGN

About the Cisco Validated Design Program

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

http://www.cisco.com/go/designzone.

Table of Contents

Executive Summary

Cisco Validated Designs (CVDs) consist of systems and solutions that are designed, tested, and documented to facilitate and improve customer deployments. These designs incorporate a wide range of technologies and products into a portfolio of solutions that have been developed to address the business needs of our customers.

Most of the modern data centers are moving away from traditional file system type storage, to object storages. Object storage offers simple management, unlimited scalability and custom metadata for objects. With its low cost per gigabyte of storage, Object storage systems are suited for archive, backup, Life sciences, video surveillance, healthcare, multimedia, message and machine data, and so on.

Cisco and Cloudian are collaborating to offer customers a scalable object storage solution for unstructured data that integrates Cisco Unified Computing System (Cisco UCS) with Cloudian HyperStore. With the power of the Cisco UCS management framework, the solution is cost effective to deploy and manage and will enable the next-generation cloud deployments that drive business agility, lower operational costs and avoid vendor lock-in.

This validated design provides the framework for designing and deploying Cloudian HyperStore 7.1.4 on Cisco UCS C240 M5L Rack Servers. The solution is validated with both Intel Xeon scalable family CPUs (Skylake) and 2nd Generation Intel Xeon scalable family CPUs (Cascade Lake). Cisco Unified Computing System provides the compute, network, and storage access components for the Cloudian HyperStore, deployed as a single cohesive system.

The reference architecture described in this document is a realistic use case for deploying Cloudian HyperStore object storage on Cisco UCS C240 M5L Rack Server. This document provides instructions for setting Cisco UCS hardware for Cloudian SDS software, installing Red Hat Linux Operating system, Installing HyperStore Software along with Performance data collected to provide scale up and scale down guidelines, issues and workarounds evolved during installation, what needs to be done to leverage High Availability from both hardware and software for Business continuity, lessons learnt, best practices evolved while validating the solution, and so on. Performance tests were run with both Intel Xeon scalable family CPUs (Skylake) and 2nd Generation Intel Xeon scalable family CPUs (Cascade Lake).

Solution Overview

Introduction

Object storage is a highly scalable system for organizing and storing data objects. Object storage does not use a file system structure, instead it ingests data as objects with unique keys into a flat directory structure and the metadata is stored with the objects instead of hierarchical journal or tree. Search and retrieval are performed using RESTful API's, which uses HTTP verbs such as GETs and PUTs. Most of the newly generated data, about 60 to 80 percent, is unstructured today and new approaches using x86 servers are proving to be more cost effective, providing storage that can be expanded as easily as data grows. Scale-out Object storage is the newest cost-effective approach for handling large amounts of data in the Petabyte and Exabyte range.

The Cloudian HyperStore is an enterprise object storage solution that offers S3 API based storage. The solution is highly scalable and durable. The software is designed to create unbounded scale-out storage systems that accommodates Petabyte scale data from multiple applications and use-cases, including both object and file-based applications. Cloudian Hyperstore can deliver a fully enterprise-ready solution that can manage different workloads and remain flexible.

The Cisco UCS® C240 M5 Rack Server delivers industry-leading performance and expandability. The Cisco UCS C240 M5 rack server is capable of addressing a wide range of enterprise workloads, including data-intensive applications such as Cloudian HyperStore. The Cisco UCS C240 M5 servers can be deployed as standalone servers or in a Cisco UCS managed environment. Cisco UCS brings the power and automation of unified computing to the enterprise, it is an ideal platform to address capacity-optimized and performance-optimized workloads.

Audience

The audience for this document includes, but is not limited to, sales engineers, field consultants, professional services, IT managers, partner engineers, IT architects, and customers who want to take advantage of an infrastructure that is built to deliver IT efficiency and enable IT innovation. The reader of this document is expected to have the necessary training and background to install and configure Red Hat Enterprise Linux, Cisco Unified Computing System, Cisco Nexus and Cisco UCS Manager, as well as a high-level understanding of Cloudian Hyperstore Software and its 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, network and security policies of the customer installation.

Purpose of this Document

This document describes the steps required to deploy Cloudian HyperStore 7.1.4 scale out object storage on Cisco UCS platform. It discusses deployment choices and best practices using this shared infrastructure platform.

Solution Summary

Cisco and Cloudian developed a solution that meets the challenges of scale-out storage. This solution uses Cloudian HyperStore Object Storage software with Cisco UCS C-Series Rack Servers powered by Intel Xeon processors. The advantages of Cisco UCS and Cloudian HyperStore combine to deliver an object storage solution

that is simple to install, scalable, high performance, robust availability, system management, monitoring capabilities and reporting.

The configuration uses the following components for the deployment:

- Cisco Unified Computing System
  - Cisco UCS 6332 Series Fabric Interconnects
  - Cisco UCS C240 M5L Rack Servers
  - Cisco UCS Virtual Interface Card (VIC) 1385
  - Cisco C220M5 servers with VIC 1387
- Cisco Nexus 9000 C9336C-FX2 Series Switches
- Cloudian HyperStore 7.1.4
- Red Hat Enterprise Linux 7.6

Technology Overview

Cisco Unified Computing System

Cisco Unified Computing System is a state-of-the-art data center platform that unites computing, network, storage access, and virtualization into a single cohesive system.

The main components of Cisco Unified Computing System are:

- Computing - The system is based on an entirely new class of computing system that incorporates rack-mount and blade servers based on Intel Xeon Processor scalable family. 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, 40-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 - The system provides consolidated access to both SAN storage and Network Attached Storage (NAS) over the unified fabric. By unifying the storage access, Cisco Unified Computing System can access storage over Ethernet (NFS or iSCSI), Fibre Channel, and Fibre Channel over Ethernet (FCoE). This provides customers with choice for storage access and investment protection. In addition, the server administrators can pre-assign storage-access policies for system connectivity to storage resources, simplifying storage connectivity, and management for increased productivity.

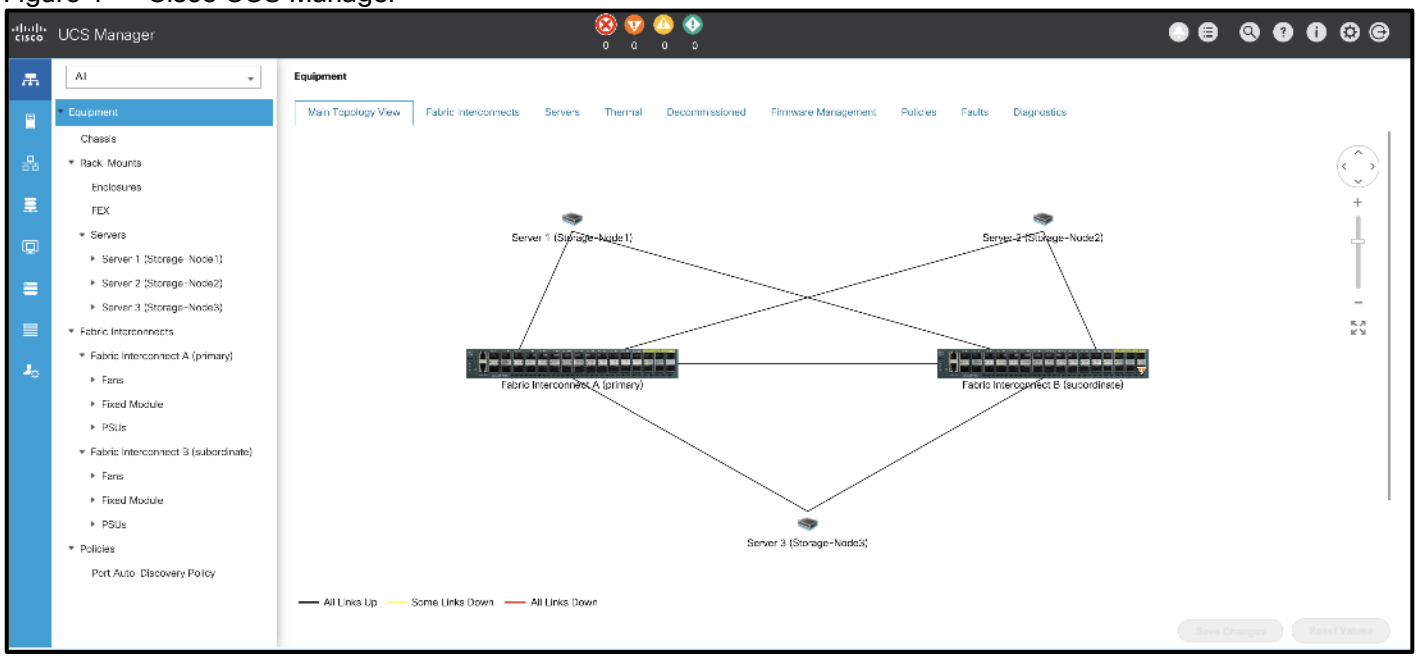
Cisco Unified Computing System is designed to deliver:

- 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 (UCSM) provides a unified, embedded management of all software and hardware components of the Cisco Unified Computing System across multiple chassis, rack servers, and thousands of virtual machines. It supports all Cisco UCS product models, including Cisco UCS B-Series Blade Servers, Cisco UCS C-Series Rack-Mount Servers, and Cisco UCS Mini, as well as the associated storage resources and networks. Cisco UCS Manager is embedded on a pair of Cisco UCS 6400, 6300 or 6200 Series Fabric Interconnects using a clustered, active-standby configuration for high availability. The manager participates in server provisioning, device discovery, inventory, configuration, diagnostics, monitoring, fault detection, auditing, and statistics collection.

Figure 1 Cisco UCS Manager



An instance of Cisco UCS Manager with all Cisco UCS components managed by it forms a Cisco UCS domain, which can include up to 160 servers. In addition to provisioning Cisco UCS resources, this infrastructure management software provides a model-based foundation for streamlining the day-to-day processes of updating, monitoring, and managing computing resources, local storage, storage connections, and network connections. By enabling better automation of processes, Cisco UCS Manager allows IT organizations to achieve greater agility and scale in their infrastructure operations while reducing complexity and risk. The manager provides flexible role and policy-based management using service profiles and templates.

Cisco UCS Manager manages Cisco UCS systems through an intuitive HTML 5 or Java user interface and a CLI. It can register with Cisco UCS Central Software in a multi-domain Cisco UCS environment, enabling centralized management of distributed systems scaling to thousands of servers. Cisco UCS Manager can be integrated with Cisco UCS Director to facilitate orchestration and to provide support for converged infrastructure and Infrastructure as a Service (IaaS). It can be integrated with Cisco Intersight which provides intelligent cloud-powered infrastructure management to securely deploy and manage infrastructure either as Software as a Service (SaaS) on Intersight.com or running on-premises with the Cisco Intersight virtual appliance.

The Cisco UCS XML API provides comprehensive access to all Cisco UCS Manager functions. The API provides Cisco UCS system visibility to higher-level systems management tools from independent software vendors (ISVs) such as VMware, Microsoft, and Splunk as well as tools from BMC, CA, HP, IBM, and others. ISVs and in-house developers can use the XML API to enhance the value of the Cisco UCS platform according to their unique requirements. Cisco UCS PowerTool for Cisco UCS Manager and the Python Software Development Kit (SDK) help automate and manage configurations within Cisco UCS Manager.

Cisco UCS 6300 Fabric Interconnects

The Cisco UCS 6300 Series Fabric Interconnects are a core part of Cisco UCS, providing both network connectivity and management capabilities for the system. The Cisco UCS 6300 Series offers line-rate, low-latency, lossless 10 and 40 Gigabit Ethernet, Fibre Channel over Ethernet (FCoE), and Fibre Channel functions.

Figure 2 Cisco UCS 6300 Fabric Interconnect

The Cisco UCS 6300 Series provides the management and communication backbone for the Cisco UCS B-Series Blade Servers, Cisco UCS 5100 Series Blade Server Chassis, and Cisco UCS C-Series Rack Servers managed by Cisco UCS. All servers attached to the fabric interconnects become part of a single, highly available management domain. In addition, by supporting unified fabric, the Cisco UCS 6300 Series provides both LAN and SAN connectivity for all servers within its domain.

From a networking perspective, the Cisco UCS 6300 Series uses a cut-through architecture, supporting deterministic, low-latency, line-rate 10 and 40 Gigabit Ethernet ports, switching capacity of 2.56 terabits per second (Tbps), and 320 Gbps of bandwidth per chassis, independent of packet size and enabled services. The product family supports Cisco® low-latency, lossless 10 and 40 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 the server through the fabric interconnect. Significant TCO savings can be achieved with an FCoE optimized server design in which network interface cards (NICs), host bus adapters (HBAs), cables, and switches can be consolidated.

The Cisco UCS 6332 32-Port Fabric Interconnect is a 1-rack-unit (1RU) Gigabit Ethernet, and FCoE switch offering up to 2.56 Tbps throughput and up to 32 ports. The switch has 32 fixed 40-Gbps Ethernet and FCoE ports.

Both the Cisco UCS 6332UP 32-Port Fabric Interconnect and the Cisco UCS 6332 16-UP 40-Port Fabric Interconnect have ports that can be configured for the breakout feature that supports connectivity between 40 Gigabit Ethernet ports and 10 Gigabit Ethernet ports. This feature provides backward compatibility to existing hardware that supports 10 Gigabit Ethernet. A 40 Gigabit Ethernet port can be used as four 10 Gigabit Ethernet ports. Using a 40 Gigabit Ethernet SFP, these ports on a Cisco UCS 6300 Series Fabric Interconnect can connect to another fabric interconnect that has four 10 Gigabit Ethernet SFPs. The breakout feature can be configured on ports 1 to 12 and ports 15 to 26 on the Cisco UCS 6332UP fabric interconnect. Ports 17 to 34 on the Cisco UCS 6332 16-UP fabric interconnect support the breakout feature.

Cisco UCS C9336C-FX2 Nexus Switches

The Cisco Nexus 9000 Series Switches include both modular and fixed-port switches that are designed to overcome these challenges with a flexible, agile, low-cost, application-centric infrastructure.

Figure 3 Cisco Nexus C9336C-FX2 Switch

The Cisco Nexus 9300 platform consists of fixed-port switches designed for top-of-rack (ToR) and middle-of-row (MoR) deployment in data centers that support enterprise applications, service provider hosting, and cloud computing environments. They are Layer 2 and 3 nonblocking 10 and 40 Gigabit Ethernet switches with up to 2.56 terabits per second (Tbps) of internal bandwidth.

The Cisco Nexus C9336C-FX2 Switch is a 1-rack-unit (1RU) switch that supports 7.2 Tbps of bandwidth and over 2.8 billion packets per second (bpps) across thirty-six 10/25/40/100 -Gbps Enhanced QSFP28 ports

All the Cisco Nexus 9300 platform switches use dual- core 2.5-GHz x86 CPUs with 64-GB solid-state disk (SSD) drives and 16 GB of memory for enhanced network performance.

With the Cisco Nexus 9000 Series, organizations can quickly and easily upgrade existing data centers to carry 40 Gigabit Ethernet to the aggregation layer or to the spine (in a leaf-and-spine configuration) through advanced and cost-effective optics that enable the use of existing 10 Gigabit Ethernet fiber (a pair of multimode fiber strands).

Cisco provides two modes of operation for the Cisco Nexus 9000 Series. Organizations can use Cisco NX-OS Software to deploy the Cisco Nexus 9000 Series in standard Cisco Nexus switch environments. Organizations also can use a hardware infrastructure that is ready to support Cisco Application Centric Infrastructure (Cisco ACI) to take full advantage of an automated, policy-based, systems management approach.

Cisco UCS C240 M5 Rack Server

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

In response to ever-increasing computing and data-intensive real-time workloads, the enterprise-class Cisco UCS C240 M5 server extends the capabilities of the Cisco UCS portfolio in a 2RU form factor. It incorporates the Intel® Xeon® Scalable processors, supporting up to 20 percent more cores per socket, twice the memory capacity, and five times more.

Non-Volatile Memory Express (NVMe) PCI Express (PCIe) Solid-State Disks (SSDs) compared to the previous generation of servers. These improvements deliver significant performance and efficiency gains that will improve your application performance. The Cisco UCS C240 M5 delivers outstanding levels of storage expandability with exceptional performance, with:

- The latest second-generation Intel Xeon Scalable CPUs, with up to 28 cores per socket
- Supports the first-generation Intel Xeon Scalable CPU, with up to 28 cores per socket
- Support for the Intel Optane DC Persistent Memory (128G, 256G, 512G)[1]
- Up to 24 DDR4 DIMMs for improved performance including higher density DDR4 DIMMs
- Up to 26 hot-swappable Small-Form-Factor (SFF) 2.5-inch drives, including 2 rear hot-swappable SFF drives (up to 10 support NVMe PCIe SSDs on the NVMe-optimized chassis version), or 12 Large-Form-Factor (LFF) 3.5-inch drives plus 2 rear hot-swappable SFF drives
- Support for 12-Gbps SAS modular RAID controller in a dedicated slot, leaving the remaining PCIe Generation 3.0 slots available for other expansion cards
- Modular LAN-On-Motherboard (mLOM) slot that can be used to install a Cisco UCS Virtual Interface Card (VIC) without consuming a PCIe slot, supporting dual 10- or 40-Gbps network connectivity
- Dual embedded Intel x550 10GBASE-T LAN-On-Motherboard (LOM) ports
- Modular M.2 or Secure Digital (SD) cards that can be used for boot

Figure 4 Cisco UCS C240 M5L Front

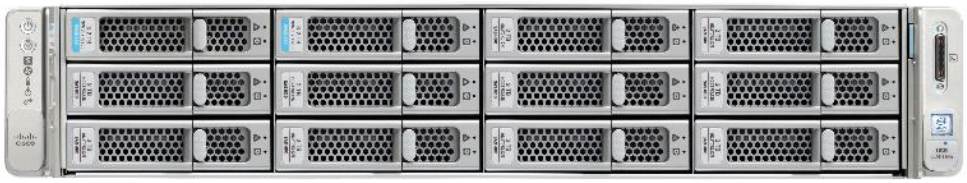


Figure 5 Cisco UCS C240 M5 Internals

Cisco UCS C240 M5 servers can be deployed as standalone servers or in a Cisco UCS managed environment. When used in combination with Cisco UCS Manager, the Cisco UCS C240 M5 brings the power and automation of unified computing to enterprise applications, including Cisco® SingleConnect technology, drastically reducing switching and cabling requirements.

Cisco UCS Manager uses service profiles, templates, and policy-based management to enable rapid deployment and help ensure deployment consistency. If also enables end-to-end server visibility, management, and control in both virtualized and bare-metal environments.

The Cisco Integrated Management Controller (IMC) delivers comprehensive out-of-band server management with support for many industry standards, including:

- Redfish Version 1.01 (v1.01)
- Intelligent Platform Management Interface (IPMI) v2.0
- Simple Network Management Protocol (SNMP) v2 and v3
- Syslog
- Simple Mail Transfer Protocol (SMTP)
- Key Management Interoperability Protocol (KMIP)
- HTML5 GUI
- HTML5 virtual Keyboard, Video, and Mouse (vKVM)
- Command-Line Interface (CLI)
- XML API

Management Software Development Kits (SDKs) and DevOps integrations exist for Python, Microsoft PowerShell, Ansible, Puppet, Chef, and more. The Cisco UCS C240 M5 is Cisco Intersight™ ready. Cisco Intersight is a new cloud-based management platform that uses analytics to deliver proactive automation and support. By combining intelligence with automated actions, you can reduce costs dramatically and resolve issues more quickly.

The Cisco UCS C240 M5 Rack Server is well-suited for a wide range of enterprise workloads, including:

- Big data and analytics
- Collaboration
- Small and medium-sized business databases
- Virtualization and consolidation
- Storage servers
- High-performance appliances

2nd Generation Intel Xeon Scalable Processors

Intel Xeon Scalable processors provide a foundation for powerful data center platforms with an evolutionary leap in agility and scalability. Disruptive by design, this innovative processor family supports new levels of platform convergence and capabilities across computing, storage, memory, network, and security resources.

Cascade Lake (CLX-SP) is the code name for the next-generation Intel Xeon Scalable processor family that is supported on the Purley platform serving as the successor to Skylake SP. These chips support up to eight-way multiprocessing, use up to 28 cores, incorporate a new AVX512 x86 extension for neural-network and deep-learning workloads, and introduce persistent memory support. Cascade Lake SP–based chips are manufactured in an enhanced 14-nanometer (14-nm++) process and use the Lewisburg chip set. Cascade Lake SP–based models are branded as the Intel Xeon Bronze, Silver, Gold, and Platinum processor families.

Cascade Lake is set to run at higher frequencies than the current and older generations of the Intel Xeon Scalable products. Additionally, it supports Intel OptaneTM DC Persistent Memory. The chip is a derivative of Intel's existing 14-nm technology (first released in 2016 in server processors). It offers 26 percent performance improvement compared to the earlier technology while maintaining the same level of power consumption.

The new Cascade Lake processors incorporate a performance-optimized multichip package to deliver up to 28 cores per CPU and up to 6 DDR4 memory channels per socket. They also support Intel Optane DC Persistent Memory and are especially valuable for in-memory computing SAP workloads.

- Cascade Lake delivers additional features, capabilities, and performance to our customers:
  - Compatibility with the Purley platform through a six-channel drop-in CPU
  - Improved core frequency through speed-path and processing improvements
  - Support for DDR4-2933 with two DIMMs per channel (DPCs) on selected SKUs and 16-Gbps devices
  - Scheduler improvements to reduce load latency
  - Additional capabilities such as Intel Optane DC Persistent Memory Module (DCPMM) support
  - Intel Deep Learning Boost with Vector Neural Network Instructions

Cisco UCS C220 M5 Rack-Mount Server

The Cisco UCS C220 M5 Rack-Mount Server is among the most versatile general-purpose enterprise infrastructure and application servers in the industry. It is a high-density 2-socket rack server that delivers industry-leading performance and efficiency for a wide range of workloads, including virtualization, collaboration, and bare-metal applications. The Cisco UCS C-Series Rack-Mount Servers can be deployed as standalone servers or as part of Cisco UCS to take advantage of Cisco's standards-based unified computing innovations that help reduce customers' TCO and increase their business agility.

The Cisco UCS C220 M5 server extends the capabilities of the Cisco UCS portfolio in a 1-Rack-Unit (1RU) form factor. It incorporates the Intel® Xeon Scalable processors, supporting up to 20 percent more cores per socket, twice the memory capacity, 20 percent greater storage density, and five times more PCIe NVMe Solid-State Disks (SSDs) compared to the previous generation of servers. These improvements deliver significant performance and efficiency gains that will improve your application performance.

Figure 6 Cisco UCS C220M5 Rack-Mount Server

The Cisco UCS C220 M5 SFF server extends the capabilities of the Cisco Unified Computing System portfolio in a 1U form factor with the addition of the Intel Xeon Processor Scalable Family, 24 DIMM slots for 2666MHz DIMMs and capacity points up to 128GB, two 2 PCI Express (PCIe) 3.0 slots, and up to 10 SAS/SATA hard disk drives (HDDs) or solid state drives (SSDs). The Cisco UCS C220 M5 SFF server also includes one dedicated internal slot for a 12G SAS storage controller card.

The Cisco UCS C220 M5 server included one dedicated internal modular LAN on motherboard (mLOM) slot for installation of a Cisco Virtual Interface Card (VIC) or third-party network interface card (NIC), without consuming a PCI slot, in addition to 2 x 10Gbase-T Intel x550 embedded (on the motherboard) LOM ports.

The Cisco UCS C220 M5 server can be used standalone, or as part of the Cisco Unified Computing System, which unifies computing, networking, management, virtualization, and storage access into a single integrated architecture enabling end-to-end server visibility, management, and control in both bare metal and virtualized environments.

Cisco UCS Virtual Interface Card 1385

The Cisco UCS Virtual Interface Card (VIC) 1385 is a Cisco® innovation. It provides a policy-based, stateless, agile server infrastructure for your data center. This dual-port Enhanced Quad Small Form-Factor Pluggable (QSFP) half-height PCI Express (PCIe) card is designed exclusively for Cisco UCS C-Series Rack Servers. The card supports 40 Gigabit Ethernet and Fibre Channel over Ethernet (FCoE). It incorporates Cisco's next-generation converged network adapter (CNA) technology and offers a comprehensive feature set, providing investment protection for future feature software releases. The card can present more than 256 PCIe standards-compliant interfaces to the host, and these can be dynamically configured as either network interface cards (NICs) or host bus adapters (HBAs). In addition, the VIC supports Cisco DatacenterVirtual Machine Fabric Extender (VM-FEX) technology. This technology extends the Cisco UCS Fabric Interconnect ports to virtual machines, simplifying server virtualization deployment.

Figure 7 Cisco UCS VIC 1385

The Cisco UCS VIC 1385 provides the following features and benefits:

- Stateless and agile platform: The personality of the card is determined dynamically at boot time using the service profile associated with the server. The number, type (NIC or HBA), identity (MAC address and World Wide Name [WWN]), failover policy, bandwidth, and quality-of-service (QoS) policies of the PCIe interfaces are all determined using the service profile. The capability to define, create, and use interfaces on demand provides a stateless and agile server infrastructure.
- Network interface virtualization: Each PCIe interface created on the VIC is associated with an interface on the Cisco UCS fabric interconnect, providing complete network separation for each virtual cable between a PCIe device on the VIC and the interface on the fabric interconnect.

VIC 1385 has a hardware classification engine. This provides support for advanced data center requirements including stateless network offloads for NVGRE and VXLAN (VMware only), low-latency features for usNIC and RDMA, and performance optimization applications such as VMQ, DPDK, and Cisco NetFlow. The Cisco UCS VIC 1385 provides high network performance and low latency for the most demanding applications:

- Big data, high-performance computing (HPC), and high-performance trading (HPT)
- Large-scale virtual machine deployments
- High-bandwidth storage targets and archives

When the VIC 1385 is used in combination with Cisco Nexus® 3000 Series Switches, big data and financial trading applications benefit from high bandwidth and low latency. When the VIC is connected to Cisco Nexus 5000 Series Switches, pools of virtual hosts scale with greater speed and agility. The Cisco Nexus 6004 Switch provides native 40-Gbps FCoE connectivity from the VIC to both Ethernet and Fibre Channel targets.

Red Hat Enterprise Linux 7.6

Red Hat® Enterprise Linux is a high-performing operating system that has delivered outstanding value to IT environments for more than a decade. More than 90 percent of Fortune Global 500 companies use Red Hat products and solutions including Red Hat Enterprise Linux. As the worlds most trusted IT platform, Red Hat Enterprise Linux has been deployed in mission-critical applications at global stock exchanges, financial institutions, leading telcos, and animation studios. It also powers the websites of some of the most recognizable global retail brands.

Red Hat Enterprise Linux:

- Delivers high-performance, reliability, and security
- Is certified by the leading hardware and software vendors

- Scales from workstations, to servers, to mainframes
- Provides a consistent application environment across physical, virtual, and cloud deployments

Designed to help organizations make a seamless transition to emerging datacenter models that include virtualization and cloud computing, Red Hat Enterprise Linux includes support for major hardware architectures, hypervisors, and cloud providers, making deployments across physical and different virtual environments predictable and secure. Enhanced tools and new capabilities in this release enable administrators to tailor the application environment to efficiently monitor and manage compute resources and security.

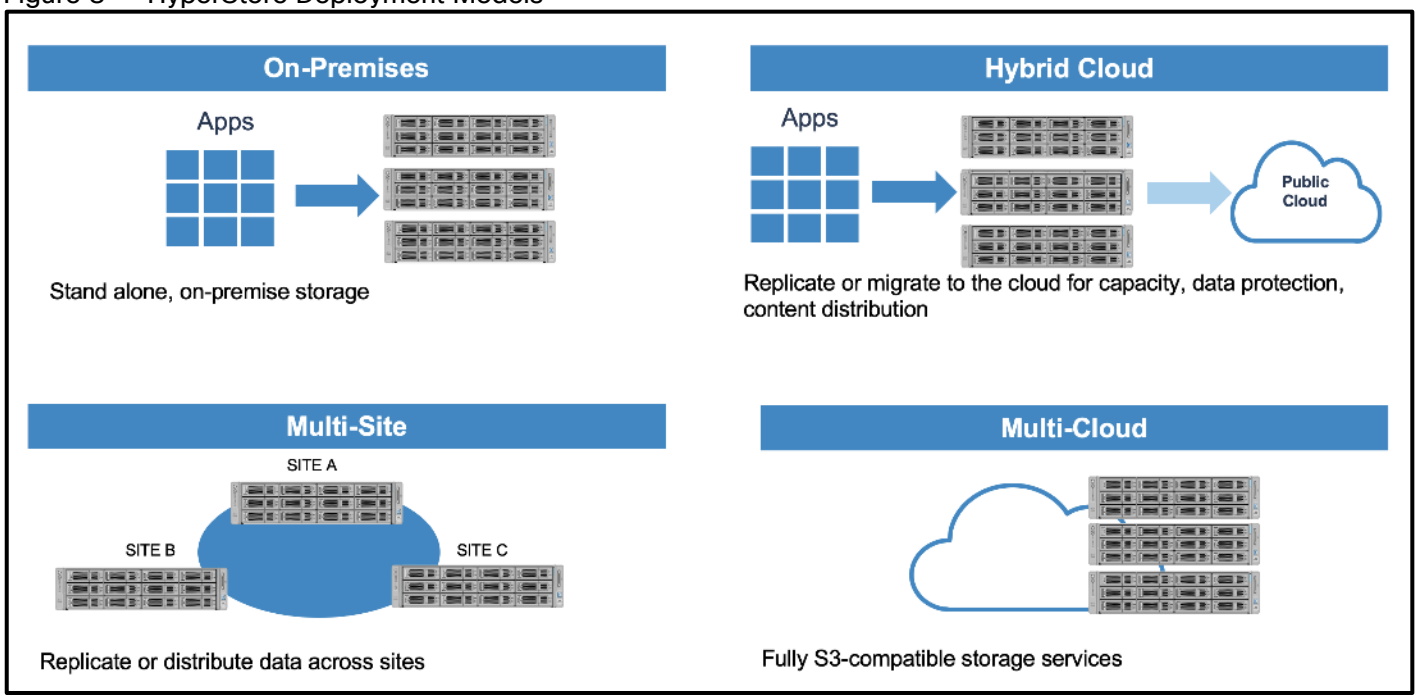
Cloudian HyperStore

Cloudian HyperStore enables data centers to provide highly cost-effective on-premise unstructured data storage repositories. Cloudian HyperStore is built on standard hardware that spans across the enterprise as well as into public cloud environments. Cloudian HyperStore is available as a stand-alone software. It easily scales to limitless capacities and offers multi-data center storage. HyperStore also has fully automated data tiering to all major public clouds, including AWS, Azure and Google Cloud Platform. It fully supports S3 applications and has flexible security options.

Cloudian HyperStore is a scale-out object storage system designed to manage massive amounts of data. It is an SDS solution that runs on the Cisco UCS platform allowing cost savings for datacenter storage while providing extreme availability and reliability.

HyperStore deployment models include on-premises storage, distributed storage, storage-as-a-service, or even other combinations (Figure 8).

Figure 8 HyperStore Deployment Models



Cloudian Object Storage

Cloudian delivers an object storage solution that provides petabyte-scalability while keeping it simple to manage. Deploy as on-premises storage or configure a hybrid cloud and automatically tier data to the public cloud.

You can view system health, manage users and groups, and automate tasks with Cloudian's web-based UI and REST API. Manage workload with a self-service portal that lets users administer their own storage. Powerful QoS capabilities help you ensure SLAs.

Cloudian makes it easy to get started. Begin with the cluster size that fits the needs and expand on demand. In Cloudian's modular, shared-nothing architecture, every node is identical, allowing the solution to grow from a few nodes to a few hundred without disruption. Performance scales linearly, too.

Cloudian HyperStore offers a 100 percent native S3 API, proven to deliver the highest interoperability in its class. Guaranteed compatible with S3-enabled applications, Cloudian gives you investment protection and peace of mind.

Get the benefits of both on-premises and cloud storage in a single management environment. Run S3-enabled applications within data center with Cloudian S3 scale-out storage. Use policies you define to automatically tier data to the public cloud. It's simple to manage and limitlessly scalable.

Get all the benefits of using the Cisco UCS platform while managing data through a single pane of glass.

Cloudian HyperStore Design

Cloudian HyperStore is an Amazon S3-compliant multi-tenant object storage system. The system utilizes a non-SQL (NoSQL) storage layer for maximum flexibility and scalability. The Cloudian HyperStore system enables any service provider or enterprise to deploy an S3-compliant multi-tenant storage cloud.

The Cloudian HyperStore system is designed specifically to meet the demands of high volume, multi-tenant data storage:

- Amazon S3 API compliance. The Cloudian HyperStore system 100 percent compatible with Amazon S3's HTTP REST API. Customer's existing HTTP S3 applications will work with the Cloudian HyperStore service, and existing S3 development tools and libraries can be used for building Cloudian HyperStore client applications.
- Secure multi-tenancy. The Cloudian HyperStore system provides the capability to have multiple users securely reside on a single, shared infrastructure. Data for each user is logically separated from other users' data and cannot be accessed by any other user unless access permission is explicitly granted.
- Group support. An enterprise or work group can share a single Cloudian HyperStore account. Each group member can have dedicated storage space, and the group can be managed by a designated group administrator.
- Quality of Service (QoS) controls. Cloudian HyperStore system administrators can set storage quotas and usage rate limits on a per-group and per-user basis. Group administrators can set quotas and rate controls for individual members of the group.
- Access control rights. Read and write access controls are supported at per-bucket and per-object granularity. Objects can also be exposed via public URLs for regular web access, subject to configurable expiration periods.
- Reporting and billing. The Cloudian HyperStore system supports usage reporting on a system-wide, group-wide, or individual user basis. Billing of groups or users can be based on storage quotas and usage rates (such as bytes in and bytes out).
- Horizontal scalability. Running on standard off-the-shelf hardware, a Cloudian HyperStore system can scale up to thousands of nodes across multiple datacenters, supporting millions of users and hundreds of petabytes of data. New nodes can be added without service interruption.
- High availability. The Cloudian HyperStore system has a fully distributed, peer-to-peer architecture, with no single point of failure. The system is resilient to network and node failures with no data loss due to the automatic replication and recovery processes inherent to the architecture. A Cloudian HyperStore geocluster can be deployed across multiple datacenters to provide redundancy and resilience in the event of a data center scale disaster.

Cloudian HyperStore Architecture

The Cloudian HyperStore is a fully distributed architecture that provides no single point of failure, data protection options (replication or erasure coding), data recovery upon a node failure, dynamic re-balancing on node addition, multi-data center and multi-region support. Figure 9 illustrates the high-level system view.

Figure 9 High-level System View

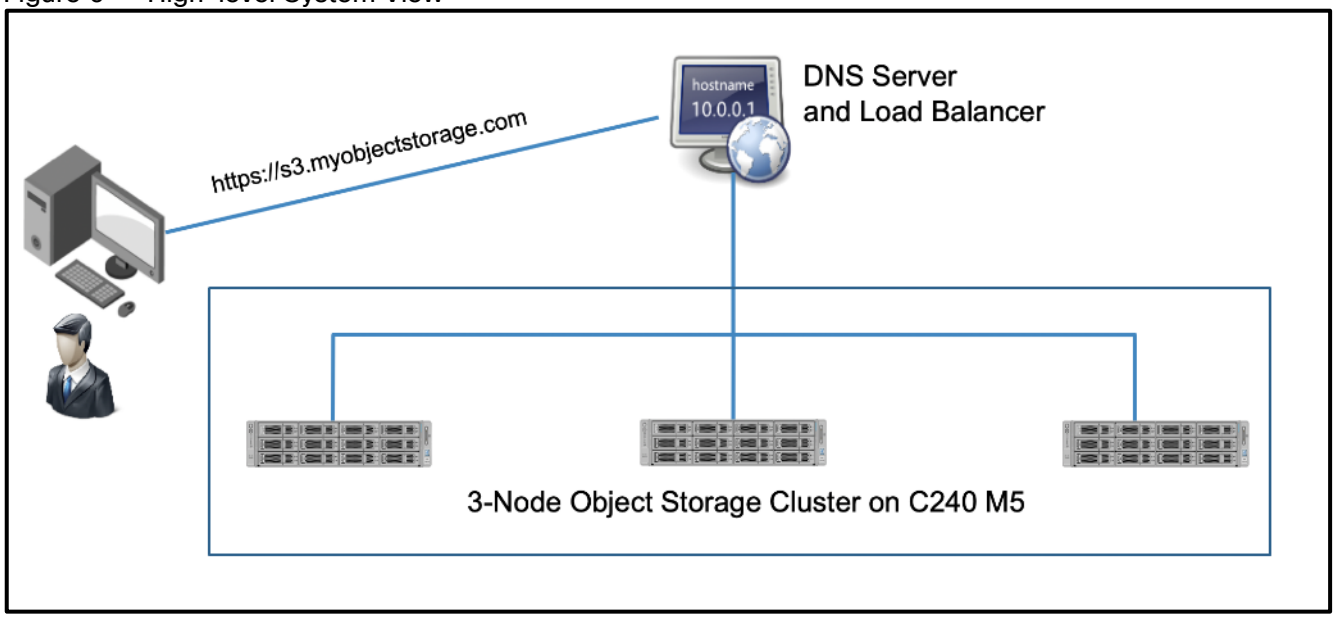
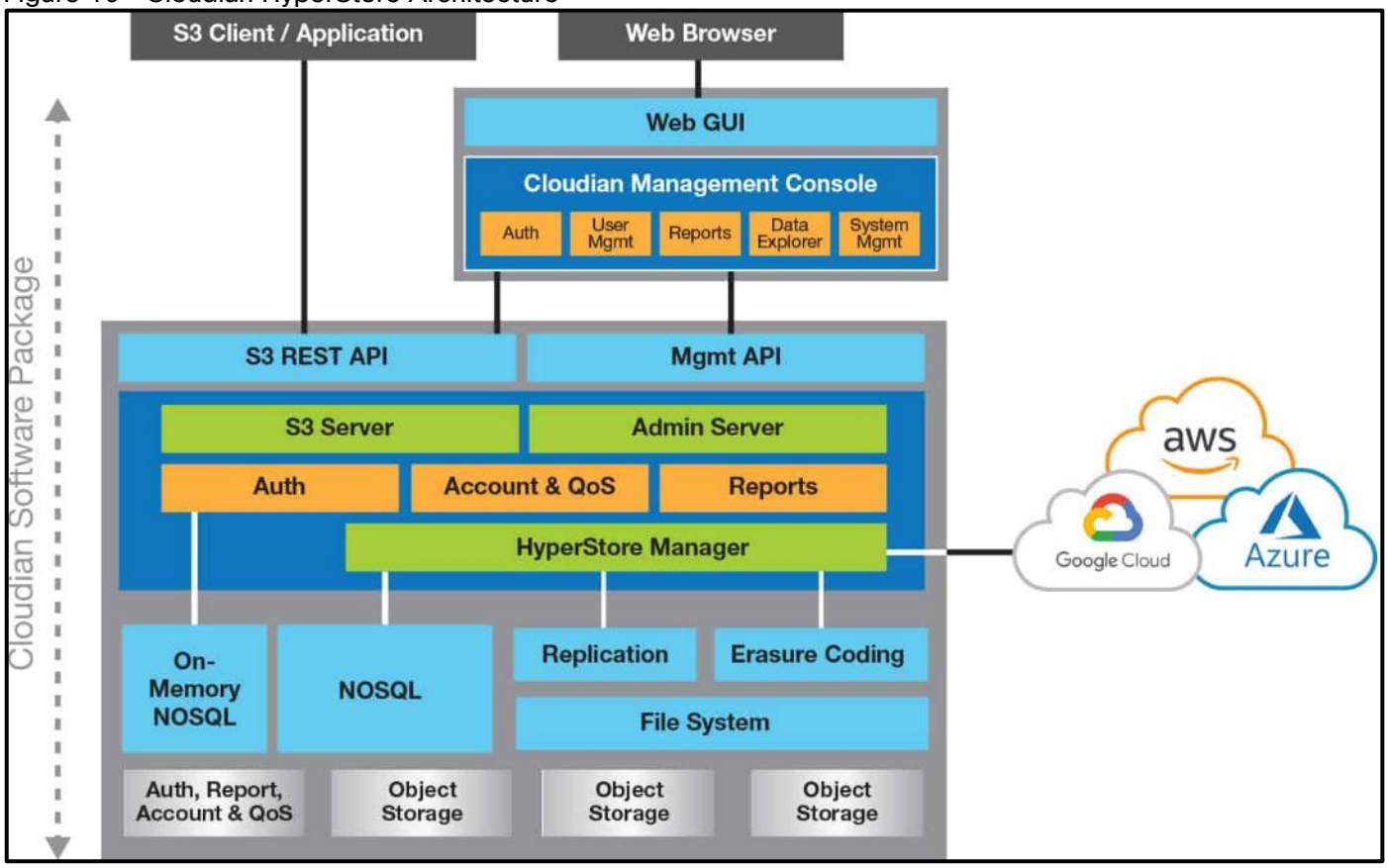


Figure 10 illustrates all service components that comprise a Cloudian HyperStore system.

Figure 10 Cloudian HyperStore Architecture



Cloudian Management Console

The Cloudian Management Console (CMC) is a web-based user interface for Cloudian HyperStore system administrators, group administrators, and end users. The functionality available through the CMC depends on the user type associated with a user's login ID (system administrative, group administrative, or regular user).

As a Cloudian HyperStore system administrator, you can use the CMC to perform the following tasks:

- Provisioning groups and users
- Managing quality of service (QoS) controls
- Creating and managing rating plans
- Generating usage data reports
- Generating bills
- Viewing and managing users' stored data objects
- Setting access control rights on users' buckets and stored objects

Group administrators can perform a limited range of administrative tasks pertaining to their own group. Regular users can perform S3 operations such as uploading and downloading S3 objects. The CMC acts as a client to the Administrative Service and the S3 Service.

Figure 11 Cloudian Management Console

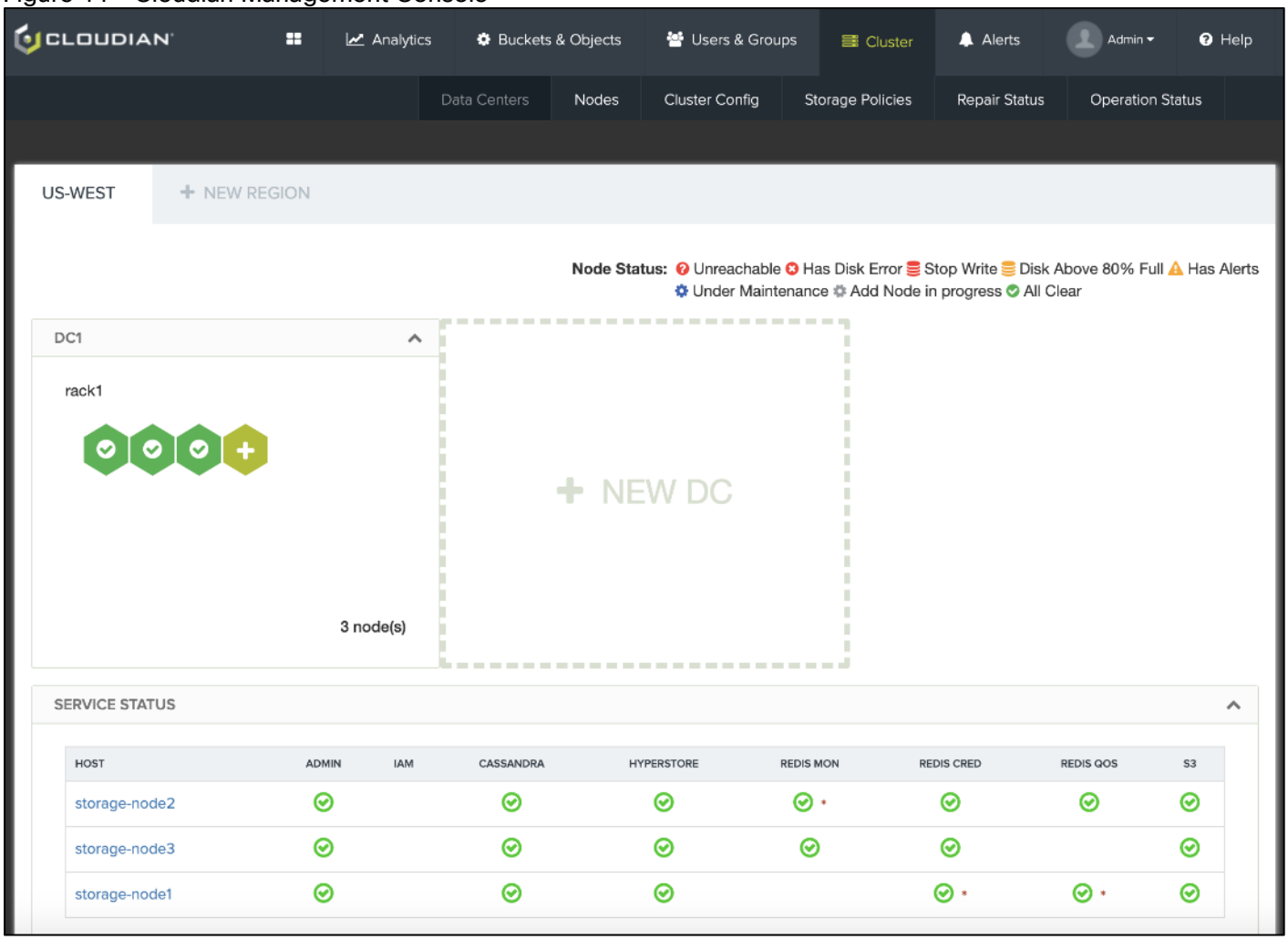


Figure 12 Snapshot of a Node from Cloudian Management Console

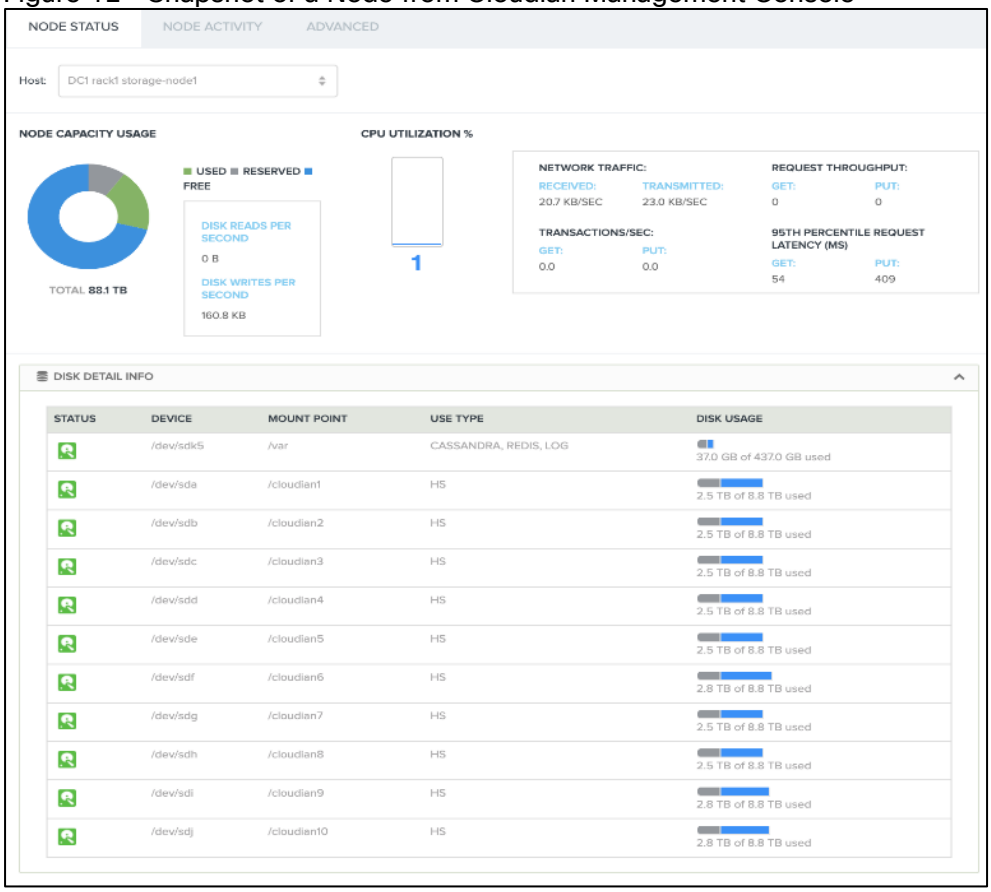


Figure 13 Snapshot of Node Activity from Cloudian Management Console

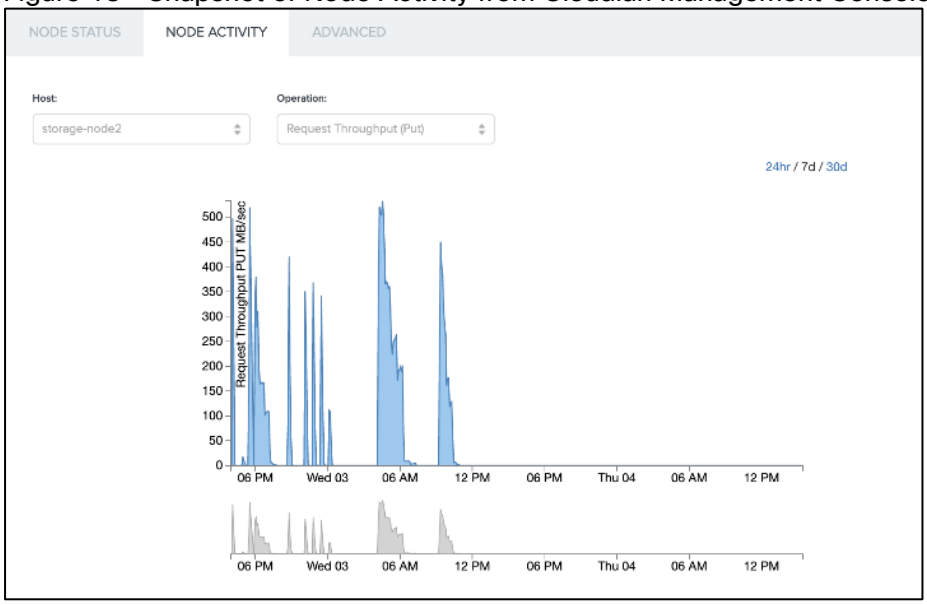


Figure 14 Cluster Usage Details from Cloudian Management Console

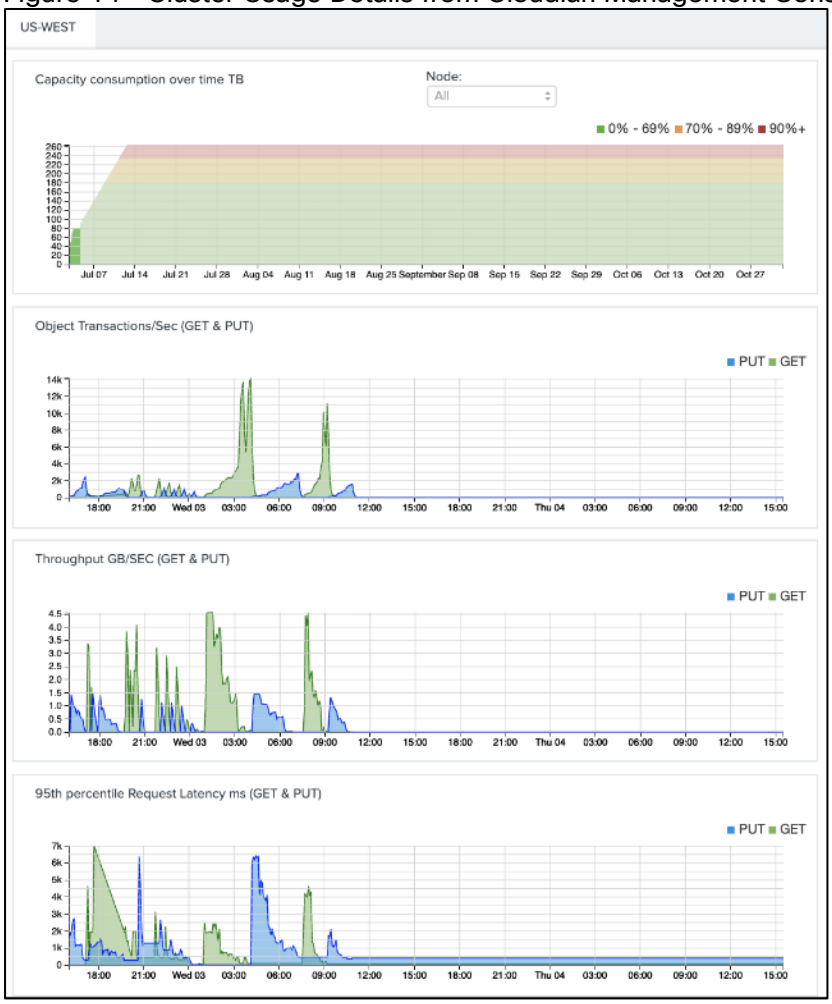
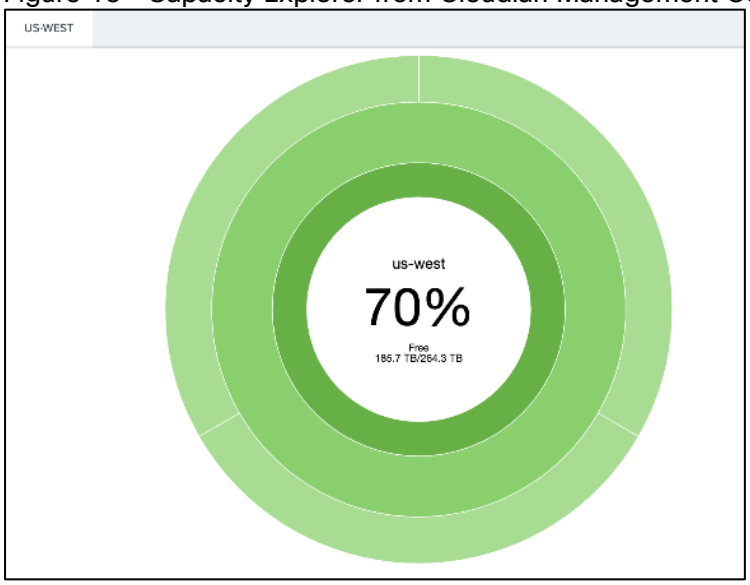


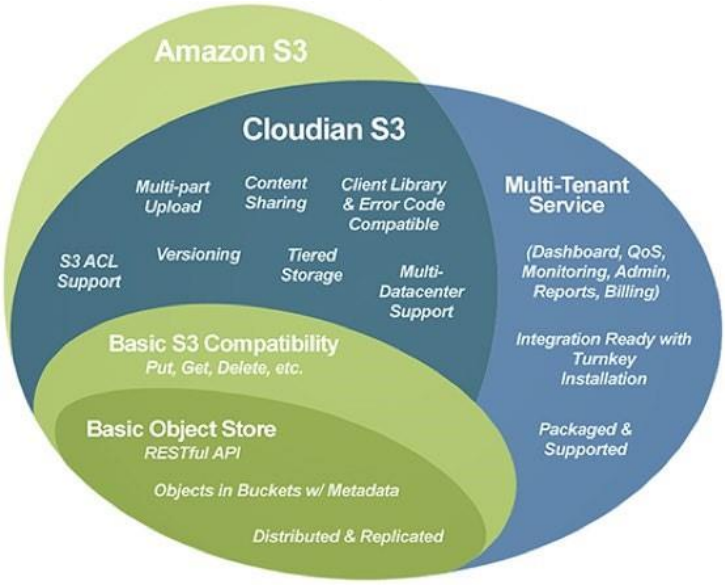
Figure 15 Capacity Explorer from Cloudian Management Console



S3 Compatible

With Amazon setting the cloud storage standard making it the largest object storage environment, and Amazon S3 API becoming the defacto standard for developers writing storage applications for cloud, it is imperative every Cloud, hybrid storage solution is S3 compliant. Cloudian HyperStore, in addition to being S3 compliant, also offers the flexibility to be on-premises object storage as well as hybrid tier to Amazon and Google clouds.

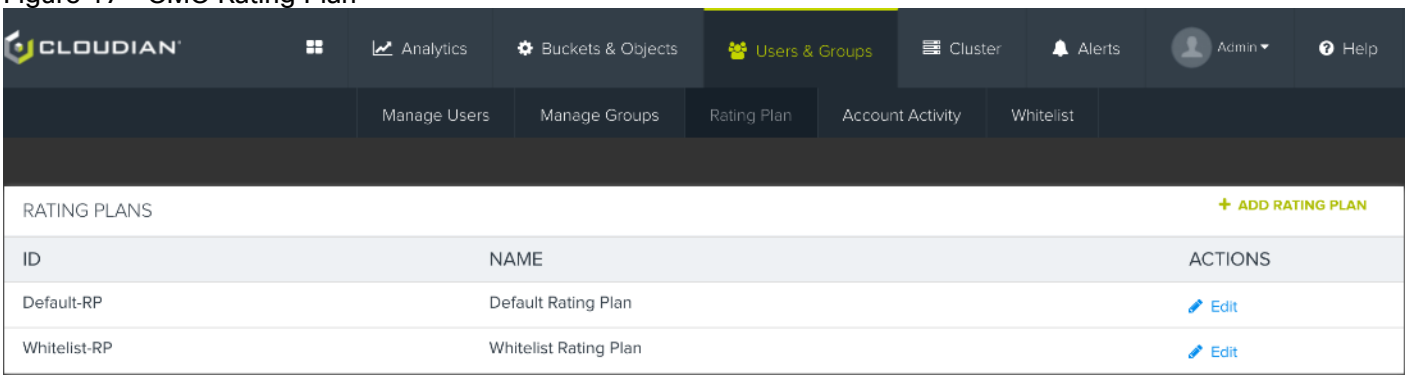
Figure 16 Cloudian S3 Compatibility Overview



Integrated Billing, Management, and Monitoring

The HyperStore system maintains comprehensive service usage data for each group and each user in the system. This usage data, which is protected by replication, serves as the foundation for HyperStore service billing functionality. The system allows the creation of rating plans that categorize the types of service usage for single users or groups for a selected service period. The CMC has a function to display a single user's bill report in a browser; HyperStore Admin API can be used to generate user or group billing data that can be ingested a third-party billing application. Cloudian HyperStore also allows for the special treatment of designated source IP addresses, so that the billing mechanism does not apply any data transfer charges for data coming from or going to these whitelisted domains.

Figure 17 CMC Rating Plan



Infinite Scalability on Demand

Cisco and Cloudian HyperStore offers on-demand infinite scalability, allowing storage space to grow as needed. As demand grows, additional storage nodes can be added across multiple DCs.

Security

Cisco and Cloudian HyperStore takes safeguarding customer data very seriously. Two server-side encryption methods (SSE/SSE-c, KeySecure) are implemented to ensure that data is always protected.

Cloudian HyperStore simplifies the data encryption process by providing transparent key management at the server or node layer. This relieves administrators from the burden of having to manage encryption keys and eliminates the risk of data loss occurring due to lost keys. Furthermore, encryption can be managed very granularly–from a large-scale to an individual object.

Data Protection

With the ISA-L Powered Erasure Coding, Cloudian HyperStore optimizes storage for all data objects, providing efficient storage redundancy with low disk space consumption.

Effortless Data Movement

Cloudian HyperStore easily manages data, stores and retrieves data on-demand (with unique features like object streaming, dynamic auto-tiering), and seamlessly moves data between on-premises cloud and Amazon S3, irrespective of data size.

Solution Design

Design Considerations

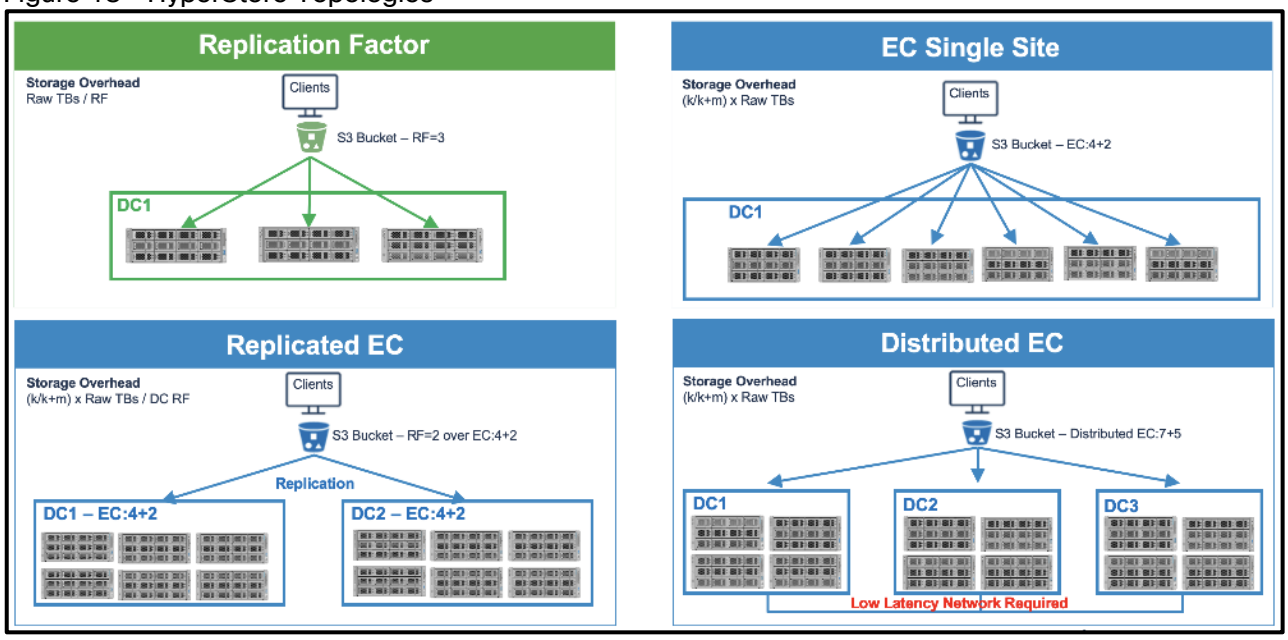
Number of Nodes of Cisco UCS C240 M5

When performance and storage capacity is not that important, a three-node configuration is recommended. This also reduces the TCO of the solution. However, as the performance and storage need increases, additional nodes can be added to the cluster.

Replication versus Erasure Coding

Central to Cloudian's data protection are its storage policies. These policies are ways of protecting data so that it's durable and highly available to users. The Cloudian HyperStore system lets you preconfigure one or more storage policies. Users, when creating a new storage bucket, can choose which preconfigured storage policy to use to protect data in that bucket. Users cannot create buckets until you have created at least one storage policy.

Figure 18 HyperStore Topologies

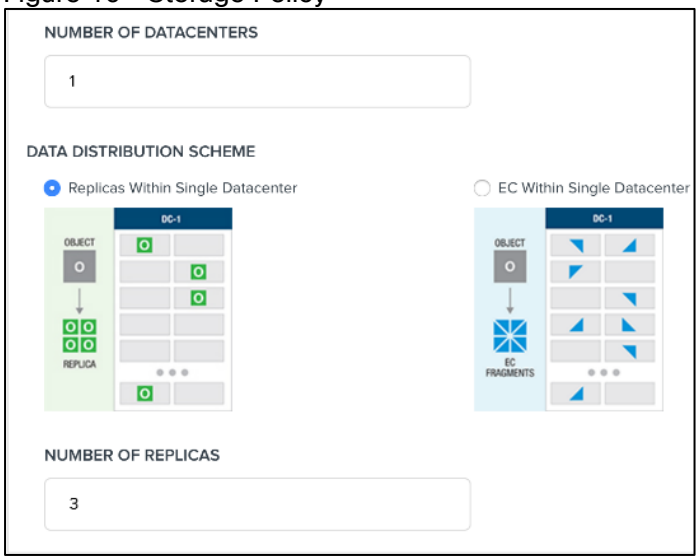


For each storage policy that you create, you can choose from the following two data protection methods:

Replication

With replication, a configurable number of copies of each data object are maintained in the system, and each copy is stored on a different node. For example, with 3X replication 3 copies of each object are stored, with each copy on a different node.

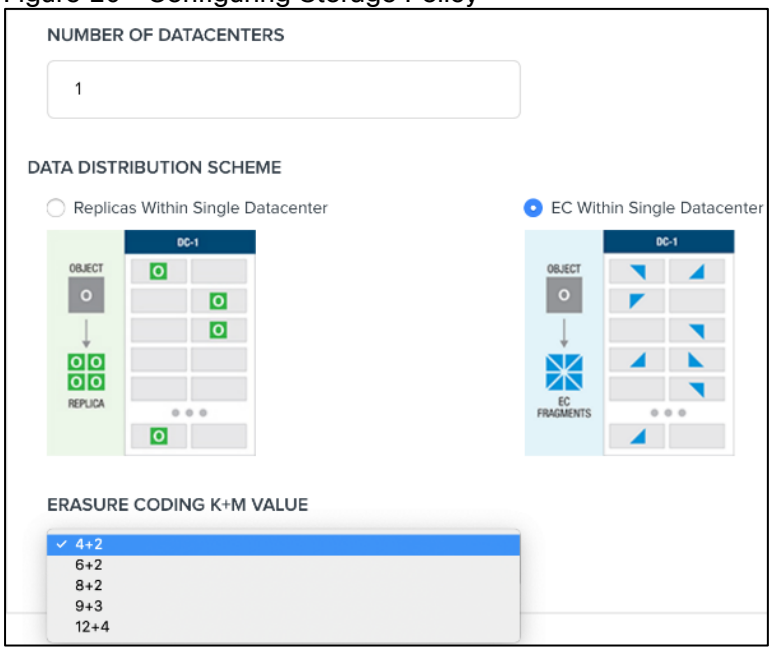
Figure 19 Storage Policy



Erasure Coding

With erasure coding, each object is encoded into a configurable number (known as the k value) of data fragments plus a configurable number (the m value) of redundant parity fragments. Each fragment is stored on a different node, and the object can be decoded from any k number of fragments. For example, in a 4:2 erasure coding configuration (4 data fragments plus 2 parity fragments), each object is encoded into a total of 6 fragments which are stored on 6 different nodes, and the object can be decoded and read so long as any 4 of those 6 fragments are available.

Figure 20 Configuring Storage Policy



Erasure coding requires less storage overhead (the amount of storage required for data redundancy) and results in somewhat longer request latency than replication. Erasure coding is best suited to large objects over a low latency network.

Supported Erasure Coding Configurations

Cloudian HyperStore supports EC, replicated EC, and distributed EC configurations.

- EC

  This configuration requires a minimum 6 nodes across a single Data Centers (DC). This supports the minimum data and parity fragments of (4+2) where 2 is the parity. Table 1 lists the default EC configuration and the default number of nodes for a single DC.

Cloudian also supports 5 nodes EC as a custom policy – EC3+2.

Table 1 Default EC Configuration and Default Number of Nodes

Nodes in the DC	EC
6	4+2
8	6+2
10	8+2
12	9+3
16	12+4

- Replicated EC

  This configuration requires a minimum of two Data Centers (DC). Each DC consists of 3 nodes each. This supports the minimum data and parity fragments of (2+1) where 1 is the parity. Table 2 lists the default replicated EC configuration and the default number of nodes per DC.

Table 2 Default Replicated EC Configuration and Default Number of Nodes

Nodes Total	DC1	DC2	EC
6	3	3	2+1
12	6	6	4+2
16	8	8	6+2
20	10	10	8+2
24	12	12	9+3

Each object is encoded into equal parts and parity fragments are replicated on each node. Each DC is a mirror image. For configurations greater than 2 DC, Distributed EC configuration is recommended. This configuration mirrors the encoded data and parity fragments to the other data centers in the configuration.

The choice among these three supported EC configurations is largely a matter of how many Cloudian HyperStore nodes in the datacenter. For a replicated EC configuration, a minimum of 3 nodes per DC are required.

- Distributed EC

  Cloudian's Distributed EC solution implements the new ISA-L Erasure Codes that is vectored and fast. ISA-L is the Intel library containing functions to improve erasure coding.

  The Cloudian Distributed Datacenterwith EC configuration requires a minimum of 3 data centers with 4 nodes each.

Data stored: DC1: 4, Dc2: 4, DC3:4, Metadata stored: Data stored: DC1: 4, DC2: 4, DC3:3

Distributed EC configuration offers the same level of protection as the replicated EC configuration with 50% less storage. The Distributed EC configuration is recommended if number of DCs involved are 3 or more.

Flash Storage

Flash Storage with SAS SSD's are used to store metadata for faster performance. The standard capacity requirement for Flash are less than 1 percent of the total data capacity. Standard design also calls for having a ratio of 1 SSD for 10 HDD.

JBOD versus RAID0 Disks

While Cloudian HyperStore as an SDS solution works with JBODs or with RAID0 disks, it is recommended to use JBOD for the solution. The 12G SAS RAID controller in C240 M5 provides up to 4G of cache that can be used for writes.

Memory Sizing

Memory sizing is based on the number of objects stored on each rack server, which is related to the average file size and the data protection scheme. Standard designs call for 384GB for the C240 M5.

Network Considerations

Cloudian Network requirements are standard Ethernet only. Please refer to the Network layout diagram in Figure 25. While Cloudian software can work on a single network interface, it is recommended to create different virtual interfaces in Cisco UCS and segregate them. A client-access network and private-cluster network are required for the operation. Cisco UCS C240 M5 has two physical ports of 40G each and the VIC allows you to create out many Virtual interfaces on each physical port.

It is recommended to have a private-cluster network on one port and the client-access networks on another port. This provides 40Gb bandwidth for each of these networks. While the client-access network requirements are minimal, every storage node can take up to 40Gb of client bandwidth requirements. Also, by having the client and cluster VIC's pinned to each fabric of the fabric interconnects, there is a minimal overhead of network traffic passing through the upstream switches for inter-node communication, if any. This unique feature of fabric interconnects and VIC's makes the design highly flexible and scalable.

Uplinks

The uplinks from fabric interconnects to upstream switches like Nexus, carry the traffic in case of FI failures or reboots. A reboot for instance is needed during a firmware upgrade. While there is complete high availability built-in the infrastructure, the performance may drop, depending on the uplink connectors from each FI to the Nexus vPC pool. If you want 'no' or a 'minimal drop', increase the uplink connectors.

Multi-Site Deployments

Like Amazon S3, the Cloudian HyperStore system supports the implementation of multiple service regions. Setting up the Cloudian HyperStore system to use multiple service regions is optional.

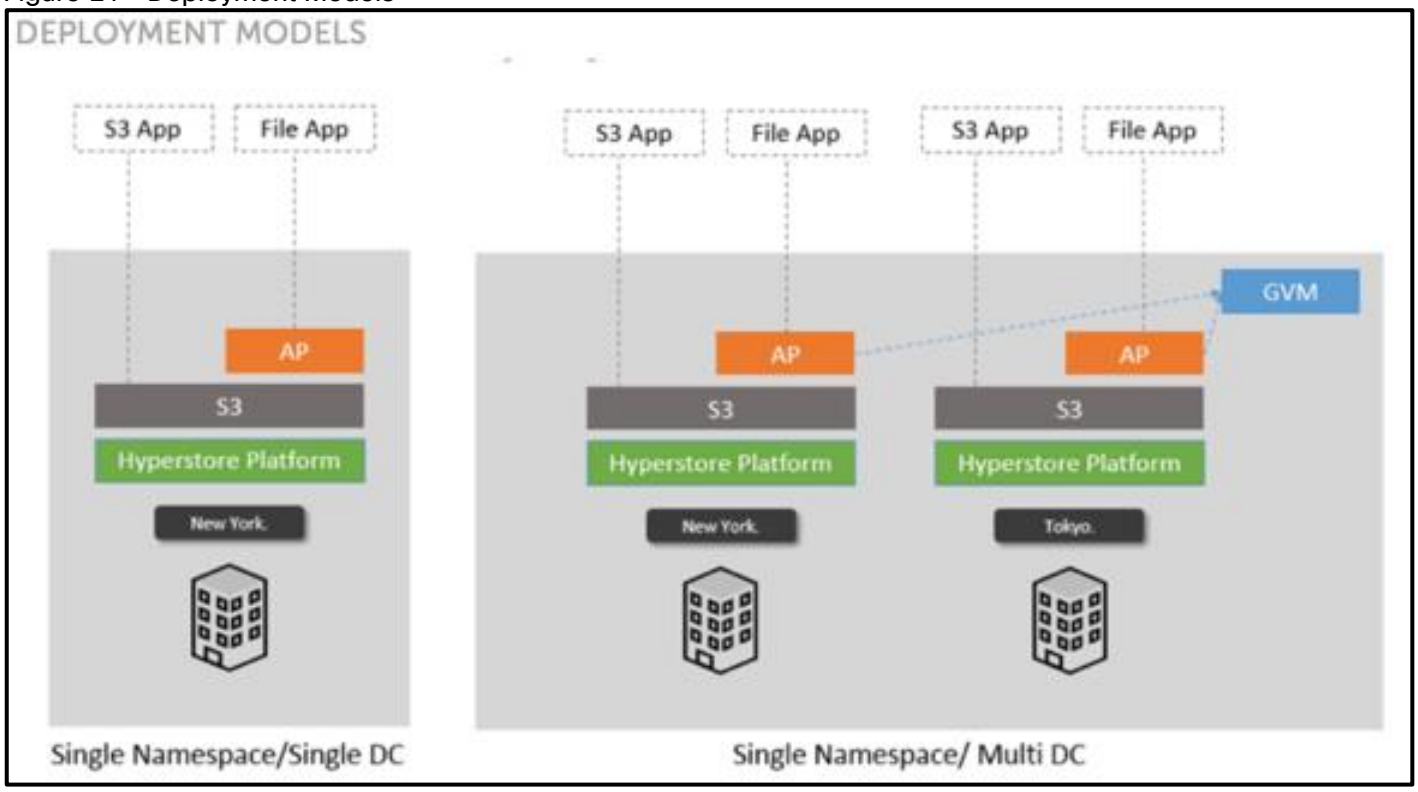
The main benefits of deploying multiple service regions are:

- Each region has its own independent Cloudian HyperStore geo-cluster for S3 object storage. Consequently, deploying multiple regions is another means of scaling-out overall Cloudian HyperStore

service offering (beyond using multiple nodes and multiple datacenters to scale out a single geo-cluster). In a multi-region deployment, different S3 datasets are stored in each region. Each region has its own token space and there is no data replication across regions.

- With a multi-region deployment, service users can choose the service region in which their storage buckets will be created. Users may choose to store their S3 objects in the region that's geographically closest to them; or they may choose one region rather than another for reasons of regulatory compliance or corporate policy.

Figure 21 Deployment Models



Designing a multi-site is beyond the scope of this document and for simplicity, only a single site deployment test bed is setup. Please contact Cisco and Cloudian if you have multi-site requirements.

Should a customer's workload and use case requirements not conform to the assumptions made while building these standard configurations, Cisco and Cloudian can work together to build custom hardware sizing to support the customer's workload.

Expansion of the Cluster

Cisco UCS hardware, along with Cloudian HyperStore, offers exceptional flexibility in order to scale-out as storage requirements change:

- Cisco UCS 6332 Fabric Interconnects have 32 ports each. Each server is connected to either of the FI's. Leaving the uplinks and any other clients directly connected to the Fabrics, 24-28 server nodes can be connected to FI pairs. If more servers are required, you should plan for a multi-domain system.

- Cisco UCS offers KVM management both in-band and out-of-band. In case out-of-band management is planned, you may have to reserve as many free IP's as needed for the servers. Planning while designing the cluster makes expansion very straightforward.
- Cisco UCS provides IP pool management, MAC pool management along with policies that can be defined once for the cluster. Any future expansion for adding nodes and so on, is just a matter of expanding the above pools.
- Cisco UCS is a template and policy-based infrastructure management tool. All the identity of the servers is stored through Service Profiles that are cloned from templates. When a template is created, a new service profile for the additional server, can be created and applied on the newly added hardware. Cisco UCS makes Infrastructure readiness, extremely simple, for any newly added storage nodes. Rack the nodes, connect the cables, and then clone and apply the service profile.
- When the nodes are ready, you may have to follow the node addition procedure per the Cloudian documentation.

The simplified management of the infrastructure with Cisco UCS and well-tested node addition from Cloudian makes the expansion of the cluster very simple.

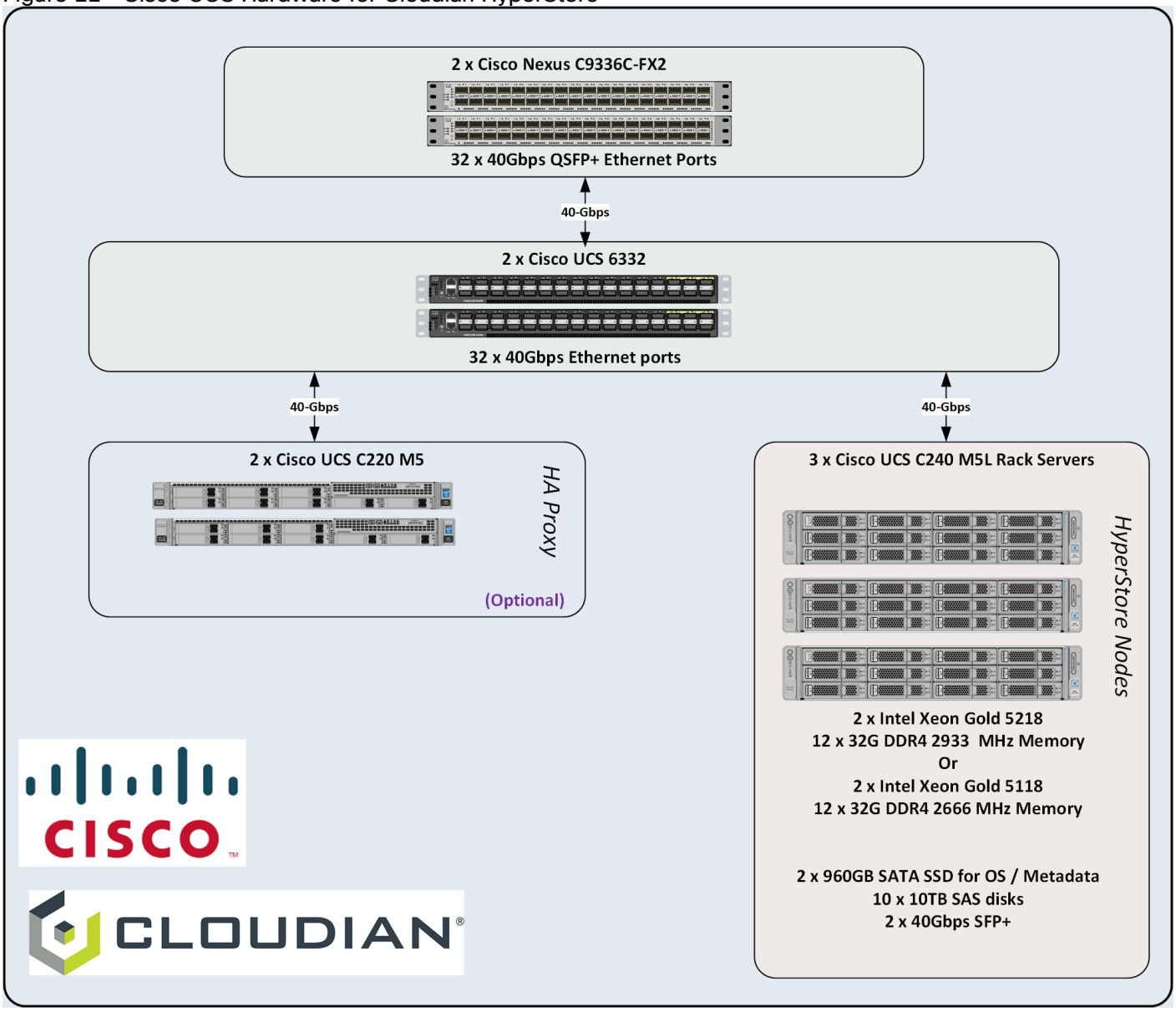
Deployment Architecture

The reference architecture use case provides a comprehensive, end-to-end example of designing and deploying Cloudian object storage on Cisco UCS C240 M5 as shown in Figure 22. This document describes the architecture and design of a Cloudian Scale-out object storage on three Cisco UCS C240 M5 Rack Servers and two Cisco UCS C220 M5S rack server as HA-proxy nodes. The whole solution is connected to a pair of Cisco UCS 6332 Fabric Interconnects and a pair of upstream network Cisco Nexus C9336C-FX2 switches.

The configuration is comprised of the following:

- 2 x Cisco Nexus 9000 C9336C-FX2 Switches
- 2 x Cisco UCS 6332 Fabric Interconnects
- 3 x Cisco UCS C240 M5L Rack Servers
- 2 x Cisco UCS C220 M5S Rack Servers (Optional for HA-Proxy)

Figure 22 Cisco UCS Hardware for Cloudian HyperStore



System Hardware and Software Specifications

Solution Overview

This solution is based on Cisco UCS, Cloudian Object, and file storage.

Software Versions

Table 3 Software Versions

Layer	Component	Version or Release
Compute (Server/Storage Nodes)	BIOS	C240M5.4.0.4d.0.0506190827

Layer	Component	Version or Release
Cisco UCS C240 M5L	CIMC Controller	4.0(4c)
Compute (HA-Proxy Nodes) Cisco UCS C220 M5S	BIOS	C220M5.4.0.4c.0.0506190754
	CIMC Controller	4.0(4c)
Network 6332 Fabric Interconnect	UCS Manager	4.0(4b)
	Kernel	5.0(3)N2(4.04a)
	System	5.0(3)N2(4.04a)
Network Nexus 9000 C9336C-FX2	BIOS	05.33
	NXOS	9.2(3)
Software	Red Hat Enterprise Linux Server	7.6 (x86_64)
	Cloudian HyperStore	7.1.4

Hardware Requirements and Bill of Materials

Table 4 lists the bill of materials used in this CVD.

Table 4 Bill of Materials

Component	Model	Quantity	Comments
Cloudian Storage Nodes	Cisco UCS C240 M5L Rack Servers	3	Per Server Node – 2 x Intel(R) Xeon(R) Gold 5118 (2.30 GHz/12 cores) or 2 x Intel(R) Xeon(R) Gold 5218 ((2.30 GHz/16 cores) – 384 GB RAM – Cisco 12G Modular Raid controller with 2GB cache – 2 x 960GB 3.5 inch Enterprise Value 6G SATA SSD (For OS and Metadata) – 10 x 10TB 12G SAS 7.2K RPM LFF HDD (512e) – Dual-port 40 Gbps VIC (Cisco UCS VIC 1385)
Cloudian HA-Proxy Node (Optional)	Cisco UCS C220 M5S Rack server	2	– 2 x Intel Xeon Silver 4110 (2.1GHz/8 Cores), 96GB RAM – Cisco 12G SAS RAID Controller – 2 x 600GB SAS for OS

Component	Model	Quantity	Comments
			– Dual-port 40 Gbps VIC
UCS Fabric Interconnects FI-6332	Cisco UCS 6332 Fabric Interconnects	2	
Switches Nexus 9000 C9336C-FX2	Cisco Nexus Switches	2	

Physical Topology and Configuration

Figure 23 illustrates the physical design of the solution and the configuration of each component.

The connectivity of the solution is based on 40 Gigabit. All components are connected via 40 Gbit QSFP cables. Between each Cisco UCS 6332 Fabric Interconnect and both Cisco Nexus C9336C-FX2 is one virtual Port Channel (vPC) configured. vPCs allow links that are physically connected to two different Cisco Nexus 9000 switches to appear to the Fabric Interconnect as coming from a single device and as part of a single port channel.

Between both Cisco Nexus 9336C-FX2 switches are 4 x 40 Gbit cabling. Each Cisco UCS 6332 Fabric Interconnect is connected via 2 x 40 Gigabit to each Cisco UCS C9336C-FX2 switch. Cisco UCS C240M5 and C220 M5 are connected via 1 x 40 Gbit to each Fabric Interconnect. The architecture is highly redundant, and system survived with little or no impact to applications under various failure test scenarios which is explained in section High Availability Tests.High Availability TestsHigh Availability TestsHigh Availability TestsHigh Availability Tests

Figure 23 Physical Topology

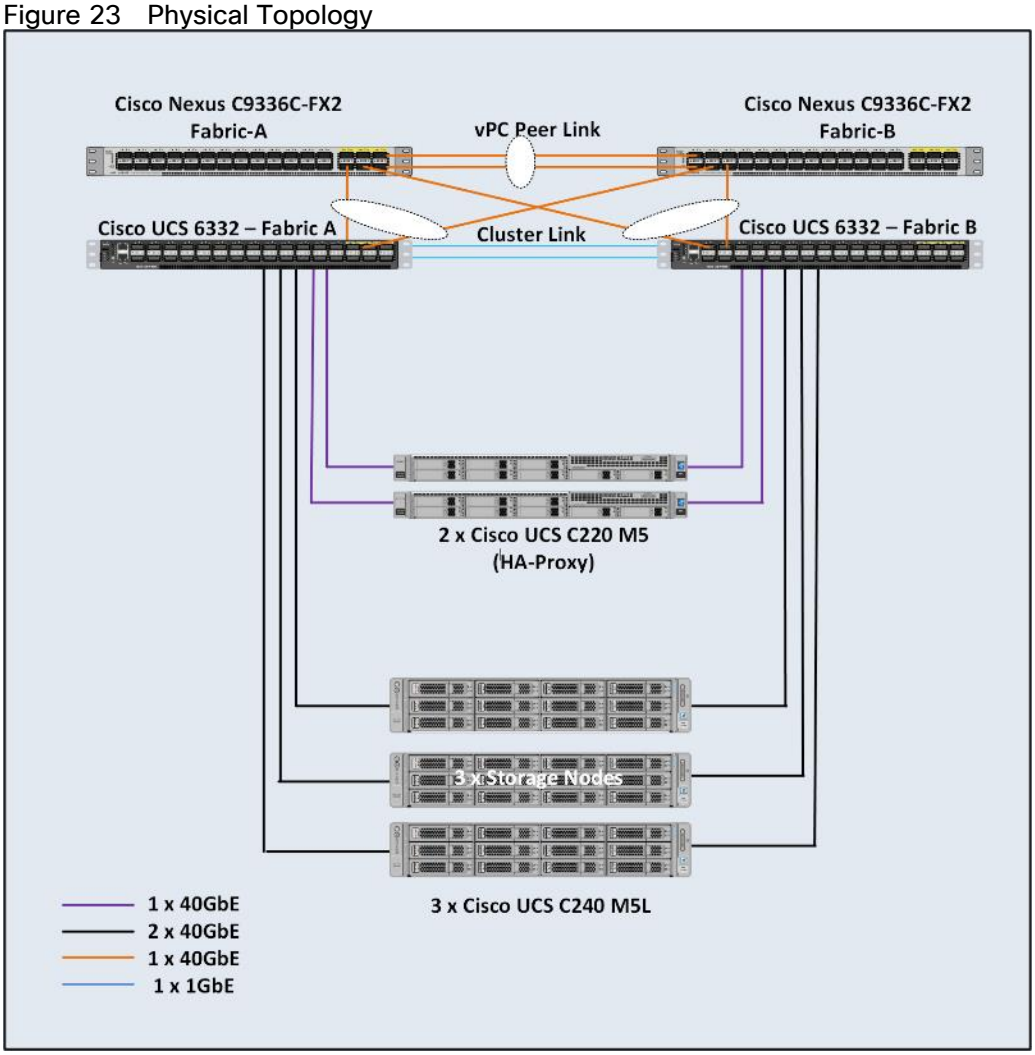
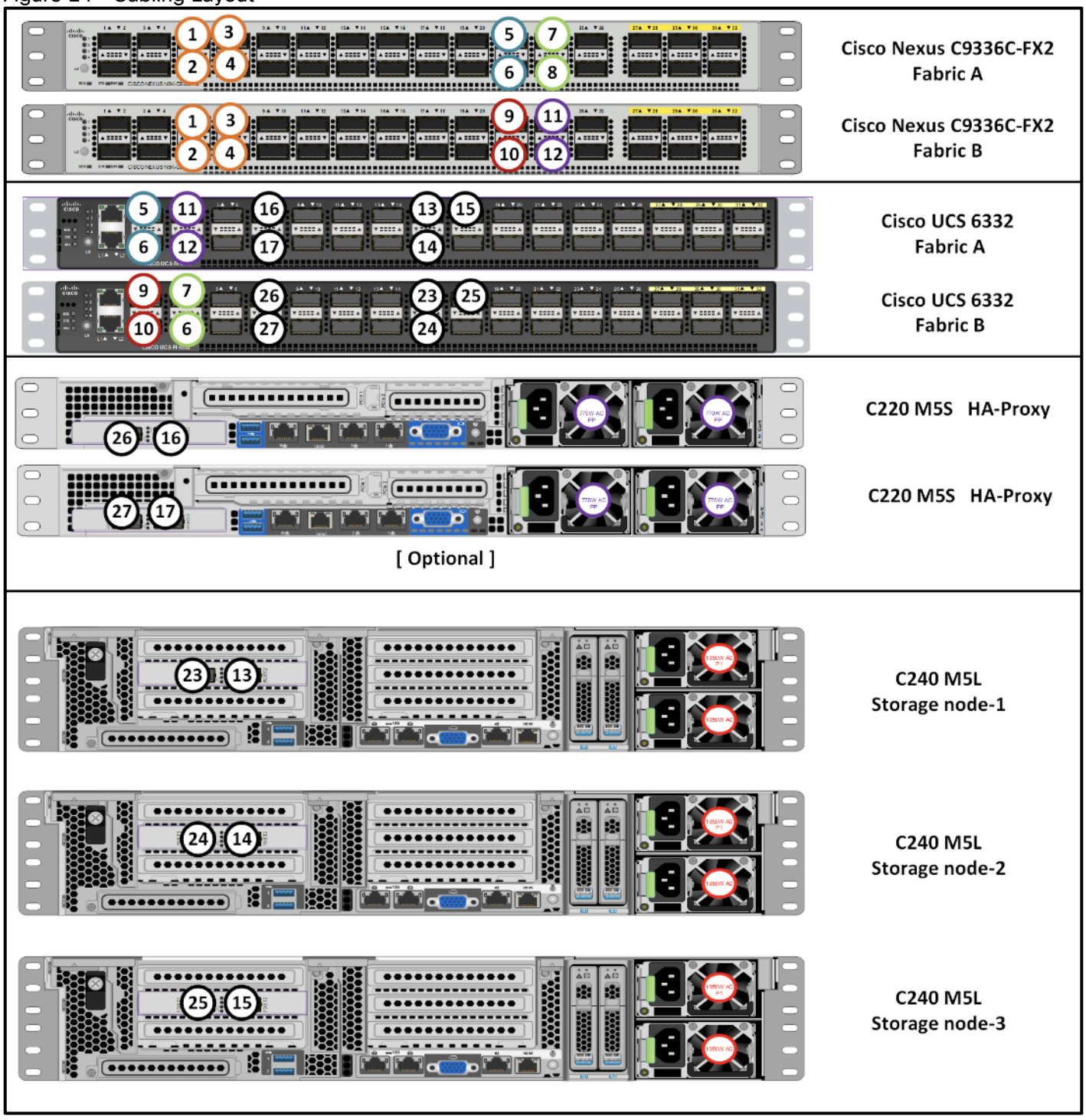


Figure 24 illustrates the actual cabling between servers and switches.

Figure 24 Cabling Layout



The exact cabling for the Cisco UCS C240 M5, Cisco UCS C220 M5, Cisco UCS 6332 Fabric Interconnect and the Nexus 9000 C9336C-FX2 is listed in Table 5

Table 5 Cabling Information

Local Device	Local Port	Connection	Remote Device	Remote Port	Cable

Local Device	Local Port	Connection	Remote Device	Remote Port	Cable
Cisco Nexus C9336C-FX2 Switch- A	Eth1/1	40GbE	Cisco UCS Fabric Interconnect A	Eth1/1	QSFP-H40G-AOC5M
	Eth1/2	40GbE	Cisco UCS Fabric Interconnect A	Eth1/2	QSFP-H40G-AOC5M
	Eth1/3	40GbE	Cisco UCS Fabric Interconnect B	Eth1/3	QSFP-H40G-AOC5M
	Eth1/4	40GbE	Cisco UCS Fabric Interconnect B	Eth1/4	QSFP-H40G-AOC5M
	Eth1/21	40GbE	Cisco Nexus C9336C-FX2 Switch- B	Eth1/21	QSFP-H40G-AOC5M
	Eth1/22	40GbE	Cisco Nexus C9336C-FX2 Switch- B	Eth1/22	QSFP-H40G-AOC5M
	Eth1/23	40GbE	Cisco Nexus C9336C-FX2 Switch- B	Eth1/23	QSFP-H40G-AOC5M
	Eth1/24	40GbE	Cisco Nexus C9336C-FX2 Switch- B	Eth1/24	QSFP-H40G-AOC5M
	Eth1/36	40GbE	Top of Rack (Upstream Network)	Any	QSFP+ 4SFP10G
	MGMT0	1GbE	Top of Rack (Management)	Any	1G RJ45
Cisco Nexus C9336C-FX2 Switch- B	Eth1/1	40GbE	Cisco UCS Fabric Interconnect B	Eth1/1	QSFP-H40G-AOC5M
	Eth1/2	40GbE	Cisco UCS Fabric Interconnect B	Eth1/2	QSFP-H40G-AOC5M
	Eth1/3	40GbE	Cisco UCS Fabric Interconnect A	Eth1/3	QSFP-H40G-AOC5M
	Eth1/4	40GbE	Cisco UCS Fabric Interconnect A	Eth1/4	QSFP-H40G-AOC5M
	Eth1/21	40GbE	Cisco Nexus C9336C-FX2 Switch- A	Eth1/21	QSFP-H40G-AOC5M
	Eth1/22	40GbE	Cisco Nexus C9336C-FX2 Switch- A	Eth1/22	QSFP-H40G-AOC5M
	Eth1/23	40GbE	Cisco Nexus C9336C-FX2	Eth1/23	QSFP-H40G-AOC5M

Local Device	Local Port	Connection	Remote Device	Remote Port	Cable
			Switch- A		
	Eth1/24	40GbE	Cisco Nexus C9336C-FX2 Switch- A	Eth1/24	QSFP-H40G-AOC5M
	Eth1/36	40GbE	Top of Rack (Upstream Network)	Any	QSFP+ 4SFP10G
	MGMT0	1GbE	Top of Rack (Management)	Any	1G RJ45
Cisco UCS 6332 Fabric Interconnect A	Eth1/1	40GbE	Cisco Nexus C9336C-FX2 Switch- A	Eth 1/1	QSFP-H40G-AOC5M
	Eth1/2	40GbE	Cisco Nexus C9336C-FX2 Switch- A	Eth 1/2	QSFP-H40G-AOC5M
	Eth1/3	40GbE	Cisco Nexus C9336C-FX2 Switch- B	Eth 1/3	QSFP-H40G-AOC5M
	Eth1/4	40GbE	Cisco Nexus C9336C-FX2 Switch- B	Eth 1/4	QSFP-H40G-AOC5M
	Eth1/7	40GbE	C240 M5 - 1	Port 1	QSFP-H40G-AOC5M
	Eth1/8	40GbE	C240 M5 - 2	Port 1	QSFP-H40G-AOC5M
	Eth1/9	40GbE	C240 M5 - 3	Port 1	QSFP-H40G-AOC5M
	Eth1/10	40GbE	C220 M5 - 1	Port 1	QSFP-H40G-AOC5M
	Eth1/11	40GbE	C220 M5 - 2	Port 1	QSFP-H40G-AOC5M
	MGMT0	40GbE	Top of Rack (Management)	Any	1G RJ45
	L1	1GbE	UCS 6332 Fabric Interconnect B	L1	1G RJ45
	L2	1GbE	UCS 6332 Fabric Interconnect B	L2	1G RJ45
Cisco UCS 6332 Fabric	Eth1/1	40GbE	Cisco Nexus C9336C-FX2 Switch- B	Eth 1/1	QSFP-H40G-AOC5M

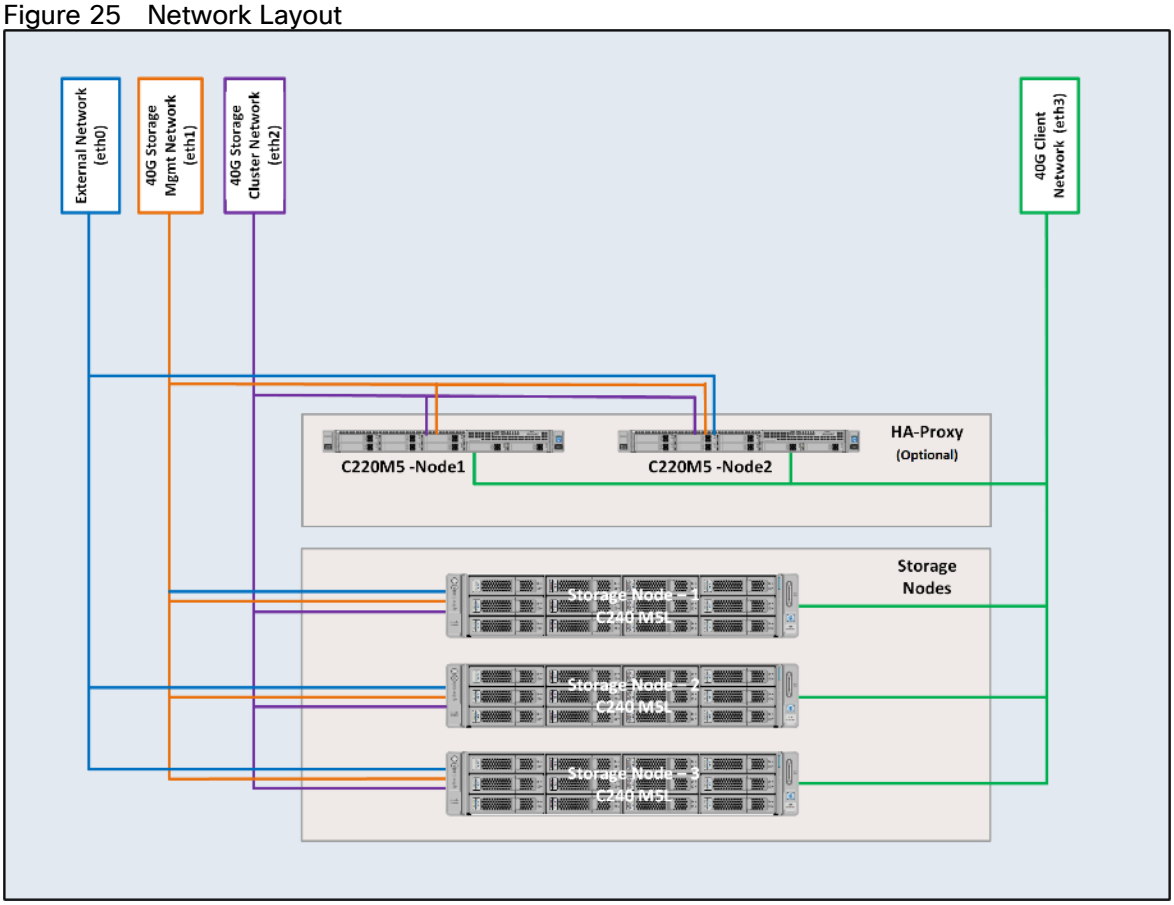
Local Device	Local Port	Connection	Remote Device	Remote Port	Cable
Interconnect B	Eth1/2	40GbE	Cisco Nexus C9336C-FX2 Switch- B	Eth 1/2	QSFP-H40G-AOC5M
	Eth1/3	40GbE	Cisco Nexus C9336C-FX2 Switch- A	Eth 1/3	QSFP-H40G-AOC5M
	Eth1/4	40GbE	Cisco Nexus C9336C-FX2 Switch- A	Eth 1/4	QSFP-H40G-AOC5M
	Eth1/7	40GbE	C240 M5 - 1	Port 2	QSFP-H40G-AOC5M
	Eth1/8	40GbE	C240 M5 - 2	Port 2	QSFP-H40G-AOC5M
	Eth1/9	40GbE	C240 M5 - 3	Port 2	QSFP-H40G-AOC5M
	Eth1/10	40GbE	C220 M5 - 1	Port 2	QSFP-H40G-AOC5M
	Eth1/11	40GbE	C220 M5 - 2	Port 2	QSFP-H40G-AOC5M
	MGMT0	40GbE	Top of Rack (Management)	Any	1G RJ45
	L1	1GbE	UCS 6332 Fabric Interconnect A	L1	1G RJ45
	L2	1GbE	UCS 6332 Fabric Interconnect A	L2	1G RJ45

Network Topology

It is important to separate the network traffic with separate virtual NIC and VLANs for outward facing(eth0), host management(eth1), Cluster(eth2) and client(eth3) traffics. eth0, eth1 and eth3 are pinned to uplink interface 0 of VIC and eth2 is pinned to uplink interface 1 to enable better traffic distribution.

Figure 25 illustrates the Network Topology used in the setup.

Figure 25 Network Layout

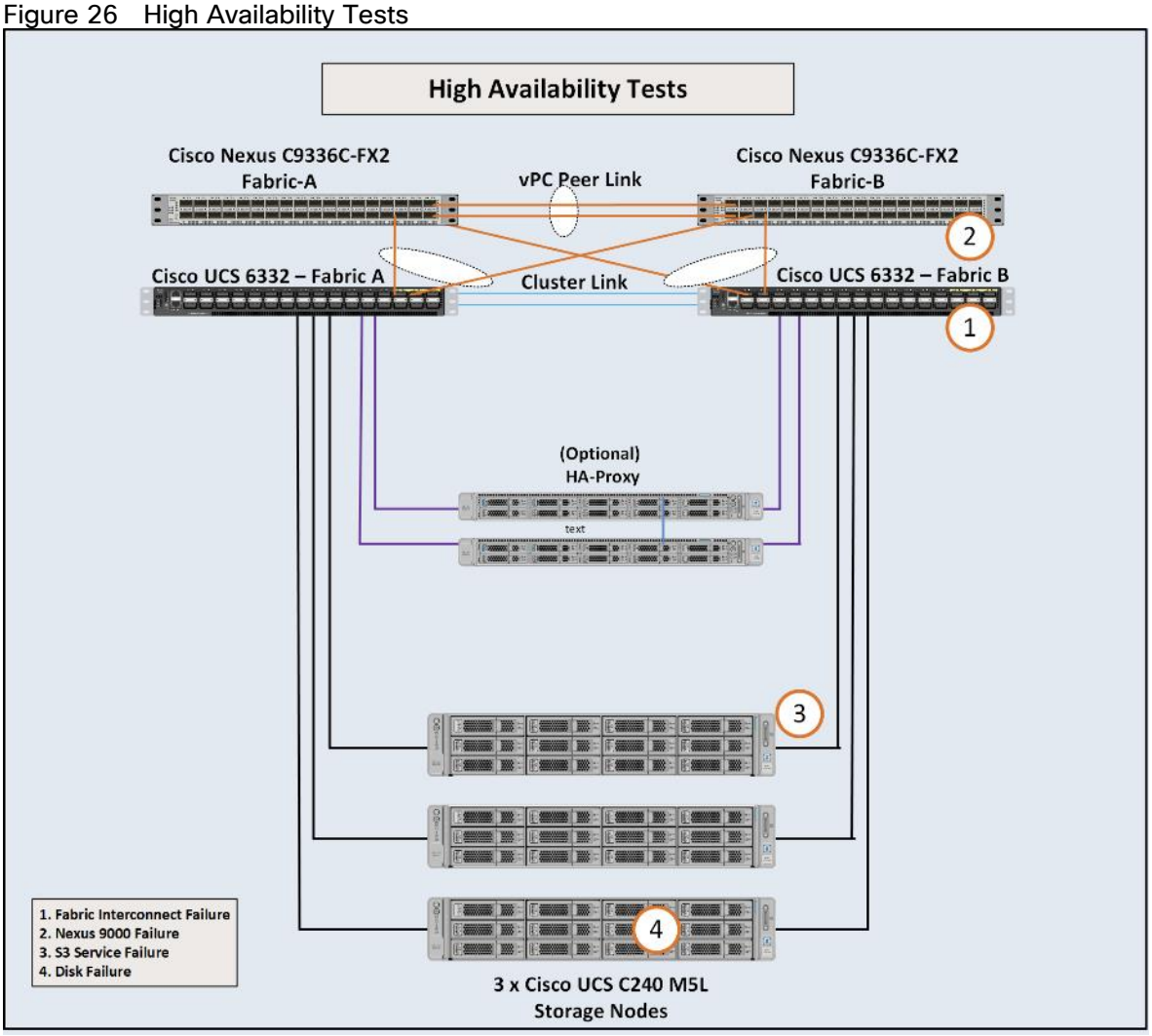


High Availability

As part of the hardware and software resiliency, random read and write load test with objects of 10MB in size will run during the failure injections. The following tests will be conducted on the test bed:

1. Fabric Interconnect failures
2. Nexus 9000 failures
3. S3 Service failures
4. Disk failures

Figure 26 High Availability Tests



Deployment of Hardware and Software

Configuration of Nexus C9336-FX2 Switch A and B

Both Cisco UCS Fabric Interconnect A and B are connected to two Cisco Nexus C9336C-FX2 switches for connectivity to Upstream Network. The following sections describe the setup of both C9336C-FX2 switches.

Initial Setup of Nexus C9336C-FX2 Switch A and B

To configure Switch A, connect a Console to the Console port of each switch, power on the switch and follow these steps:

1. Type **yes**.
2. Type **n**.
3. Type **n**.
4. Type **n**.
5. Enter the switch name.
6. Type **y**.
7. Type your IPv4 management address for Switch A.
8. Type your IPv4 management netmask for Switch A.
9. Type **y**.
10. Type your IPv4 management default gateway address for Switch A.
11. Type **n**.
12. Type **n**.
13. Type **y** for ssh service.
14. Press `<Return>` and then `<Return>`.
15. Type **y** for ntp server.
16. Type the IPv4 address of the NTP server.
17. Press `<Return>`, then `<Return>` and again `<Return>`.
18. Check the configuration and if correct then press `<Return>` and again `<Return>`.

The complete setup looks like the following:

```
          ---- System Admin Account Setup ----
Do you want to enforce secure password standard (yes/no) [y]: no
  Enter the password for admin:
  Confirm the password for admin:
         ---- Basic System Configuration Dialog VDC: 1 ----

This setup utility will guide you through the basic configuration of
the system. Setup configures only enough connectivity for management
of the system.
Please register Cisco Nexus9000 Family devices promptly with your
supplier. Failure to register may affect response times for initial
service calls. Nexus9000 devices must be registered to receive
entitled support services.

Press Enter at anytime to skip a dialog. Use ctrl-c at anytime
to skip the remaining dialogs.

 Would you like to enter the basic configuration dialog (yes/no): yes
  Create another login account (yes/no) [n]: no
  Configure read-only SNMP community string (yes/no) [n]: no
  Configure read-write SNMP community string (yes/no) [n]: no
  Enter the switch name : N9K-Cloudian-Fab-A
  Continue with Out-of-band (mgmt0) management configuration? (yes/no) [y]: yes
    Mgmt0 IPv4 address : 173.36.220.13
    Mgmt0 IPv4 netmask : 255.255.255.0
  Configure the default gateway? (yes/no) [y]: yes
    IPv4 address of the default gateway : 173.36.220.1
  Configure advanced IP options? (yes/no) [n]: no
  Enable the telnet service? (yes/no) [n]: no
  Enable the ssh service? (yes/no) [y]: yes
    Type of ssh key you would like to generate (dsa/rsa) [rsa]: rsa
    Number of rsa key bits <1024-2048> [1024]: 1024
  Configure the ntp server? (yes/no) [n]: yes
    NTP server IPv4 address : 171.68.38.65
  Configure default interface layer (L3/L2) [L2]: L2
  Configure default switchport interface state (shut/noshut) [noshut]: shut
  Configure CoPP system profile (strict/moderate/lenient/dense) [strict]:
The following configuration will be applied:
  no password strength-check
  switchname N9K-Cloudian-Fab-A
vrf context management
ip route 0.0.0.0/0 173.36.220.1
exit
  no feature telnet
  ssh key rsa 1024 force
  feature ssh
  ntp server 171.68.38.65
  system default switchport
  system default switchport shutdown
  copp profile strict
interface mgmt0
ip address 173.36.220.13 255.255.255.0
no shutdown

Would you like to edit the configuration? (yes/no) [n]: no
Use this configuration and save it? (yes/no) [y]: yes
 [########################################] 100%
```

```
Copy complete, now saving to disk (please wait)...
Copy complete.
User Access Verification
N9K-Cloudian-Fab-A login:
```

19. Repeat steps 1-18 for the Nexus C9336C-FX2 Switch B except for configuring a different IPv4 management address 173.36.220.14 as described in step 7.

Enable Features on Nexus C9336C-FX2 Switch A and B

To enable the features UDLD, VLAN, HSRP, LACP, VPC, and Jumbo Frames, connect to the management interface via ssh on both switches and follow these steps on both Switch A and B:

Switch A

```
N9K-Cloudian-Fab-A# configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
N9K-Cloudian-Fab-A(config)# feature udld
N9K-Cloudian-Fab-A(config)# feature interface-vlan
N9K-Cloudian-Fab-A(config)# feature hsrp
N9K-Cloudian-Fab-A(config)# feature lacp
N9K-Cloudian-Fab-A(config)# feature vpc
N9K-Cloudian-Fab-A(config)# system jumbomtu 9216
N9K-Cloudian-Fab-A(config)# exit
N9K-Cloudian-Fab-A(config)# copy running-config startup-config
```

Switch B

```
N9K-Cloudian-Fab-B# configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
N9K-Cloudian-Fab-B(config)# feature udld
N9K-Cloudian-Fab-B(config)# feature interface-vlan
N9K-Cloudian-Fab-B(config)# feature hsrp
N9K-Cloudian-Fab-B(config)# feature lacp
N9K-Cloudian-Fab-B(config)# feature vpc
N9K-Cloudian-Fab-B(config)# system jumbomtu 9216
N9K-Cloudian-Fab-B(config)# exit
N9K-Cloudian-Fab-B(config)# copy running-config startup-config
```

Configure VLANs on Nexus C9336C-FX2 Switch A and B

To configure the same VLANs Storage-Management, Storage-Cluster, Client Network, and External Management as previously configured in the Cisco UCS Manager GUI, follow these steps on Switch A and Switch B:

Switch A

```
N9K-Cloudian-Fab-A# config terminal
Enter configuration commands, one per line. End with CNTL/Z.
N9K-Cloudian-Fab-A(config)# vlan 10
N9K-Cloudian-Fab-A(config-vlan)# name Storage-Management
N9K-Cloudian-Fab-A(config-vlan)# no shut
N9K-Cloudian-Fab-A(config-vlan)# exit
N9K-Cloudian-Fab-A(config)# vlan 30
N9K-Cloudian-Fab-A(config-vlan)# name Storage-Cluster
N9K-Cloudian-Fab-A(config-vlan)# no shut
N9K-Cloudian-Fab-A(config-vlan)# exit
```

```
N9K-Cloudian-Fab-A(config)# vlan 220
N9K-Cloudian-Fab-A(config-vlan)# name External-Mgmt
N9K-Cloudian-Fab-A(config-vlan)# no shut
N9K-Cloudian-Fab-A(config-vlan)# exit
N9K-Cloudian-Fab-A(config)# vlan 20
N9K-Cloudian-Fab-A(config-vlan)# name Client-Network
N9K-Cloudian-Fab-A(config-vlan)# no shut
N9K-Cloudian-Fab-A(config-vlan)# exit


N9K-Cloudian-Fab-A(config)# interface vlan10
N9K-Cloudian-Fab-A(config-if)# description Storage-Mgmt
N9K-Cloudian-Fab-A(config-if)# no shutdown
N9K-Cloudian-Fab-A(config-if)# no ip redirects
N9K-Cloudian-Fab-A(config-if)# ip address 192.168.10.253/24
N9K-Cloudian-Fab-A(config-if)# no ipv6 redirects
N9K-Cloudian-Fab-A(config-if)# hsrp version 2
N9K-Cloudian-Fab-A(config-if)# hsrp 10
N9K-Cloudian-Fab-A(config-if-hsrp)# preempt
N9K-Cloudian-Fab-A(config-if-hsrp)# priority 10
N9K-Cloudian-Fab-A(config-if-hsrp)# ip 192.168.10.1
N9K-Cloudian-Fab-A(config-if-hsrp)# exit
N9K-Cloudian-Fab-A(config-if)# exit


N9K-Cloudian-Fab-A(config)# interface vlan30
N9K-Cloudian-Fab-A(config-if)# description Storage-Cluster
N9K-Cloudian-Fab-A(config-if)# no shutdown
N9K-Cloudian-Fab-A(config-if)# no ip redirects
N9K-Cloudian-Fab-A(config-if)# ip address 192.168.30.253/24
N9K-Cloudian-Fab-A(config-if)# no ipv6 redirects
N9K-Cloudian-Fab-A(config-if)# hsrp version 2
N9K-Cloudian-Fab-A(config-if)# hsrp 30
N9K-Cloudian-Fab-A(config-if-hsrp)# preempt
N9K-Cloudian-Fab-A(config-if-hsrp)# priority 10
N9K-Cloudian-Fab-A(config-if-hsrp)# ip 192.168.30.1
N9K-Cloudian-Fab-A(config-if-hsrp)# exit
N9K-Cloudian-Fab-A(config)# interface vlan20
N9K-Cloudian-Fab-A(config-if)# description Client-Network
N9K-Cloudian-Fab-A(config-if)# no shutdown
N9K-Cloudian-Fab-A(config-if)# no ip redirects
N9K-Cloudian-Fab-A(config-if)# ip address 192.168.20.253/24
N9K-Cloudian-Fab-A(config-if)# no ipv6 redirects
N9K-Cloudian-Fab-A(config-if)# hsrp version 2
N9K-Cloudian-Fab-A(config-if)# hsrp 20
N9K-Cloudian-Fab-A(config-if-hsrp)# preempt
N9K-Cloudian-Fab-A(config-if-hsrp)# priority 10
N9K-Cloudian-Fab-A(config-if-hsrp)# ip 192.168.20.1
N9K-Cloudian-Fab-A(config-if-hsrp)# exit
N9K-Cloudian-Fab-A(config-if)# exit
```

Switch B

```
N9K-Cloudian-Fab-B# config terminal
Enter configuration commands, one per line. End with CNTL/Z.
N9K-Cloudian-Fab-B(config)# vlan 10
N9K-Cloudian-Fab-B(config-vlan)# name Storage-Management
N9K-Cloudian-Fab-B(config-vlan)# no shut
```

```
N9K-Cloudian-Fab-B(config-vlan)# exit
N9K-Cloudian-Fab-B(config)# vlan 30
N9K-Cloudian-Fab-B(config-vlan)# name Storage-Cluster
N9K-Cloudian-Fab-B(config-vlan)# no shut
N9K-Cloudian-Fab-B(config-vlan)# exit
N9K-Cloudian-Fab-B(config)# vlan 220
N9K-Cloudian-Fab-B(config-vlan)# name External-Mgmt
N9K-Cloudian-Fab-B(config-vlan)# no shut
N9K-Cloudian-Fab-B(config-vlan)# exit
N9K-Cloudian-Fab-B(config)# vlan 20
N9K-Cloudian-Fab-B(config-vlan)# name Client-Network
N9K-Cloudian-Fab-B(config-vlan)# no shut
N9K-Cloudian-Fab-B(config-vlan)# exit
N9K-Cloudian-Fab-B(config)# interface vlan10
N9K-Cloudian-Fab-B(config-if)# description Storage-Mgmt
N9K-Cloudian-Fab-B(config-if)# no ip redirects
N9K-Cloudian-Fab-B(config-if)# ip address 192.168.10.254/24
N9K-Cloudian-Fab-B(config-if)# no ipv6 redirects
N9K-Cloudian-Fab-B(config-if)# hsrp version 2
N9K-Cloudian-Fab-B(config-if)# hsrp 10
N9K-Cloudian-Fab-B(config-if-hsrp)# preempt
N9K-Cloudian-Fab-B(config-if-hsrp)# priority 5
N9K-Cloudian-Fab-B(config-if-hsrp)# ip 192.168.10.1
N9K-Cloudian-Fab-B(config-if-hsrp)# exit
N9K-Cloudian-Fab-B(config-if)# exit

N9K-Cloudian-Fab-B(config)# interface vlan30
N9K-Cloudian-Fab-B(config-if)# description Storage-Cluster
N9K-Cloudian-Fab-B(config-if)# no ip redirects
N9K-Cloudian-Fab-B(config-if)# ip address 192.168.30.254/24
N9K-Cloudian-Fab-B(config-if)# no ipv6 redirects
N9K-Cloudian-Fab-B(config-if)# hsrp version 2
N9K-Cloudian-Fab-B(config-if)# hsrp 30
N9K-Cloudian-Fab-B(config-if-hsrp)# preempt
N9K-Cloudian-Fab-B(config-if-hsrp)# priority 5
N9K-Cloudian-Fab-B(config-if-hsrp)# ip 192.168.30.1
N9K-Cloudian-Fab-B(config-if-hsrp)# exit
N9K-Cloudian-Fab-B(config)# interface vlan20
N9K-Cloudian-Fab-B(config-if)# description Client-Network
N9K-Cloudian-Fab-B(config-if)# no ip redirects
N9K-Cloudian-Fab-B(config-if)# ip address 192.168.20.254/24
N9K-Cloudian-Fab-B(config-if)# no ipv6 redirects
N9K-Cloudian-Fab-B(config-if)# hsrp version 2
N9K-Cloudian-Fab-B(config-if)# hsrp 20
N9K-Cloudian-Fab-B(config-if-hsrp)# preempt
N9K-Cloudian-Fab-B(config-if-hsrp)# priority 5
N9K-Cloudian-Fab-B(config-if-hsrp)# ip 192.168.20.1
N9K-Cloudian-Fab-B(config-if-hsrp)# exit
N9K-Cloudian-Fab-B(config-if)# exit
N9K-Cloudian-Fab-B(config)# copy running-config startup-config
```

Configure vPC and Port Channels on Nexus C9336C-FX2 Switch A and B

To enable vPC and Port Channels on both Switch A and B, follow these steps:

vPC and Port Channels for Peerlink on Switch A

```
N9K-Cloudian-Fab-A(config)# vpc domain 2
N9K-Cloudian-Fab-A(config-vpc-domain)# peer-keepalive destination
173.36.220.14
N9K-Cloudian-Fab-A(config-vpc-domain)# peer-gateway
N9K-Cloudian-Fab-A(config-vpc-domain)# exit
N9K-Cloudian-Fab-A(config)# interface port-channel 1
N9K-Cloudian-Fab-A(config-if)# description vPC peerlink for N9K-Cloudian-Fab-A
and N9K-Cloudian-Fab-B
N9K-Cloudian-Fab-A(config-if)# switchport
N9K-Cloudian-Fab-A(config-if)# switchport mode trunk
N9K-Cloudian-Fab-A(config-if)# spanning-tree port type network
N9K-Cloudian-Fab-A(config-if)# speed 40000
N9K-Cloudian-Fab-A(config-if)# vpc peer-link
N9K-Cloudian-Fab-A(config-if)# exit
N9K-Cloudian-Fab-A(config)# interface ethernet 1/21
N9K-Cloudian-Fab-A(config-if)# description connected to peer N9K-Cloudian-Fab-B
port 21
N9K-Cloudian-Fab-A(config-if)# switchport
N9K-Cloudian-Fab-A(config-if)# switchport mode trunk
N9K-Cloudian-Fab-A(config-if)# speed 40000
N9K-Cloudian-Fab-A(config-if)# channel-group 1 mode active
N9K-Cloudian-Fab-A(config-if)# exit
N9K-Cloudian-Fab-A(config)# interface ethernet 1/22
N9K-Cloudian-Fab-A(config-if)# description connected to peer N9K-Cloudian-Fab-B
port 22
N9K-Cloudian-Fab-A(config-if)# switchport
N9K-Cloudian-Fab-A(config-if)# switchport mode trunk
N9K-Cloudian-Fab-A(config-if)# speed 40000
N9K-Cloudian-Fab-A(config-if)# channel-group 1 mode active
N9K-Cloudian-Fab-A(config-if)# exit
N9K-Cloudian-Fab-A(config)# interface ethernet 1/23
N9K-Cloudian-Fab-A(config-if)# description connected to peer N9K-Cloudian-Fab-B
port 23
N9K-Cloudian-Fab-A(config-if)# switchport
N9K-Cloudian-Fab-A(config-if)# switchport mode trunk
N9K-Cloudian-Fab-A(config-if)# speed 40000
N9K-Cloudian-Fab-A(config-if)# channel-group 1 mode active
N9K-Cloudian-Fab-A(config-if)# exit
N9K-Cloudian-Fab-A(config)# interface ethernet 1/24
N9K-Cloudian-Fab-A(config-if)# description connected to peer N9K-Cloudian-Fab-B
port 24
N9K-Cloudian-Fab-A(config-if)# switchport
N9K-Cloudian-Fab-A(config-if)# switchport mode trunk
N9K-Cloudian-Fab-A(config-if)# speed 40000
N9K-Cloudian-Fab-A(config-if)# channel-group 1 mode active
N9K-Cloudian-Fab-A(config-if)# exit
```

vPC and Port Channels for Peerlink on Switch B

```
N9K-Cloudian-Fab-B(config)# vpc domain 2
N9K-Cloudian-Fab-B(config-vpc-domain)# peer-keepalive destination
173.36.200.13
N9K-Cloudian-Fab-B(config-vpc-domain)# interface port-channel 1
N9K-Cloudian-Fab-B(config-if)# description vPC peerlink for N9K-Cloudian-Fab-A
and N9K-Cloudian-Fab-B
```

```
N9K-Cloudian-Fab-B(config-if)# switchport
N9K-Cloudian-Fab-B(config-if)# switchport mode trunk
N9K-Cloudian-Fab-B(config-if)# spanning-tree port type network
N9K-Cloudian-Fab-B(config-if)# speed 40000
N9K-Cloudian-Fab-B(config-if)# vpc peer-link
N9K-Cloudian-Fab-B(config-if)# exit
N9K-Cloudian-Fab-B(config)# interface ethernet 1/21
N9K-Cloudian-Fab-B(config-if)# description connected to peer N9K-Cloudian-Fab-A
port 21
N9K-Cloudian-Fab-B(config-if)# switchport
N9K-Cloudian-Fab-B(config-if)# switchport mode trunk
N9K-Cloudian-Fab-B(config-if)# speed 40000
N9K-Cloudian-Fab-B(config-if)# channel-group 1 mode active
N9K-Cloudian-Fab-B(config-if)# exit
N9K-Cloudian-Fab-B(config)# interface ethernet 1/22
N9K-Cloudian-Fab-B(config-if)# description connected to peer N9K-Cloudian-Fab-A
port 22
N9K-Cloudian-Fab-B(config-if)# switchport
N9K-Cloudian-Fab-B(config-if)# switchport mode trunk
N9K-Cloudian-Fab-B(config-if)# speed 40000
N9K-Cloudian-Fab-B(config-if)# channel-group 1 mode active
N9K-Cloudian-Fab-B(config-if)# exit
N9K-Cloudian-Fab-B(config)# interface ethernet 1/23
N9K-Cloudian-Fab-B(config-if)# description connected to peer N9K-Cloudian-Fab-A
port 23
N9K-Cloudian-Fab-B(config-if)# switchport
N9K-Cloudian-Fab-B(config-if)# switchport mode trunk
N9K-Cloudian-Fab-B(config-if)# speed 40000
N9K-Cloudian-Fab-B(config-if)# channel-group 1 mode active
N9K-Cloudian-Fab-B(config-if)# exit
N9K-Cloudian-Fab-B(config)# interface ethernet 1/24
N9K-Cloudian-Fab-B(config-if)# description connected to peer N9K-Cloudian-Fab-A
port 24
N9K-Cloudian-Fab-B(config-if)# switchport
N9K-Cloudian-Fab-B(config-if)# switchport mode trunk
N9K-Cloudian-Fab-B(config-if)# speed 40000
N9K-Cloudian-Fab-B(config-if)# channel-group 1 mode active
N9K-Cloudian-Fab-B(config-if)# exit
```

vPC and Port Channels for Uplink from UCS Fabric A & B on Nexus Switch A

```
N9K-Cloudian-Fab-A(config)# interface port-channel 25
N9K-Cloudian-Fab-A(config-if)# description vPC for UCS FI-A ports 1 to 2
N9K-Cloudian-Fab-A(config-if)# vpc 25
N9K-Cloudian-Fab-A(config-if)# switchport
N9K-Cloudian-Fab-A(config-if)# switchport mode trunk
N9K-Cloudian-Fab-A(config-if)# switchport trunk allowed vlan 10,20,30,220
N9K-Cloudian-Fab-A(config-if)# spanning-tree port type edge trunk
Edge port type (portfast) should only be enabled on ports connected to a single
 host. Connecting hubs, concentrators, switches, bridges, etc...  to this
 interface when edge port type (portfast) is enabled, can cause temporary
bridging loops.
 Use with CAUTION
N9K-Cloudian-Fab-A(config-if)# mtu 9216
N9K-Cloudian-Fab-A(config-if)# exit
```

```
N9K-Cloudian-Fab-A(config)# interface port-channel 26
N9K-Cloudian-Fab-A(config-if)# description vPC for UCS FI-B ports 3 to 4
N9K-Cloudian-Fab-A(config-if)# vpc 26
N9K-Cloudian-Fab-A(config-if)# switchport
N9K-Cloudian-Fab-A(config-if)# switchport mode trunk
N9K-Cloudian-Fab-A(config-if)# switchport trunk allowed vlan 10,20,30,220
N9K-Cloudian-Fab-A(config-if)# spanning-tree port type edge trunk
Edge port type (portfast) should only be enabled on ports connected to a single
 host. Connecting hubs, concentrators, switches, bridges, etc...  to this
 interface when edge port type (portfast) is enabled, can cause temporary
bridging loops.
 Use with CAUTION
N9K-Cloudian-Fab-A(config-if)# mtu 9216
N9K-Cloudian-Fab-A(config-if)# exit
N9K-Cloudian-Fab-A(config)# interface ethernet 1/1
N9K-Cloudian-Fab-A(config-if)# switchport
N9K-Cloudian-Fab-A(config-if)# switchport mode trunk
N9K-Cloudian-Fab-A(config-if)# description Uplink from UCS FI-A ports 1
N9K-Cloudian-Fab-A(config-if)# switchport trunk allowed vlan 10,20,30,220
N9K-Cloudian-Fab-A(config-if)# mtu 9216
N9K-Cloudian-Fab-A(config-if)# channel-group 25 mode active
N9K-Cloudian-Fab-A(config-if)# exit
N9K-Cloudian-Fab-A(config)# interface ethernet 1/2
N9K-Cloudian-Fab-A(config-if)# switchport
N9K-Cloudian-Fab-A(config-if)# switchport mode trunk
N9K-Cloudian-Fab-A(config-if)# description Uplink from UCS FI-A port 2
N9K-Cloudian-Fab-A(config-if)# mtu 9216
N9K-Cloudian-Fab-A(config-if)# switchport trunk allowed vlan 10,20,30,220
N9K-Cloudian-Fab-A(config-if)# channel-group 25 mode active
N9K-Cloudian-Fab-A(config-if)# exit
N9K-Cloudian-Fab-A(config)#
N9K-Cloudian-Fab-A(config)# interface ethernet 1/3
N9K-Cloudian-Fab-A(config-if)# switchport
N9K-Cloudian-Fab-A(config-if)# switchport mode trunk
N9K-Cloudian-Fab-A(config-if)# description Uplink from UCS FI-B port 3
N9K-Cloudian-Fab-A(config-if)# switchport trunk allowed vlan 10,20,30,220
N9K-Cloudian-Fab-A(config-if)# mtu 9216
N9K-Cloudian-Fab-A(config-if)# channel-group 26 mode active
N9K-Cloudian-Fab-A(config-if)# exit
N9K-Cloudian-Fab-A(config)# interface ethernet 1/4
N9K-Cloudian-Fab-A(config-if)# switchport
N9K-Cloudian-Fab-A(config-if)# switchport mode trunk
N9K-Cloudian-Fab-A(config-if)# description Uplink from UCS FI-B port 4
N9K-Cloudian-Fab-A(config-if)# switchport trunk allowed vlan 10,20,30,220
N9K-Cloudian-Fab-A(config-if)# mtu 9216
N9K-Cloudian-Fab-A(config-if)# channel-group 26 mode active
N9K-Cloudian-Fab-A(config-if)# exit
N9K-Cloudian-Fab-A(config)#
```

vPC and Port Channels for Uplink from Fabric A and B on Nexus Switch B

```
N9K-Cloudian-Fab-B(config-if)# vpc 26
N9K-Cloudian-Fab-B(config-if)# no vpc 26
N9K-Cloudian-Fab-B(config-if)# vpc 26
N9K-Cloudian-Fab-B(config-if)# mtu 9216
```

```
N9K-Cloudian-Fab-B(config-if)# exit
N9K-Cloudian-Fab-B(config)# interface ethernet 1/1
N9K-Cloudian-Fab-B(config-if)# switchport
N9K-Cloudian-Fab-B(config-if)# switchport mode trunk
N9K-Cloudian-Fab-B(config-if)# description Uplink from UCS FI-B port 1
N9K-Cloudian-Fab-B(config-if)# switchport trunk allowed vlan 10,20,30,220
N9K-Cloudian-Fab-B(config-if)# mtu 9216
N9K-Cloudian-Fab-B(config-if)# channel-group 26 mode active
N9K-Cloudian-Fab-B(config-if)# exit
N9K-Cloudian-Fab-B(config)# interface ethernet 1/2
N9K-Cloudian-Fab-B(config-if)# switchport
N9K-Cloudian-Fab-B(config-if)# switchport mode trunk
N9K-Cloudian-Fab-B(config-if)# switchport trunk allowed vlan 10,20,30,220
N9K-Cloudian-Fab-B(config-if)# mtu 9216
N9K-Cloudian-Fab-B(config-if)# channel-group 26 mode active
N9K-Cloudian-Fab-B(config-if)# exit
N9K-Cloudian-Fab-B(config)# interface ethernet 1/3
N9K-Cloudian-Fab-B(config-if)# switchport
N9K-Cloudian-Fab-B(config-if)# switchport mode trunk
N9K-Cloudian-Fab-B(config-if)# switchport trunk allowed vlan 10,20,30,220
N9K-Cloudian-Fab-B(config-if)# mtu 9216
N9K-Cloudian-Fab-B(config-if)# channel-group 25 mode active
N9K-Cloudian-Fab-B(config-if)# exit
N9K-Cloudian-Fab-B(config)# interface ethernet 1/4
N9K-Cloudian-Fab-B(config-if)# switchport
N9K-Cloudian-Fab-B(config-if)# switchport mode trunk
N9K-Cloudian-Fab-B(config-if)# switchport trunk allowed vlan 10,20,30,220
N9K-Cloudian-Fab-B(config-if)# mtu 9216
N9K-Cloudian-Fab-B(config-if)# channel-group 25 mode active
N9K-Cloudian-Fab-B(config-if)# exit
N9K-Cloudian-Fab-B(config)#
```

Verification Check of Nexus C9336C-FX2 Configuration for Switch A and B

Switch A

```
N9K-Cloudian-Fab-A# show vpc brief
Legend:
                (*) - local vPC is down, forwarding via vPC peer-link

vPC domain id                     : 2
Peer status                       : peer adjacency formed ok
vPC keep-alive status             : peer is alive
Configuration consistency status  : success
Per-vlan consistency status       : success
Type-2 consistency status         : success
vPC role                          : primary
Number of vPCs configured         : 2
Peer Gateway                      : Enabled
Dual-active excluded VLANs        : -
Graceful Consistency Check        : Enabled
Auto-recovery status              : Disabled
Delay-restore status              : Timer is off.(timeout = 30s)
Delay-restore SVI status          : Timer is off.(timeout = 10s)
Operational Layer3 Peer-router    : Disabled
Virtual-peerlink mode             : Disabled
```

```
vPC Peer-link status
---------------------------------------------------------------------
id    Port   Status Active vlans
--    ----   ------ --------------------------------------------------
1     Po2    up     1,10,20,30,220

vPC status
----------------------------------------------------------------------------
Id    Port          Status Consistency Reason                Active vlans
--    ------------  ------ ----------- ------                --------------
50    Po50          up     success     success               1,10,20,30,220

51    Po51          up     success     success               1,10,20,30,220




N9K-Cloudian-Fab-A# show port-channel summary
Flags:  D - Down        P - Up in port-channel (members)
        I - Individual  H - Hot-standby (LACP only)
        s - Suspended   r - Module-removed
        b - BFD Session Wait
        S - Switched    R - Routed
        U - Up (port-channel)
        p - Up in delay-lacp mode (member)
        M - Not in use. Min-links not met
------------------------------------------------------------------------------
Group Port-       Type     Protocol  Member Ports
      Channel
------------------------------------------------------------------------------
2     Po2(SU)     Eth      LACP      Eth1/21(P)   Eth1/22(P)   Eth1/23(P)
                                     Eth1/24(P)
50    Po50(SU)    Eth      LACP      Eth1/1(P)    Eth1/2(P)
51    Po51(SU)    Eth      LACP      Eth1/3(P)    Eth1/4(P)
N9K-Cloudian-Fab-A#
```

Switch B

```
N9K-Cloudian-Fab-B# show vpc brief
Legend:
                (*) - local vPC is down, forwarding via vPC peer-link

vPC domain id                     : 2
Peer status                       : peer adjacency formed ok
vPC keep-alive status             : peer is alive
Configuration consistency status  : success
Per-vlan consistency status       : success
Type-2 consistency status         : success
vPC role                          : secondary
Number of vPCs configured         : 2
Peer Gateway                      : Enabled
Dual-active excluded VLANs        : -
Graceful Consistency Check        : Enabled
Auto-recovery status              : Disabled
Delay-restore status              : Timer is off.(timeout = 30s)
Delay-restore SVI status          : Timer is off.(timeout = 10s)
```

```
Operational Layer3 Peer-router    : Disabled
Virtual-peerlink mode             : Disabled

vPC Peer-link status
---------------------------------------------------------------------
id    Port   Status Active vlans
--    ----   ------ --------------------------------------------------
1     Po2    up     1,10,20,30,220

vPC status
----------------------------------------------------------------------------
Id    Port          Status Consistency Reason                Active vlans
--    ------------  ------ ----------- ------                ---------------
50    Po50          up     success     success               1,10,20,30,220

51    Po51          up     success     success               1,10,20,30,220
```

```
N9K-Cloudian-Fab-B# show port-channel summary
Flags:  D - Down        P - Up in port-channel (members)
        I - Individual  H - Hot-standby (LACP only)
        s - Suspended   r - Module-removed
        b - BFD Session Wait
        S - Switched    R - Routed
        U - Up (port-channel)
        p - Up in delay-lacp mode (member)
        M - Not in use. Min-links not met
------------------------------------------------------------------------------
Group Port-       Type     Protocol  Member Ports
      Channel
------------------------------------------------------------------------------
2     Po2(SU)     Eth      LACP      Eth1/21(P)   Eth1/22(P)   Eth1/23(P)
                                     Eth1/24(P)
50    Po50(SU)    Eth      LACP      Eth1/3(P)    Eth1/4(P)
```

Fabric Interconnect Configuration

This section provides the details to configure a fully redundant, highly available Cisco UCS 6332 fabric configuration:

- Initial setup of the Fabric Interconnect A and B
- Connect to Cisco UCS Manager using virtual IP address of using the web browser
- Launch Cisco UCS Manager
- Enable server and uplink ports
- Start discovery process
- Create pools and policies for service profile template
- Create chassis and storage profiles
- Create Service Profile templates and appropriate Service Profiles

- Associate Service Profiles to servers

Initial Setup of Cisco UCS 6332 Fabric Interconnects

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

Configure Fabric Interconnect A

To configure Fabric A, follow these steps:

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

Example Setup for Fabric Interconnect A

```
                ---- Basic System Configuration Dialog ----

  This setup utility will guide you through the basic configuration of
  the system. Only minimal configuration including IP connectivity to
  the Fabric interconnect and its clustering mode is performed through these
steps.

  Type Ctrl-C at any time to abort configuration and reboot system.
  To back track or make modifications to already entered values,
  complete input till end of section and answer no when prompted
  to apply configuration.

  Enter the configuration method. (console/gui) ? console
  Enter the setup mode; setup newly or restore from backup. (setup/restore) ?
setup
  You have chosen to setup a new Fabric interconnect. Continue? (y/n): y
  Enforce strong password? (y/n) [y]: n
  Enter the password for admin:
  Confirm the password for admin:
  Is this Fabric interconnect part of a cluster(select 'no' for standalone)?
(yes/no) [n]: yes
  Enter the switch fabric (A/B): A
  Enter the system name:  CLOUDIAN-FI-6332
  Physical Switch Mgmt0 IP address : 173.36.220.15
  Physical Switch Mgmt0 IPv4 netmask : 255.255.255.0
  IPv4 address of the default gateway : 173.36.220.1
  Cluster IPv4 address : 173.36.220.17
  Configure the DNS Server IP address? (yes/no) [n]: no
  Configure the default domain name? (yes/no) [n]: no
  Join centralized management environment (UCS Central)? (yes/no) [n]: no

  Following configurations will be applied:
```

```
    Switch Fabric=A

    System Name= CLOUDIAN-FI-6332

    Enforced Strong Password=no

    Physical Switch Mgmt0 IP Address=173.36.220.15

    Physical Switch Mgmt0 IP Netmask=255.255.255.0

    Default Gateway=173.36.220.1

    Ipv6 value=0


    Cluster Enabled=yes

    Cluster IP Address=173.36.220.17

    NOTE: Cluster IP will be configured only after both Fabric Interconnects are
initialized.

          UCSM will be functional only after peer FI is configured in clustering
mode.


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

  Applying configuration. Please wait.

 Configuration file - Ok


Cisco UCS 6300 Series Fabric Interconnect

CLOUDIAN-FI-6332-A login:
```

Configure Fabric Interconnect B

To configure Fabric Interconnect B, follow these steps:

1. Connect to the console port on the second Cisco UCS 6332 Fabric Interconnect.
2. When prompted to enter the configuration method, enter **console** to continue.
3. The installer detects the presence of the partner Fabric Interconnect and adds this fabric interconnect to the cluster. Enter **y** to continue the installation.
4. Enter the admin password that was configured for the first Fabric Interconnect.
5. Enter the Mgmt0 IPv4 address.
6. Answer **yes** to save the configuration.

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

Example Setup for Fabric Interconnect B

```
                ---- Basic System Configuration Dialog ----


  This setup utility will guide you through the basic configuration of

  the system. Only minimal configuration including IP connectivity to

  the Fabric interconnect and its clustering mode is performed through these
steps.


  Type Ctrl-C at any time to abort configuration and reboot system.

  To back track or make modifications to already entered values,

  complete input till end of section and answer no when prompted

  to apply configuration.


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


  Installer has detected the presence of a peer Fabric interconnect. This Fabric
interconnect will 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 IPv4 Address: 173.36.220.15

    Peer Fabric interconnect Mgmt0 IPv4 Netmask: 255.255.255.0

    Cluster IPv4 address          : 173.36.220.17


    Peer FI is IPv4 Cluster enabled. Please Provide Local Fabric Interconnect
Mgmt0 IPv4 Address

  Physical Switch Mgmt0 IP address : 173.36.220.16


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

  Applying configuration. Please wait.

  Configuration file - Ok
```

```
Cisco UCS 6300 Series Fabric Interconnect

CLOUDIAN-FI-6332-B login:
```

Log into Cisco UCS Manager

To log into Cisco UCS Manager, follow these steps:

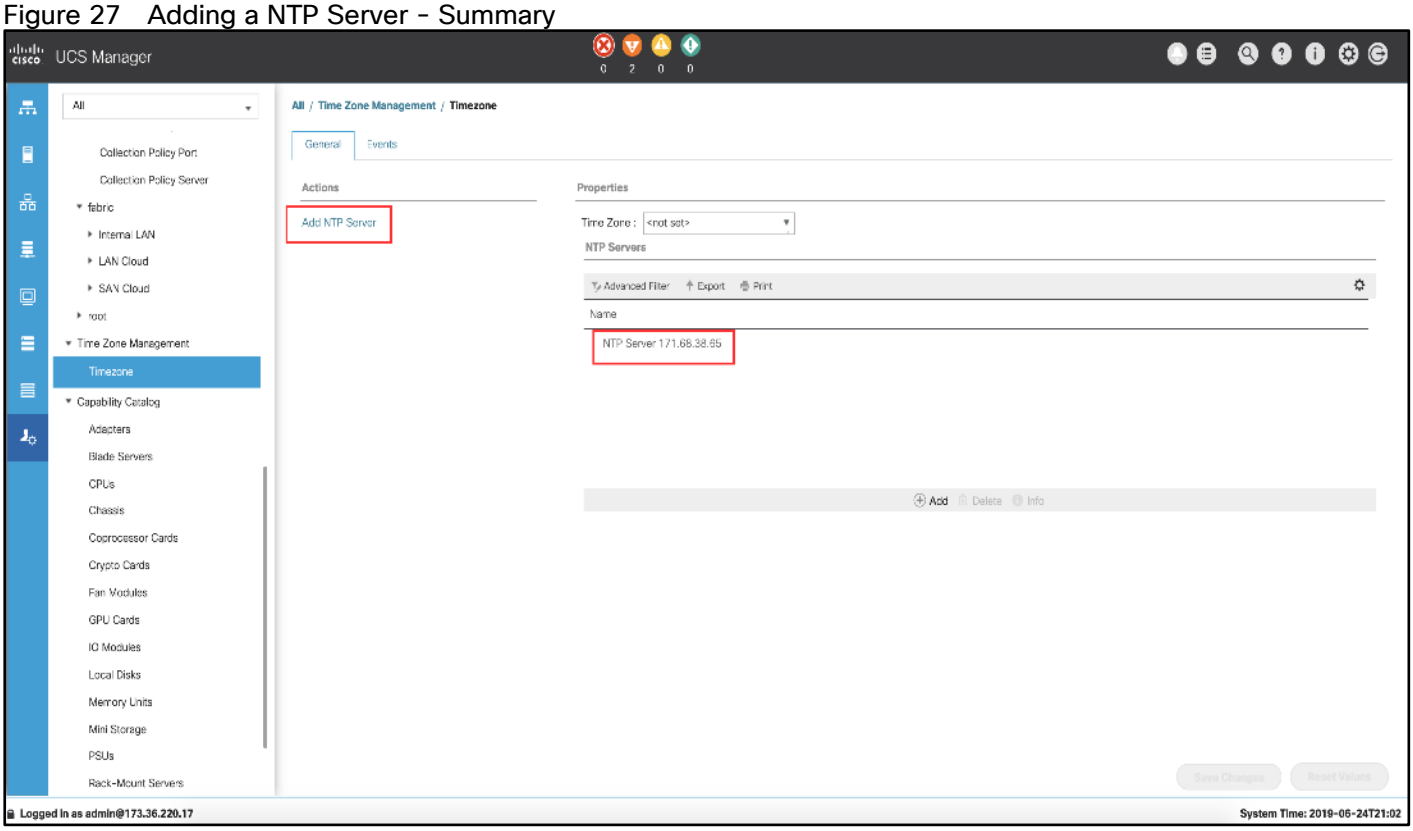
1. Open a Web browser and navigate to the Cisco UCS 6332 Fabric Interconnect cluster address.
2. Click the Launch link to download the Cisco UCS Manager software.
3. If prompted to accept security certificates, accept as necessary.
4. Click Launch UCS Manager HTML.
5. When prompted, enter admin for the username and enter the administrative password.
6. Click Login to log in to the Cisco UCS Manager.

Configure NTP Server

To configure the NTP server for the Cisco UCS environment, follow these steps:

1. Select Admin tab.
2. Select Time Zone Management.
3. Select Time Zone.
4. Under Properties select your time zone.
5. Select Add NTP Server.
6. Enter the IP address/DNS name of the NTP server.
7. Select OK.

Figure 27 Adding a NTP Server – Summary



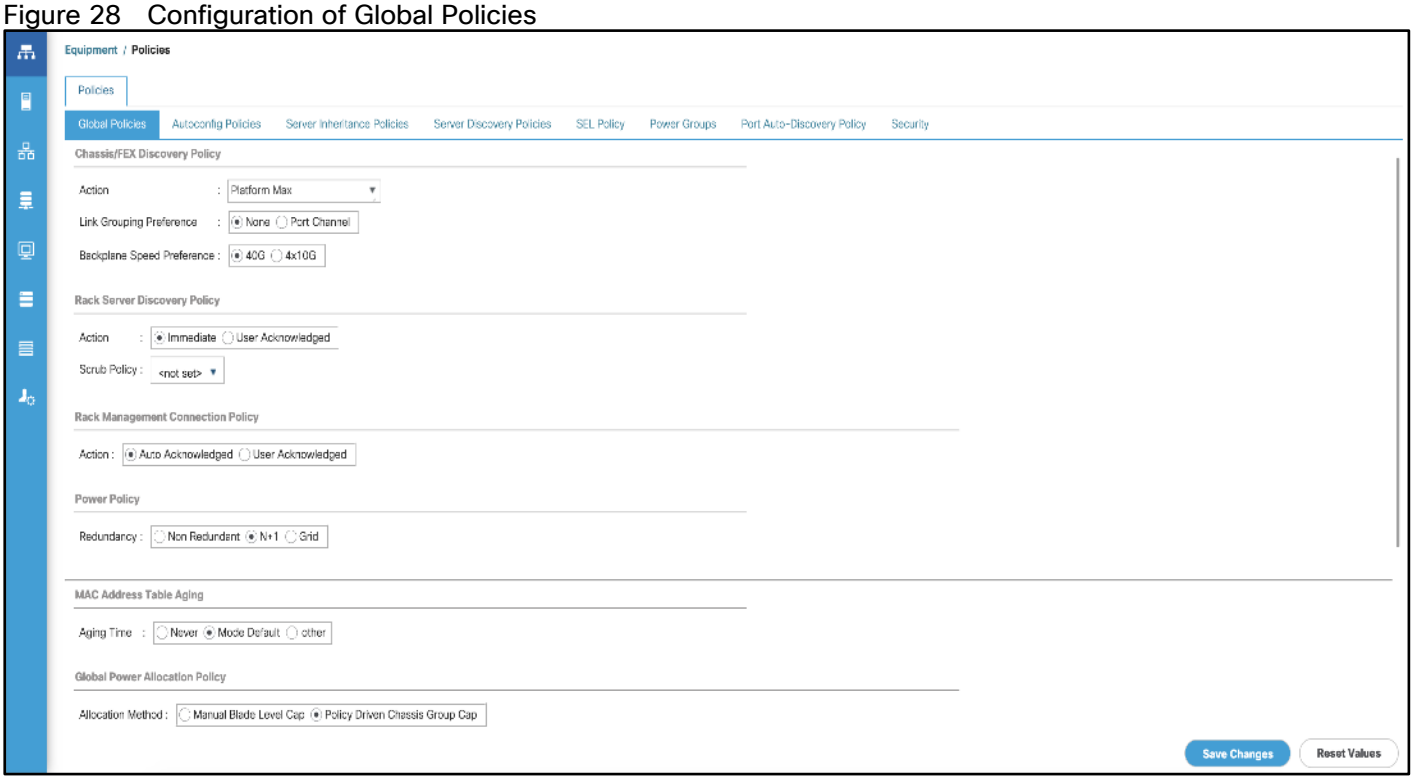
Initial Base Setup of the Environment

Configure Global Policies

To configure the global policies, follow these steps:

1. Select the `Equipment` tab of the window.
2. Select `Policies` on the right site.
3. Select Global Policies.
4. Under Chassis/FEX Discovery Policy select `Platform Max` under Action.
5. Select `40G` under Backplane Speed Preference.
6. Under Rack Server Discovery Policy select `Immediate` under Action.
7. Under Rack Management Connection Policy select `Auto Acknowledged` under Action.
8. Under Power Policy select `Redundancy N+1`.
9. Under Global Power Allocation Policy select `Policy Driven Chassis Group Cap`.
10. Select Save Changes.

Figure 28 Configuration of Global Policies

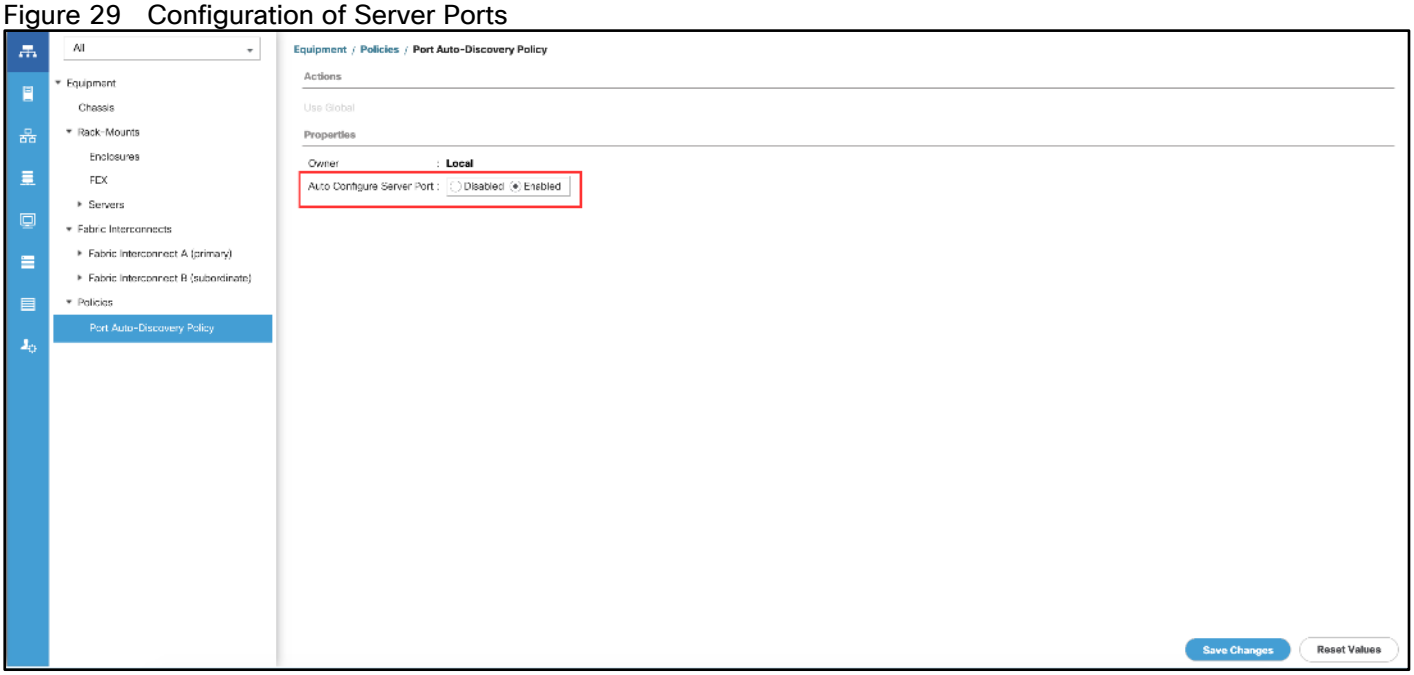


Enable Fabric Interconnect Server Ports

To enable server ports, follow these steps:

1. Select the **Equipment** tab.
2. Select Equipment > Policies > Port-Auto Discovery Policy
3. Click **Enabled** Under Properties
4. Click **Save Changes** to Configure Server Ports Automatically for FI-A and FI-B.

Figure 29 Configuration of Server Ports

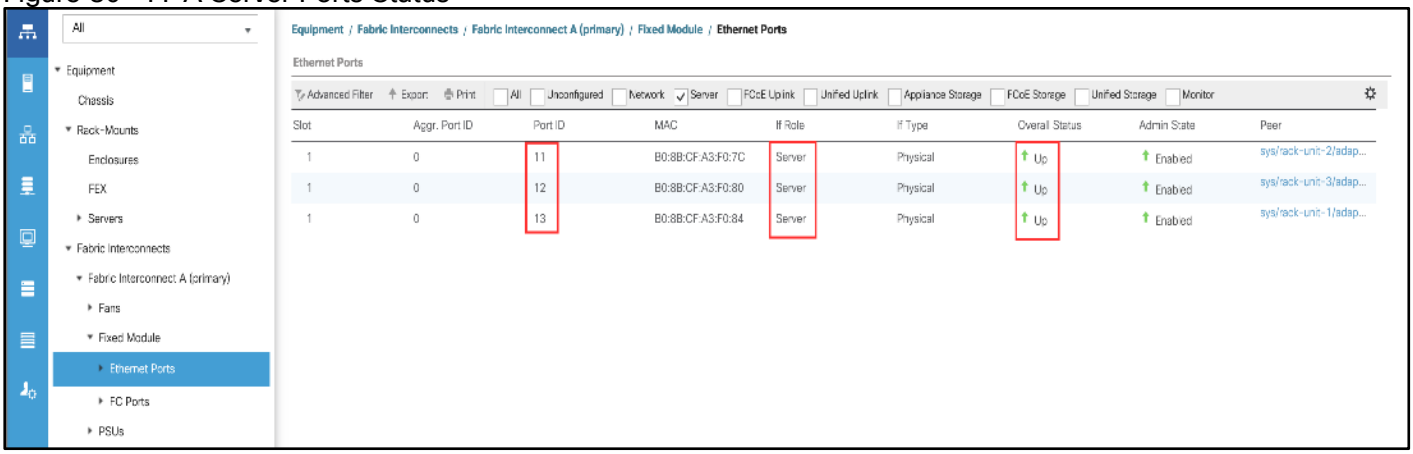


5. Verify the ports Server port on Fabric Interconnect A.

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

7. Click **Ethernet Ports** section.

Figure 30 FI-A Server Ports Status

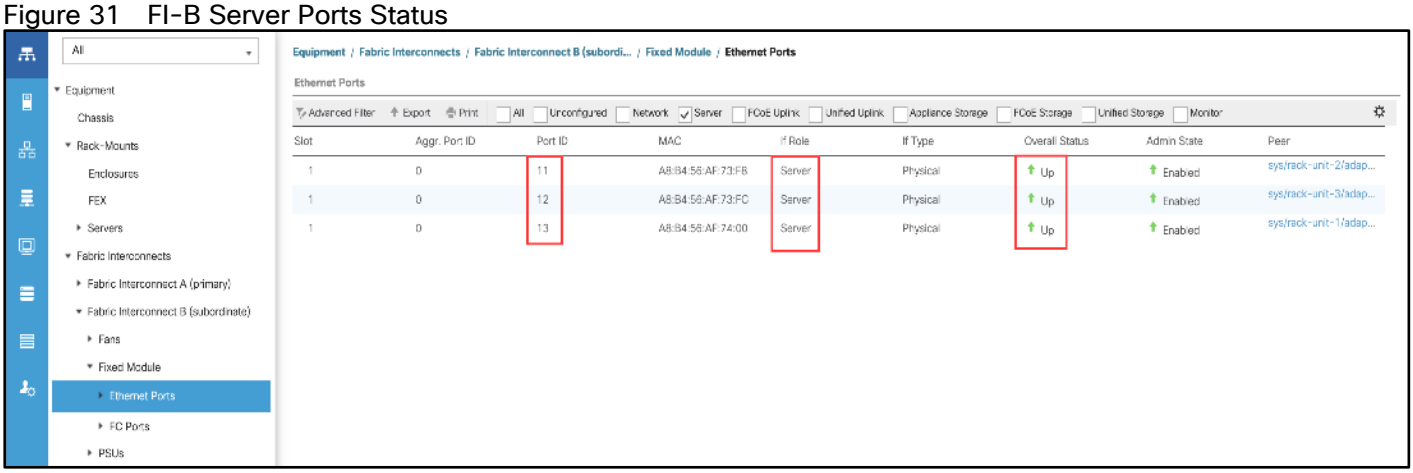


8. Verify the ports Server port on Fabric Interconnect A.

9. Select Equipment > Fabric Interconnects > Fabric Interconnect B (subordinate) > Fixed Module.

10. Click **Ethernet Ports** section.

Figure 31 FI-B Server Ports Status

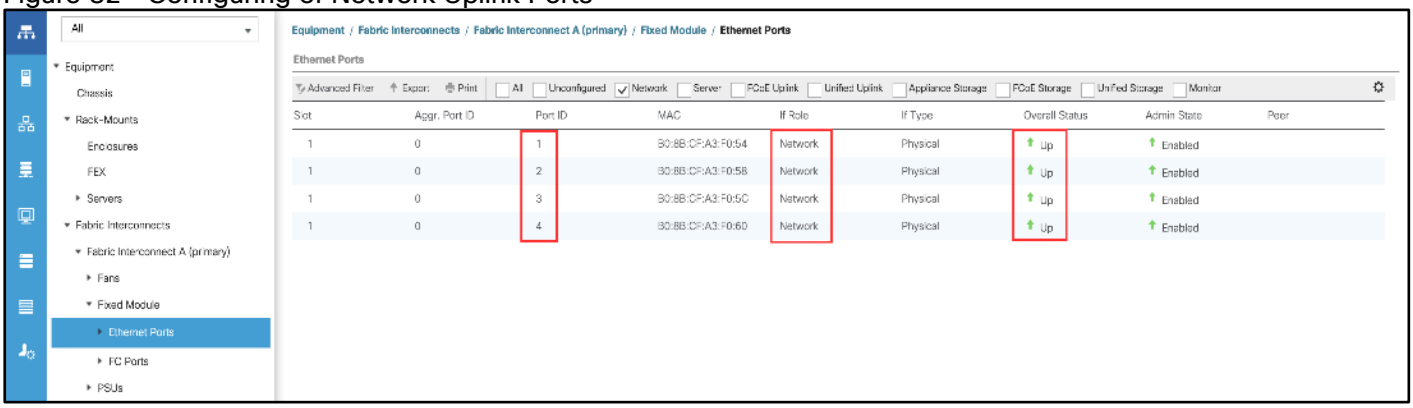


Enable Fabric Interconnect A Ports for Uplinks

To enable uplink ports, follow these steps:

1. Select the **Equipment** tab.
2. Select Equipment > Fabric Interconnects > Fabric Interconnect A (primary) > Fixed Module.
3. Click **Ethernet Ports** section.
4. Select Ports 1-4, right-click and then select **Configure as Uplink Port**.
5. Click **Yes** and then **OK**.
6. Repeat steps 1-5 for Fabric Interconnect B.

Figure 32 Configuring of Network Uplink Ports



Label Servers for Identification

For better identification, label each server by following these steps:

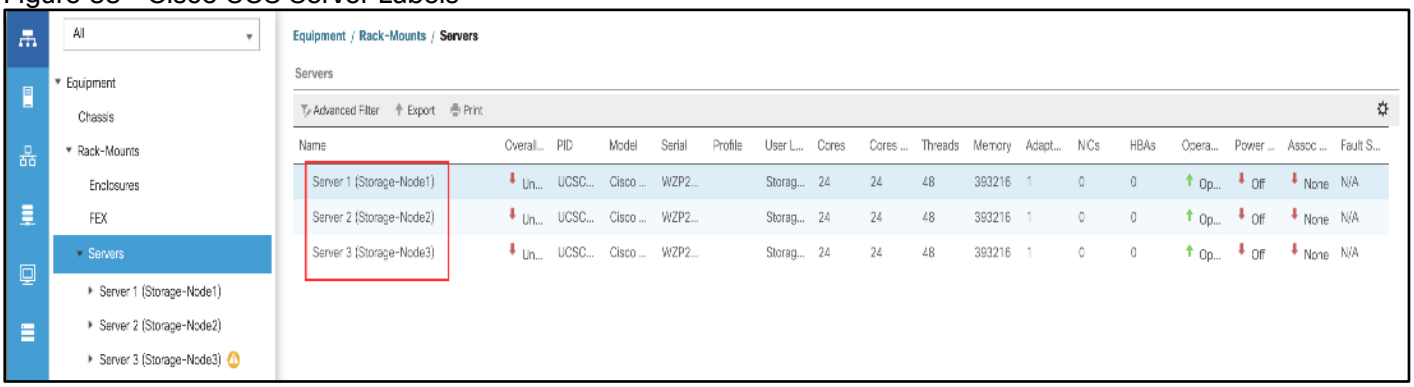
1. Select the **Equipment** tab.
2. Select Rack-Mounts > Servers > Server 1.

3. In the **Properties** section on the right go to **User Label** and add **Storage-Node1** to the field.
4. Repeat steps 1-3 for **Server 2** and **Server 3** according to Table 6 .
5. Click Save Changes.

Table 6 Labeling Servers

Server	Name
Server 1	Storage-Node1
Server 2	Storage-Node2
Server 3	Storage-Node3

Figure 33 Cisco UCS Server Labels

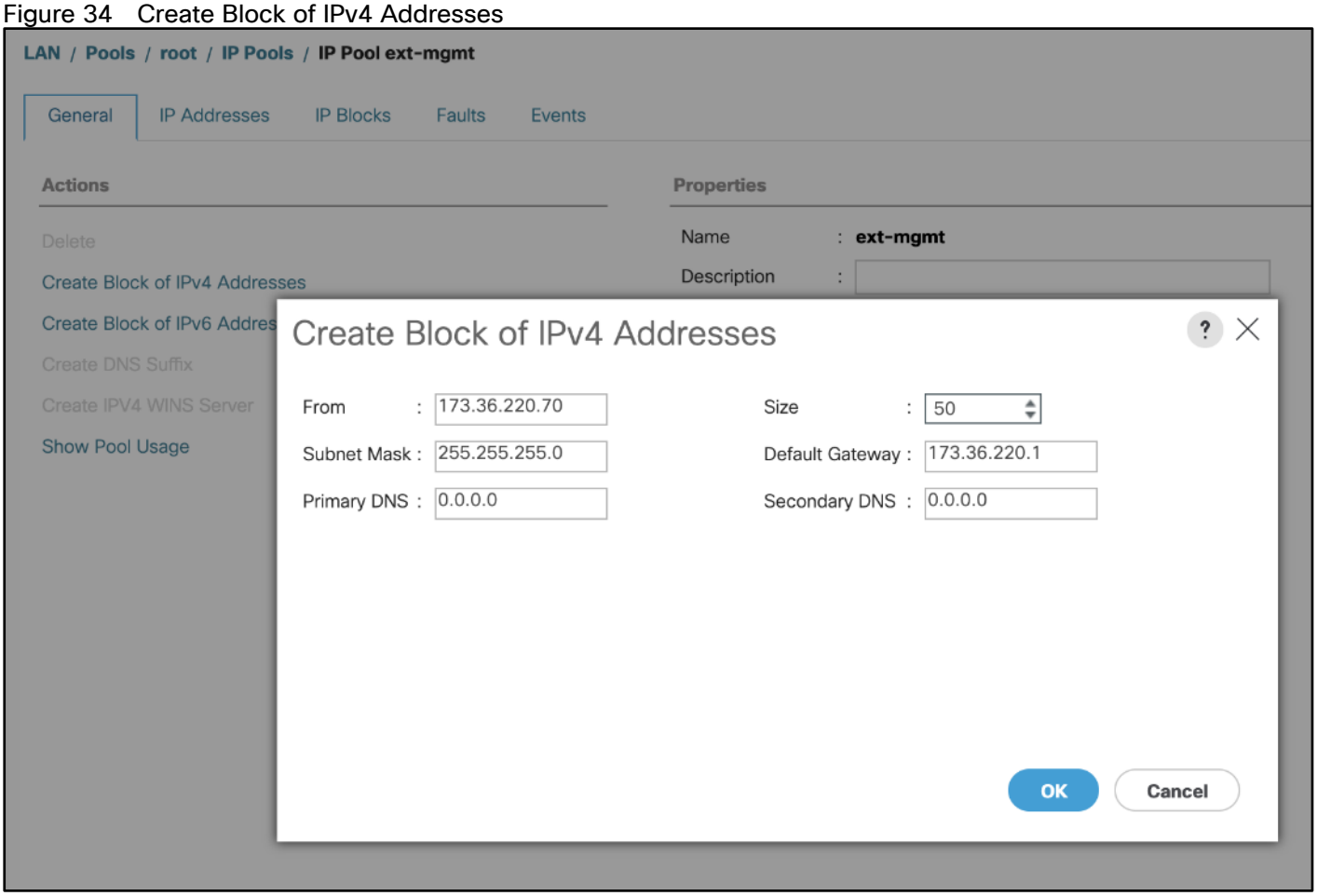


Create KVM IP Pool

To create a KVM IP Pool, follow these steps:

1. Select the **LAN** tab.
2. Go to LAN > Pools > root > IP Pools > IP Pool ext-mgmt.
3. Click on Create Block of IPv4 Addresses.
4. Enter an IP Address in the **From** field.
5. Enter **Size** 50.
6. Enter your `Subnet Mask`.
7. Fill in your `Default Gateway`.
8. Enter your **Primary DNS** and **Secondary DNS** if needed.
9. Click `OK`.

Figure 34 Create Block of IPv4 Addresses

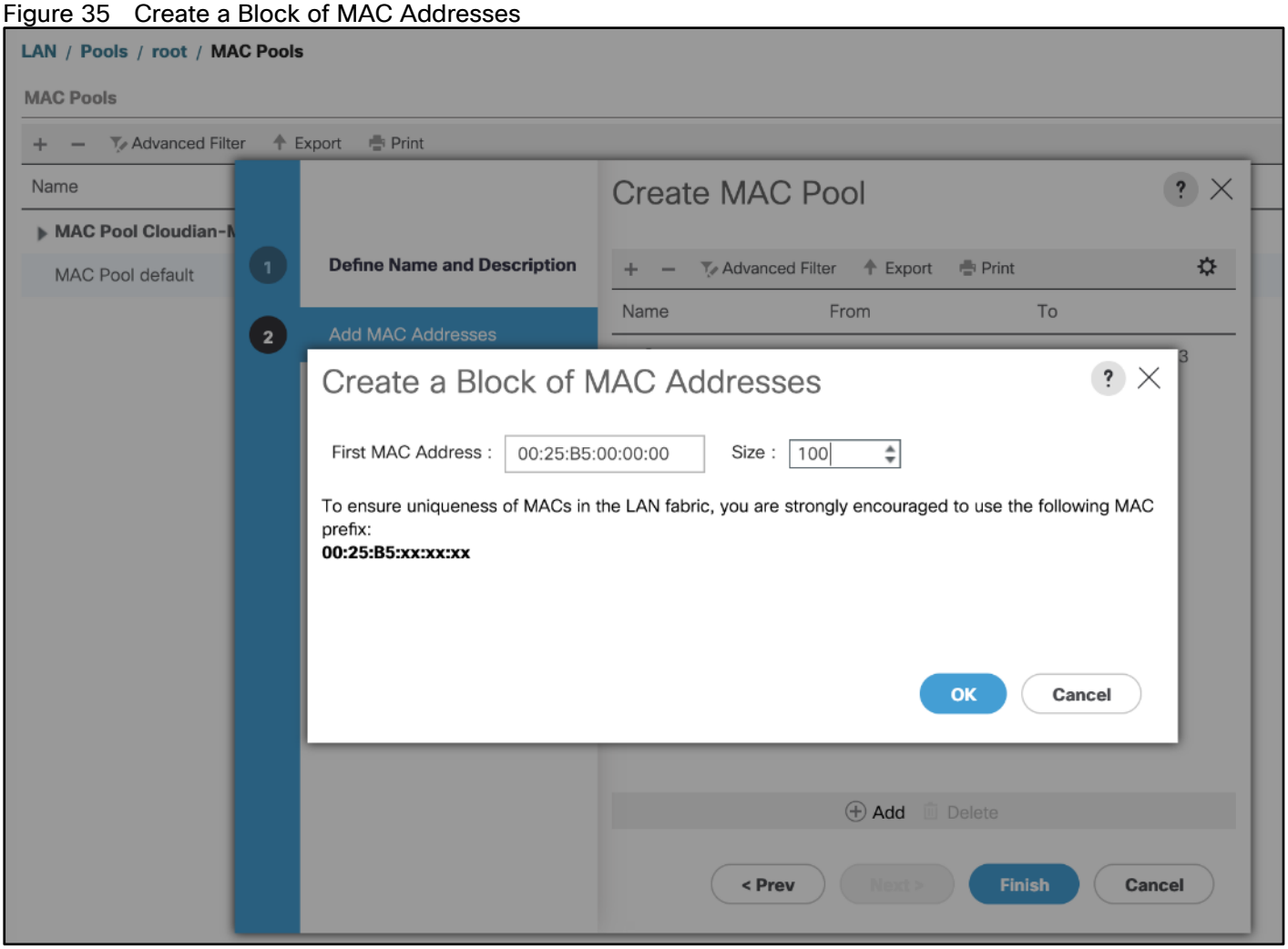


MAC Pool

To create a MAC Pool, follow these steps:

1. Select the **LAN** tab.
2. Go to LAN > Pools > root > Mac Pools and right-click Create MAC Pool.
3. Type in Cloudian-MAC-Pools for Name.
4. (Optional) Enter a **Description** of the MAC Pool.
5. Set Assignment Order as Sequential.
6. Click **Next**.
7. Click **Add**.
8. Specify a starting MAC address.
9. Specify a size of the MAC address pool, which is sufficient to support the available server resources, for example, 100.

Figure 35 Create a Block of MAC Addresses



10. Click **OK**.

11. Click **Finish**.

Create UUID Pool

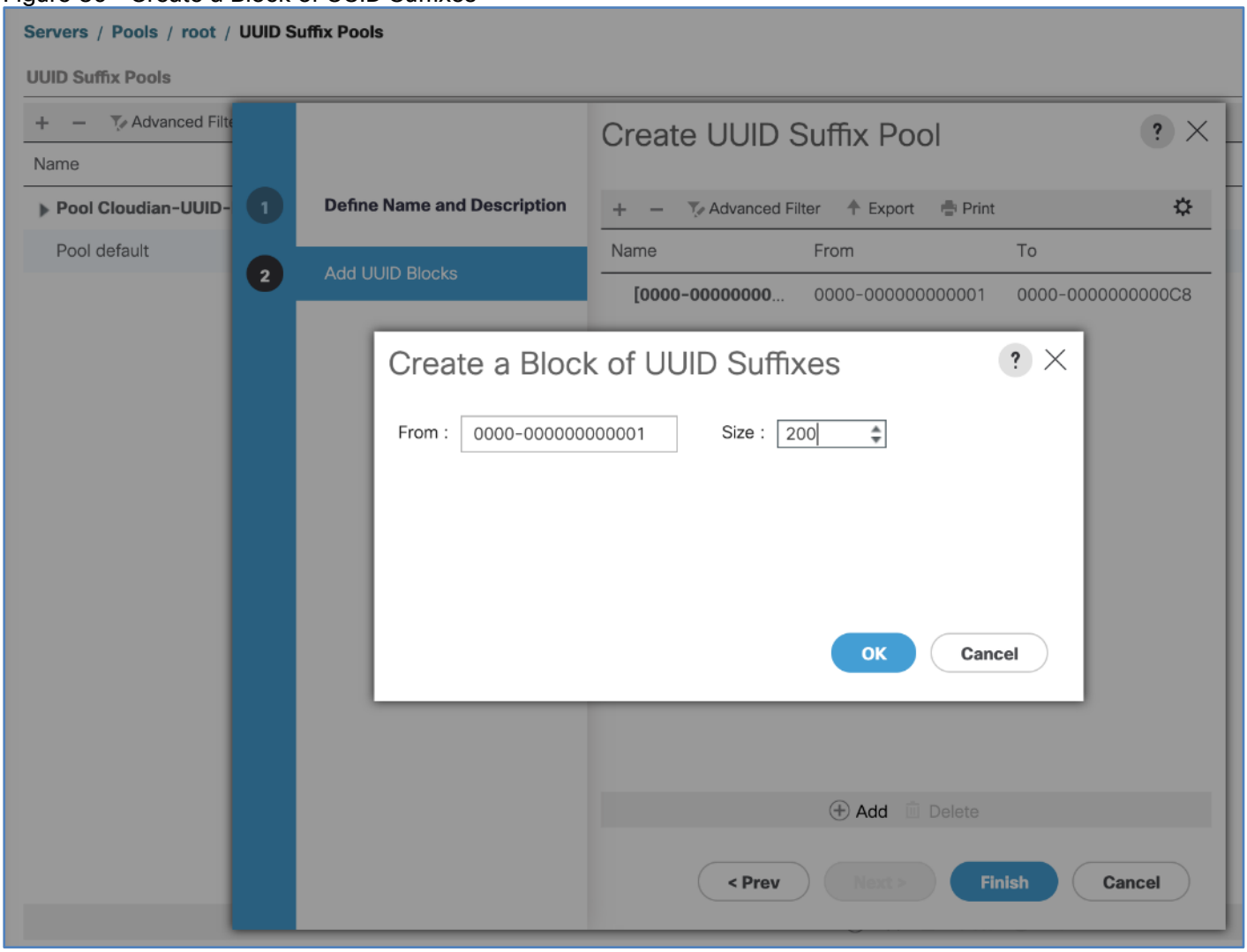
To create a UUID Pool, follow these steps:

1. Select the **Servers** tab.
2. Go to Servers > Pools > root > UUID Suffix Pools and right-click Create UUID Suffix Pool.
3. Type in `Cloudian-UUID-Pools` for Name.
4. (Optional) Enter a **Description** of the MAC Pool.
5. Set Assignment Order to Sequential and click Next.
6. Click **Add**.

7. Specify a starting UUID Suffix.

8. Specify a size of the UUID suffix pool, which is sufficient to support the available server resources, for example, 50.

Figure 36 Create a Block of UUID Suffixes



9. Click **OK**.

10. Click **Finish** and then click **OK**.

Create VLANs

As mentioned previously, it is important to separate the network traffic with VLANs for Storage-Management traffic and Storage-Cluster traffic, External traffic and Client traffic (Optional). Table 7 lists the configured VLANs.

Client traffic is optional. We used Client traffic to validate the functionality of S3 connectors.

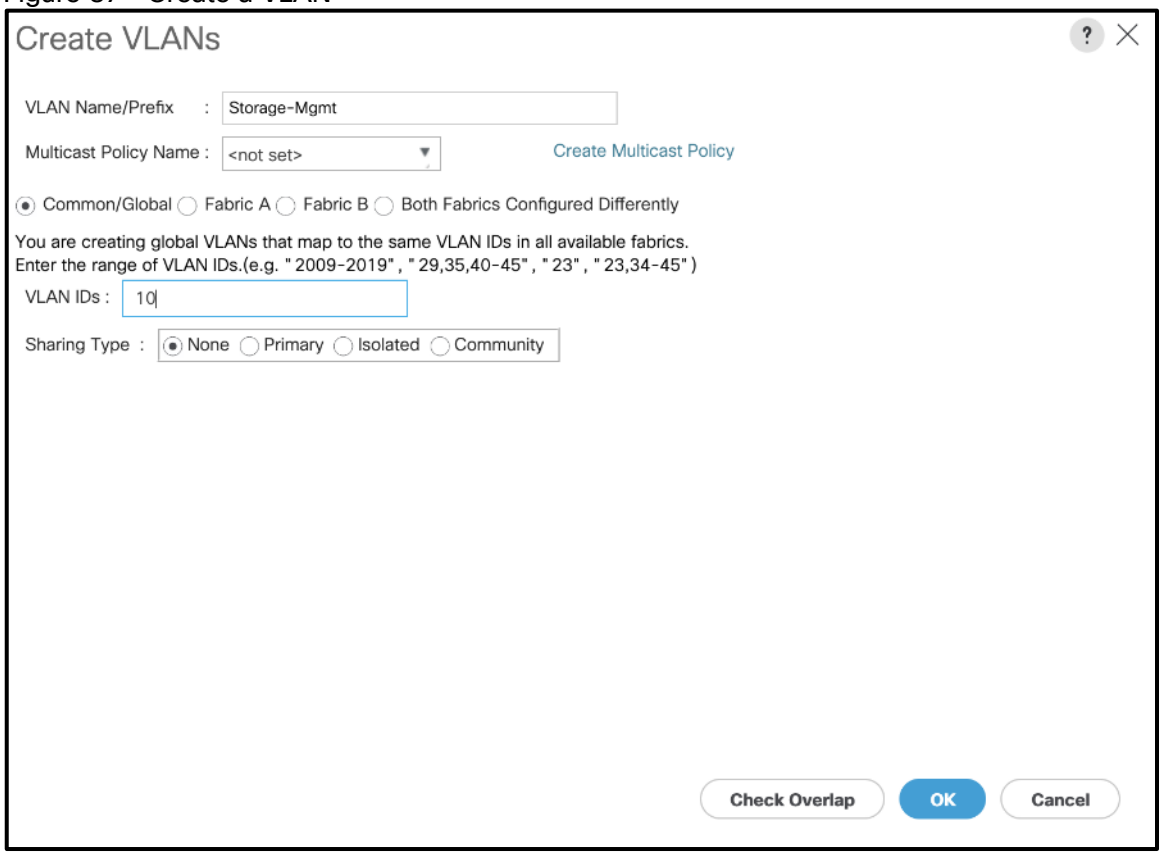
Table 7 VLAN Configurations

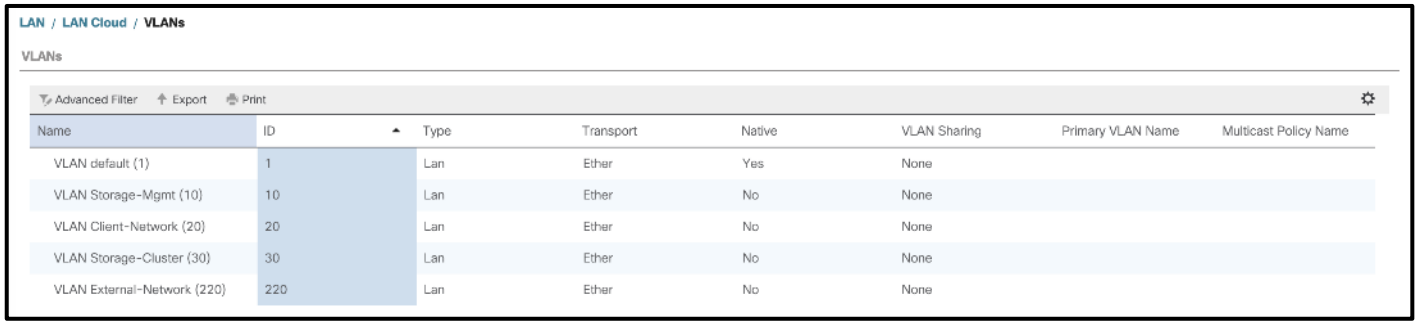
VLAN	Name	Function
10	Storage-Management	Storage Management traffic for Storage Nodes
20	Client-Network (optional)	Client traffic for Storage Nodes
30	Storage-Cluster	Storage Cluster traffic and Storage Nodes
220	External-Network	External Public Network for all UCS Servers

To configure VLANs in the Cisco UCS Manager GUI, follow these steps:

1. Select **LAN** in the UCSM GUI.
2. Select LAN > LAN Cloud > VLANs and right-click Create VLANs.
3. Enter Storage-Mgmt for the VLAN Name.
4. Keep Multicast Policy Name as <not set>.
5. Select **Common/Global** for Public.
6. Enter 10 in the **VLAN IDs** field.
7. Click **OK** and then click **Finish**.

Figure 37 Create a VLAN





8. Repeat steps 1-7 for rest of the VLANs Storage-Cluster External-Network and Client-Network.

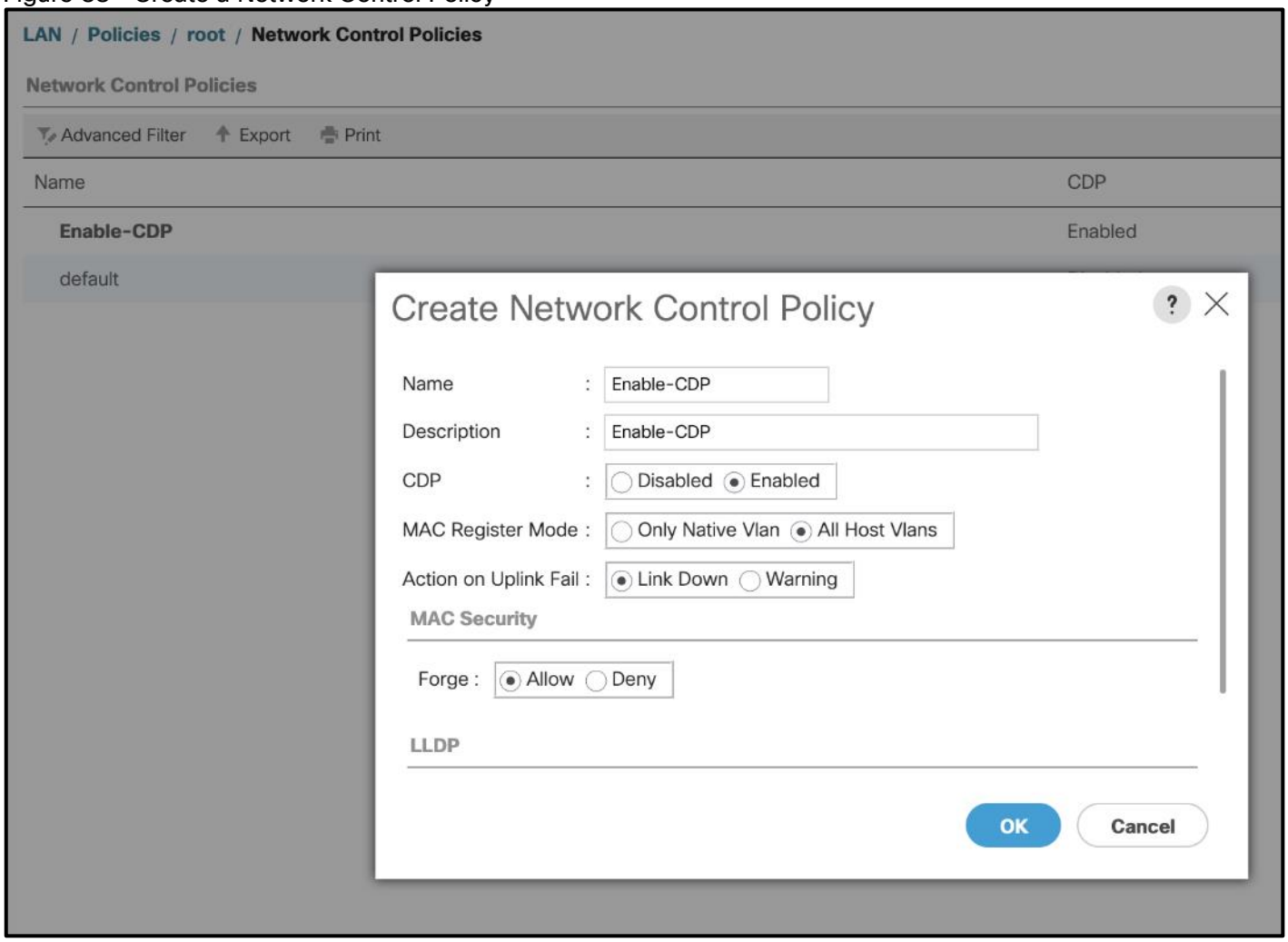
Enable CDP

To enable Network Control Policies, follow these steps:

1. Select the **LAN** tab in the Cisco UCS Manager GUI.
2. Go to LAN > Policies > root > Network Control Policies and right-click Create Network-Control Policy.
3. Type in **Enable-CDP** in the **Name** field.
4. (Optional) Enter a description in the **Description** field.

5. Click **Enabled** under **CDP**.
6. Click All Hosts VLANs under MAC Register Mode.
7. Leave everything else untouched and click **OK**.
8. Click **OK**.

Figure 38 Create a Network Control Policy



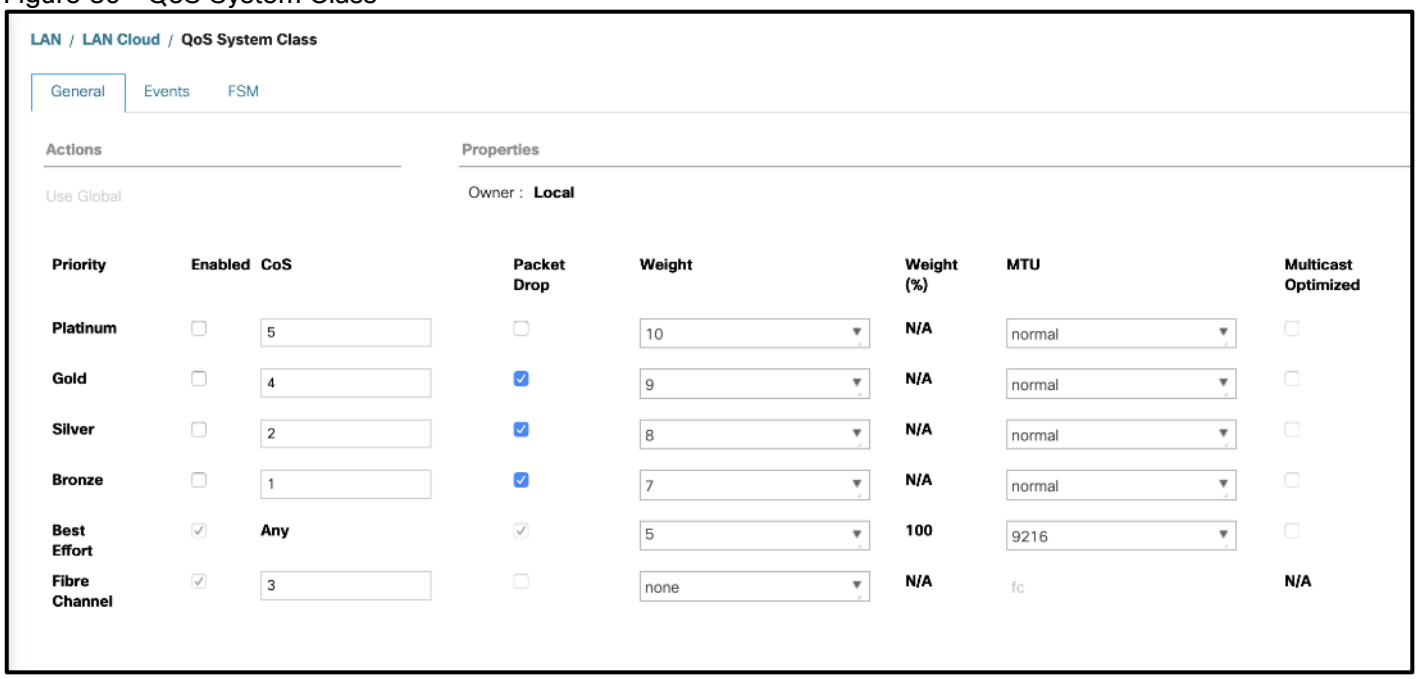
QoS System Class

To create a Quality of Service System Class, follow these steps:

1. Select the **LAN** tab in the Cisco UCS Manager GUI.
2. Go to LAN > LAN Cloud > QoS System Class.
3. Best Effort MTU as 9216.
4. Set Fibre Channel Weight to None.

5. Click **Save Changes** and then click **OK**.

Figure 39 QoS System Class



vNIC Template Setup

Based on the previous section of creating VLANs, the next step is to create the appropriate vNIC templates. For Cloudian Storage we need to create four different vNICs, depending on the role of the server. Table 8 provides an overview of the configuration.

Table 8 vNIC Table

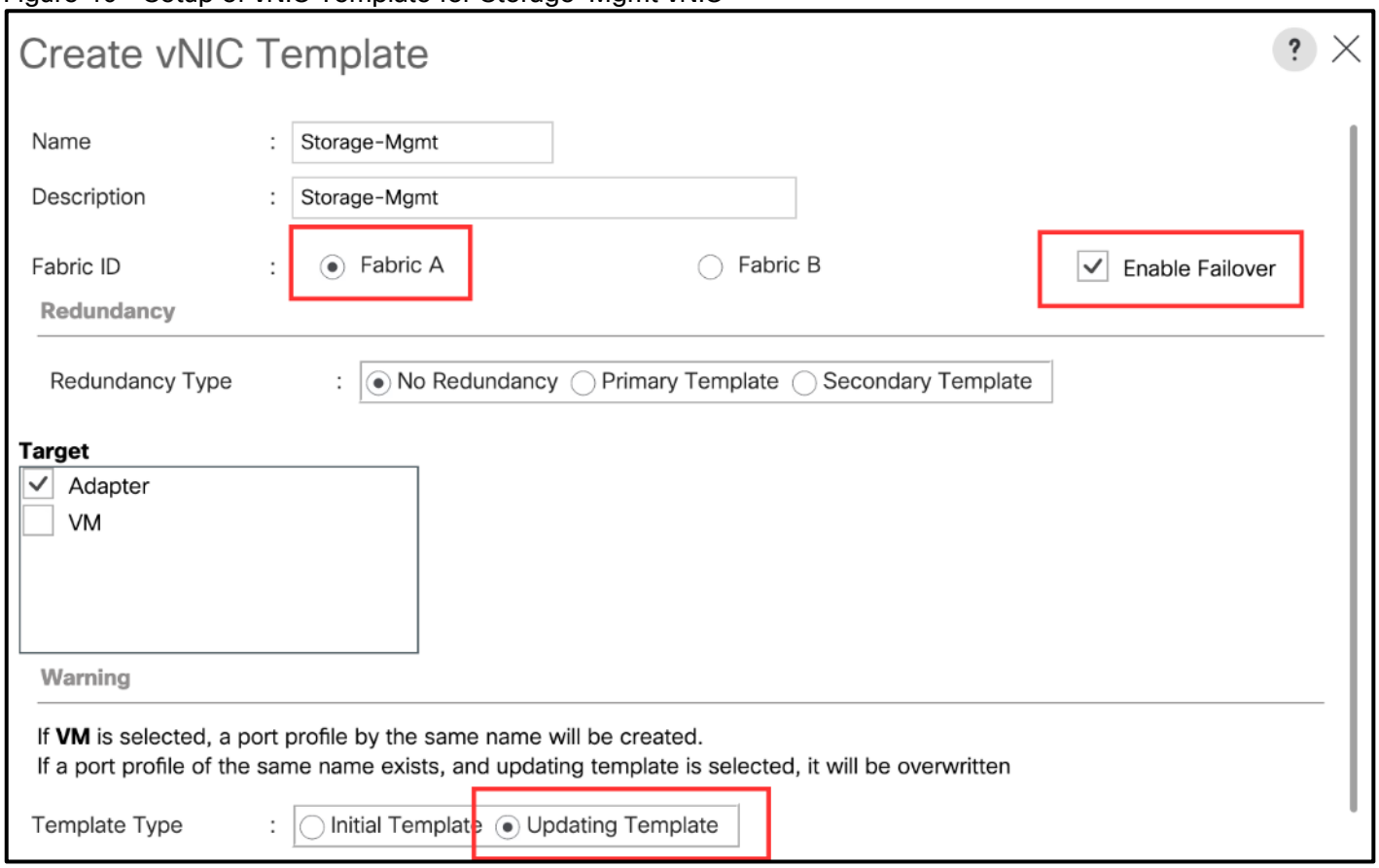
vNIC Name	Fabric	Failover	VLAN Name / ID	MTU Size	MAC Pool	Network Control Policy
Storage-Mgmt	A	Yes	Storage-Mgmt (10)	9000	Cloudian-MAC-Pools	Enable-CDP
Storage-Cluster	B	Yes	Storage-Cluster (30)	9000	Cloudian-MAC-Pools	Enable-CDP
External-Network	A	Yes	External-Network (220)	1500	Cloudian-MAC-Pools	Enable-CDP
Client-Network	A	Yes	Client-Network (20)	9000	Cloudian-MAC-Pools	Enable-CDP

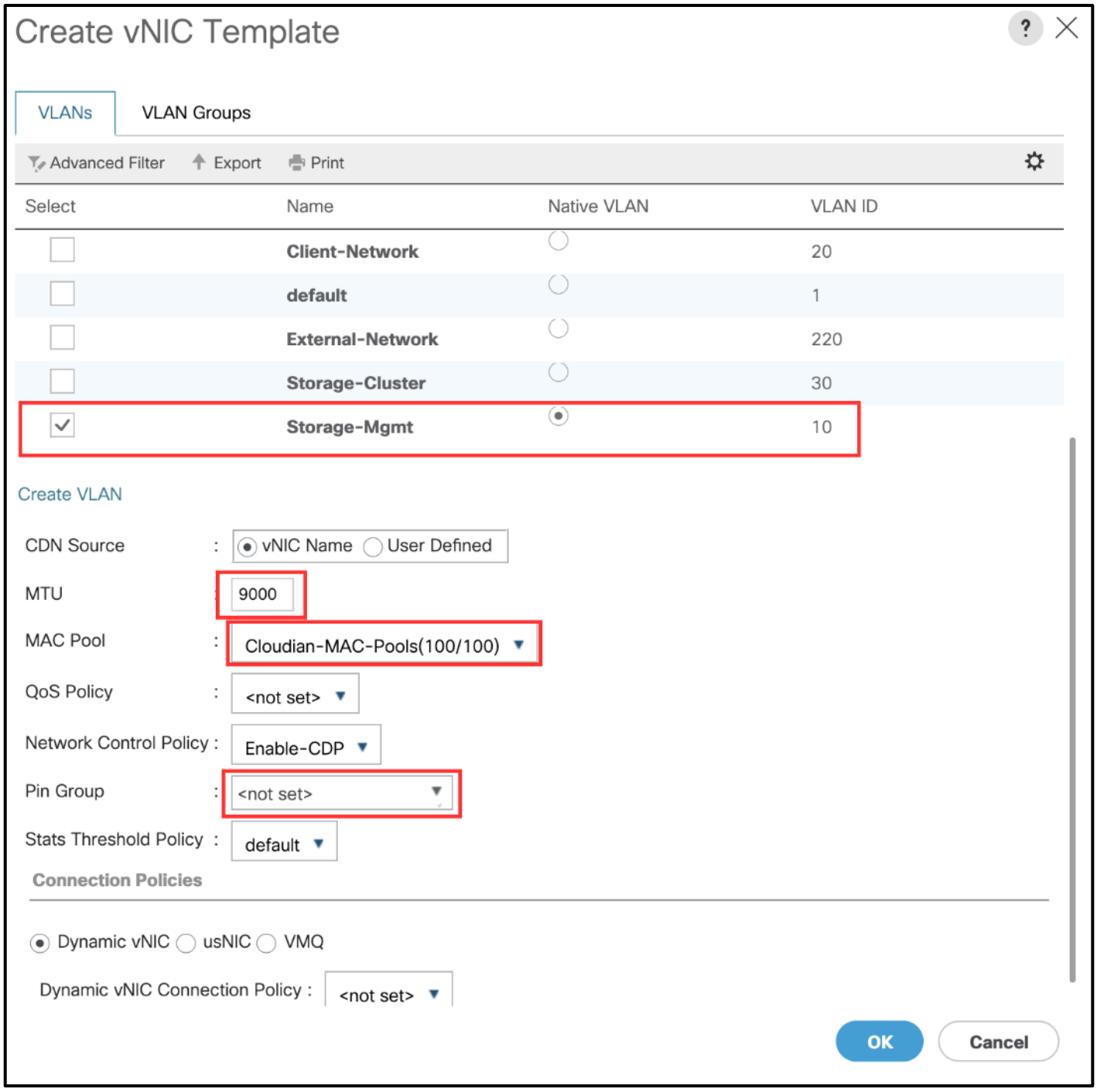
To create the appropriate vNICs, follow these steps:

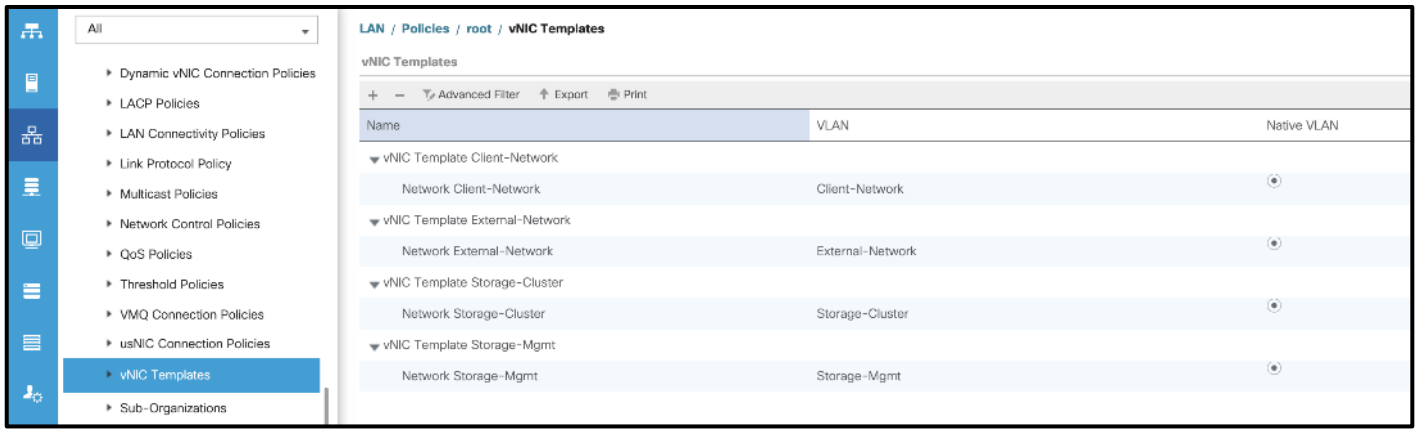
1. Select the **LAN** tab in the Cisco UCS Manager GUI.
2. Go to LAN > Policies > root > vNIC Templates and right-click Create vNIC Template.
3. Type in **Storage-Mgmt** in the **Name** field.

4. (Optional) Enter a description in the **Description** field.
5. Click Fabric A as Fabric ID and enable failover.
6. Template Type as **Updating Template**
7. Select **default** as **VLANs** and click **Native VLAN**.
8. Select **Cloudian-MAC-Pools** as MAC Pool.
9. Select Enable-CDP as Network Control Policy.
10. Click **OK** and then click **OK**.

Figure 40 Setup of vNIC Template for Storage-Mgmt vNIC




Create vNIC Template
VLANs
VLAN Groups
Advanced Filter
Export
Print
Select
Name
Native VLAN
VLAN ID
Client-Network
20
default
1
External-Network
220
Storage-Cluster
30
Storage-Mgmt
10
Create VLAN
CDN Source : vNIC Name User Defined
MTU : 9000
MAC Pool : Cloudian-MAC-Pools(100/100)
QoS Policy : <not set>
Network Control Policy : Enable-CDP
Pin Group : <not set>
Stats Threshold Policy : default
Connection Policies
Dynamic vNIC usNIC VMQ
Dynamic vNIC Connection Policy : <not set>
OK
Cancel



11. Repeat steps 1-10 for the vNICs Storage-Cluster External-Network and Client-Network. Make sure you select the correct Fabric ID, VLAN and MTU size according to Table 7 .

Ethernet Adapter Policy Setup

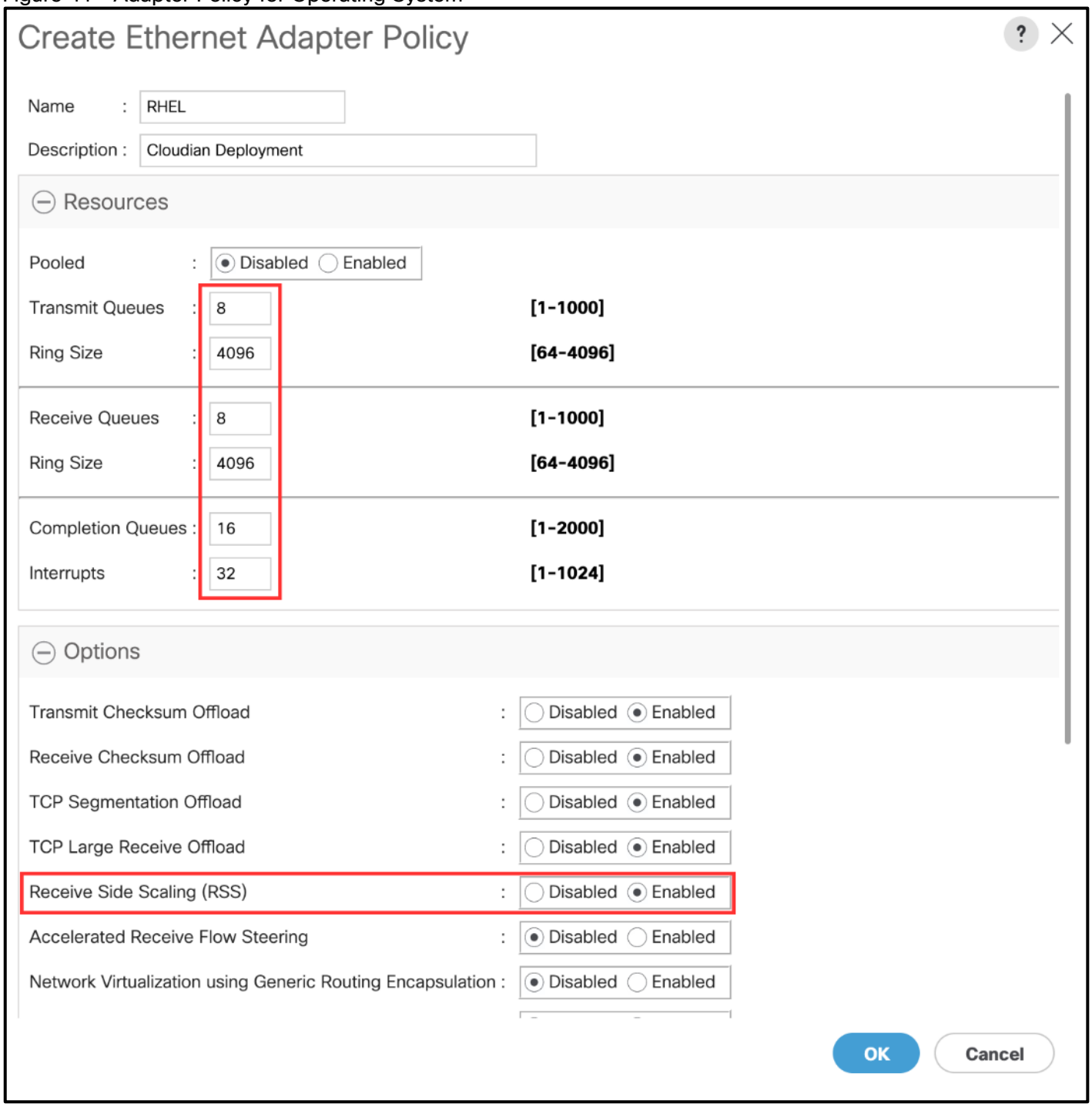
By default, Cisco UCS provides a set of Ethernet adapter policies. These policies include the recommended settings for each supported server operating system. Operating systems are sensitive to the settings in these policies.

Cisco UCS best practice is to enable Jumbo Frames MTU 9000 for any Storage facing Networks (Storage-Mgmt and Storage-Cluster). Enabling jumbo frames on specific interfaces and modifying Tx and Rx values guarantees 39Gb/s bandwidth on the UCS fabric.

To create a specific adapter policy for Red Hat Enterprise Linux, follow these steps:

1. Select the **Server** tab in the Cisco UCS Manager GUI.
2. Go to Servers > Policies > root > Adapter Policies and right-click Create Ethernet Adapter Policy**.**
3. Type in **RHEL** in the **Name** field.
4. (Optional) Enter a description in the **Description** field.
5. Under **Resources** type in the following values:
   - Transmit Queues: 8
   - Ring Size: 4096
   - Receive Queues: 8
   - Ring Size: 4096
   - Completion Queues: 16
   - Interrupts: 32
6. Under Options enable Receive Side Scaling (RSS).
7. Click **OK** and then click **OK** again.

Figure 41 Adapter Policy for Operating System



Boot Policy Setup

To create a Boot Policy, follow these steps:

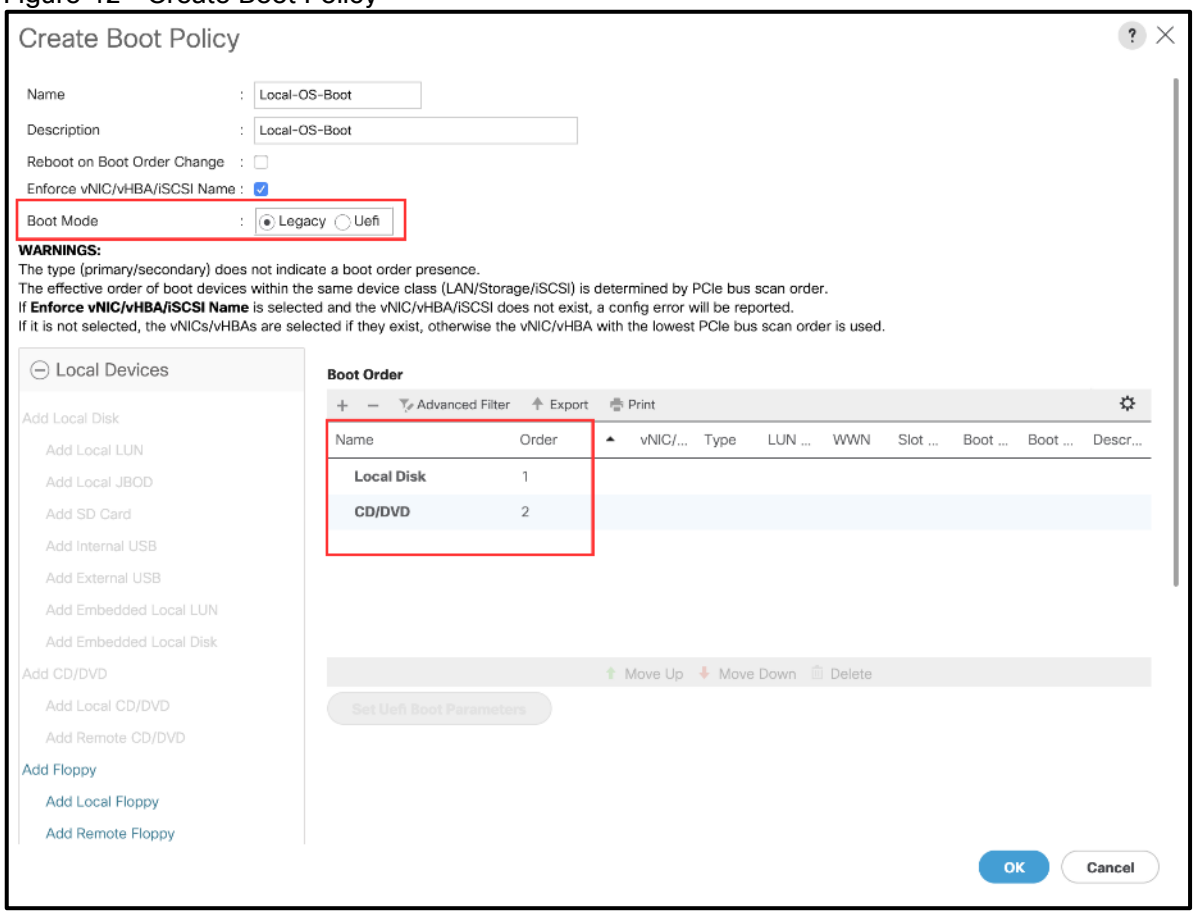
1. Select the **Servers** tab.

2. Go to Servers > Policies > root > Boot Policies and right-click Create Boot Policy.

3. Type in a **Local-OS-Boot** in the **Name** field.

4. (Optional) Enter a description in the **Description** field.

Figure 42 Create Boot Policy



5. Click Add CD/DVD and click OK.

6. Click Local Disk > Add Local LUN and Set Type as Any and click OK.

7. Click **OK**.

Create LAN Connectivity Policy Setup

To create a LAN Connectivity Policy, follow these steps:

1. Select the **LAN** tab.

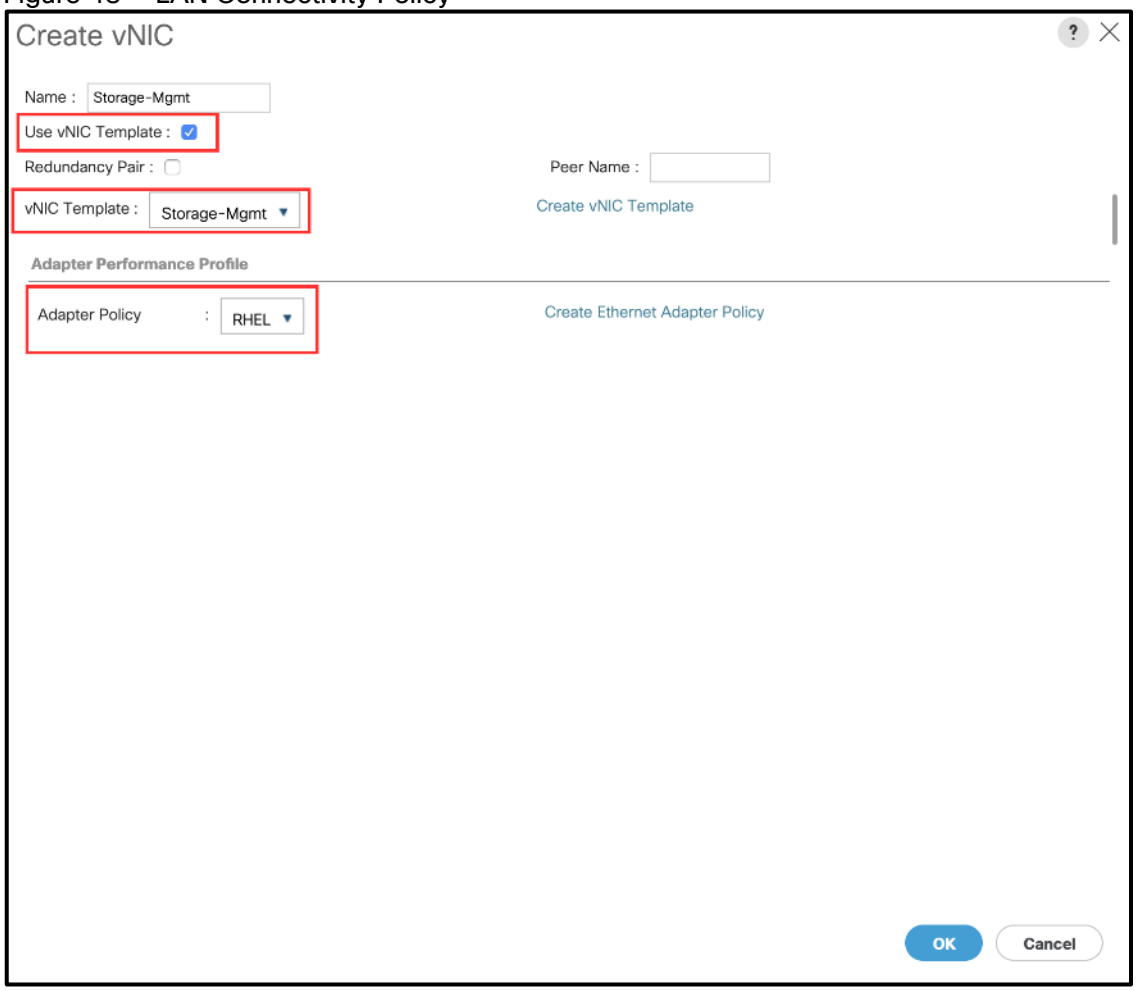
2. Go to LAN > Policies > root > LAN Connectivity Policies and right-click Create LAN Connectivity Policy for Rack Servers.

3. Type in **Storage-Node** in the **Name** field.

4. (Optional) Enter a description in the **Description** field.

5. Click Add.
6. Type in Storage-Mgmt in the name field.
7. Click Use vNIC Template.
8. Select vNIC template for Storage-Mgmt from drop-down list.
9. If you are using Jumbo Frame MTU 9000, select the default Adapter Policy, previously created as RHEL from the drop-down list.

Figure 43 LAN Connectivity Policy



10. Repeat steps 1-9 for the remaining networks Storage-Cluster, External-Network, and Client-Network Make sure you choose Adapter Policy as RHEL for VNIC interface Storage-Cluster.

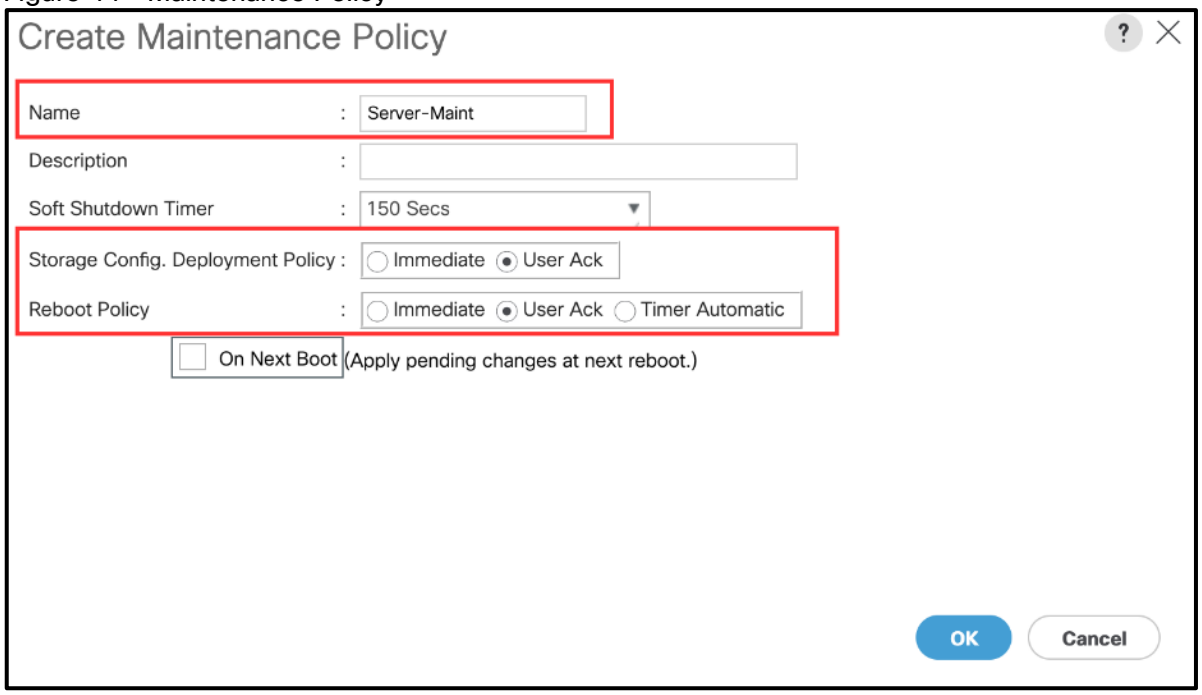
Create Maintenance Policy Setup

To setup a Maintenance Policy, follow these steps:

1. Select the **Servers** tab.
2. Go to Servers > Policies > root > Maintenance Policies and right-click Create Maintenance Policy.

3. Type in a **Server-Maint** in the Name field.
4. (Optional) Enter a description in the **Description** field.
5. Click User Ack under Reboot Policy.
6. Click **OK** and then click **OK** again.
7. Create Maintenance Policy.

Figure 44 Maintenance Policy

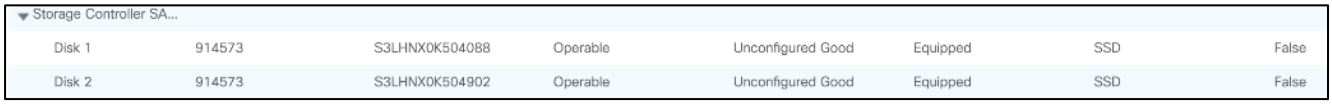


Create Storage Profiles

Set Disks for Cisco UCS C240 M5 Servers to Unconfigured-Good

To prepare the OS drives reserved from the Cisco UCS C240 M5 servers for storage profiles, make sure the disks must be converted from JBOD to Unconfigured-Good. To convert the disks, follow these steps:

1. Select the **Equipment** tab in the Cisco UCS Manager GUI.
2. For Cisco UCS C240 M5 servers, Go to Equipment > Rack-Mounts > Servers > Server1 > Inventory > Storage > Disks.
3. Select Disk1 and Disk2; right-click **Set JBOD to Unconfigured-Good**.

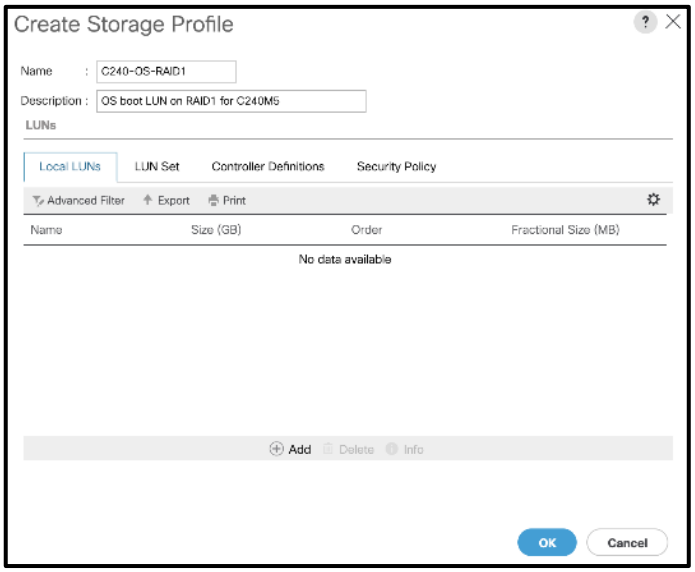


4. Repeat steps 1-3 for the other Cisco UCS C240 M5 Servers.

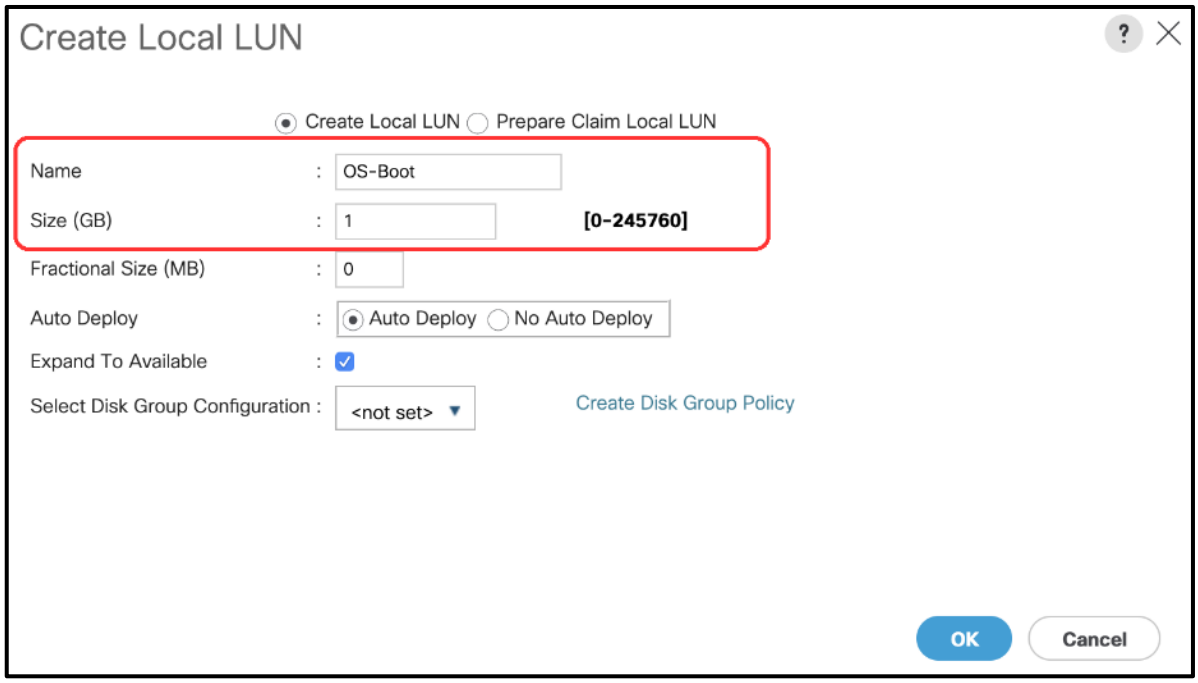
Create Storage Profiles for Cisco UCS C240 M5 Rack Server

To create the Storage Profile for the top node of the Cisco UCS C240 Rack Server, follow these steps:

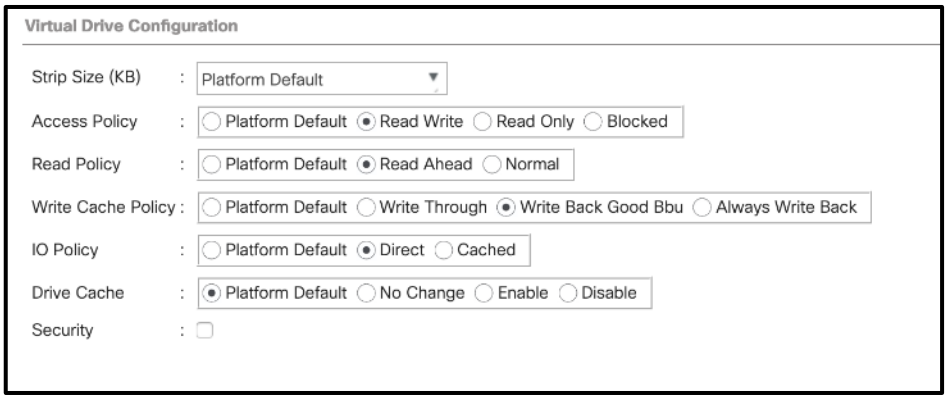
1. Select **Storage** in the Cisco UCS Manager GUI.
2. Go to Storage > Storage Profiles and right-click Create Storage Profile.
3. Type in **C240-OS-RAID1** in the **Name** field.
4. (Optional) Enter a description in the **Description** field.



5. Click **Add**.
6. Type in **OS-Boot** in the **Name** field.
7. Configure as follows:
   - Create Local LUN
   - Size (GB) = 1
   - Fractional Size (MB) = 0
   - Auto Deploy
   - Select Expand To Available

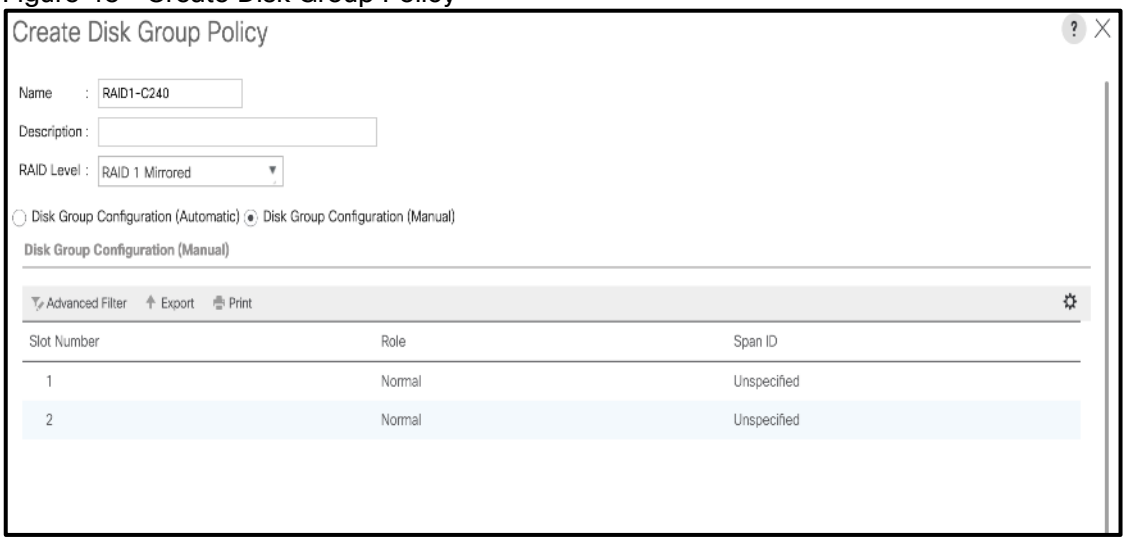


8. Click Create Disk Group Policy to Create RAID1 LUN.
9. Type in **RAID1-C240** in the Name field.
10. (Optional) Enter a description in the **Description** field.
11. RAID Level = RAID 1 Mirrored.
12. Select Disk Group Configuration (Manual).
13. Click **Add**.
14. Type in **1** for **Slot Number**.
15. Click **OK** and then again **Add**.
16. Type in **2** for **Slot Number**.
17. Under Change Virtual Drive Configuration:
    a. Modify Access Policy as Read Write and Read Policy as Read Ahead.
    b. Modify Write Cache Policy as Write Back Good BBU and IO Policy as Direct.



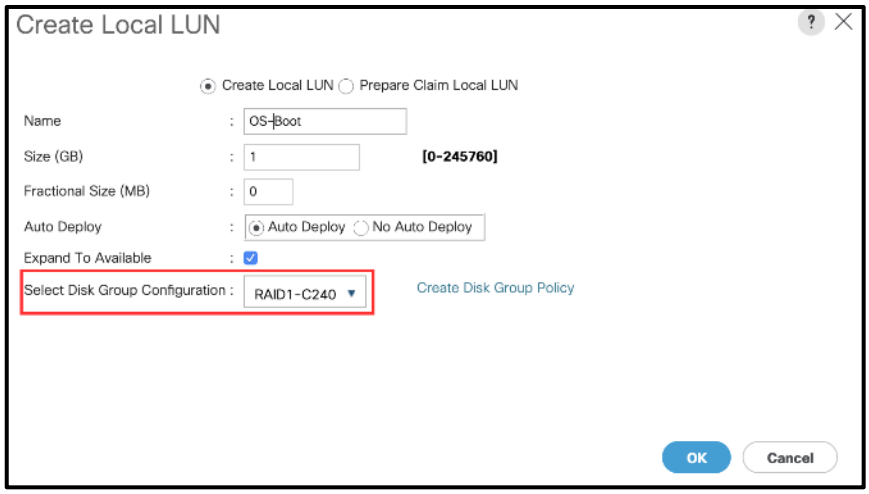
18. Click **OK** and then click **OK** again.

Figure 45 Create Disk Group Policy



19. Select your previously created Disk Group Policy for the Boot with the radio button under **Select Disk Group Configuration**.

20. Select Disk Group Configuration.



21. Click **OK**, click **OK** again, and then click **OK**.

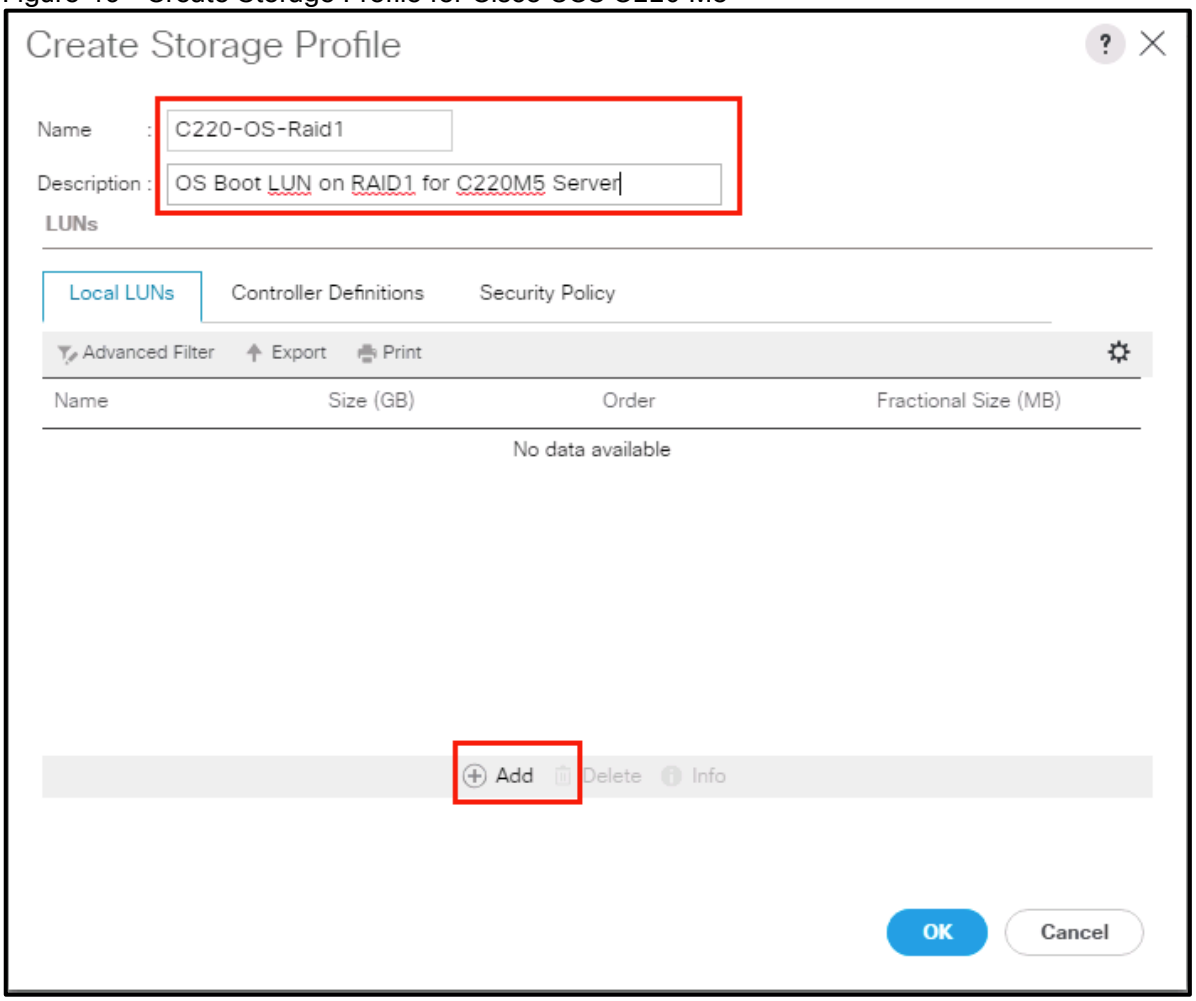
22. Repeat steps 1-21 to create a storage profile for Server-2 and Server-3.

Create Storage Profile for Cisco UCS C220 M5 Rack-Mount Servers

To create a Storage Profile for the Cisco UCS C220 M5, follow these steps:

1. Select **Storage** of the UCSM GUI.
2. Go to Storage > Storage Profiles and right-click Create Storage Profile.
3. Type in **C220-OS-RAID1** in the **Name** field.
4. (Optional) Enter a description in the **Description** field.
5. Click **Add**.

Figure 46 Create Storage Profile for Cisco UCS C220 M5



6. Type in **Boot** in the **Name** field.

7. Configure as follows:

   a. Create Local LUN

   b. Size (GB) = 1

   c. Fractional Size (MB) = 0

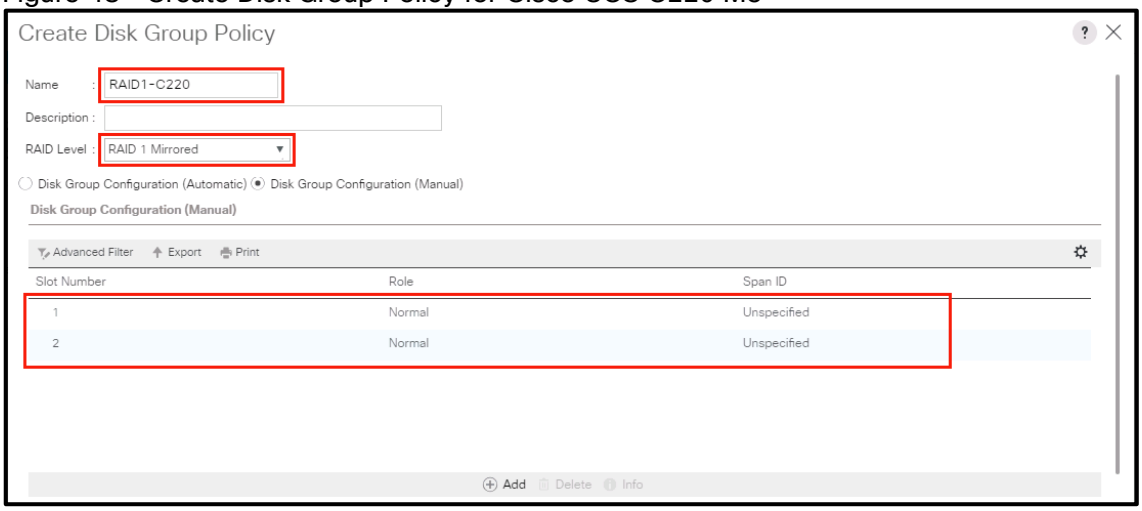
   d. Select Expand To Available

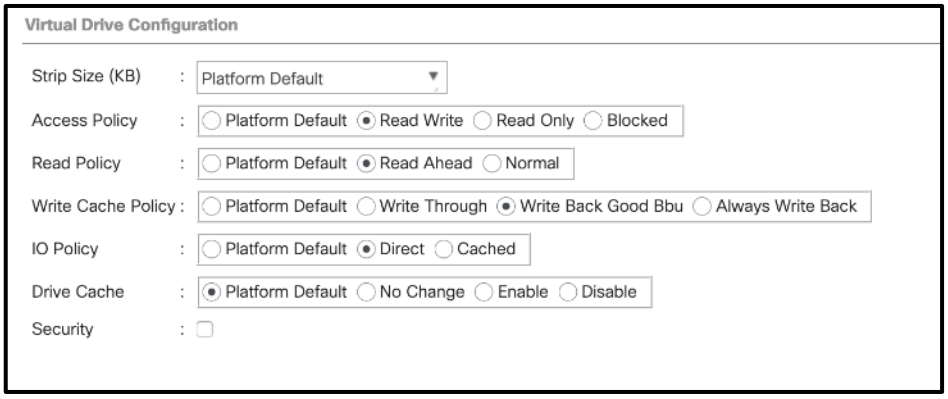
   e. Auto Deploy

Figure 47 Create Local LUN

8. Click Create Disk Group Policy to Create RAID1 LUN.
9. Type in **RAID1-C220** in the **Name** field.
10. (Optional) Enter a description in the **Description** field.
11. RAID Level = RAID 1 Mirrored.
12. Select Disk Group Configuration (Manual).
13. Click **Add**.
14. Type in **1** for **Slot Number**.
15. Click **OK** and then again **Add**.
16. Type in **2** for **Slot Number**.
17. Under Change Virtual Drive Configuration:
    a. Modify Access Policy as Read Write and Read Policy as Read Ahead.
    b. Modify Write Cache Policy as Write Back Good BBU and IO Policy as Cache.
18. Click **OK** and then click **OK** again.

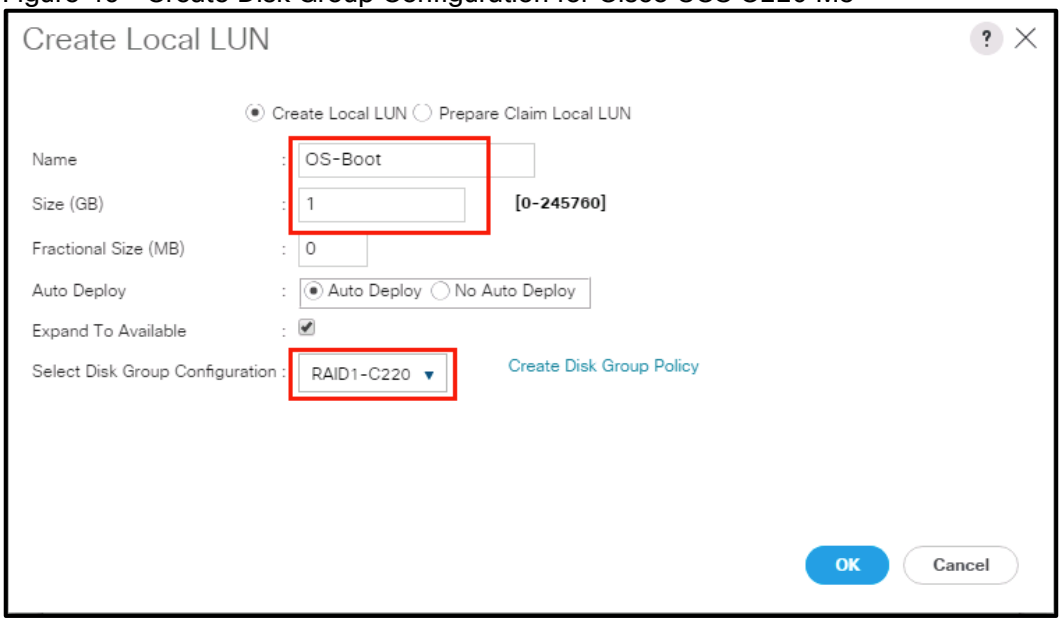
Figure 48 Create Disk Group Policy for Cisco UCS C220 M5





19. Select the previously created Disk Group Policy for the Cisco UCS C220 M5 Boot Disks under **`Select Disk Group Configuration`**.

Figure 49 Create Disk Group Configuration for Cisco UCS C220 M5



20. Click **OK** and then click **OK** and again click **OK**.

Creating a Service Profile Template for Cisco UCS C240 M5 Rack Server

To create a Service Profile Template, follow these steps:

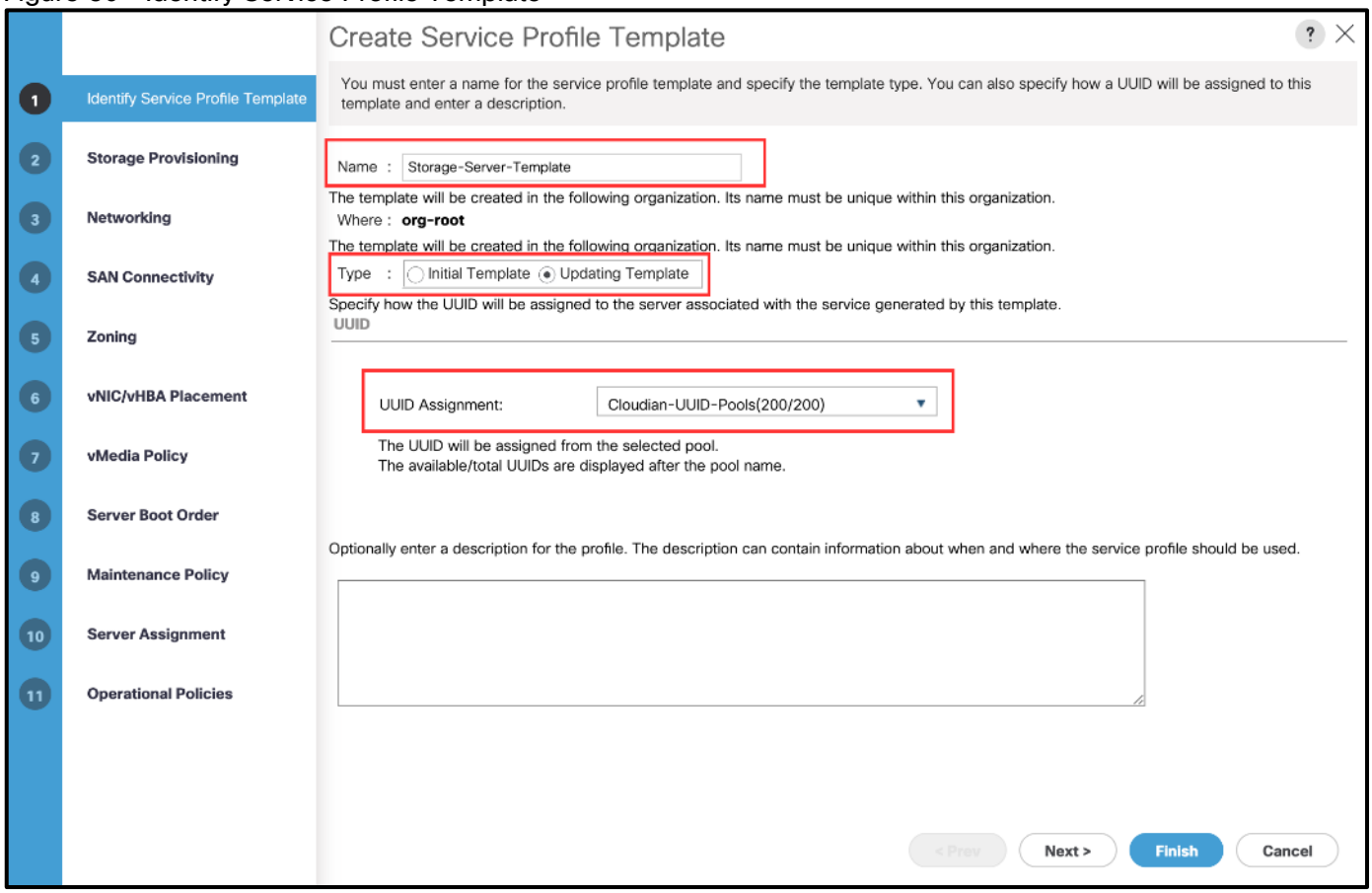
1. Select **Servers** of the Cisco UCS Manager GUI.
2. Go to Servers > Service Profile Templates > root and right-click Create Service Profile Template.

Identify Service Profile Template

To identify the Service Profile template, follow these steps:

1. Type in Storage-Server-Template in the Name field.
2. Select Template Type **Updating Template**
3. In the **UUID Assignment** section, select the UUID Pool you created in the beginning.
4. (Optional) Enter a description in the **Description** field.

Figure 50 Identify Service Profile Template



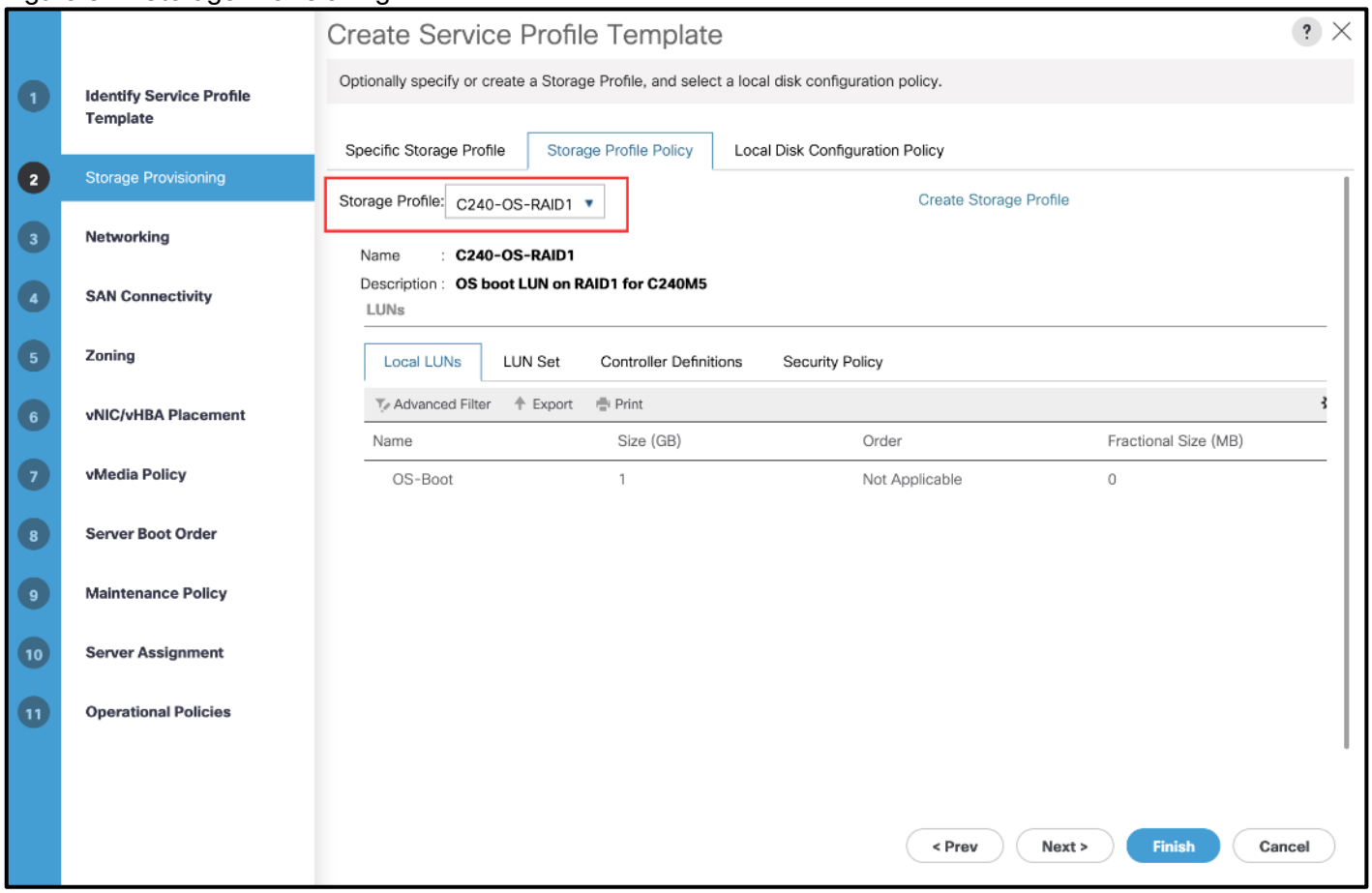
5. Click **Next**.

Storage Provisioning

To provision the storage profile, follow these steps:

1. Go to the **Storage Profile Policy** tab and select the Storage Profile **C240-OS-RAID1** for the top node of the Cisco UCS C240 Rack Server you created before.
2. Click **Next**.

Figure 51 Storage Provisioning

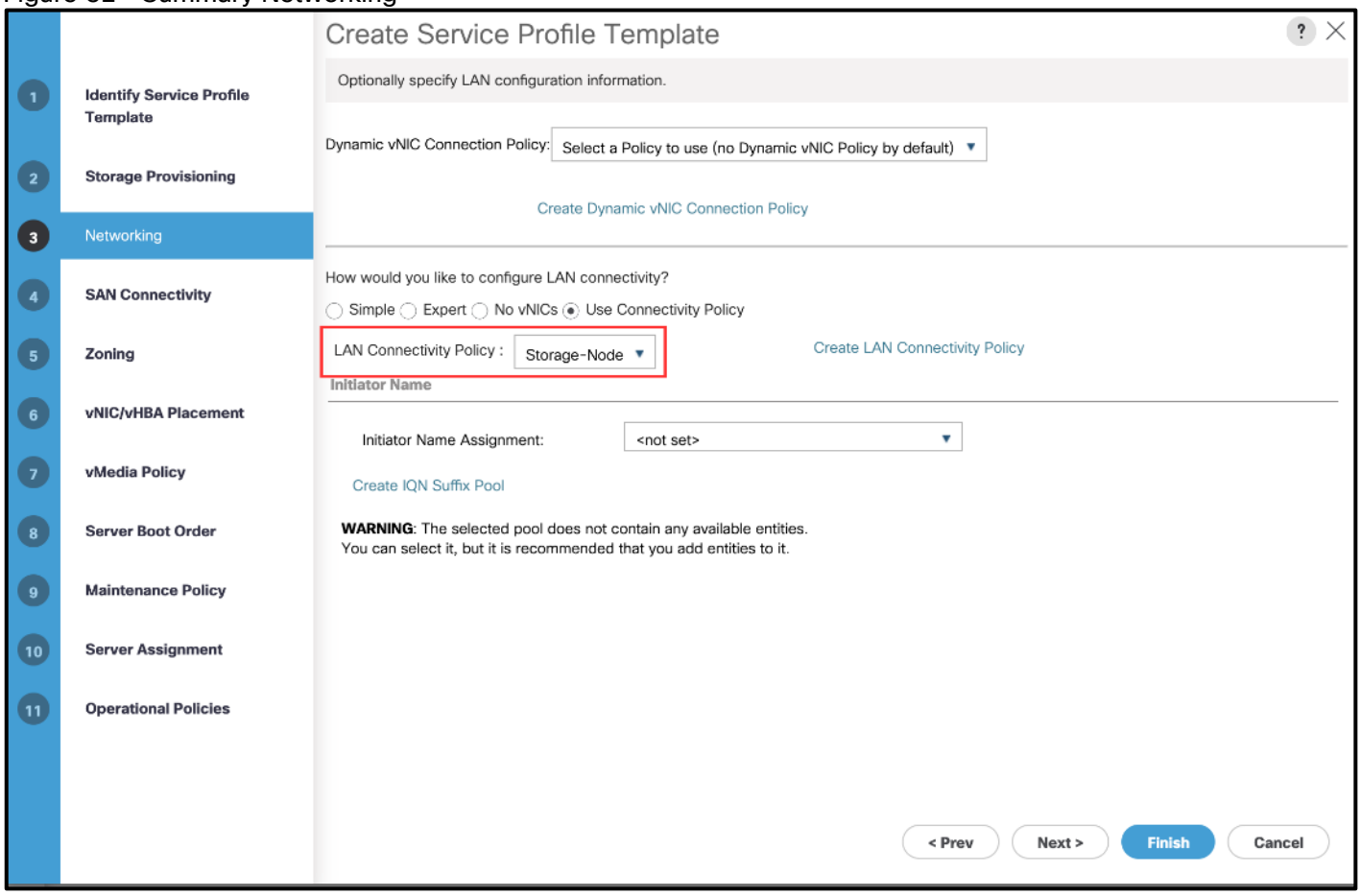


Networking

To configure networking, follow these steps:

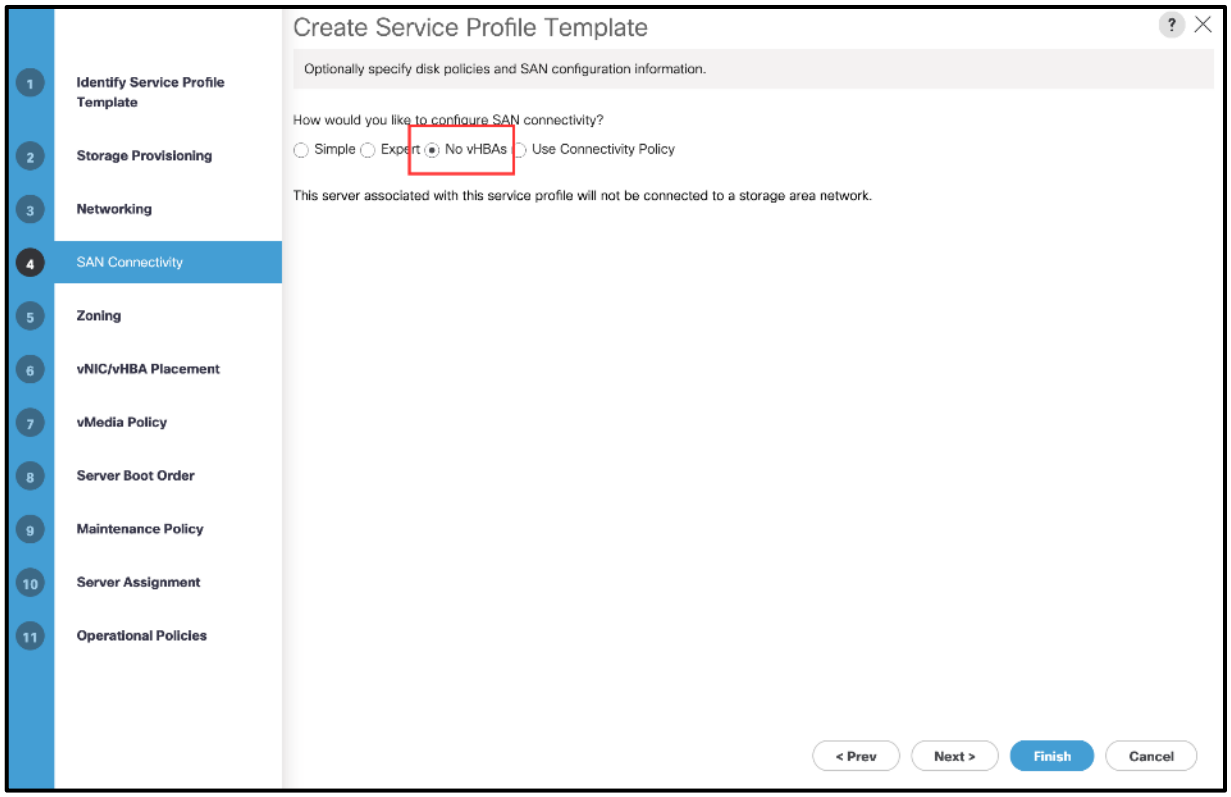
1. Keep the Dynamic vNIC Connection Policy field at the default.
2. Select LAN connectivity to Use Connectivity Policy created before.
3. From LAN Connectivity drop-down list, select Storage-Node created before and click Next.

Figure 52 Summary Networking



3. Click **Next** to continue with SAN Connectivity.

4. Select No vHBA for How would you like to configure SAN Connectivity?



5. Click **Next** to continue with Zoning.

6. Click **Next**.

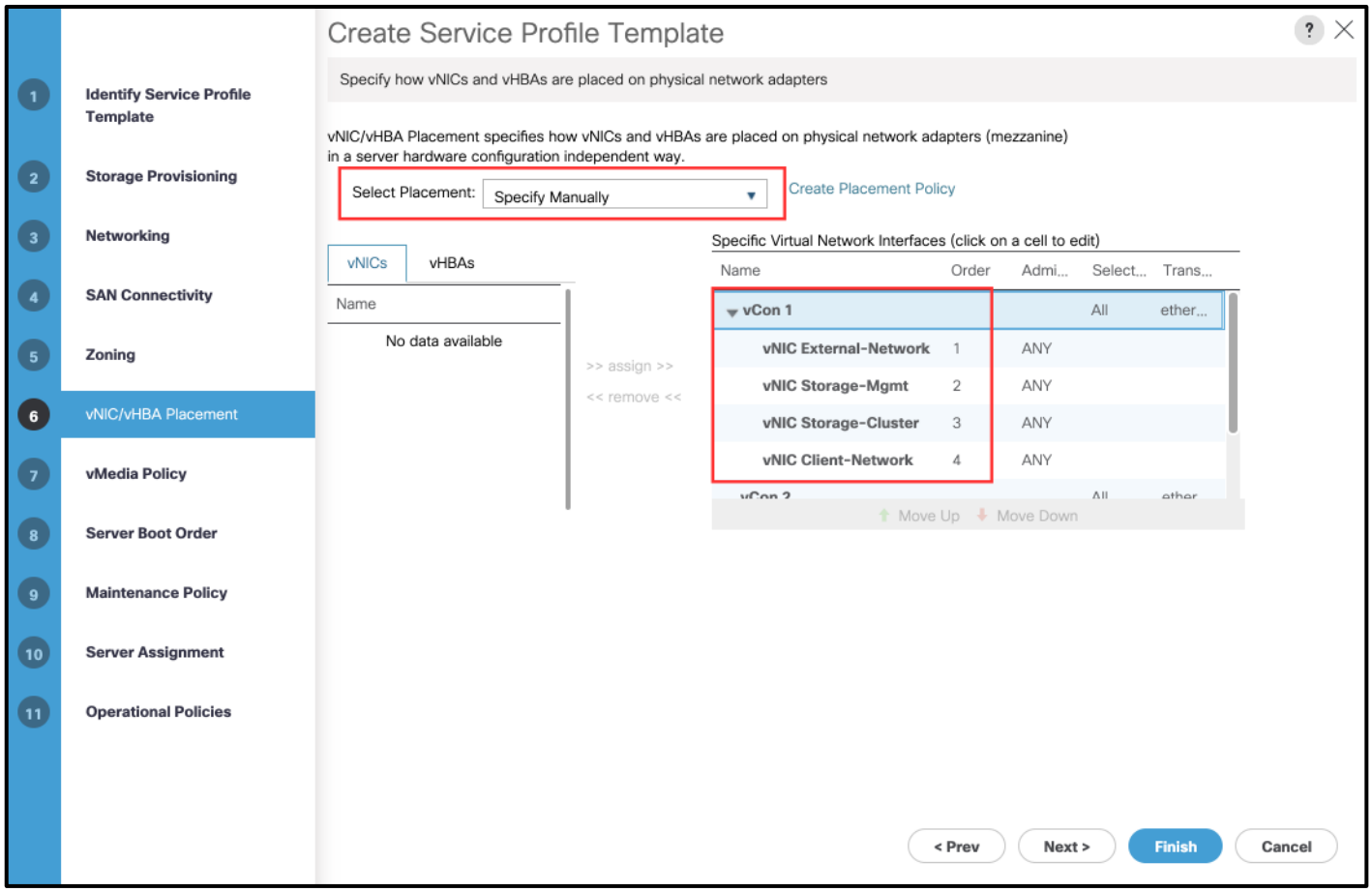
vNIC/vHBA Placement

To configure the vNIC/vHBA placement, follow these steps:

1. Select **Specify Manually** form the drop-down list.

2. Under PCI order section, Sort all the vNICs.

3. Make sure the vNICs order are listed as External-Network > 1, then followed by Storage-Mgmt > 2 and Storage-Cluster > 3 Client-Network > 4.

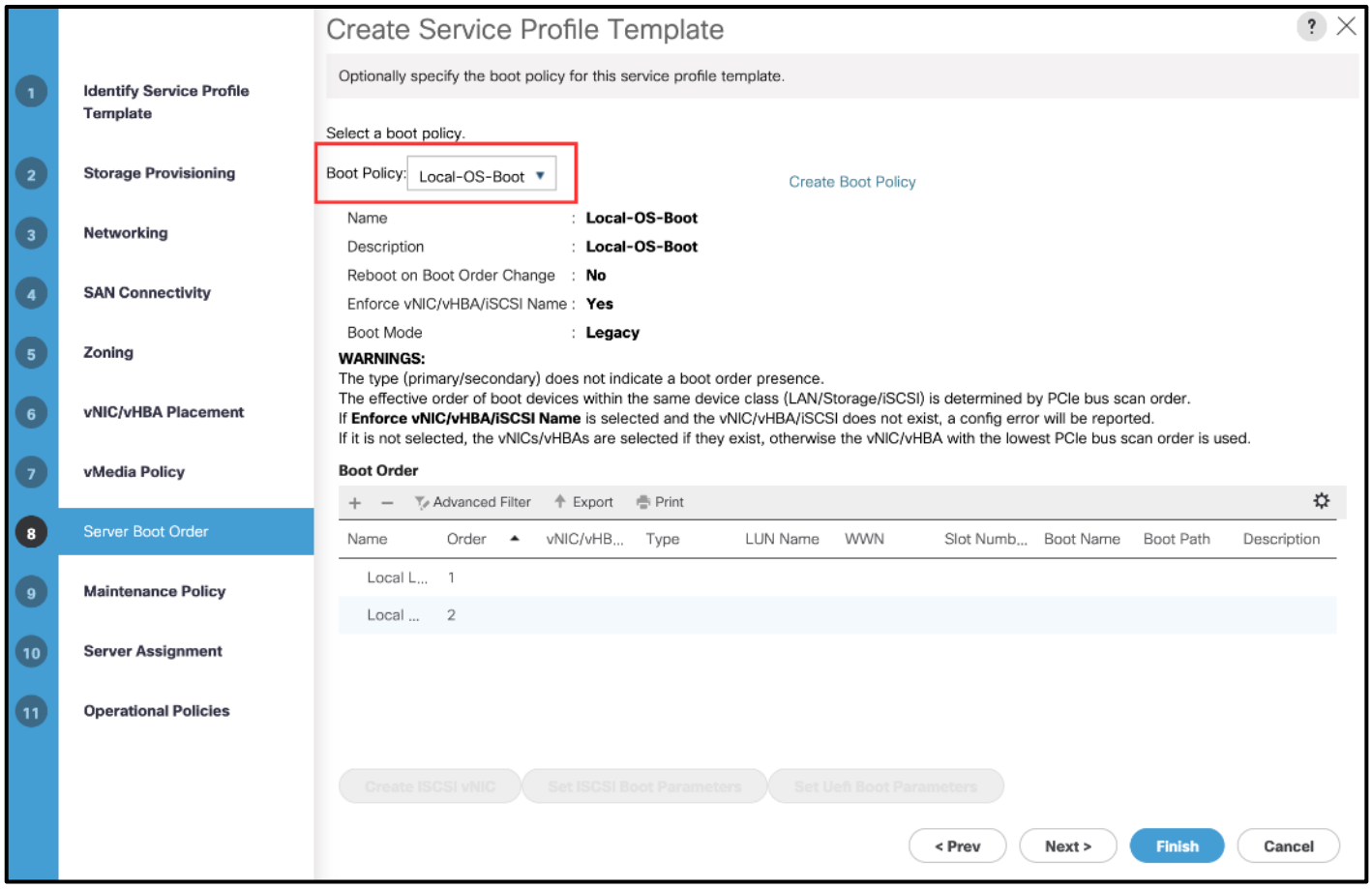


4. Click **Next** to continue with vMedia Policy.
5. Click **Next**.

Server Boot Order

To configure the server boot order, follow these steps:

1. Select the Boot Policy Local-OS-Boot under Boot Policy.
2. Server Boot Order.
3. Click Next.

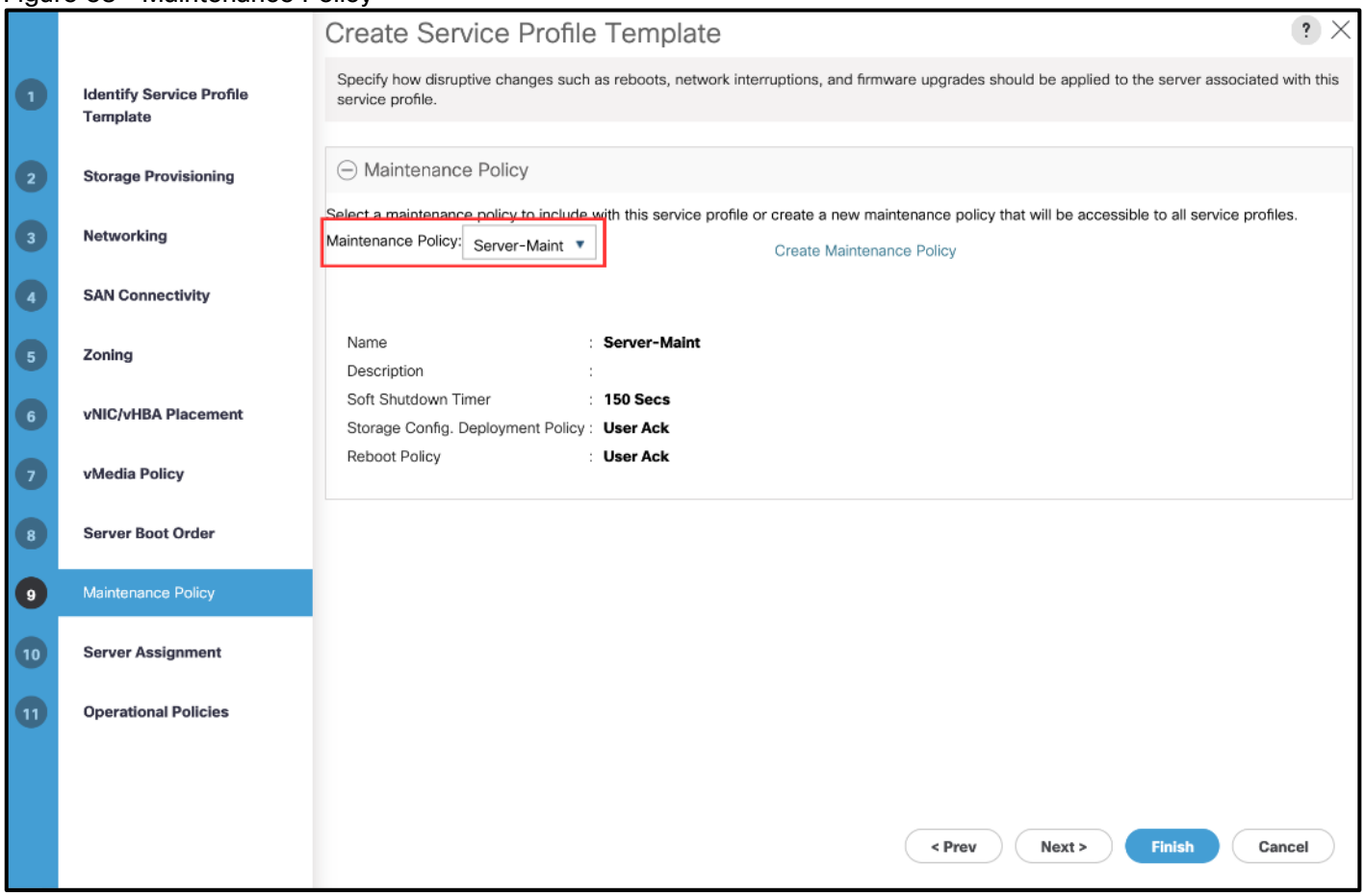


Maintenance Policy

To configure the maintenance policy, follow these steps:

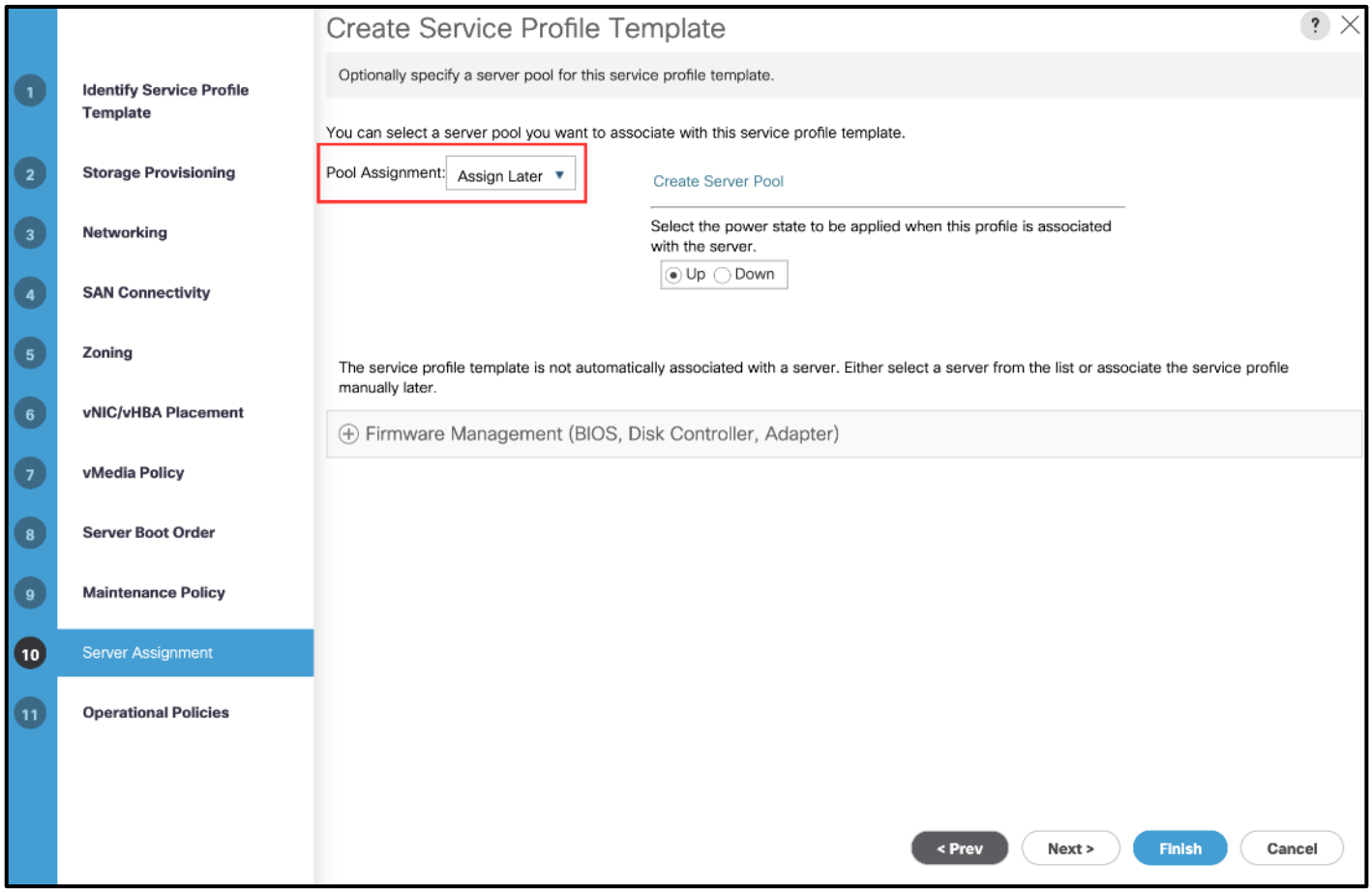
1. Select the Maintenance Policy you created before under Maintenance Policy.

Figure 53 Maintenance Policy



2. Click **Next**.

3. Under Server Assignment, make sure Assign Later is selected for Pool Assignment.



4. Click **Next**.

5. Click **Finish** and then click **OK**.

Create Service Profiles from Template

This section details how to create the appropriate Service Profiles from the previous Service Profile Templates. To create the first profile for the Server1 of the Cisco UCS C240 Rack Server, follow these steps:

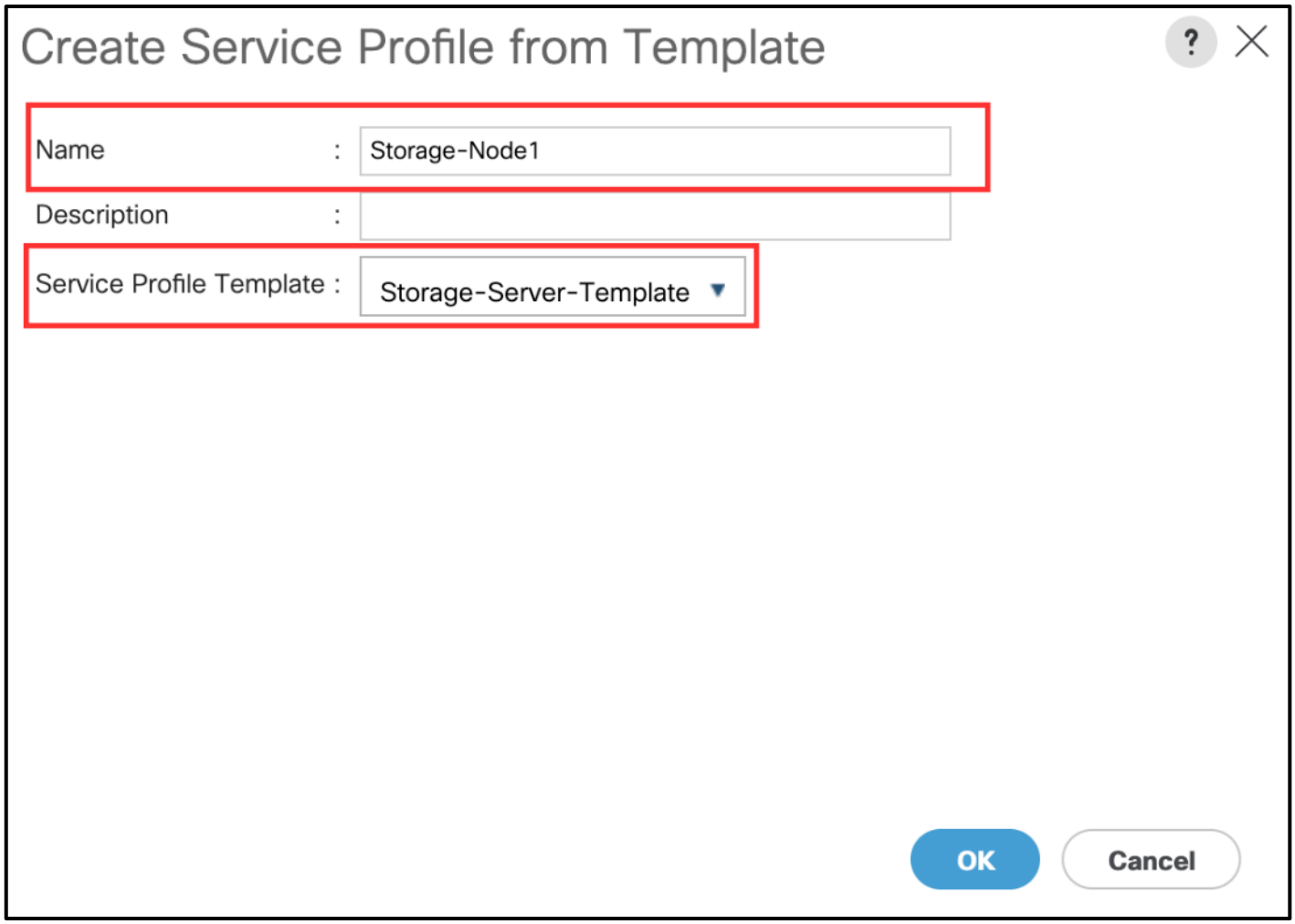
1. Select **Servers** in the Cisco UCS Manager GUI.

2. Go to Servers > Service Profiles and right-click Create Service Profile from Template.

3. Type in **Storage-Node1** in the Name Prefix field.

4. Choose **Storage-Server-Template** as the **Service Profile Template** you created before for the top node of the Cisco UCS C240 Rack Server.

5. Click **OK** and then click **OK** again.

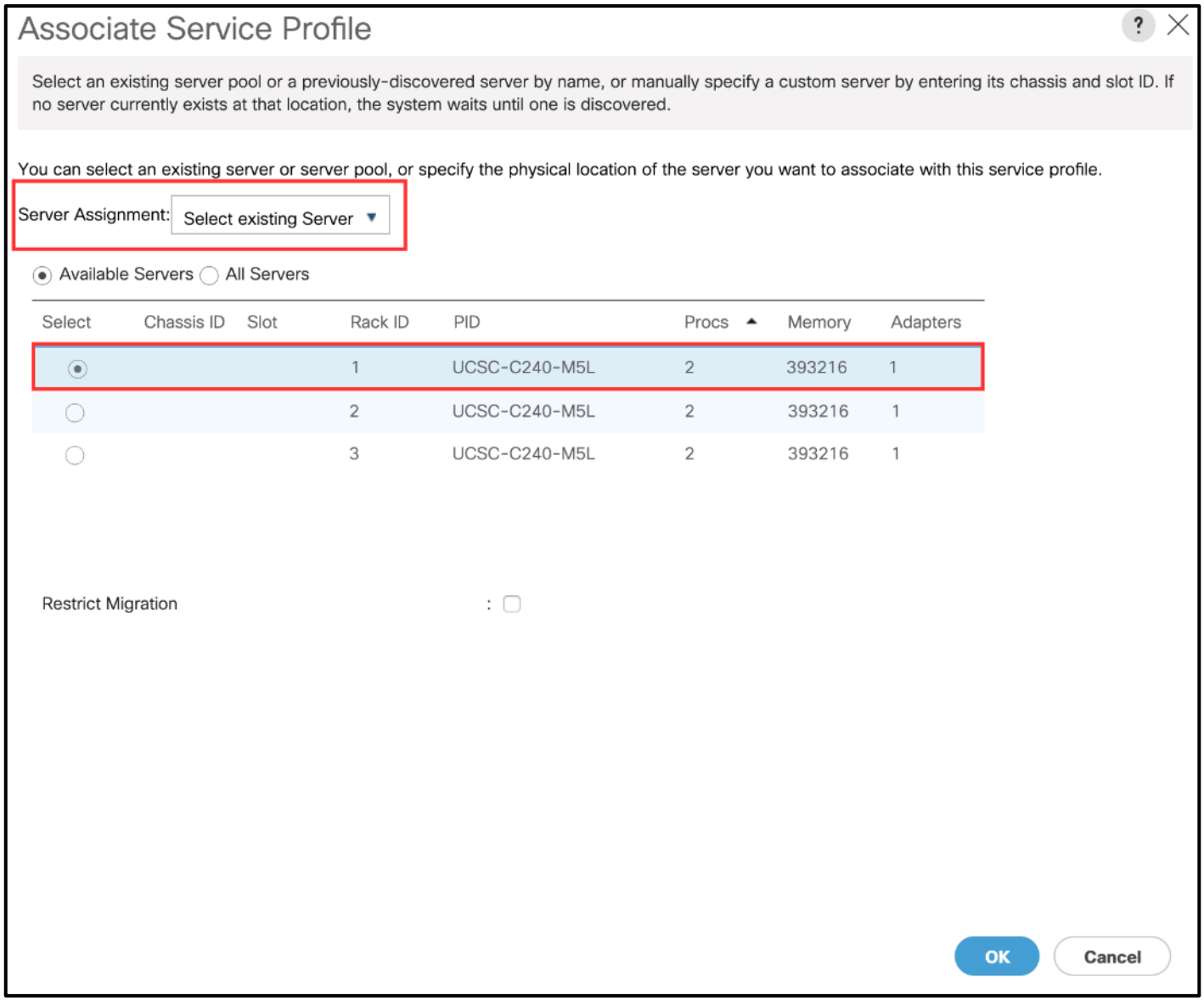


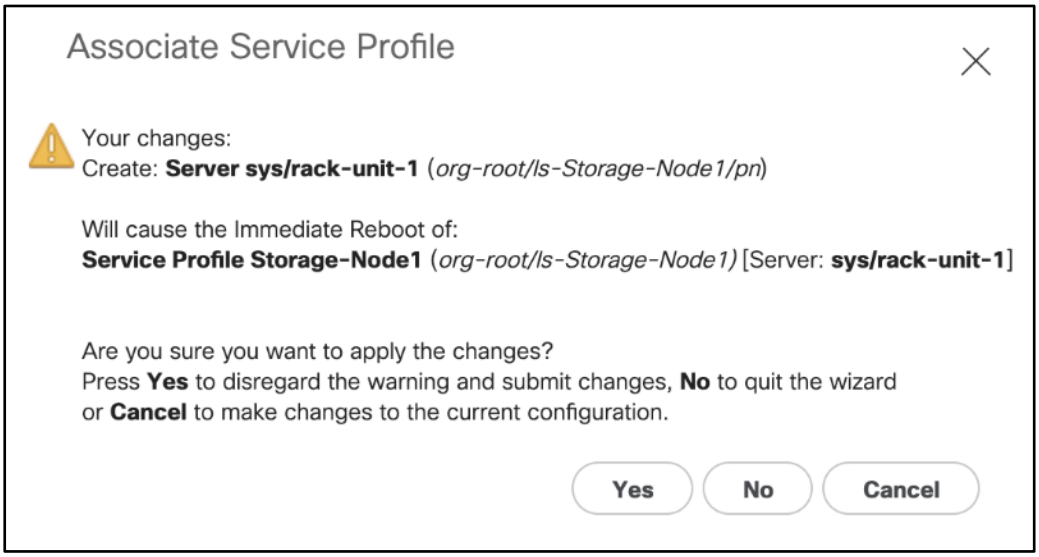
6. Repeat steps 1-5 to create Service Profiles with the names **Storage-Node2 and Storage-Node3** for the remaining Cisco UCS C240 M5 server nodes from the Template Storage-Server-Template.

Associating a Service Profile for Cisco UCS C240 M5 Server

To associate all the Storage-NodeX Service Profiles to the Cisco UCS C240 M5 Rack Servers, follow these steps:

1. Select **Servers** in the Cisco UCS Manager GUI.
2. Go to Servers > Service Profiles and right-click Storage-Node1 Service profile created previously.
3. Click Change Server Profile Association.
4. From the Server Assignment drop-down list choose Select Existing Server.
5. Click the radio button Available Servers.
6. From the list, choose Rack ID-1 for Storage-Node1.
7. Click OK.


Associate Service Profile
Select an existing server pool or a previously-discovered server by name, or manually specify a custom server by entering its chassis and slot ID. If no server currently exists at that location, the system waits until one is discovered.
You can select an existing server or server pool, or specify the physical location of the server you want to associate with this service profile.
Server Assignment:
Select existing Server
Available Servers
All Servers
Select
Chassis ID
Slot
Rack ID
PID
Procs
Memory
Adapters
1
UCSC-C240-M5L
2
393216
1
2
UCSC-C240-M5L
2
393216
1
3
UCSC-C240-M5L
2
393216
1
Restrict Migration
:
OK
Cancel



8. Repeat steps 1-7 to the Associate Remaining Service profiles Storage-NodeX for the Cisco UCS C240 M5 Rack server as listed in the table below.

Service Profile Template	Service Profile	Rack ID
Storage-Server-Template	Storage-Node1	1
	Storage-Node2	2
	Storage-Node3	3

Create Service Profile for Cisco UCS C220 M5 Server for HA-Proxy Node

To create a Service Profile, follow these steps:

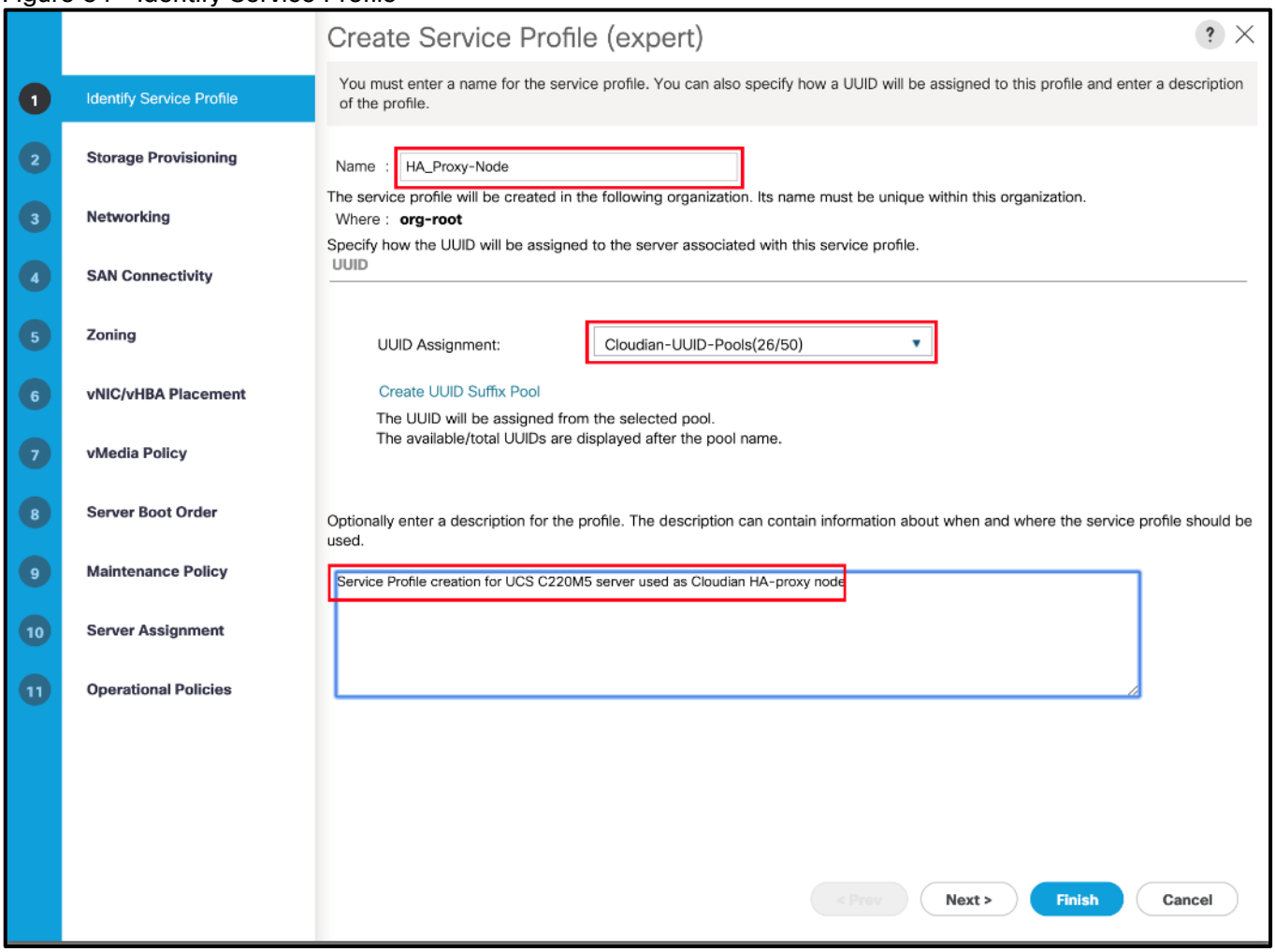
1. Select **Servers** of the Cisco UCS Manager GUI.
2. Go to Servers > Service Profile > root and right-click to choose Create Service Profile (expert).

Identify Service Profile

To identify the service profile, follow these steps:

1. Type in the Name field.
2. In the **UUID Assignment** section, select the UUID Pool you created in the beginning.
3. (Optional) Enter a description in the **Description** field.

Figure 54 Identify Service Profile



4. Click **Next**.

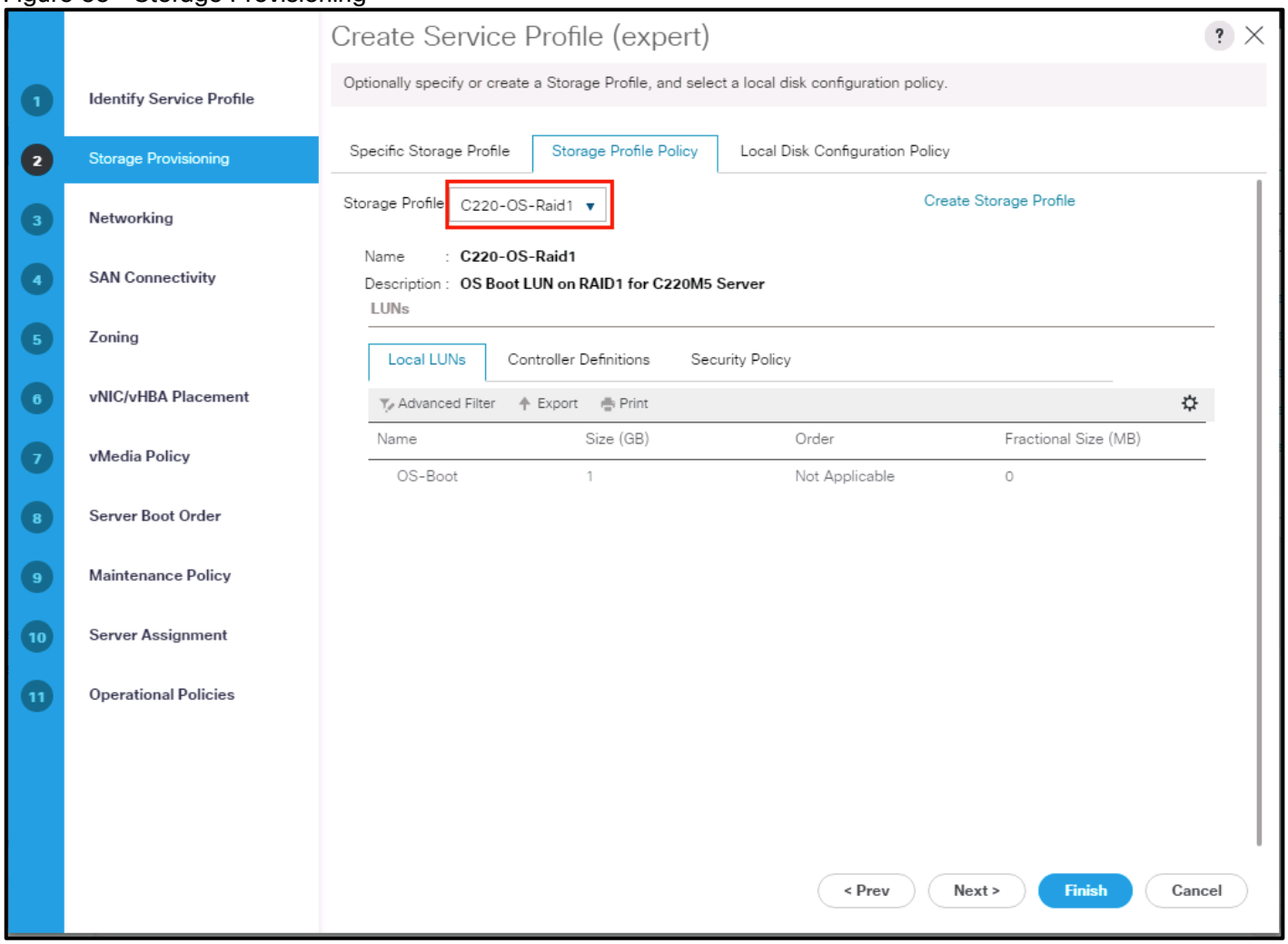
Storage Provisioning

To configure the storage provisioning, follow these steps:

1. Go to the **Storage Profile Policy** tab and select the Storage Profile **C220-OS-Raid1** for the top node of the Cisco UCS C240 Rack Server you created before.

2. Click **Next**.

Figure 55 Storage Provisioning



Networking

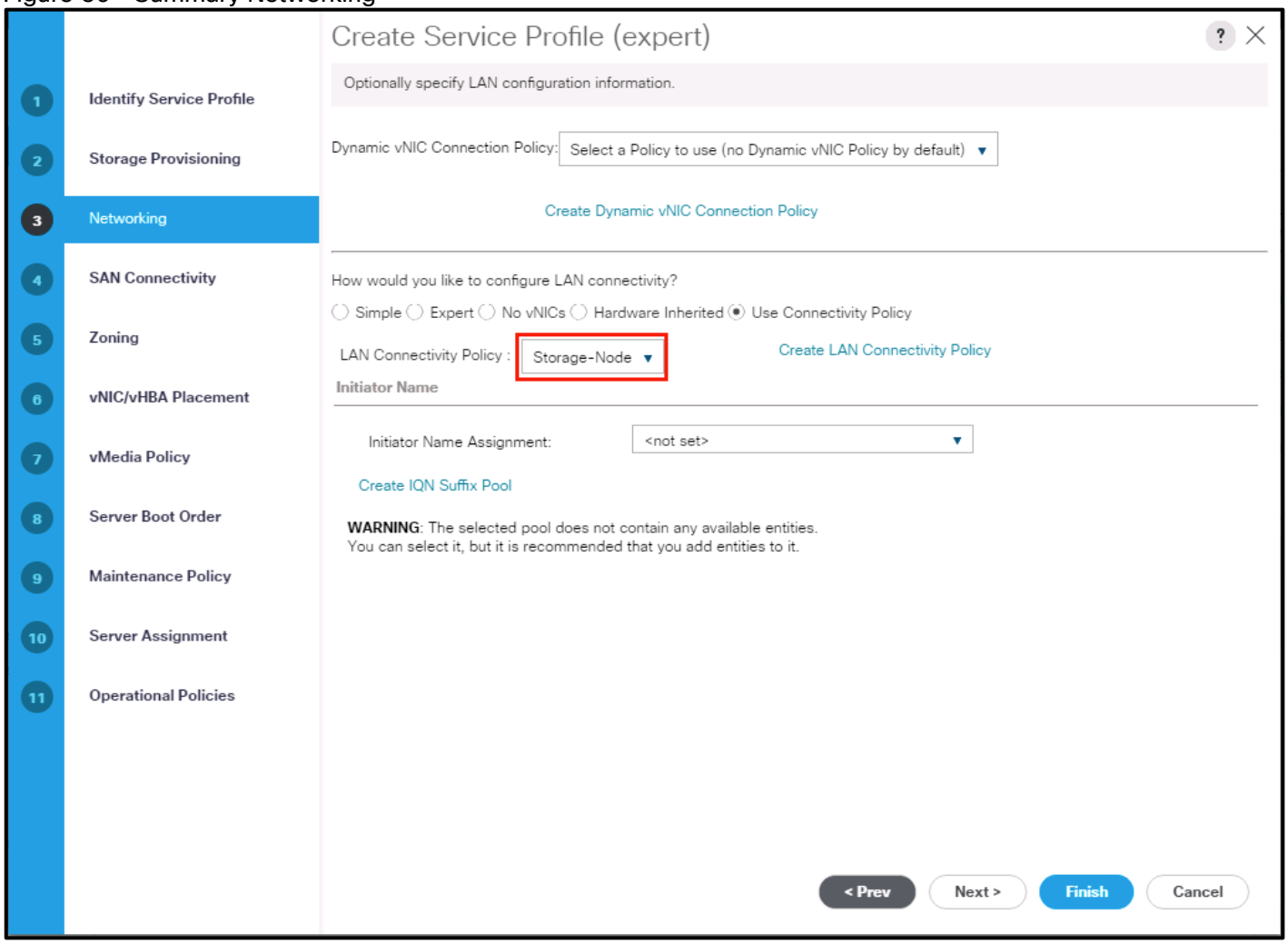
To configure networking, follow these steps:

1. Keep the Dynamic vNIC Connection Policy field at the default.
2. Select LAN connectivity to Use Connectivity Policy created previously.
3. From the LAN Connectivity drop-down list, select Storage-Node previously created.

HA-Proxy Node and Storage-Nodes use the same VNIC interfaces.

4. Click Next.

Figure 56 Summary Networking

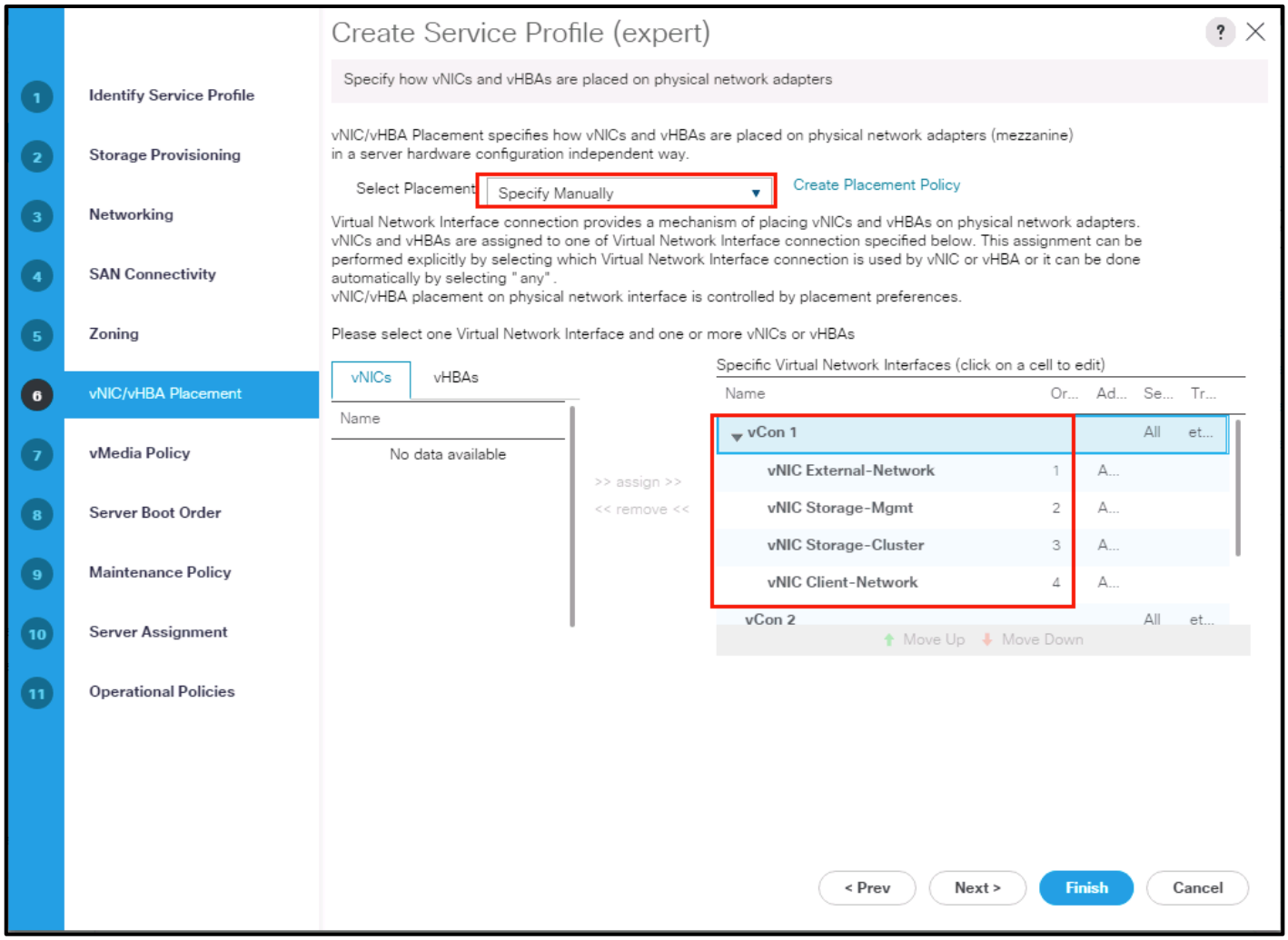


5. Click **Next** to continue with SAN Connectivity.
6. Select No vHBA for How would you like to configure SAN Connectivity?
7. Click **Next** to continue with Zoning.
8. Click **Next**.

vNIC/vHBA Placement

To configure the vNIC/vHBA placement, follow these steps:

1. Select **Specify Manually** form the drop-down list.
2. Under PCI order section, Sort all the vNICs.
3. Make sure the vNICs order is listed as External-Network > 1, then followed by Storage-Mgmt > 2 and Storage-Cluster > 3 Client-Network > 4.



4. Click **Next** to continue with vMedia Policy.

5. Click **Next**.

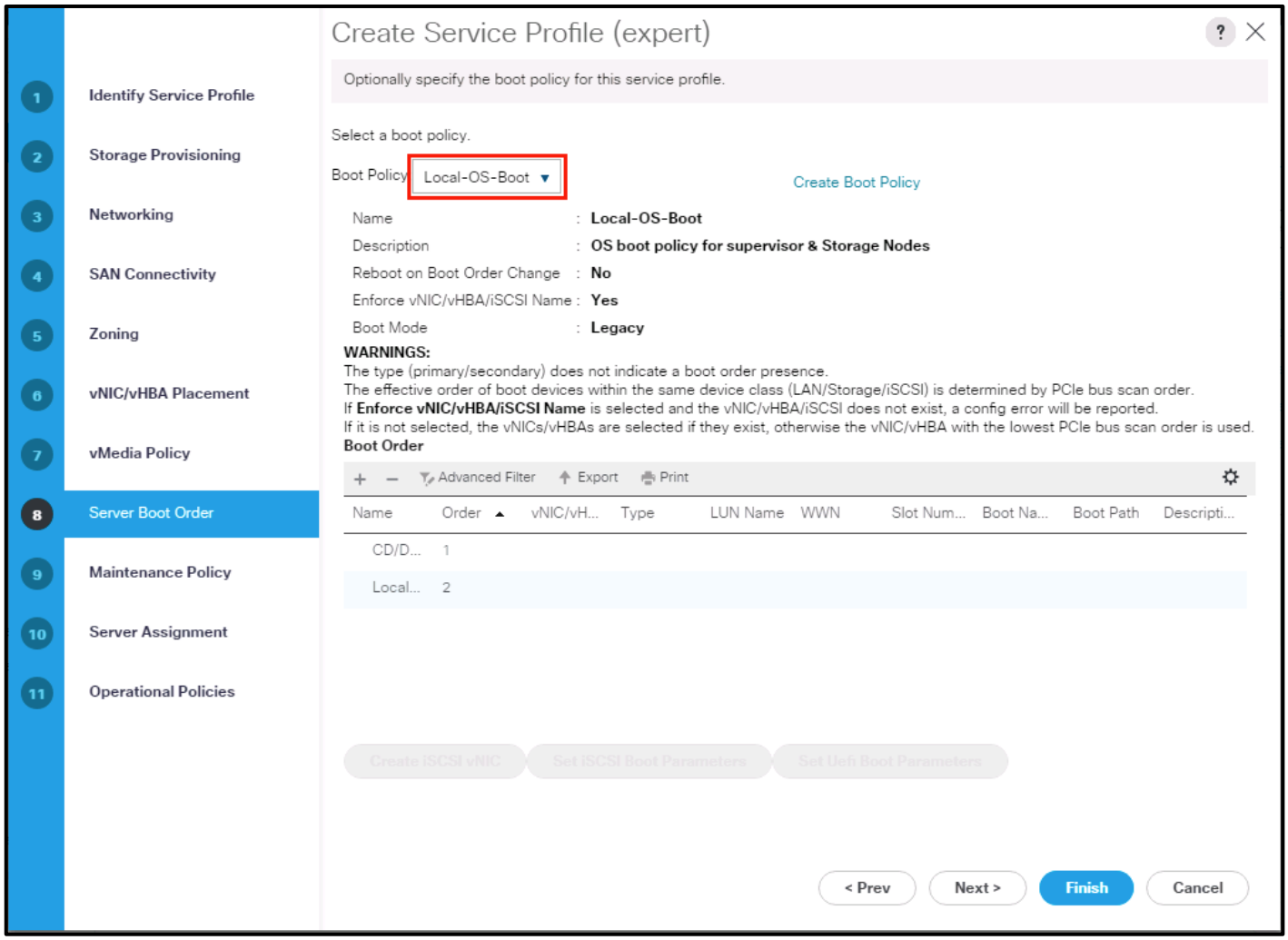
Server Boot Order

To configure the server boot order, follow these steps:

1. Select the Boot Policy Local-OS-Boot you created before under Boot Policy.

2. Server Boot Order.

3. Click Next.

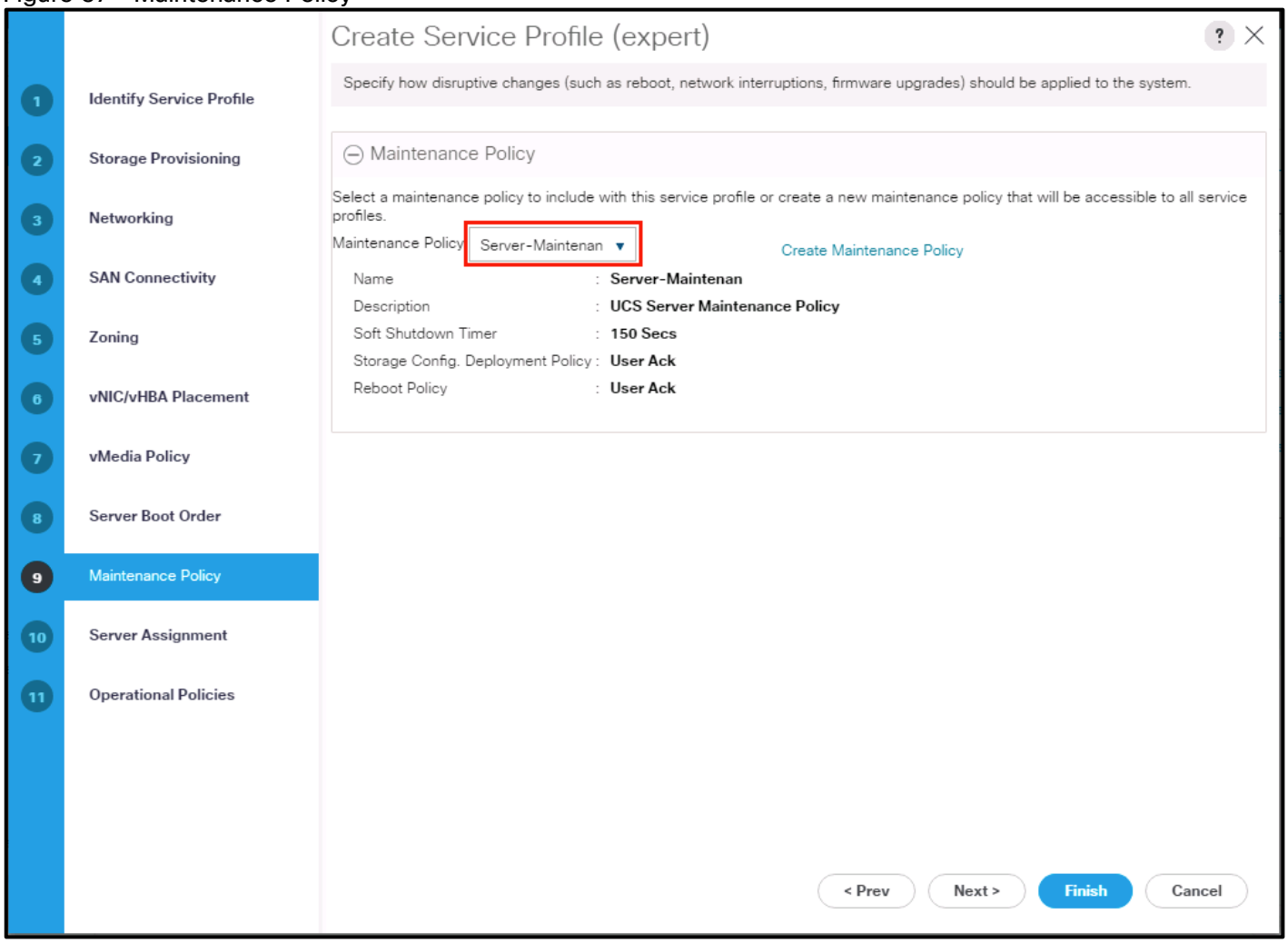


Maintenance Policy

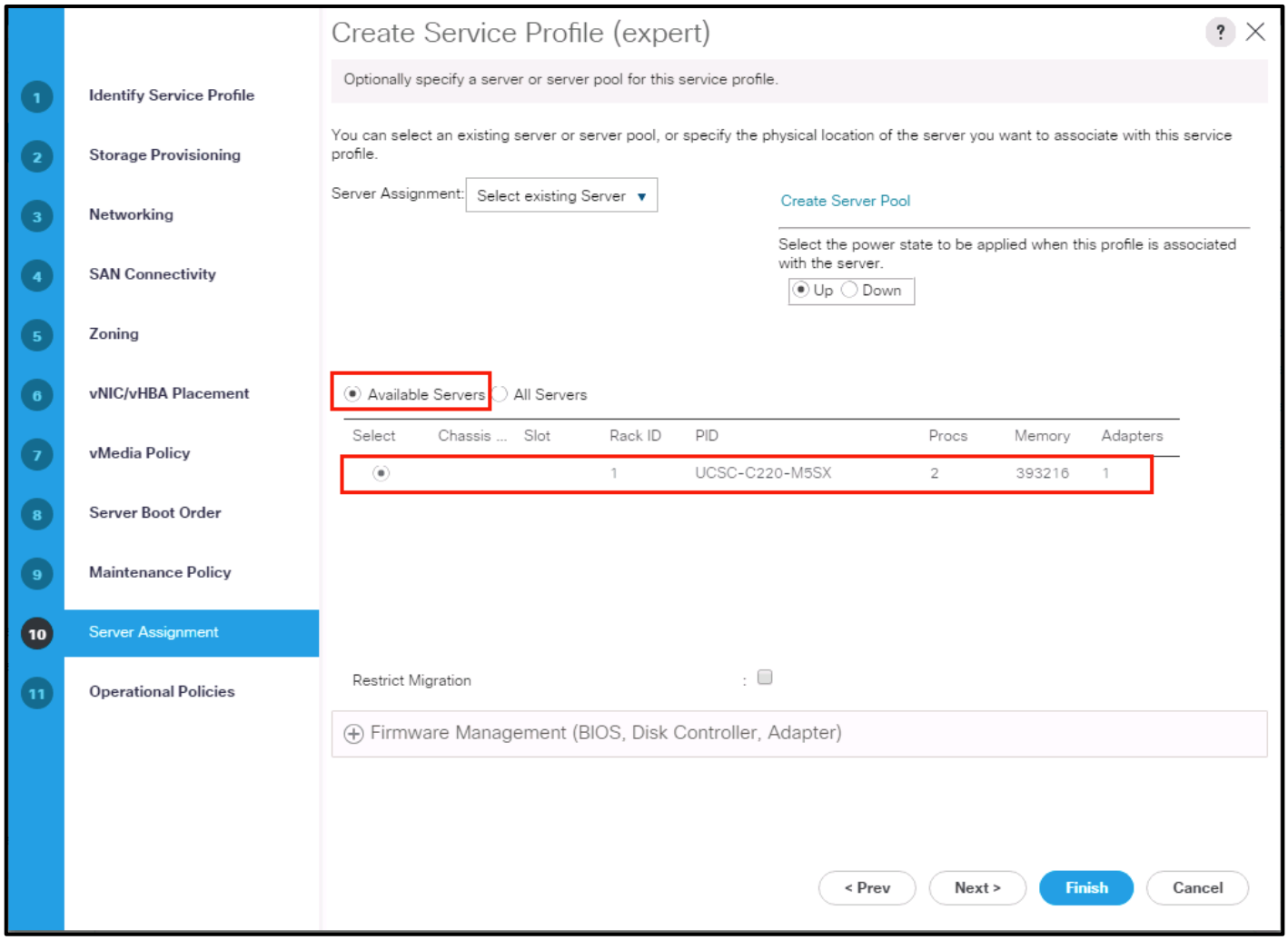
To configure the maintenance policy, follow these steps:

1. Select the Maintenance Policy you created before under Maintenance Policy.

Figure 57 Maintenance Policy



2. Click **Next**.
3. From the Server Assignment drop-down list, choose Select existing Server.
4. Click the Available Servers radio button.
5. From the Server list, select Rack ID 1 radio button for the Cisco UCS C220 M5 Server. This will Associate the service profile.



6. Click **Next**.

Operational Policies

To configure the operational policies, follow these steps:

1. Click **Finish** and then click **OK** and click Yes.
2. After Successful creation of HA_Proxy-Node Service profile, the Cisco UCS C220 M5 server will start the Service profile association.

Create Port Channel for Network Uplinks

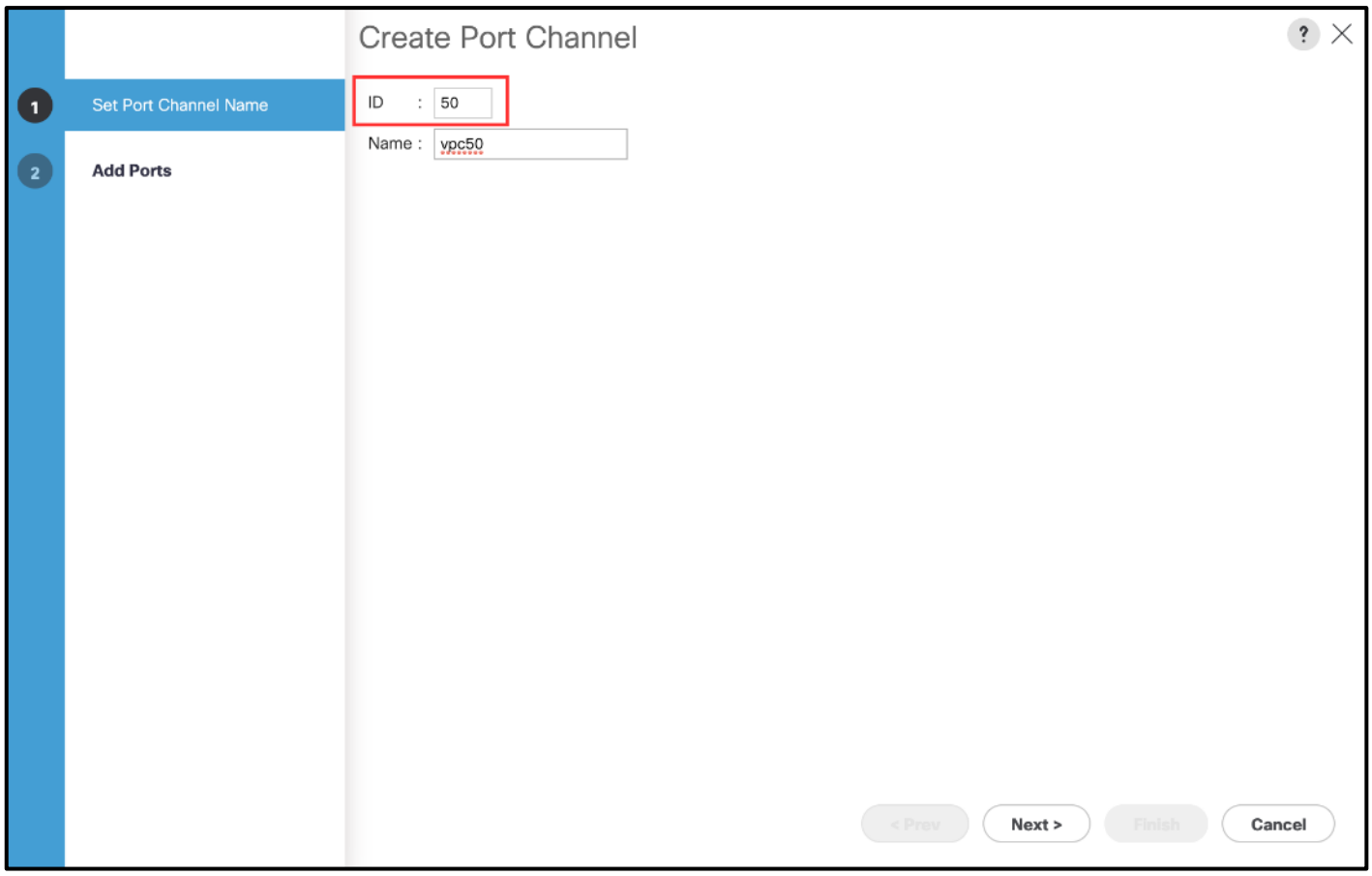
Create Port Channel for Fabric Interconnect A/B

To create Port Channels to the connected Nexus C9336C-FX2 switches, follow these steps:

1. Select the **LAN** tab in the Cisco UCS Manager GUI.
2. Go to LAN > LAN Cloud > Fabric A > Port Channels and right-click Create Port Channel.

3. Type in **`ID 50`**.

4. Type in **`vPC50`** in the `Name` field.

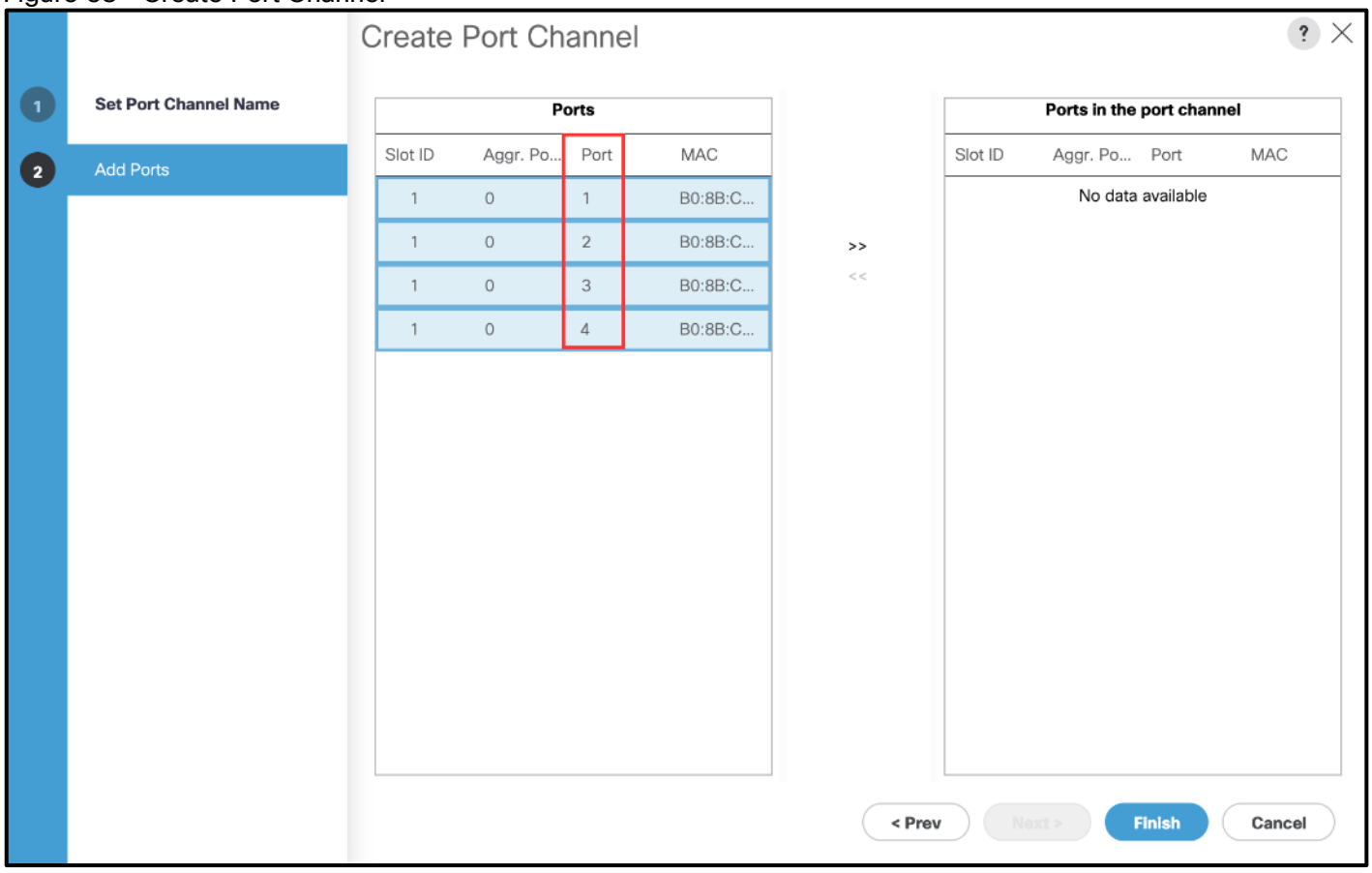


5. Click Next.

6. Select the available ports on the left **`1-4`** and assign them with >> to **`Ports in the Port Channel`**.

7. The Add Ports window will prompt you to confirm the selection, click Yes.

Figure 58 Create Port Channel



8. Click **Finish** and then click **OK**.

9. Repeat steps 1-8 for Fabric B under LAN > LAN Cloud > Fabric B > Port Channels and right-click Create Port Channel.

10. Type in **ID 51**.

11. Type in **vPC51** name in the Name field.

12. Click **Next**.

13. Select the available ports on the left **1,2,3 and 4** and assign them with >> to **Ports in the Port Channel**.

14. Click **Finish** and then click **OK**.

The formal setup of the Cisco UCS Manager environment and both Cisco Nexus C9336C-FX2 switches is completed and next is the installation of the Red Hat Enterprise Linux 7.6 Operating System.

Install Red Hat Enterprise Linux 7.6 Operating System

This section provides the detailed procedures to install Red Hat Enterprise Linux 7.6 on Cisco UCS C220 M5 and Cisco UCS C240 Rack Server. The installation uses the KVM console and virtual Media from Cisco UCS Manager.

This requires RHEL 7.6 DVD/ISO media for the installation.

Install RHEL 7.6 on Cisco UCS C220 M5 and Cisco UCS C240 M5 Server

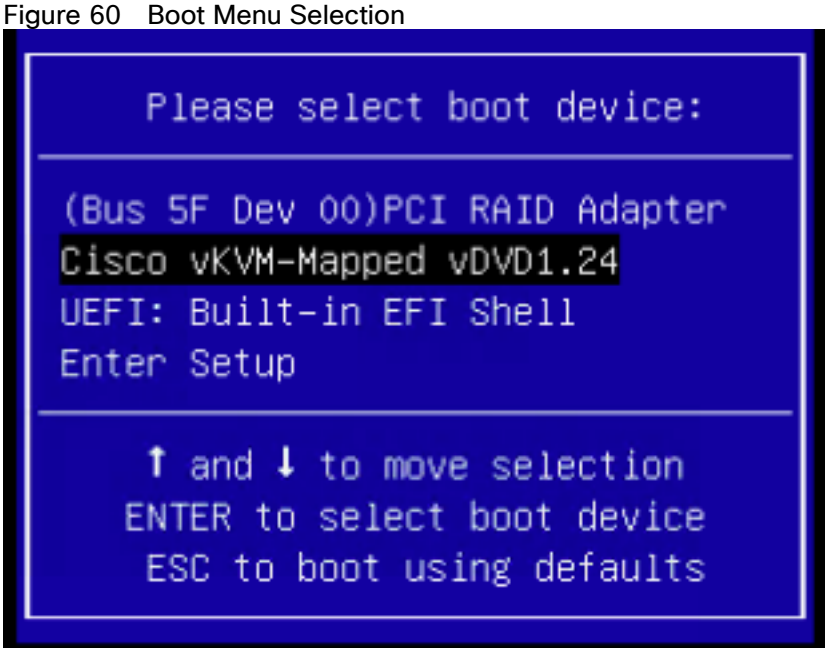
To install Red Hat Linux 7.6 operating system on Cisco UCS C220 M5, follow these steps:

1. Log into the Cisco UCS Manager and select the **Equipment** tab.
2. Go to Equipment > Rack-Mounts > Server > Server 1 (HA-Proxy) and right-click KVM Console.
3. Launch KVM Console.
4. Click the **Activate Virtual Devices** in the Virtual Media tab.
5. In the UCS KVM window, select the Virtual Media tab and then click **CD/DVD**.
6. Click Choose File and Browse to the Red Hat Enterprise Linux 7.6 installation ISO image and select then click **Map Drive.**

Figure 59 Red Hat Enterprise Linux 7.6 ISO image

7. In the KVM window, select the **Macros > Static Macros > Ctrl-Alt-Del** button.
8. Click **OK** and then click **OK** to reboot the system.
9. In the boot screen with the Cisco Logo, press **F6** for the boot menu.
10. When the Boot Menu appears, select **Cisco vKVM-Mapped vDVD1.24**

Figure 60 Boot Menu Selection



11. When the Red Hat Enterprise Linux 7.6 installer appears, press the Tab button for further configuration options.

12. At the prompt type:

```
inst.ks=ftp://192.168.10.220/storage-node1.cfg net.ifnames=0 biosdevname=0
ip=192.168.10.240::192.168.10.1:255.255.255.0:storage-node1:eth1:none
```

We prepared a Linux Kickstart file with all necessary options for an automatic install. The Kickstart file is located on a server in the same subnet.

The Kickstart file for the Cisco UCS C220 M5 server for HA-Proxy is in Appendix A. This Kickstart file for the Cisco UCS C240 M5 Server for Storage Nodes is in Appendix B.

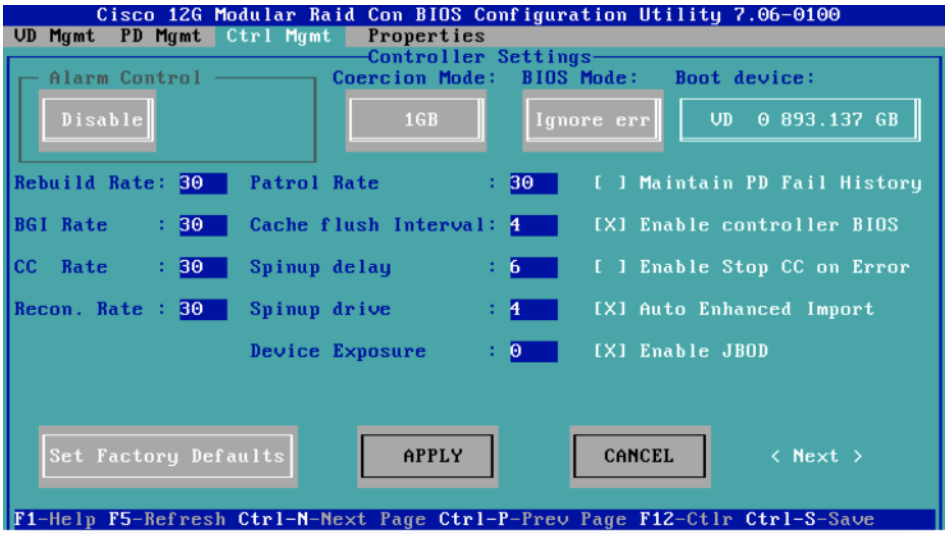
13. Repeat steps 1-12 to install RHEL7.6 on all the UCS C240 M5 Rack servers.

Post reboot, if the Operating System does not boot, it might be because the boot flag is not set for the RAID drive. You'll need to set it manually, to do so, follow these steps:

1. Reboot the host.

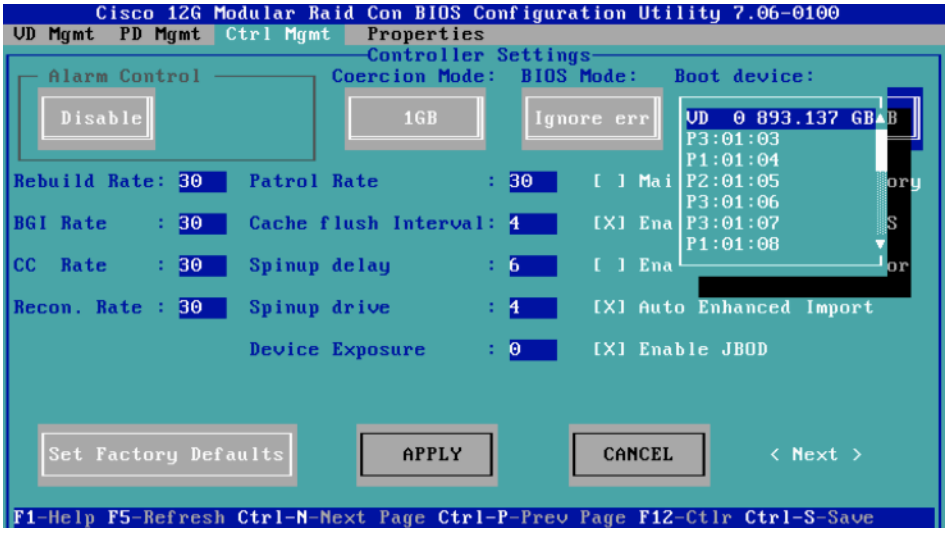
2. Enter Cisco 12G Modular RAID controller BIOS Configuration Utility by pressing Ctrl-R.

3. Press Ctrl-N and select Ctrl Mgmt.



4. Navigate to Boot Device by pressing Tab.

5. Press Enter and select the correct virtual drive to boot from



6. Press Ctrl-S to save

7. Reboot the server again.

Cloudian Hyperstore Preparation

Once the OS is installed, login as root with the defined password in the kickstart file. The Cloudian Hyperstore installation will be completed as root.

Software Version

This CVD guide is based on Cloudian HyperStore 7.1.4 but will support any version upgrades for 7.x.

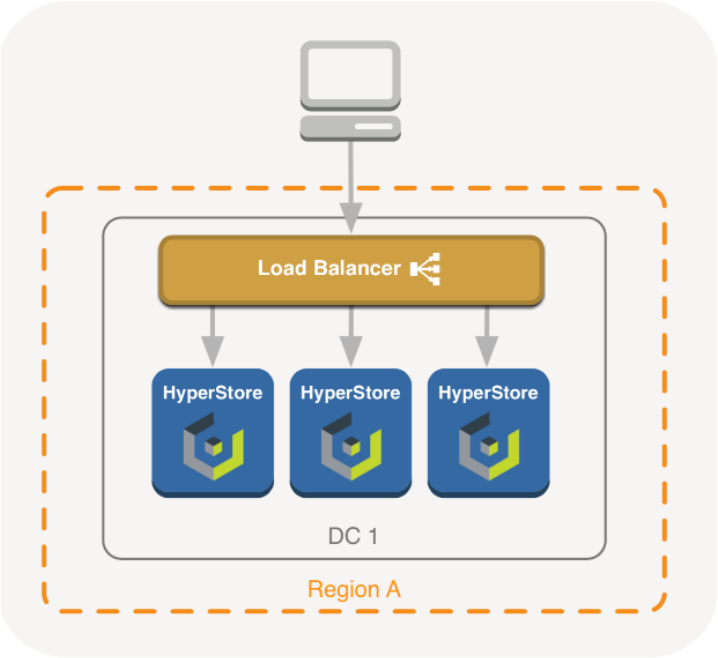
Load-Balancer Requirements

Cloudian HyperStore requires using a Load-Balancer or a VIP manager to ensure high-availability across the platform. Cloudian HyperStore supports working with most load-balancers and VIP managers.

In the Cloudian HyperStore High Availability architecture, the cluster is typically fronted by a Load Balancer. The purpose of the Load Balancer is to monitor the health of a node so that traffic is not routed to a node that is unhealthy or offline, as well as balance the workload evenly across cluster nodes. There must be a component that can redirect the work and there must be a mechanism to monitor for failure and transition the system if an interruption is detected. Without a Load Balancer, a node that is offline would still receive requests from clients. Those requests would then just fail. In general, Load Balancers will distribute requests to nodes that belong to a pool of available service members. A Load Balancer will also perform frequent health checks against pool member nodes to ensure they are healthy and able to support new traffic.

All Cloudian HyperStore S3, Admin-API and CMC services should be configured with a Load Balancer to ensure any kind of High Availability. There are many Load Balancing solutions that are available for use. Commercial examples are F5, A10 Networks, KEMP, Loadbalancer.org and Citrix NetScaler. Open source Load Balancer software exist as well, one popular example is HAProxy. Most of the Load Balancing technologies operate in a similar manner, some enterprise solutions and DNS services like Amazon Route53 however also include support for GEO based load distribution (known as Gl Global Server Load Balancing).

Figure 61 Load Balancing Example



Concepts of Load Balancing

Round-Robin DNS

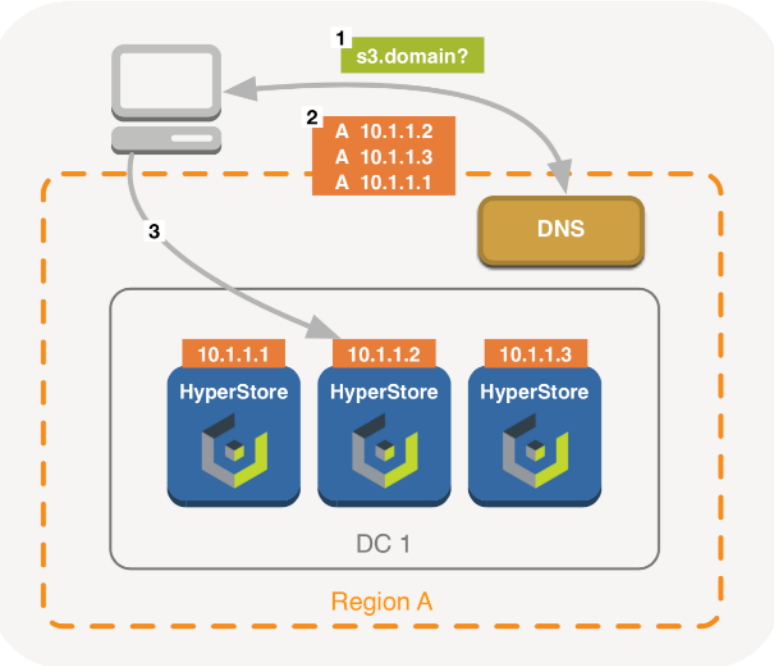
Round-robin DNS (RR-DNS), is a method where a series of A records is registered in DNS, each by the same name. The following is an example:

Whenever a client requests s3.domain in DNS, the reply will contain a certain order of the above records. The next request will however be answered with a rotated list of those records. This way, traffic is automatically balanced across the mentioned addresses.

s3.domain IN A 10.1.1.1 s3.domain IN A 10.1.1.2
s3.domain IN A 10.1.1.3

RR-DNS is very simple to implement as DNS is already available everywhere, but without combining it with other HA solutions, it isn't very useful in itself; If any of those nodes are offline, we would still be directing requests to them and so, those requests would fail.

Figure 62 Basic Round-Robing DNS Example

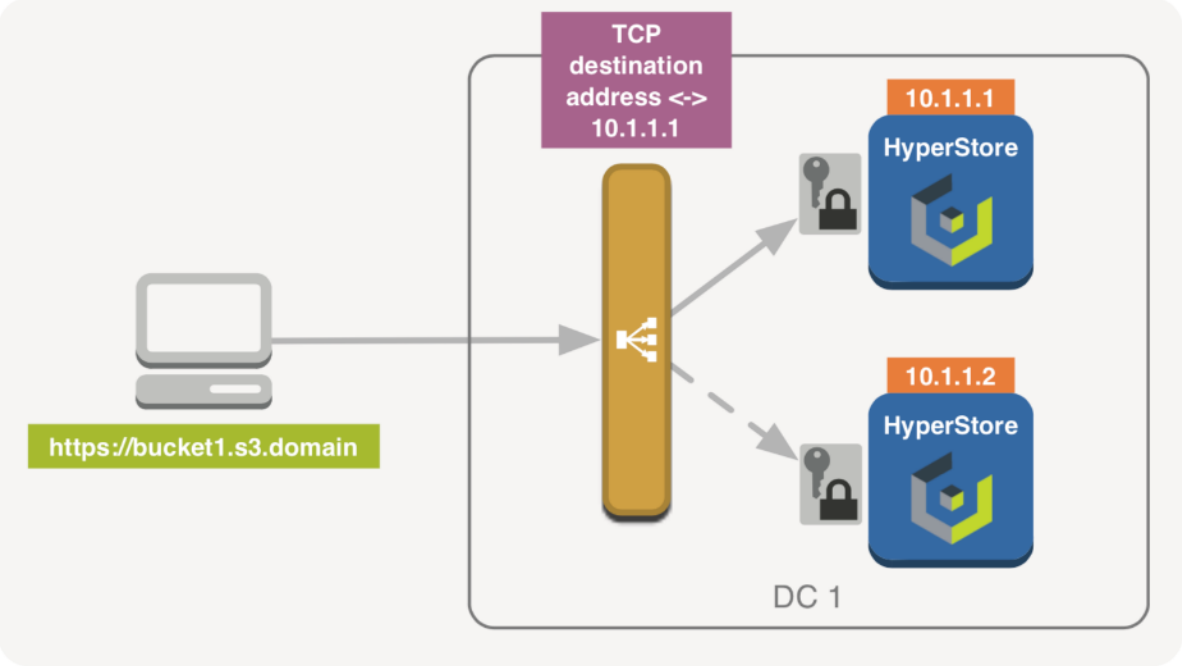


Even if we would dynamically update those DNS records according to the monitored health of the nodes, a pitfall might still be that we are relying on DNS and have no control over caching of those DNS records client-side, or on intermediate caching nameservers (read: low TTL values are often discarded on large DNS resolvers). In scenario's where we do have control over the nameservers our S3 clients are using, RR-DNS and dynamic DNS updates might be a proper solution when implemented correctly. If the Cloudian HyperStore services would be published externally, in most cases it will be a better solution to combine RR-DNS with other HA and/or Load Balancing technologies.

Layer-4 Load Balancing

Layer 4 Load Balancing operates on the transport layer in the OSI model. This means that although the TCP connection is established on the Load Balancer, anything above that (like HTTP) is tunneled across both sides. The Load Balancer can therefor make a balancing decision based on anything in the TCP header, however it cannot look inside the payload or perform more advanced things like injecting a session cookie or inspect URI.

Figure 63 Layer 4 Balancing Example with TLS Termination on HyperStore Nodes Only

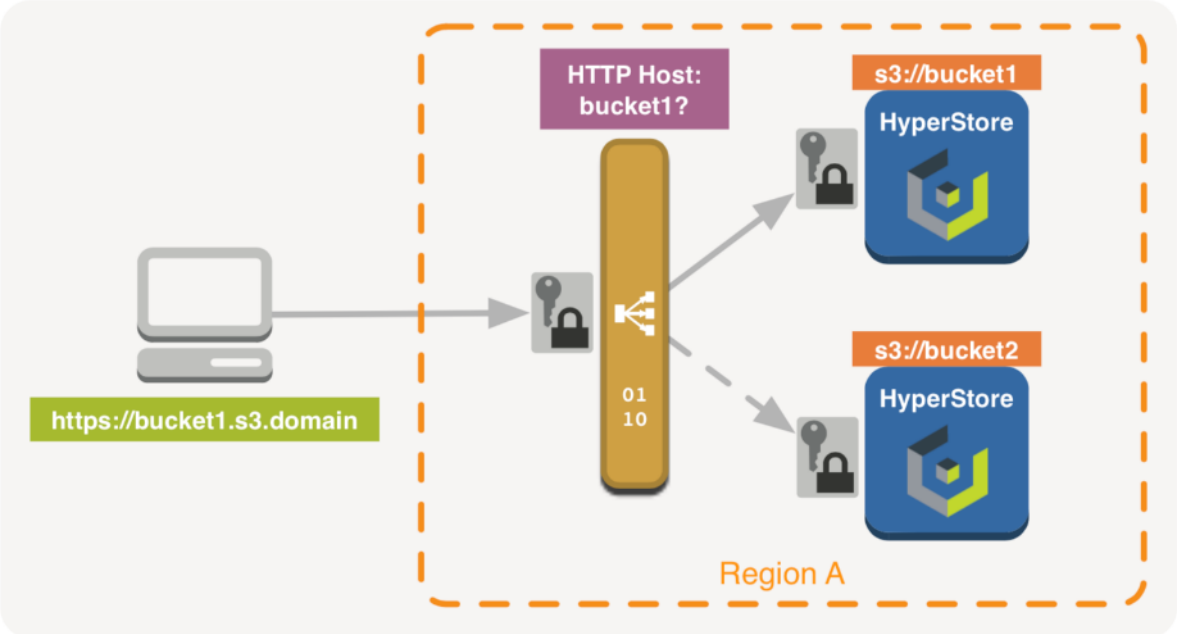


The layer 4 or TCP mode is sufficient and has the advantage since it operates on a much lower level, Load Balancer resources are likely to become the first bottleneck (there are limits of course, scaling the Load Balancing layer itself is covered later). With TLS, termination will happen on the back-end nodes only. This mode fits the scale-out nature of HyperStore best, as all crypto calculation involved with TLS will be spread evenly across all HyperStore nodes as well (although the overhead involved is not nearly what it used to be, due to dedicated instructions (AES) available in many CPUs and optimizations in TLS handshake). No change to SSL certificate management is required, certificates are still only managed on the HyperStore installer node (puppet master) as described in chapter Setting up HTTPS/SSL for the S3 Service in the Cloudian Documentation.

Layer-7 Load Balancing

Layer 7 Load Balancing operates on the application layer in the OSI model. In this HTTP mode, all incoming connections are established on the Load Balancer, the payload is inspected, and new HTTP sessions are created between the Load Balancer and available back-end nodes. This mode is heavier on resources than layer 4 and HTTP mode is typically used when one needs to make balance decisions based on e.g. HTTP headers or inspect cookies to maintain a session to a back-end.

Figure 64 Layer 7 Balancing – SSL Certificates Need to be Managed on LB



With HyperStore, this doesn't add too much value since the balancing algorithm for S3 and API can be random or based on the least connections to a backend node. For CMC requests, you do need to configure stickiness to a backend node but basing that on a source IP address is usually sufficiently random.

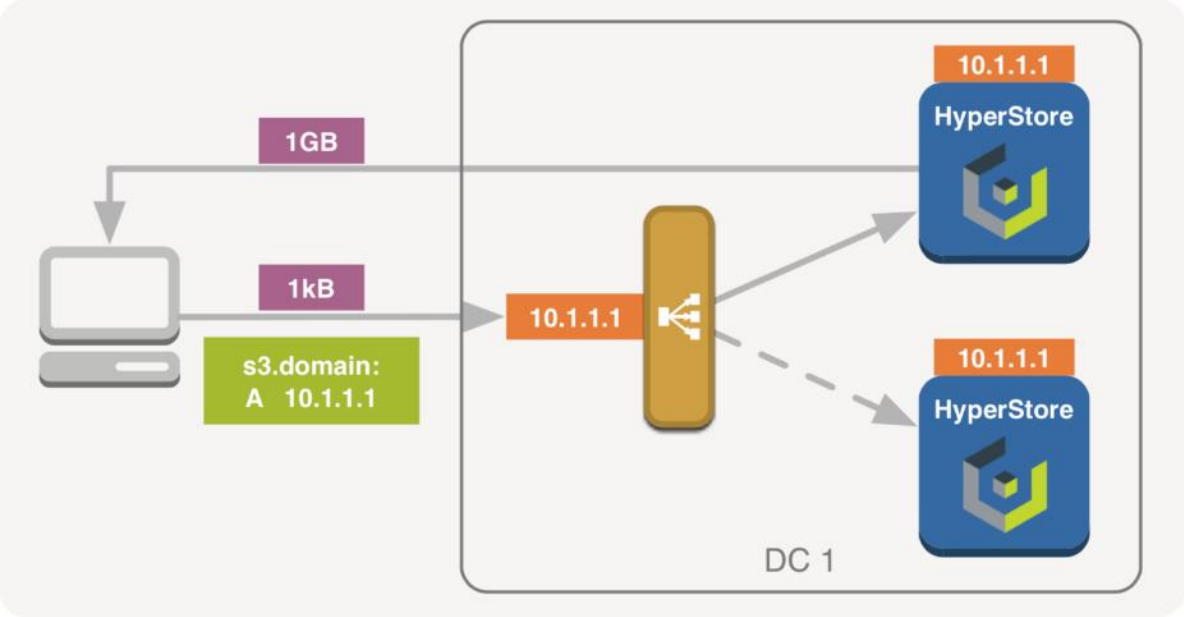
In HTTP mode, SSL certificates need to be managed both on the HyperStore installer node and on all Load Balancers involved. One exception is when traffic from the Load Balancers to the HyperStore nodes is not required to be encrypted. In that case, SSL certificates only need to be maintained on the Load Balancers (Usually referred to as SSL offloading).

Direct Routing

A downside to Load Balancing is that all traffic needs to pass the Load Balancer, both ways, especially with a solution like an Object Store, where you are combining both scaleout and large data transfers. The odds are that a Load Balancer will become the first bottleneck in the chain.

One (partial) solution to that is a concept called Direct Routing, also known as Direct Server Return (DSR). With Direct Routing, a backend node does not rely on the Load Balancer to send its reply to the client. Instead, the backend nodes have the VIP (Virtual IP) or Load Balancer address attached to their local interface (ARP replies for that address will need to be switched off) and are able to send a TCP reply directly back to the client with the source address (the VIP) the client is expecting. One example is the LVS project, but some commercial Load Balancers also support Direct Routing.

Figure 65 1GB GET Object Example with Direct Routing



This relieves the Load Balancer from (often large) GET request replies returning to the client, however all PUT requests still need to pass the Load Balancer on their way in. All traffic still passes other network peripherals like Routers, unless S3 clients and HyperStore or the Load Balancers are on the same layer 2 network.

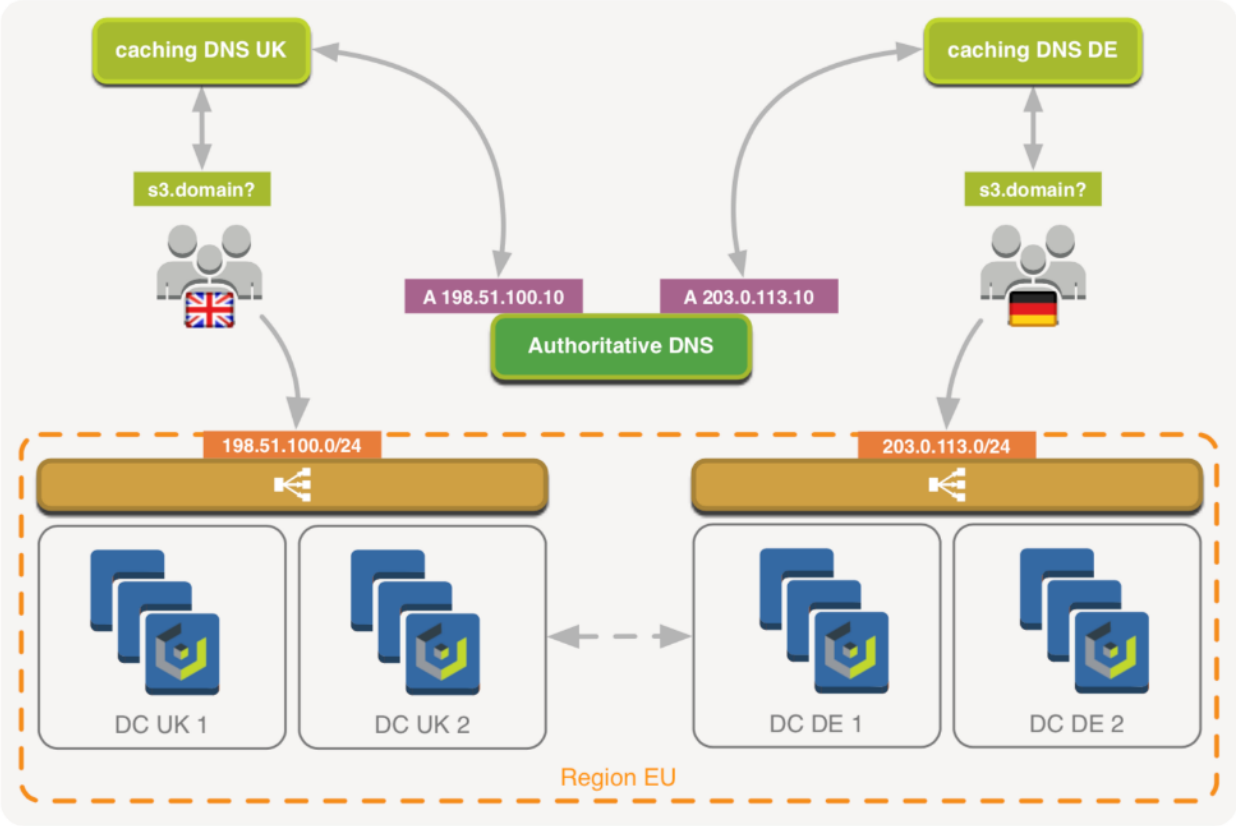
Direct Routing is not a setting you can just turn off and on, instead it's a mechanism acting on multiple levels within a network and often requires some low-level magic to configure correctly.

Direct Routing cannot be combined with L7 balancing or SSL offloading and on Linux it's only available through the Linux Virtual Server project

Global Server Load Balancing

Global Server Load Balancing (GSLB) is a mechanism designed to provide Disaster Recovery, load distribution and/or ensure shortest path or best response between client and data center. Essentially, the technology itself isn't that complicated; Based on, any number of things really but usually, geographical location and/or availability of a data center, a user receives a DNS reply which routes the request accordingly. Although GSLB does provide load distribution and is present in several commercially available Load Balancers, GSLB is built on top of DNS and is not necessarily, or typically, a physical Load Balancer.

Figure 66 GSLB Example – Geographically Spread Balancing



In this example (Figure 66), users from the UK would receive a DNS record pointing to UK-based data centers and likewise, German users receive an address pointing to a German DC. Just like directing users to specific geographical areas, GSLB can also be used to directly return a pool of addresses of healthy HyperStore nodes instead. GSLB solutions typically monitor health of a destination and manage the DNS records returned by an authoritative DNS nameserver, either for availability purposes, balancing, localization or a combination. A viable architecture could for example be based on Amazon's Route53 for handling Geo- based DNS and have traditional Load Balancers in front of HyperStore in each location.

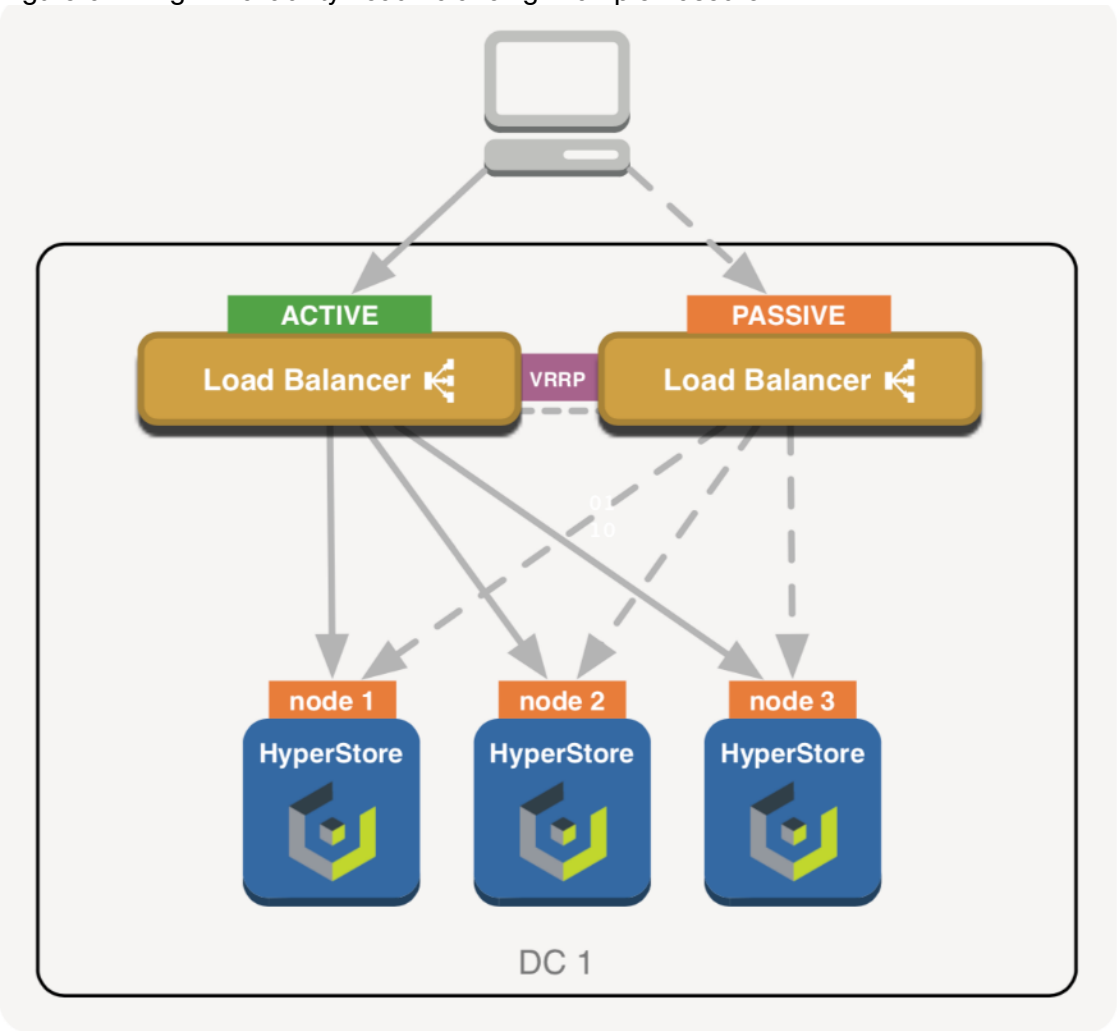
Like with RR-DNS or anything based on DNS, aggressive caching of DNS records may become an issue depending on the overall architecture and infrastructure between client and S3 service.

High Availability

Load Balancing does not necessarily equal High Availability. When using multiple backends but just a single Load Balancer, the Load Balancer becomes the Single Point of Failure. This is usually resolved by adding another Load Balancer and enabling a failover mechanism between both Load Balancers. Most enterprise Load Balancers will support such a failover mechanism, often based on protocols like Virtual Router Redundancy Protocol (VRRP).

An open source solution often deployed in combination with LVS or HAProxy is Keepalive. Based on VRRP, Keepalive can be configured to allow a pool of floating IP addresses where each Virtual IP (VIP) will only be active on a single node at any given time. Whenever a node would fail, in this case a Load Balancer, any VRRP address attached to that node would be seamlessly migrated to any of the remaining nodes.

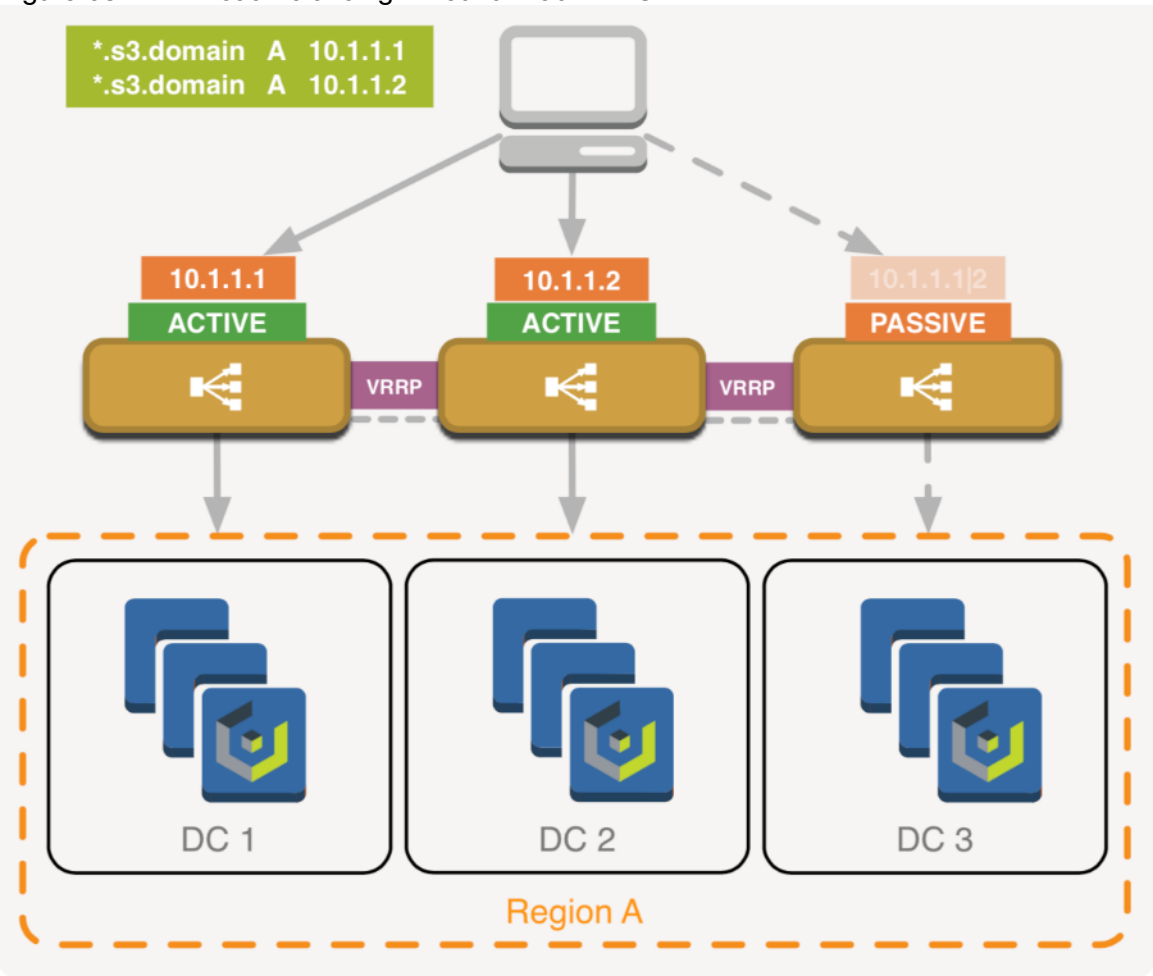
Figure 67 High Availability Load Balancing Example Based on VRRP



When working with VRRP-like protocols, make sure Multicast is allowed on the relevant ports, either by disabling IGMP Snooping for those ports or joining the correct IGMP group.

When deploying a Load Balancing layer in front of HyperStore, especially when that layer is spread across multiple data centers, a single active Load Balancer might not be preferred or even viable. This is where Round-Robin DNS is actually very useful; When scaling the Load Balancing layer horizontally and using multiple Load Balancers in an Active-Active setup, you can now make use of RR-DNS so that multiple DNS records are spread evenly across multiple, active Load Balancers. Effectively creating an N+1 setup on Load Balancer level.

Figure 68 N+1 Load Balancing + Round-Robin DNS



Why not run all Load Balancers in an Active mode and direct traffic to them? It's usually a best practice to keep an N+1 setup, especially when N is a relatively low number, since you won't have any real insight in how your remaining Load Balancers will handle the load, number of connections, and so on, until any one of the Load Balancers fail (and it turns out they weren't able to handle, for example, 33% more connections or additional throughput). As with any High-Availability technology, setting it up properly can get rather complex and usually involves configuration on multiple levels, often including switch port fine-tuning as well.

For that reason, this document does not provide in-depth details on how to set up high availability between your Load Balancers but does explain the technologies involved and some basic examples.

Load Balancing HyperStore

HyperStore Services

The HyperStore services that should be balanced are: S3, Cloudian Management Console (CMC) and Admin-API. Besides advertising those services to any clients, all HyperStore nodes within the cluster will also benefit and make use of S3 and Admin-API being highly available. All other, internal services like Cassandra and Redis are cluster-aware, meaning that as part of the HyperStore installation they've received topology information and know how to communicate directly to all other nodes. These internal services do not need to be taken into consideration when architecting Load Balancing within your network.

The only HyperStore services that need to be balanced are: S3, CMC and Admin-API.

Figure 69 Balancing HyperStore Services

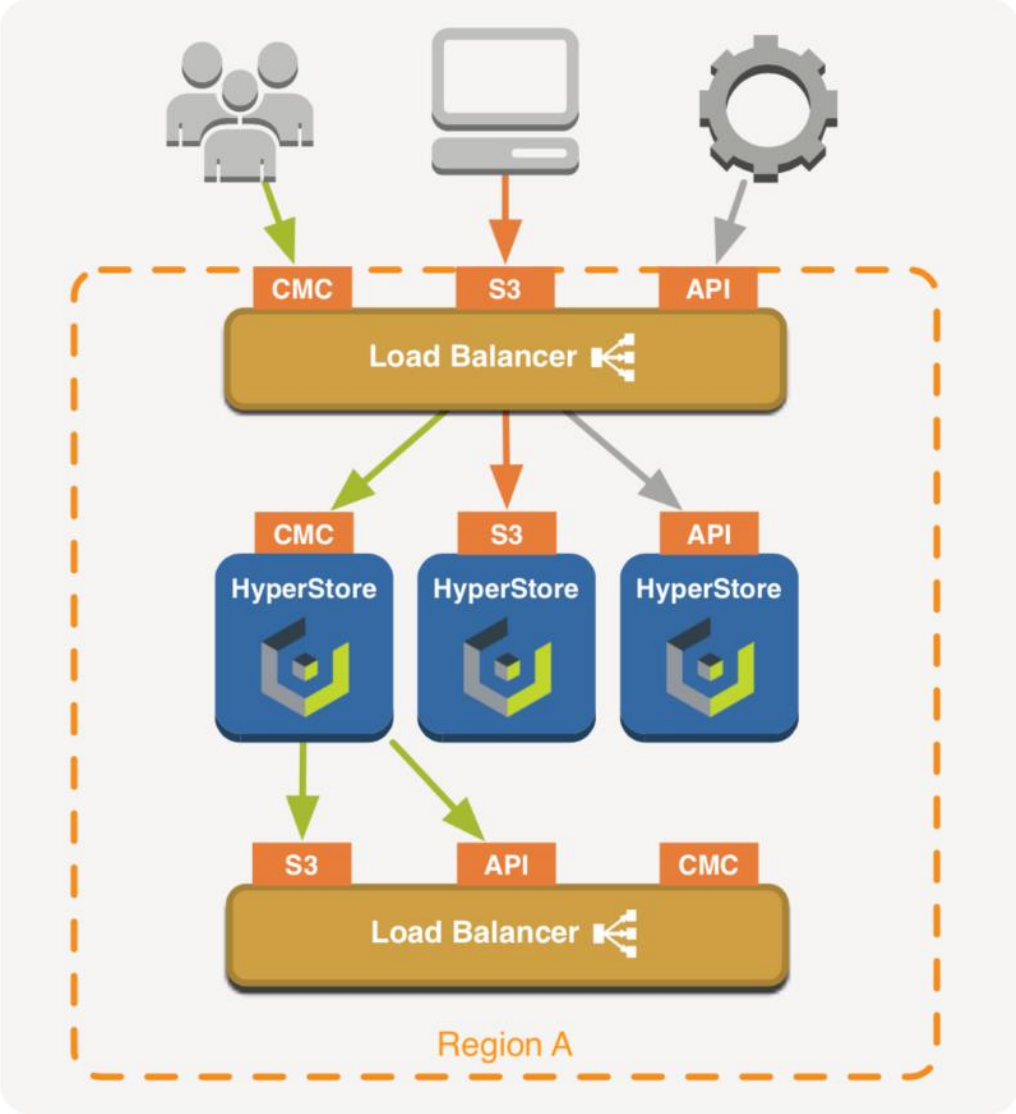
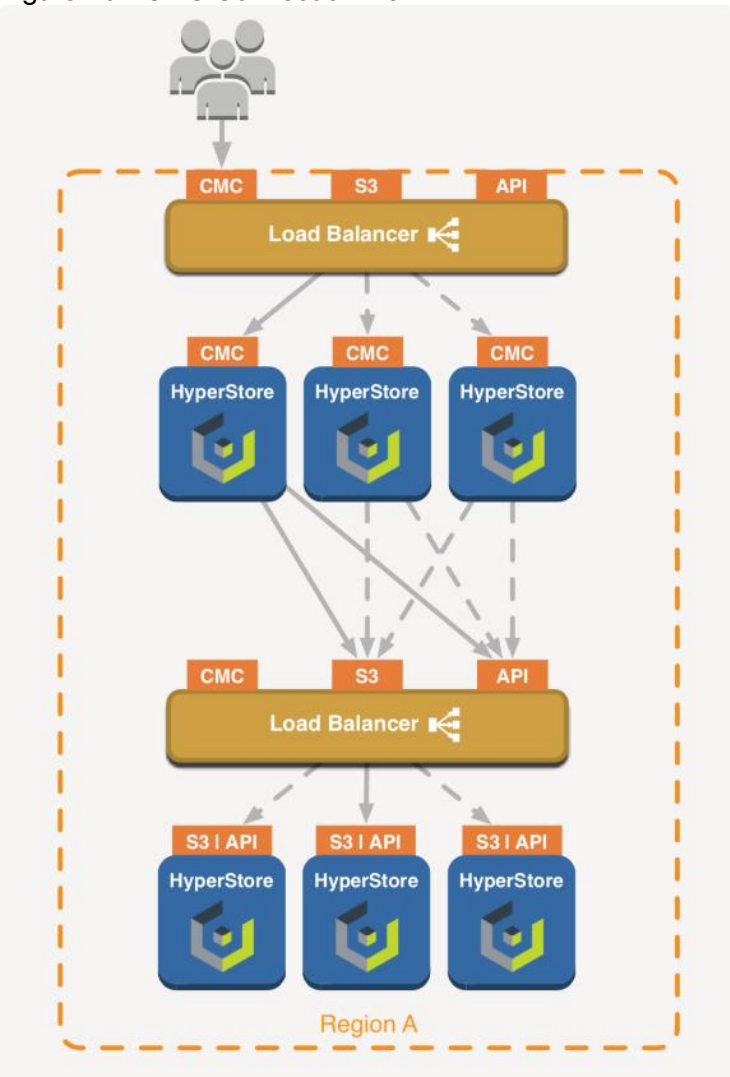


Figure 69 illustrates a simplified view on how different clients connect to S3, CMC and the Admin-API. As you can see, the Cloudian Management Console can be considered a client as well. The CMC connects via the Load Balancer to both S3 and Admin-API services, which should in turn be balanced across all HyperStore nodes.

Note that all S3, CMC, and Admin-API services are running on every HyperStore node. Figure 70 illustrates a client connecting to the CMC.

Figure 70 CMC Connection Flow



Make sure the Admin-API is made highly available, since the CMC communicates directly with the API

HyperStore Configuration

For detailed configuration instructions on how to prepare HyperStore and DNS for High Availability please see chapter DNS Set-Up in the Cloudian documentation. You will need to make sure all S3, CMC, and Admin-API DNS records point to the Load Balancer.

When installing HyperStore with option configure-dnsmasq (no-dnsmasq is default), a simple resolver will be installed on each HyperStore node, and all required records will be added to dnsmasq automatically. Note that for production use, it is not recommended to install with dnsmasq enabled. Instead, you need to make sure the following records are all present in DNS before installation:

In this single Region example, 10.1.1.10 is the IP address attached to the Load Balancer, region1 is the Region name and domain is the Domain name. During installation of HyperStore these hostnames can all be customized.

When installing with dnsmasq you need to do the following:

1. Customize template: /etc/cloudian- <version>-puppet/modules/dnsmasq/templates/dnsmasq.conf.erb to reflect the above DNS example.

2. Push the update to the cluster.

3. Restart service dnsmasq.

HAProxy Examples

HAProxy – Basic Configuration

As mentioned earlier, HAProxy is a widely used Load Balancer solution, used by, for example, Twitter, Amazon AWS, GitHub and Netflix and is available as Open Source, Enterprise version, as an appliance (ALOHA) and also available in the Loadbalancer.org appliances. HAProxy is known to be very performant, stable and feature-rich (for in an-depth explanation of all options, features and syntax please review to the online documentation).

Installing HAProxy is very easy. When installing on a RedHat-based distribution, all that is required is to run the following commands:

```
sudo yum update
sudo yum –y install haproxy systemctl enable haproxy.service
```

When installing on a Debian-derivative, the command is:

```
sudo apt-get update
sudo apt-get install haproxy
```

Set ENABLED=1 in /etc/default/haproxy :

s3-region1.domain IN A 10.1.1.10 *.s3-region1.domain s3-website-region1.domain *.s3-website-region1.domain IN A IN A IN A 10.1.1.10 10.1.1.10 10.1.1.10
s3-admin.domain IN A 10.1.1.10 cmc.domain IN A 10.1.1.10

Move the default HAProxy configuration file /etc/haproxy/haproxy.cfg aside and create a new one. This document assumes there are three Cloudian nodes to balance the load across. With Cloudian HyperStore all HTTP REST API services run on every node so the configuration is quite simple. The node IP's here are assumed to be 10.1.1.11, 10.1.1.12, and 10.1.1.13.

Configuration – Global Section

```
global
log /dev/log local0
log /dev/log local1 notice chroot /var/lib/haproxy user haproxy
group haproxy spread-checks 5 tune.bufsize 32768 tune.maxrewrite 1024 maxconn 16384
daemon
```

Since you are running inside a chroot environment, the local syslog server would need to create a listening socket in /var/lib/haproxy/dev. In rsyslog the syntax would be: $AddUnixListenSocket /var/lib/haproxy/dev/log.

Configuration – Defaults Section

```
   defaults
       log global
mode tcp
maxconn 8192 timeout connect 5s timeout client 1m timeout server 1m timeout check 5s
balance leastconn
```

Add option tcplog to the defaults section to log every connection to each front-end

Configuration – Admin Statistics

```
# admin stats on port 8080 listen stats
bind :8080
mode http
stats enable
maxconn 128
stats uri /
stats realm Haproxy\ Statistics stats auth admin:public
```

For production use the statistics page should be reachable over TLS only and a proper password should be configured.

Configuration – Backend CMC

```
# Cloudian CMC
listen cmc.cloudian-hyperstore
bind :8888
mode http
http-request replace-value Host (.*):8888 \1:8443 http-request redirect code 302
location
https://%[hdr(host)]%[capture.req.uri]
listen https.cmc.cloudian-hyperstore bind :8443
mode tcp
stick-table type ip size 100k expire 30m
stick on src
option httpchk HEAD /Cloudian/login.htm
description Cloudian HyperStore CMC - HTTPS
server cloudian-node1 10.1.1.11:8443 check check-ssl verify none
inter 5s rise 1 fall 2
server cloudian-node2 10.1.1.12:8443 check check-ssl verify none
inter 5s rise 1 fall 2
server cloudian-node3 10.1.1.13:8443 check check-ssl verify none
inter 5s rise 1 fall 2
```

CMC balance algorithm needs to be sticky (stick-table, stick on src)

Configuration – Backend S3 HTTP

```
# Cloudian S3 services
listen s3.cloudian-hyperstore
bind :80
mode tcp
option httpchk HEAD /.healthCheck
description Cloudian HyperStore S3
server cloudian-node1 10.1.1.11:80 check inter 5s rise 1 fall 2 server cloudian-
node2 10.1.1.12:80 check inter 5s rise 1 fall 2 server cloudian-node3 10.1.1.13:80
check inter 5s rise 1 fall 2
```

HyperStore S3 includes a health check page, reachable over HTTP method HEAD.

Configuration – Backend S3 HTTPS

```
# Cloudian S3 services - HTTPS listen https.s3.cloudian-hyperstore
bind :443
mode tcp
option httpchk HEAD /.healthCheck
description Cloudian HyperStore S3 - HTTPS
server cloudian-node1 10.1.1.11:443 check check-ssl verify none
inter 5s rise 1 fall 2
server cloudian-node2 10.1.1.12:443 check check-ssl verify none
inter 5s rise 1 fall 2
server cloudian-node3 10.1.1.13:443 check check-ssl verify none
inter 5s rise 1 fall 2
```

When a CA-Verified certificate is used for S3, the verify none should be omitted.

Configuration – Backend Admin API

```
# Cloudian Admin-API
listen api.cloudian-hyperstore
bind :19443
mode tcp
option httpchk HEAD /.healthCheck HTTP/1.0\r\nAuthorization:\ Basic\
c3lzYWRtaW46cHVibGlj
description Cloudian HyperStore API
server cloudian-node1 10.1.1.11:19443 check check-ssl verify none inter 5s rise 1
fall 2
server cloudian-node2 10.1.1.12:19443 check check-ssl verify none inter 5s rise 1
fall 2
server cloudian-node3 10.1.1.13:19443 check check-ssl verify none inter 5s rise 1
fall 2
```

When you have customized the Admin-API credentials make sure to replace the base64 encoded string shown in the example (which is the base64 version of the default credentials sysadmin:public).

The encoded credentials can be generated on the command line: echo –n <username>:<password> | base64

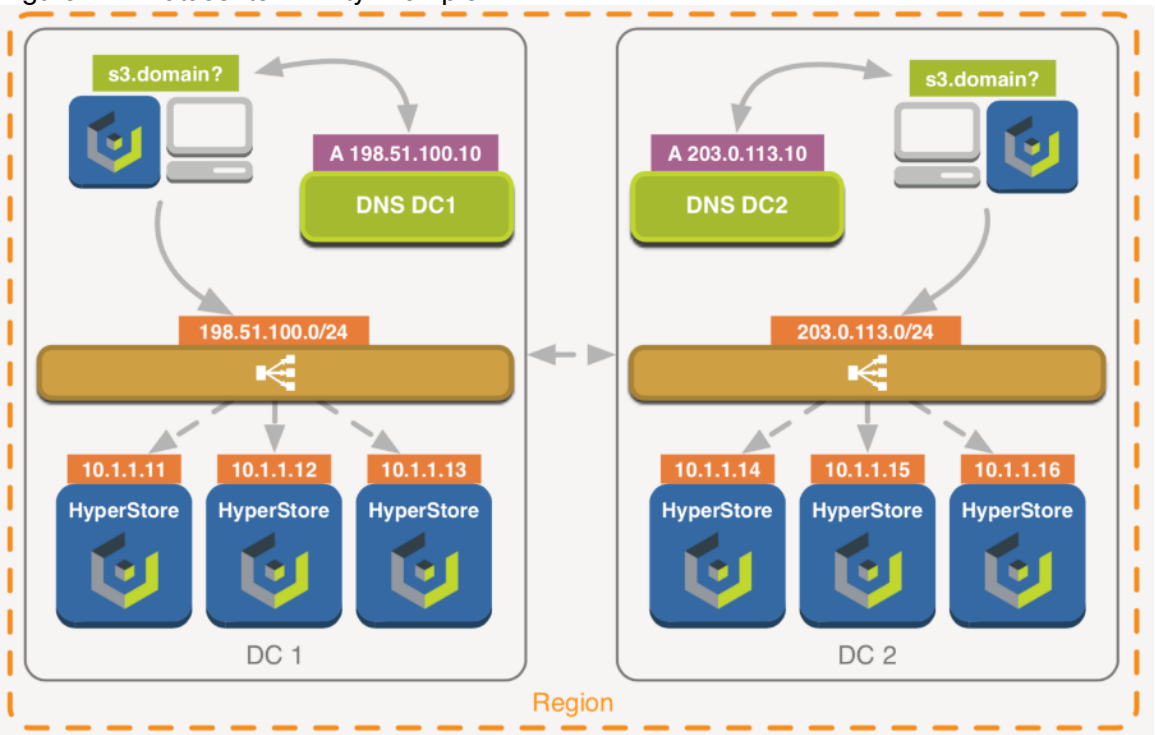
HAProxy – Location Affinity

HAProxy by itself does not provide any GSLB-like capabilities, and the following example won't be very useful when HyperStore is running public services, for example, a publicly reachable S3 endpoint, let's say the public Storage as a Service use case. However, imagine all your S3 clients are internal applications within a single S3 Region which spreads across multiple data centers, and you want to use some form of location affinity because you don't prefer an application in DC1, connecting to an S3 endpoint in DC2 half of the time.

When using a single S3 endpoint across multiple data centers we could create complex rules, inspect HTTP headers and base a routing decision on, for example, the name of the Bucket or even the subnet the application is connecting from by defining ACL's and multiple back-ends. However, this might not always help every scenario and may also create a configuration more complex than desired.

Instead, what you could do is simply run different DNS nameservers in both data centers and register all service records to point to the closest Load Balancer. This way a client connecting from within DC 1 would always be directed to the Load Balancer in DC 1. The same applies to DC 2.

Figure 71 DatacenterAffinity Example



The nameservers do not necessarily need to run on dedicated servers and could be installed on the Load Balancer nodes. The following is an example setup of the S3 service in HAProxy, combined with installing and running the light-weight DNS (amongst others) server dnsmasq.

HAProxy Configuration – S3 DC1

```
listen s3.cloudian-hyperstore bind :80
mode tcp
option httpchk HEAD /.healthCheck
description Cloudian HyperStore S3 DC1
server cloudian-node1 10.1.1.11:80 check inter 5s rise 1 fall 2 server cloudian-
node2 10.1.1.12:80 check inter 5s rise 1 fall 2 server cloudian-node3 10.1.1.13:80
check inter 5s rise 1 fall 2 server cloudian-node4 10.1.1.14:80 check inter 5s rise
1 fall 2
backup
server cloudian-node5 10.1.1.15:80 check inter 5s rise 1 fall 2
backup
server cloudian-node6 10.1.1.16:80 check inter 5s rise 1 fall 2
backup
```

HAProxy Configuration – S3 DC2

```
listen s3.cloudian-hyperstore bind :80
mode tcp
option httpchk HEAD /.healthCheck
description Cloudian HyperStore S3 DC1
server cloudian-node4 10.1.1.14:80 check inter 5s rise 1 fall 2 server cloudian-
node5 10.1.1.15:80 check inter 5s rise 1 fall 2 server cloudian-node6 10.1.1.16:80
check inter 5s rise 1 fall 2 server cloudian-node1 10.1.1.11:80 check inter 5s rise
1 fall 2
```

```
backup
server cloudian-node2 10.1.1.12:80 check inter 5s rise 1 fall 2
backup
server cloudian-node3 10.1.1.13:80 check inter 5s rise 1 fall 2
backup
```

By adding the remote nodes as backup back-end servers, whenever local HyperStore nodes fail, the remote HyperStore nodes become active. Alternatively, a single backup entry could be used by pointing it to the VIP of the remote Load Balancer.

Install DNSMASQ

To install dnsmasq on RedHat-based distributions, issue the following commands:

```
sudo yum update
sudo yum –y install dnsmasq systemctl enable dnsmasq.service
```

DNSMASQ Configuration

Leave the default configuration as-is. Make sure that you're not allowing recursion to requests from the internet. As recursion is allowed by default in dnsmasq, you can either set the interface option to only listen on the specified, internal interface, or an intermediate firewall should be configured not to allow external DNS traffic to pass in. To listen only on a specified interface, create a file /etc/dnsmasq.d/custom.conf and add the following inside (adjust Interface name, VLAN 10 in this example):

```
interface=enp0s3.10 bind-interfaces
```

Now add the following configuration to that same file, one in each data center (adjust addresses, region and domain to match your environment)

DNSMASQ Configuration – DC1

```
address=/.s3-region.domain/198.51.100.10 address=/s3-region.domain/198.51.100.10
address=/.s3-website-region.domain/198.51.100.10 address=/cmc.domain/198.51.100.10
address=/s3-admin.domain/198.51.100.10
```

DNSMASQ Configuration – DC2

```
address=/.s3-region.domain/203.0.113.10 address=/s3-region.domain/203.0.113.10
address=/.s3-website-region.domain/203.0.113.10 address=/cmc.domain/203.0.113.10
address=/s3-admin.domain/203.0.113.10
```

After saving all files, restart both dnsmasq and HAProxy to apply all configurations. At this point, all HyperStore nodes can now be reconfigured to use the Load Balancers (or other nodes if you installed dnsmasq on separate servers) as resolvers. In the same way, all clients and applications within the same data centers that connect to HyperStore, can now be pointed to use dnsmasq as resolver(s) as well (or use DNS delegation in your existing DNS infrastructure).

Some commercial Load Balancers like Citrix NetScaler, F5 GTM and Loadbalancer.org come equipped with GSLB- or GSLB-like features (like location affinity based on subnet of incoming requests).

Proxy Protocol

By using HAProxy and most other Load Balancers or proxies, one will lose the source IP address of the actual client performing the request. For logging purposes, this could be circumvented by using the X-Forwarded-For header sent by the Load Balancer. However, this would only work when using HTTP level balancing but more importantly, it would still not cover more advanced S3 features such as using conditions based on IP addresses in S3 Bucket Policies, and IP addresses and/or subnets used in HyperStore Rating Plan whitelists.

HAProxy can be set up to run in full transparent mode (TPROXY) but that may require recompiling the Linux kernel with TPROXY support, recompiling HAProxy, marking packets with IPtables and adding custom routing tables. Moreover, in full transparent mode all HyperStore nodes will need to use the Load Balancers as their default gateway which is not typically preferred.

However, to avoid too much complexity around this issue, the HAProxy team developed the following PROXY protocol:

The PROXY protocol provides a convenient way to safely transport connection information, such as a client's address across multiple layers of NAT or TCP proxies. It is designed to require minor changes to existing components and to limit the performance impact caused by the processing of the transported information

Between the HAProxy and the backend nodes, an additional PROXY header is passed within a Datagram and processed by the application running on the backend nodes. This contains the original source IP of the actual client. The mechanism does need to be supported by the application receiving the PROXY protocol; currently HyperStore S3 supports the PROXY protocol but the CMC does not (yet).

If visibility of client IP addresses are a strict requirement for both S3 AND the CMC. A suggested configuration is to run the CMC in HTTP mode and using the X-Forwarded-For header and enabling the PROXY protocol for S3 in TCP mode

There are other commercial Load Balancing appliances that also support the PROXY protocol. F5, ALOHA and Loadbalancer.org are known to support PROXY as well.

Enable Proxy for S3

As PROXY needs to be enabled and used on both client as server, when enabled on HyperStore it creates additional listening sockets for PROXY on dedicated ports (81 for S3 over HTTP and 4431 if S3 HTTPS is enabled).

On the HAProxy level, all that is required is to add the send-proxy option to the S3 backend nodes and point those to the PROXY-enabled port on HyperStore. For example:

```
server cloudian-node1 10.1.1.11:81 check send-proxy inter 5s rise 1 fall 2
```

Subsequently, for S3 HTTPS;:

```
server cloudian-node1 10.1.1.11:4431 check send-proxy check-ssl verify none inter 5s
rise 1 fall 2
```

On HyperStore, you will enable the PROXY protocol by setting the following to true:

```
/etc/cloudian-7.0-puppet/manifests/extdata/common.csv:
s3_proxy_protocol_enabled,true
```

Refer to chapter “Pushing Configuration File Edits to the Cluster and Restarting Services” in the official Cloudian HyperStore documentation on how to apply these changes to the cluster and restart the HyperStore S3 service. After this change to HyperStore, reload the HAProxy service to apply the changes on the Load Balancer(s).

To minimize downtime, make the required changes to HAProxy and HyperStore first and push those across the cluster, but only restart S3 and HAProxy services afterwards and around the same time. Both sides need to have PROXY either enabled or disabled to communicate.

DNS Requirements

Cloudian HyperStore uses Service Endpoints Names to ensure client requests are correctly resolved and handled. Table 9 lists the services to define in DNS and that need to be accessible to clients in order to successfully connect and use the Object Store.

Table 9 DNS Requirements

Service Endpoint	DNS Host A Record Example	Ports	Description
s3 Service Endpoint	s3-region1.cisco.cloudian.local	80, 443	Default s3 service endpoint that should resolve to the load balancer
s3 Wildcard Service Endpoint	*.s3-region1.cisco.cloudian.local	80, 443	Wildcard s3 service endpoint that should resolve to the load balancer
s3 Admin Service Endpoint	s3-admin.cisco.cloudian.local	19443	Admin service endpoint that should resolve to the load balancer
Cloudian Management Console	cmc.cisco.cloudian.local	8888, 8443	CMC service endpoint that should resolve to the load balancer

The Load Balancer should forward the traffic to all HyperStore nodes in the data center in a round-robin way. The traffic for the Cloudian Management Console (CMC) should be configured with sticky sessions enabled.

Prepare the Master Node

The master node is used to push binaries and configurations to the nodes. THe basic directories need to be created and the system_setup script needs to be downloaded.

Create folders for Cloudian installation:

```
# mkdir -p /root/CloudianTools /root/CloudianPackages/
```

Download and execute HyperStore system_setup script:

```
# cd /root/CloudianTools/ && yum install -y wget && wget
https://s3.cloudianhyperstore.com/downloads/Scripts/system_setup.sh && chmod +x
system_setup.sh && ./system_setup.sh
```

Select D - Download HyperStore Files

```
# System Setup
   1) Configure Networking
   2) Change Timezone
   5) Change root Password
   B) BMC Configuration
```

```
   D) Download HyperStore Files
        Please Download or place the HyperStore files in '/root/CloudianPackages'
   S) Script Settings
   A) About system_setup2.sh
   X) Exit
```

Select EA Version

```
System Setup » HyperStore Downloader
Downloading HyperStore Version Information ... Done
Which HyperStore release would you like to download? (v6-GA/GA/FTP/EA) EA
Downloading HyperStore Binary v7.1.4 ... Done
Downloading HyperStore Binary v7.1.4 (md5) ... Done
Downloading HyperStore Documentation v7.1.4 ... Done
Downloading HyperStore Documentation v7.1.4 (md5) ... Done
Downloading HyperStore Release Notes ... Done
Downloading HyperStore Installation License ... Done
Press any key to continue ...
```

Once the HyperStore binary is downloaded exit the script and extract the binary using the by Cloudian provided License file (.lic)

```
#./CloudianHyperStore-7.1.4.bin cloudian_289001406012.lic
Extracting package contents for installation...
Extraction completed.
*** Cloudian HyperStore(R) Cloud Storage System ***
*** Checking required packages: Oracle Java jdk-1.8.0_172, Puppetserver (JVM) 1.2.0,
Puppet 3.8.7, Facter 2.4.6, Python 2.7.8, Ruby ***, bind-utils
The Cloudian Hyperstore install script will now install: Java, Puppet 3.8.7, Puppet-
server 3.8.7, Python 2.7.8, facter 2.4.6, puppetserver 1.2.0, bind_utils
Self Extracting Installer
*** Running Installer for Cloudian Pre-requisite packages ***
*** Completed Installation of Cloudian Pre-requisite packages ***
Unpackaging Cloudian configuration files...
Creating Puppet configuration root directory /etc/cloudian-7.1.4-puppet ...
Successfully created Puppet configuration root directory /etc/cloudian-7.1.4-puppet.
Default templates stored for future upgrades in
/root/CloudianPackages/orig_templates/cloudian-7.1.4-puppet.tar.gz.
Default csv's stored for future upgrades in
/root/CloudianPackages/orig_csvs/cloudian-7.1.4-puppet-csvs.tar.gz.
To install Cloudian HyperStore software:
   1. Compose a network survey file.  A sample survey file, sample-survey.csv,
      is provided for your reference.
   2. Run cloudianInstall.sh
Your staging directory is /root/CloudianPackages
```

Network Best Practices

The best practice is to create a network for client access and cluster communication. The interfaces can be bonded and be configured on separate VLANS when desired. The internal cluster communication interface should not be used as the interface for default routing.

Cloudian supports the following bonding options:

- Balanced Round Robin
- Active Backup
- Balance XOR
- Broadcast
- 802.3ad
- Balance TLB
- Balance ALB

This configuration is not necessary for this CVD.

Cloudian recommends using an MTU size of 1500 if object storage is exposed to Internet. An MTU size of 9000 can be used if the cluster is not serving clients over the Internet and the entire network infra-structure including load balancers have been adjusted accordingly.

HyperStore nodes can communicate with each other via JMX, and when they do, after initial connection established on the designated JMX, a random port is used for continued communication. Therefore, there cannot be any port restrictions on communication between HyperStore nodes. Consequently, the HyperStore installation will abort if firewalled, SELinux, or iptables is running on a host

The ports marked in italics below should be exposed to public traffic

Table 10 Overview of Hyperstore Network Ports

Service	Listening Port	Interface(s) Binded To	Purpose
Cloudian Management Console (CMC)	*8888*	All	Requests from administrators' or end users' browsers via HTTP
	8443	All	Requests from administrators' or end users' browsers via HTTPS
S3 Service	*80*	All	Requests from the CMC or other S3 client applications via HTTP
	443	All	Requests from the CMC or other S3 client applications via HTTPS
	81	All	Requests relayed by an HAProxy load balancer using the PROXY Protocol (if enabled by configuration; see **s3_proxy_protocol_enabled** in common.csv)
	4431	All	Requests relayed by an HAProxy load balancer using the PROXY Protocol with SSL (if enabled by configuration)
	19080	All	JMX access
IAM Service	*16080*	All	Requests from the CMC or other IAM clients via HTTP
	16443	All	Requests from the CMC or other IAM clients via HTTPS
	19084	All	JMX access
Admin Service	18081	All	Requests from the CMC or other Admin API clients via HTTP
	19443	All	Requests from the CMC or other Admin API clients via HTTPS (Note: The CMC by default

Create the survey.csv File

The survey.csv file is used to identify the nodes that will be used for installing the HyperStore cluster. The survey.csv file includes the following information for each node:

- Region name
- Hostname
- IP that resolves to the hostname
- Datacentername
- Rack name
- Interface name for internal cluster communication

Since the interface used for internal cluster communication must be defined in the survey.csv file, the correct interface name has to be verified.

To verify the interface name for internal cluster network, run the following:

```
ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 173.36.220.21  netmask 255.255.255.0  broadcast 173.36.220.255
        ether 00:25:b5:00:00:00  txqueuelen 1000  (Ethernet)
        RX packets 1619574  bytes 413914654 (394.7 MiB)
        RX errors 0  dropped 18487  overruns 0  frame 0
        TX packets 369478  bytes 183662471 (175.1 MiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

eth1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 9000
        inet 192.168.10.21  netmask 255.255.255.0  broadcast 192.168.10.255
        ether 00:25:b5:00:00:01  txqueuelen 1000  (Ethernet)
        RX packets 814772  bytes 86586538 (82.5 MiB)
        RX errors 0  dropped 18487  overruns 0  frame 0
        TX packets 999  bytes 64112 (62.6 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

eth2: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 9000
        inet 192.168.30.21  netmask 255.255.255.0  broadcast 192.168.30.255
        ether 00:25:b5:00:00:02  txqueuelen 1000  (Ethernet)
        RX packets 4659026459  bytes 20594509693265 (18.7 TiB)
        RX errors 0  dropped 18487  overruns 0  frame 0
        TX packets 4734599130  bytes 21643340207599 (19.6 TiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

eth3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 9000
        inet 192.168.20.21  netmask 255.255.255.0  broadcast 192.168.20.255
        ether 00:25:b5:00:00:03  txqueuelen 1000  (Ethernet)
        RX packets 2127629901  bytes 10874339372550 (9.8 TiB)
        RX errors 0  dropped 18487  overruns 0  frame 0
        TX packets 2863788128  bytes 21777867239126 (19.8 TiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

When the interface name for internal cluster communication has been identified, the system_setup script can be executed to create the survey.csv file:

```
System Setup
   1) Configure Networking
   2) Change Timezone
   3) Setup Disks
   4) Setup Survey.csv File
        Survey File '/root/CloudianPackages/survey.csv' Not Found
   5) Change root Password
   6) Install & Configure Prerequisites
   9) Prep New Node to Add to Cluster
   B) BMC Configuration
   S) Script Settings
   A) About system_setup.sh
   X) Exit
Choice:   4) Setup Survey.csv File
```

Add the correct hostname, desired region name, data center name, rack name and interface for internal cluster communication. Add additional entries as needed.

```
System Setup » Survey File
  Using '/root/CloudianPackages/survey.csv'

   C) Create New File

   P) Return to the Previous Menu

Choice: c
System Setup » Survey File » Create Survey File

Creating Directory '/root/CloudianPackages' ... Already Exists
Creating File '/root/CloudianPackages/survey.csv' ... Done

Would you like to add entries now? (Yes/No) Yes
System Setup » Survey File » Add Entry

No Entries Found
Region  Hostname  IP Address  Datacenter Rack  Interface

Lines in red are commented out in the survey file.

Region Name: region1
Hostname: storage-node1
Attempting auto IP resolution for storage-node1 ... Done
IP Address: 192.168.10.21
DatacenterName: DC1
Rack name: Rack1
Internal Interface (optional): eth2

Adding entry to /root/CloudianPackages/survey.csv ... Done

Would you like to add another entry? (Yes/No) [Yes]
```

When completed the survey.csv will look similar like this:

```
region1,storage-node1,192.168.10.21,DC1,Rack1,eth2
region1,storage-node2,192.168.10.22,DC1,Rack1,eth2
region1,storage-node3,192.168.10.23,DC1,Rack1,eth2
```

To install multiple data centers with the initial installation, make sure to correctly specify the DatacenterName in the fourth tab of the survey.csv.

Prepare Cluster Nodes

Now the survey.csv file is completed and IP communication between the nodes has been established and verified, the prerequisites can be installed on each node by running option 6 from the system_setup script.

```
System Setup

6) Install & Configure Prerequisites
```

Next, the script will prompt you to provide the root password for each node and will setup ssh certificates.

```
Would you like to perform this on all nodes listed in your survey file? (Yes/No) Yes

If your root password is the same on all (or most) nodes in the cluster, you can
supply it as a cluster password
If you do not want to supply a password, each server will prompt for one when
connecting.

Cluster Password:
```

When the prerequisites have been successfully installed, it's time to format and mount the data drives on each cluster node. As part of the prerequisites installation the system_setup script has been placed under /root/CloudianTools/system_setup.sh for each node.

Log into each node and execute /root/CloudianTools/system_setup.sh to configure the hostname, domain name and configure additional network interfaces, bonds and VLANS if you have not done so already.

Select 1 Configure Networking

```
System Setup
   1) Configure Networking
```

Select 1 to 4 to adjust network interfaces as needed, VLANS and Bondings can be created as well. When finished D to set domain name

```
System Setup » Networking

      Interface  IP Address                     State  Type      Mode  Master  Speed
   1) eth0       173.36.220.21/24              Up     Ethernet  --    --      40
Gb/s
                 fe80::225:b5ff:fe00:a000/64
   2) eth1       192.168.10.21/24              Up     Ethernet  --    --      40
Gb/s
                 fe80::225:b5ff:fe00:a001/64
   3) eth2       192.168.30.21/24              Up     Ethernet  --    --      40
Gb/s
                 fe80::225:b5ff:fe00:a002/64
   4) eth3       192.168.20.21/24              Up     Ethernet  --    --      40
Gb/s
                 fe80::225:b5ff:fe00:a003/64
   5) lo:1       192.168.10.100/32             Down   Ethernet  --    --      --

        Select a number from the list above to edit an interface's configuration

   D) Change Domain Name (<unset>)          H) Change Hostname (storage-node1)

   B) Create Bond Interface                 V) Create VLAN Interface

   N) Restart Networking                    R) Refresh Interface Details
```

The data drives can be formatted and mounted automatically by running option 3 of the system_setup script individually on each node and selecting the drives that are going to be used to store data.

Select option 3

```
System Setup
```

```
   1) Configure Networking
   2) Change Timezone
   3) Setup Disks
   4) Setup Survey.csv File
   5) Change root Password
   6) Install & Configure Prerequisites
   7) Run Commands on each Cluster Node
   8) Copy Local File to each Cluster Node
   9) Prep New Node to Add to Cluster

   B) BMC Configuration

   R) Run Pre-installation Checks

   S) Script Settings
   A) About system_setup2.sh

   X) Exit
```

Select the drives that are to be used for data storage:

```
System Setup » Setup Disks
  Selected Disks: sda sdaa sdab sdb sdc sdd sde sdf sdg sdh sdi sdj sdk sdl sdm sdn
sdo sdp sdq sdr sds sdt sdu sdv sdw sdx sdy sdz

      Device   Size    Dependencies            Device   Size    Dependencies
      1) sda     9.1T    0                     2) sdb     9.1T    0
   3) sdc     9.1T    0                     4) sdd     9.1T    0
   5) sde     9.1T    0                     6) sdf     9.1T    0
   7) sdg     9.1T    0                     8) sdh     9.1T    0
   9) sdi     9.1T    0                    10) sdj     9.1T    0
  11) sdk     893.1G  5                    12) sdl     223.6G  0
  13) sdm     223.6G  0

   C) Configure Selected Disks

   T) Toggle Selection for all disks

   R) Refresh Disks

   P) Return to the Previous Menu
```

Alternatively, the drives can be remotely formatted from the master node with the command below, please ensure to grep for the correct disk size:

```
for i in {1..3}; do ssh -t -i /root/CloudianPackages/cloudian-installation-key
storage-node${i} /root/CloudianTools/system_setup.sh --configure-disks $(lsblk -d |
grep 9.1T |awk '{print $1}' |sed -r -e ':a;N;$!ba;s~\n~ ~g' ) <<<'y'; done
```

Verify that all disks are correctly mounted on all nodes:

```
for i in {1..3}; do ssh -t -i /root/CloudianPackages/cloudian-installation-key
storage-node${i} df -h |grep -c /cloudian; done

10
Connection to storage-node1 closed.
```

```
10
Connection to storage-node2 closed.
10
Connection to storage-node3 closed.
```

Cloudian Hyperstore Installation

Software Installation

When it has been verified that all drives on all nodes have been successfully mounted, the pre-installation check is run to ensure all requirements for installation have been met and there are no conditions that would cause the installation to fail. The pre-install check can be run from the /root/CloudianTools/systemsetup.sh and selecting option R Run Pre-installation Checks:

```
System Setup

   R) Run Pre-installation Checks

System Setup » Pre-installation Checklist

OK   found survey file /root/CloudianPackages/survey.csv
OK   All 1 Data Center(s) contain a minimum of 3 nodes
OK   entry found in hosts file for node storage-node1

Total checks performed: 152. Warnings: 0, Errors: 0
Press any key to continue ...
```

The pre-installation check should finish without errors or warnings. If errors are encountered, the generated output should provide more information about what the error is; do not proceed with the installation until all the errors are resolved.

After the pre-installation check has run successfully and no errors are found, the Cloudian HyperStore installer can be executed. The installer can be executed with the -s option to specify the survey.csv file name, not using this option will prompt you to enter the correct survey file name.

```
./cloudianInstall.sh -s survey.csv

Cloudian HyperStore(R) 7.1.4 Installation/Configuration
    ------------------------------------------------------

    0 )  Run Pre-Installation checks
    1 )  Install Cloudian HyperStore
    2 )  Cluster Management
    3 )  Upgrade From a Previous Version
    4 )  Advanced Configuration Options
    5 )  Uninstall Cloudian HyperStore
    6 )  Help
    x )  Exit
```

The Cloudian HyperStore installer provides multiple usage options that can be listed by executing the help; ./cloudianInstall -h

Select option 1 Install Cloudian HyperStore and answer yes to use the Cloudian-installation-key that was already created.

```
Setup Access to Hosts in Cluster
```

```
--------------------------------
Processing cluster host information in survey.csv file.
Connectivity check to all (3) hosts defined in survey file.
Able to ping all 3 hosts defined in survey.csv file.
Check and setup password-less SSH access to hosts.
Would you like to use key ./cloudian-installation-key? (yes/no) [yes]:
Installation key cloudian-installation-key is now being copied to all nodes ...
Installation key ./cloudian-installation-key.pub copied to all agent hosts
successfully.
Will now copy key cloudian-installation-key to this host storage-node1.
Password-less SSH access to hosts setup.
Installation requirements check on hosts defined in survey file survey.csv.
Installing prerequisite packages on agent node 192.168.10.21.  This could take a
minute.
Installing prerequisite packages on agent node 192.168.10.22.  This could take a
minute.
Installing prerequisite packages on agent node 192.168.10.23.  This could take a
minute.
```

Next, the installer asks for a default interface for the internal cluster communication. The interface that was previously entered in the survey.csv file will take precedence over the default internal interface. The default internal interface is only used when no interface is defined in the survey.csv file.

```
Configure cluster
-----------------
Select only one from this list of known interfaces: eth0,eth1,eth2,eth3,lo:1.
Leave it blank you wish to use the default network interface.

   Please enter one of the interface names [eth0,eth1,eth2,eth3,lo:1] for internal
services:  []: eth2
   Using eth2 for all internal traffic.
```

Provide the Top-Level Domain that will be used by the cluster:

```
Cloudian HyperStore(R) S3 service endpoints are based on your desired top
level DNS domain name.  For example, yourcompany.com.
Please enter your top level domain name [cisco.cloudian.local]:

Region [region1] Cassandra cluster name: Cloudianregion1
```

Keep the metadata replication strategy at 3 by accepting the default value or specifying DC1:3:

```
Please enter the service metadata replication strategy for region1 [DC1:3]:
```

Enter a local NTP time source or use an external NTP server:

```
NTP time server(s) for region region1:
Please enter your NTP time server(s)
[0.centos.pool.ntp.org,1.centos.pool.ntp.org,2.centos.pool.ntp.org,3.centos.pool.ntp
.org]:

NTP time server(s) for region :
0.centos.pool.ntp.org,1.centos.pool.ntp.org,2.centos.pool.ntp.org,3.centos.pool.ntp.
org
```

Accept the default entries for the service endpoints based on the Top Level Domain or enter a custom endpoint name for s3 service, s3-website, s3-admin and CMC:

```
Service endpoints for region region1:
Region [region1] S3 service domain URLs(comma separated) [s3-
region1.cisco.cloudian.local]:
Region [region1] S3 Web site end point [s3-website-region1.cisco.cloudian.local]:
Admin endpoint [s3-admin.cisco.cloudian.local]:
S3 Admin service endpoint: s3-admin.cisco.cloudian.local
Domain name of your Cloudian Management Console service [cmc.cisco.cloudian.local]:
Cloudian Management Console service endpoint: cmc.cisco.cloudian.local
```

When the installation has completed successfully it will display the predefined CMC url to manage the cluster:

```
http://cmc.cisco.cloudian.local:8888
```

Generate HTTPS Certificate and Signing Request

By default, Cloudian HyperStore is configured for HTTP access only, HTTPS can be setup by generating a self-signed certificate that in parallel will also create a Certificate Signing Request (CSR) in the same directory, if your Keystore file is named cloudian.jks for example, then the CSR file will be named cloudian.csr.

To Generate a Certificate and a Certificate Signing Request select option 4 Advanced Configuration Options from the installation menu.

```
Cloudian HyperStore(R) 7.1.4 Installation/Configuration
    ------------------------------------------------------

    0 )  Run Pre-Installation checks
    1 )  Install Cloudian HyperStore
    2 )  Cluster Management
    3 )  Upgrade From a Previous Version
    4 )  Advanced Configuration Options
    5 )  Uninstall Cloudian HyperStore
    6 )  Help
    x )  Exit
```

Select option e Generate a self-signed certificate in a JKS keystore:

```
Advanced Configuration Options
    ------------------------------

    a  )  Change server role assignments
    b  )  Change S3, Admin and CMC ports
    c  )  Change S3, S3-Website, Admin, or CMC endpoints
    d  )  Configure diagnostic data collection options
    e  )  Generate a self-signed certificate in a JKS keystore
    f  )  Enable and configure HTTPS access on S3 server [OK]
    g  )  Import Java keystore to CMC
    h  )  Remove existing Puppet SSL certificates
    i  )  Start or stop Puppet daemon
    j  )  Remove Puppet access lock
    k  )  Enable or disable DNSMASQ
    l  )  Configure Performance Parameters on Nodes
    m  )  Generate a self-signed certificate for IAM in a JKS keystore
```

```
    n  )  Enable and configure HTTPS access for IAM
    r  )  Exclude host(s) from configuration push and service restarts
    x  )  Return to Main Menu

Provide the keystore name and password to use.

Generate a self-signed certificate in a JKS keystore
--------------------------------------------------

   Generating self-signed certificate for region region1

Please enter key store name [cloudian.jks]:

If you plan to store multiple certificates in your key store, you must provide
an alias for each certificate stored.

Please enter alias name []: cloudians3
Please enter the password for cloudian.jks [testpass]:
Please enter the key store manager password for cloudian.jks [testpass]:
```

Provide the wildcard domain name(s) that you want to use for the certificate and complete the organization identity fields:

```
Common name is the URL(FQDN or IP address) for SSL connection.  For S3 service
bucket name is a part of the FQDN.  You will need to generate and verify the
certificate as a wildcard.  For example, *.s3.cloudian.com.

Please enter comma-separated domain names [*.s3-region1.cisco.cloudian.local]:
Enter your organizational unit name []: cisco
Enter the name of your organization []: cisco-cloudian
Enter the name of your City or Locality []: San Jose
Enter the name of your State or Province []: CA
Enter the two-letter country code for this unit []: US

Certificate generated for region region1.


CSR location for this example.
/etc/cloudian-7.1.4-puppet/modules/baselayout/files/cloudian.csr
```

When intending to use an official certificate submit the generated CSR file to your preferred Certificate Authority for signing, using the instructions from the CA.

Import SSL certificate in Keystore

When intending to use a self-signed certificate this step can be skipped.

Copy all the certificates that you received from the CA into the /etc/cloudian-7.1.4-puppet/modules/baselayout/files directory.

From the same directory, issue the following commands to import the Root CA Certificate and Intermediate CA Certificate into your Keystore file.

```
Example for GoDaddy as the CA:
/usr/java/default/bin/keytool -import -trustcacerts -alias root -file <Root CA
Certificate File> -keystore cloudian.jks
/usr/java/default/bin/keytool -import -trustcacerts -alias intermediate -file
<Intermediate CA Certificate File> -keystore cloudian.jks
/usr/java/default/bin/keytool -import -trustcacerts -alias GoDaddyRoot -file
gdrootg2_cross.crt -keystore cloudian.jks
/usr/java/default/bin/keytool -import -trustcacerts -alias GoDaddyCrossCA -file
gd_cross_intermediate.crt -keystore cloudian.jks
/usr/java/default/bin/keytool -import -trustcacerts -alias GoDaddyG2CA -file
gdig2.crt -keystore cloudian.jks
```

Issue the following command to import your CA-signed TLS/SSL Certificate into your Keystore file:

```
 [files]# /usr/java/default/bin/keytool -import -trustcacerts -alias cloudians3 -
file cloudianS3.crt
-keystore cloudian.jks
```

Enable HTTPS access on s3

When the self-signed or CA-signed certificate has been created and imported to the keystore, HTTPS can be enabled for s3. To enable HTTPS for s3, select option f Enable and configure HTTPS access on S3 server from the Advanced Configuration Options menu from within the installer.

```
Advanced Configuration Options
    ------------------------------

    a  )  Change server role assignments
    b  )  Change S3, Admin and CMC ports
    c  )  Change S3, S3-Website, Admin, or CMC endpoints
    d  )  Configure diagnostic data collection options
    e  )  Generate a self-signed certificate in a JKS keystore
    f  )  Enable and configure HTTPS access on S3 server
    g  )  Import Java keystore to CMC
    h  )  Remove existing Puppet SSL certificates
    i  )  Start or stop Puppet daemon
    j  )  Remove Puppet access lock
    k  )  Enable or disable DNSMASQ
    l  )  Configure Performance Parameters on Nodes
    m  )  Generate a self-signed certificate for IAM in a JKS keystore
    n  )  Enable and configure HTTPS access for IAM
    r  )  Exclude host(s) from configuration push and service restarts
    x  )  Return to Main Menu
```

Follow the onscreen instructions to enable HTTPS, make sure to provide the correct key store name, password and alias.

```
Enable and configure HTTPS access on S3 server
----------------------------------------------


HTTPS access to Cloudian HyperStore(R) S3 server is not enabled.
Do you wish to enable HTTPS access on S3 server? (yes/no) [no]: yes
HTTPS on Cloudian HyperStore(R) S3 server is enabled.

The key store is the file name of your identity store that will contain the server
```

```
private key and corresponding server public certificate (self-signed or CA
verified).

Please enter name of your key store [cloudian.jks]:
Certificate alias to use []: cloudians3

The trust keystore is the file name of the identity store that will contain the
client
certificates for SSL mutual authentication.  If not set, the system will look for
client certificates in the keystore.
```

Please enter trust keystore [cloudian.jks]:

```
   Passwords are obfuscated in configuration files. If the password your enter
is not prefixed with 'OBF:', then password obfuscation is automatically performed
using Jetty utility.

Please enter password for the keystore cloudian.jks
[OBF:1ytc1vu91v2p1y831y7v1v1p1vv11yta]:
Please enter password for the trust keystore cloudian.jks
[OBF:1ytc1vu91v2p1y831y7v1v1p1vv11yta]:
Please enter keystore manager password [OBF:1ytc1vu91v2p1y831y7v1v1p1vv11yta]:

Please enter path name in which to store keystore file [/opt/cloudian/conf]:
Please enter path name in which to store trust keystore file [/opt/cloudian/conf]:

Please enter connection maximum idle time in (ms) [60000]:
Please enter connections at which system is considered to have low resource [1000]:
Please enter low resource connection idle time in (ms) [5000]
```

To complete the HTTPS configuration the changes have to be pushed out to all cluster nodes using puppet. From the main installer menu select option 2 Cluster Management.

```
    Cloudian HyperStore(R) 7.1.4 Installation/Configuration
    -------------------------------------------------------

    0 )  Run Pre-Installation checks
    1 )  Install Cloudian HyperStore
    2 )  Cluster Management
    3 )  Upgrade From a Previous Version
    4 )  Advanced Configuration Options
    5 )  Uninstall Cloudian HyperStore
    6 )  Help
    x )  Exit
```

Select option b Push Configuration Settings to Cluster.

```
Cluster Management
    ------------------

    a )  Review Cluster Configuration
    b )  Push Configuration Settings to Cluster
    c )  Manage Services
    d )  Run Validation Tests
    x )  Return to Main Menu
```

Select default empty value to send configuration to all nodes in the cluster.

```
Run Puppet to configure agent nodes
-----------------------------------

region region1 contains the following hosts: storage-node2 storage-node3 storage-
node4 storage-node5 storage-node6 storage-node1
Enter a comma-separated list of hosts in region1 to execute agents on? [empty for
all] []:

All Puppet agent runs completed successfully in region1 region.
Puppet agent run ended for region1.

Press any key to continue ...

As final step the s3 service has to be restarted on all nodes. From within the
Cluster Management menu, select option c Manage Services.

Cluster Management
------------------

    a )  Review Cluster Configuration
    b )  Push Configuration Settings to Cluster [OK]
    c )  Manage Services
    d )  Run Validation Tests
    x )  Return to Main Menu
```

From within the Service Management Menu, select option 5 S3 Service and enter restart.

```
Service Management
-------------------------------------
    0 ) All services
    1 ) Redis Credentials
    2 ) Redis QOS
    3 ) Cassandra
    4 ) HyperStore service
    5 ) S3 service
    6 ) Redis Monitor
    7 ) Cloudian Agent
    8 ) DNSMASQ
    9 ) Cloudian Management Console (CMC)
    P ) Puppet service (status only)
    X ) Quit
```

You can execute the following list of commands:

```
start,stop,status,restart,version,force-stop,node-start,node-stop

Select a service to manage: 5

Enter command: (start,stop,status,restart,version) restart
Executing Cloudian S3 service command restart ...

On host storage-node2:
```

```
/etc/init.d/cloudian-s3 restart => Restarting cloudian-s3 (via systemctl):  [  OK  ]


On host storage-node3:
/etc/init.d/cloudian-s3 restart => Restarting cloudian-s3 (via systemctl):  [  OK  ]


On host storage-node1:
/etc/init.d/cloudian-s3 restart => Restarting cloudian-s3 (via systemctl):  [  OK  ]


Press any key to continue ...
```

For more information on how to Install Cloudian HyperStore, refer to the Cloudian HyperStore Installation guide.

Cloudian Hyperstore Configuration

Log into the Cloudian Management Console (CMC)

To login to the CMC, point a web browser to the predefined CMC URL on HTTP port 8888 or HTTPS port 8443 and follow these steps::

http://cmc.cisco.cloudian.local:8888

https://cmc.cisco.cloudian.local:8443

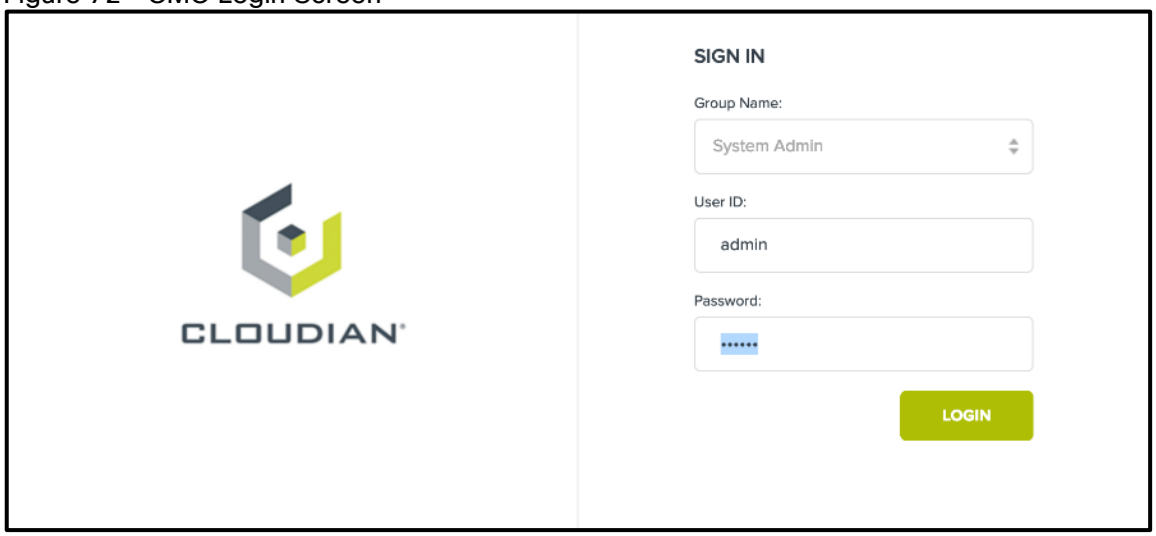
When using HTTP, the browser will be redirected to the HTTPS port for secure login.

In addition to the predefined CMC URL, all IP addresses of all nodes can be used to access the CMC.

1. Login with the default admin credentials:

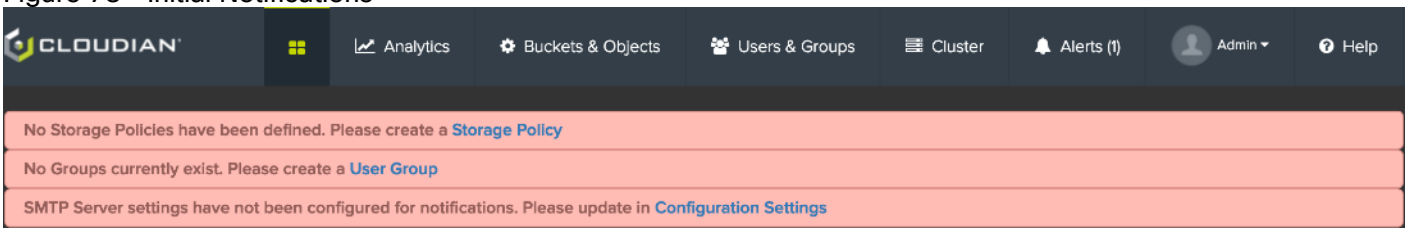
```
Group Name: System Admin
User ID:       admin
Password:      public
```

Figure 72 CMC Login Screen



2. When logged into the CMC, the system needs to be configured with one or more Storage Policies, Groups and Users and settings to enable SMTP notifications.

Figure 73 Initial Notifications



Create a Storage Policy

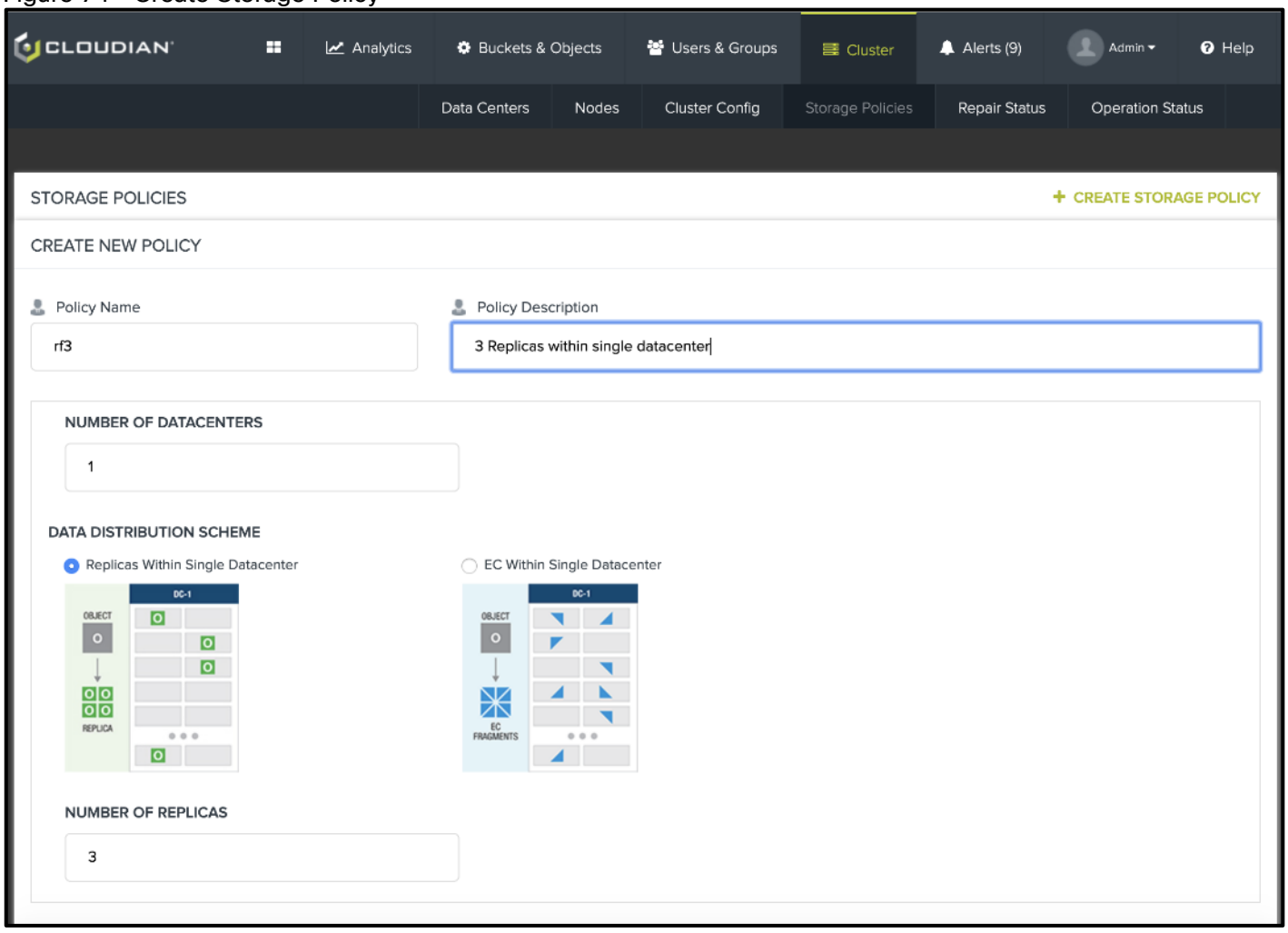
To create a storage policy, follow these steps:

1. Click the pink bar No Storage Policies have been defined or alternatively go to the Cluster tab and click theStorage Policies tab and then click CREATE STORAGE POLICY.

2. Provide a name for the Storage Policy in the Policy Name field, followed by a description. In this setup the cluster has a single data center (DC) and exists out of 3 nodes, which provides the option to use Replication with number of replicas as 3 which will be used for this example.

3. To use the 3-way replication within the Single Datacenterusing the protection scheme for this storage policy, select Replicas Within Single Datacenter under DATA DISTRIBUTION SCHEME. Enter 3 for NUMBER OF REPLICAS.

Figure 74 Create Storage Policy



4. Select the desired consistency level for the 3-way replication Storage Policy. The consistency level of QUORUM provides a strong consistency as a read or write operation and must succeed on a set number of replica copies before a successful response is returned to the client application. This enables flexibility in how stringent you want your replication policy to be.

By default, the created storage policies are available to each group/tenant. When a group is specified, the storage policy will only be visible to the defined group(s).

Storage policies can be configured with the compression algorithms provided below.

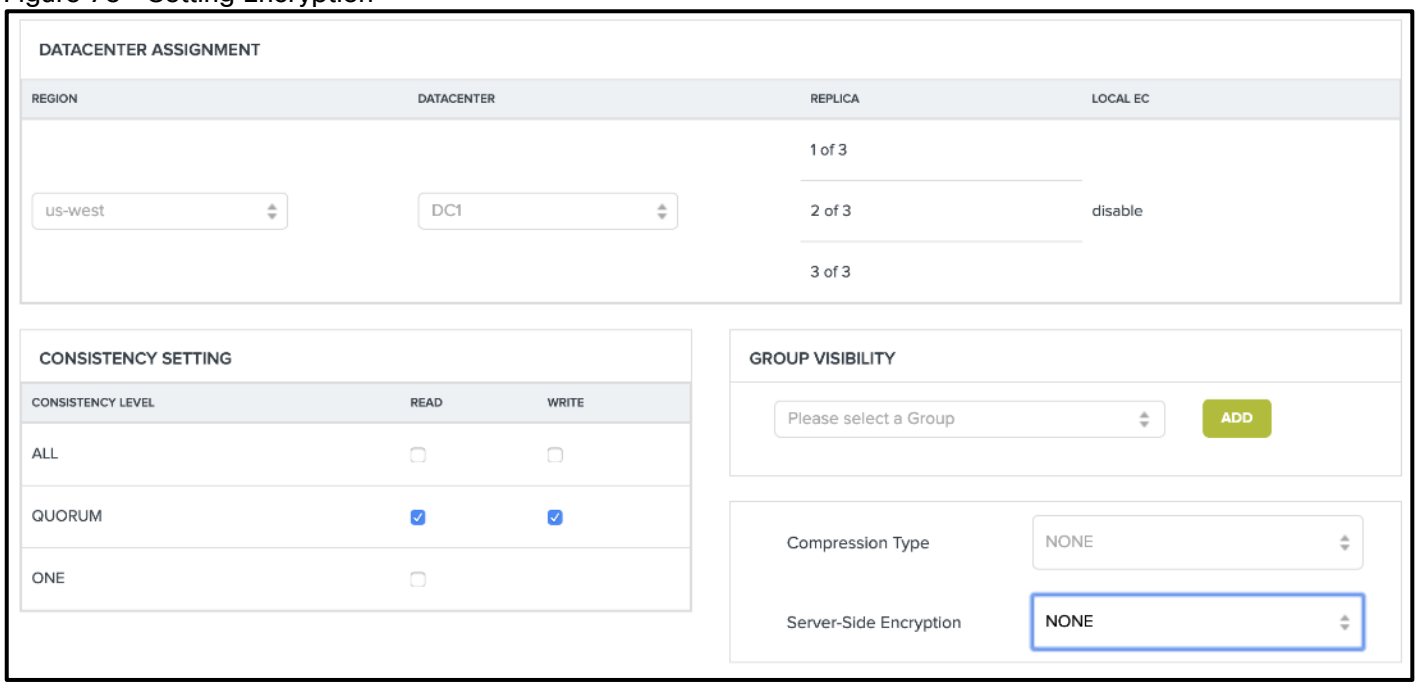
Be aware that CPU cycles will be wasted when the data is placed in a bucket using storage policy with compression enabled, however that data is not compressible.

Supported Compression Algorithms:

- SNAPPY
- ZLIB
- LZ4

Encryption at rest can be enabled and forced at the bucket level by setting the Server-side Encryption box to SSE.

Figure 75 Setting Encryption



The first Storage Policy that is created will be the default Storage Policy for all Users and Groups. The default Storage Policy can be changed when multiple Storage Policies exist on the Cluster. For more information on setting up Storage Policies in HyperStore please refer to the HyperStore Admin Guide.

Storage Policies can also be created through the admin API, see the Admin API section of the HyperStore admin guide

HyperStore supports multiple storage policies on the same hardware; the type of storage policies you can create depends on the number of nodes and the DC's in the cluster.

The minimum amount of data replicas is 3; this is to protect the data and ensure there is always a quorum.

For increased read performance, select a Quorum of ONE for read. Please not that this comes at the cost of having a small window of time where potentially old data can be read.

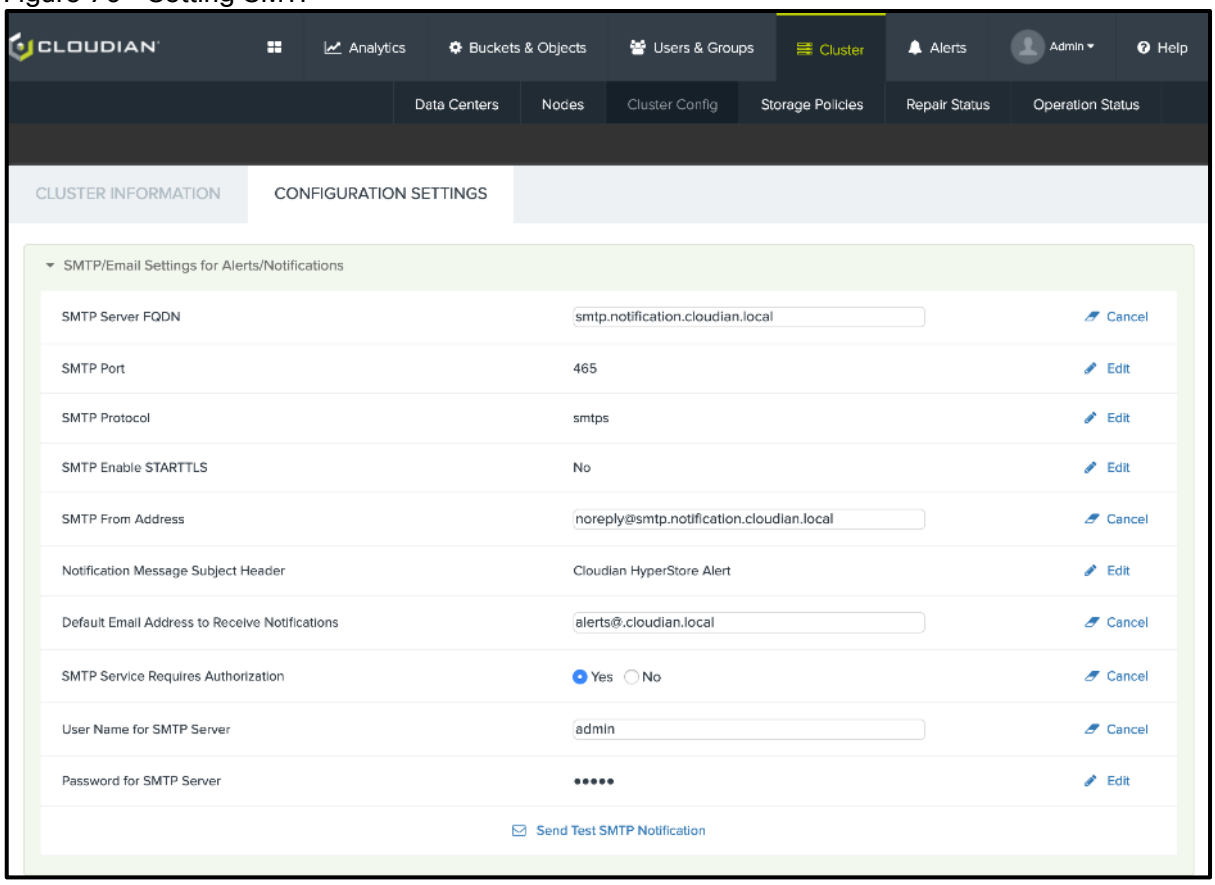
For more information on setting up Storage Policies in HyperStore, refer to the HyperStore Admin Guide

Setup Alerts and Notifications

Cloudian HyperStore should be configured to send out alerts and notification for events to ensure proper action can be taken in a timely manner. Cloudian HyperStore supports alerts through SMTP and SNMP. To create alert rules, the SMTP email settings and/or SNMP server details have to be completed. To do so, follow these steps:

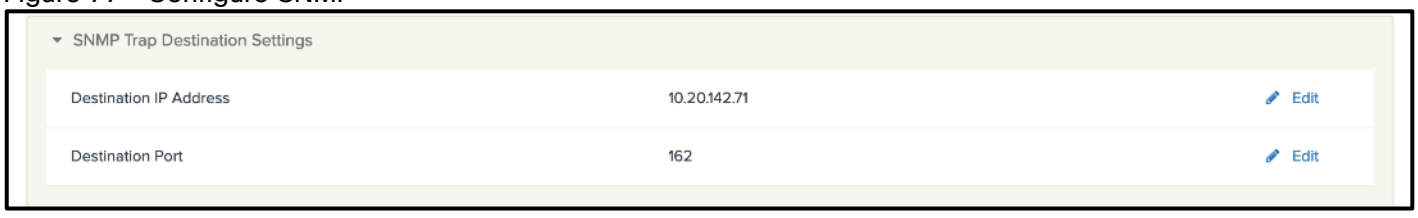
1. To setup the SMTP email details, go to Cluste,r click Cluster Config, then click the CONFIGURATION SET-TINGS tab and complete the SMTP/Email Settings for Alerts/Notifications.

Figure 76 Setting SMTP



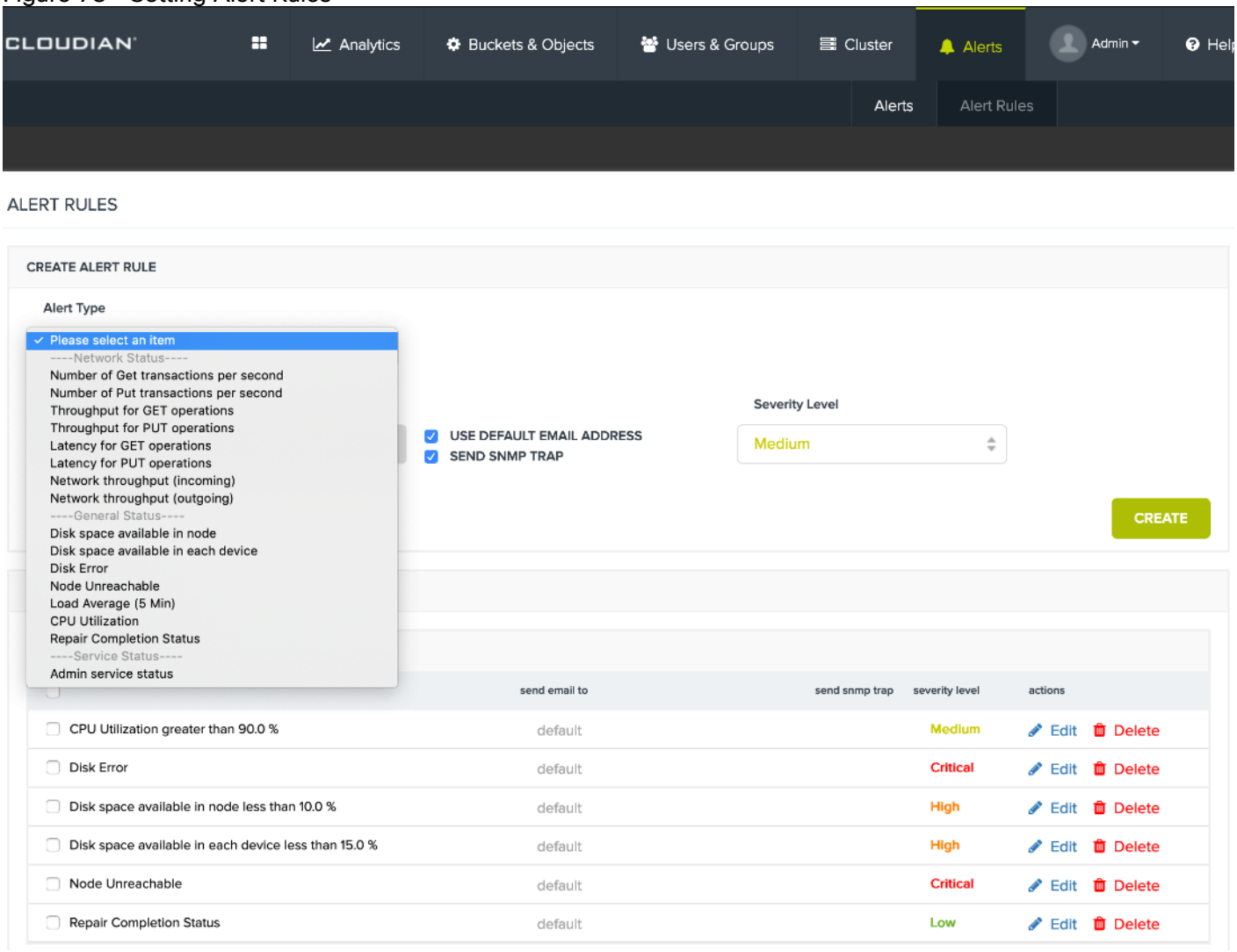
2. To setup SNMP, complete the SNMP Trap Destination Settings.

Figure 77 Configure SNMP



3. Once the SMTP and/or SNMP details have been configured, alert rules can be created to trigger notification events. To setup an alert rule, go to Alerts, click Alert Rules, select an Alert Type from the drop-down list, specify the condition, severity level and the alert destination.

Figure 78 Setting Alert Rules



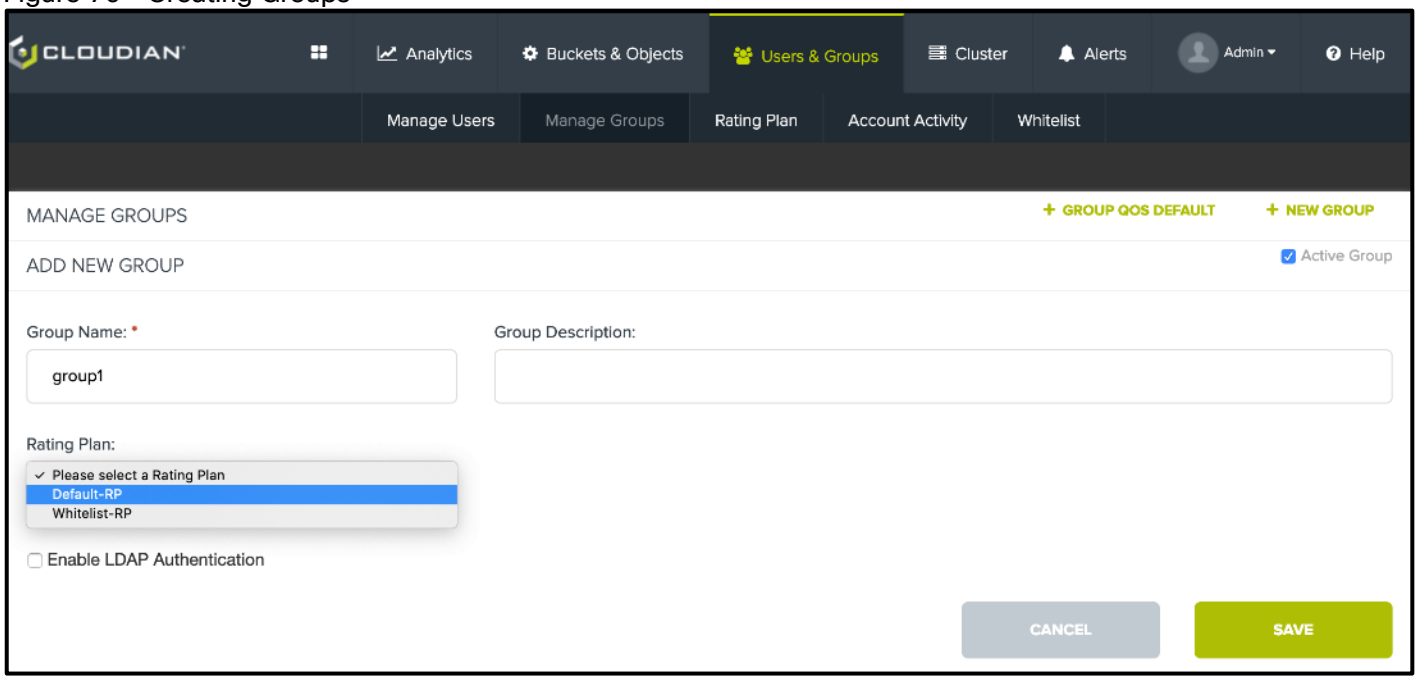
For more information on setting up Alerts in HyperStore, refer to the HyperStore Admin Guide.

Create a Group and User

To be able to create a user, a group/tenant has to be created first. To create a group, follow these steps:

1. Click the Users & Groups tab in the CMC then click the Manage Groups tab. Click +NEW GROUP and provide a name and rating plan.

Figure 79 Creating Groups



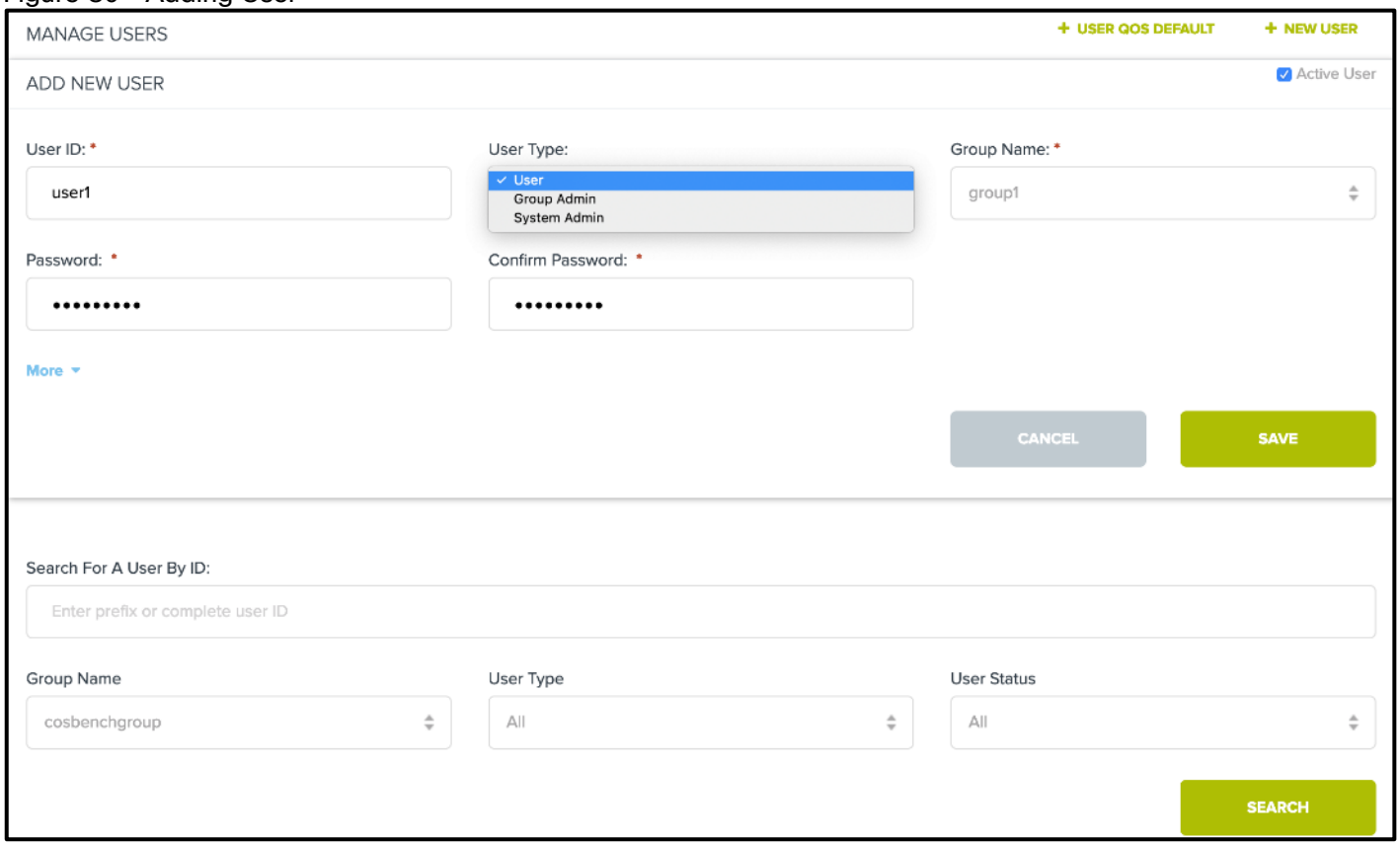
LDAP authentication can be enabled to authenticate users that login to the CMC and automatically create S3 credentials. Each group/tenant can connect to its own LDAP server or Active Directory forest.

For more information on creating groups in HyperStore, refer to the HyperStore Admin Guide.

When the group has been created, you can create a user that you can add to the group. To create a user, follow these steps:

1. Click Manage Users and then click +NEW USER. Provide a User ID, select the User Type, Group Name and provide a password.

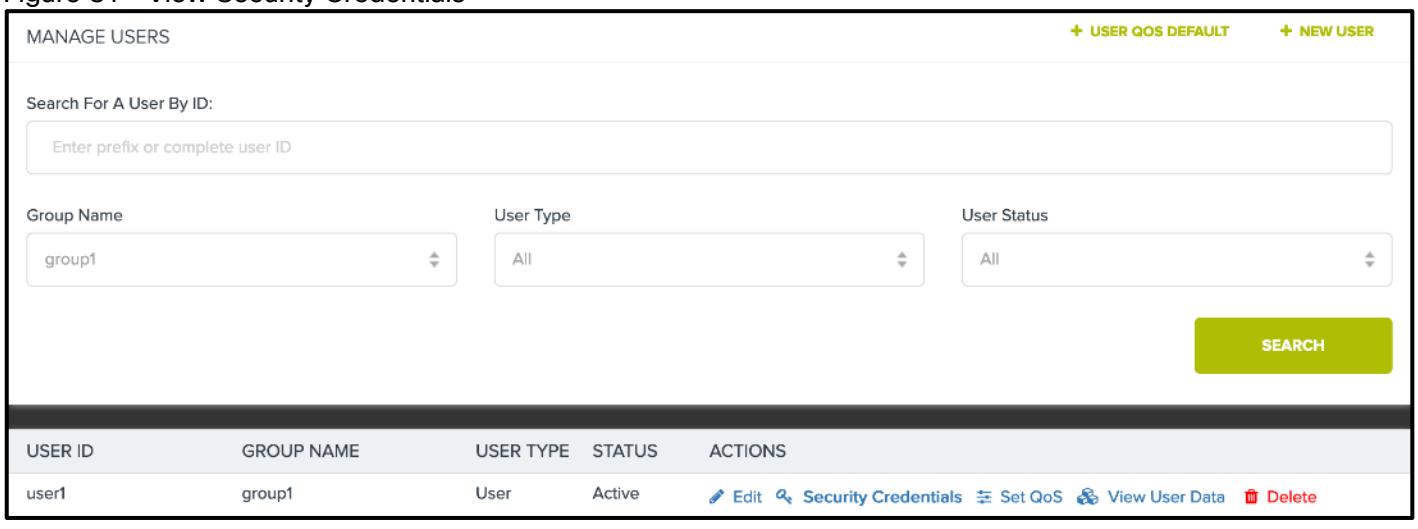
Figure 80 Adding User



For more information on creating users in HyperStore, refer to the HyperStore Admin Guide.

2. Click Security Credentials to view and copy the ACCESS and SECRET key of the newly created user.

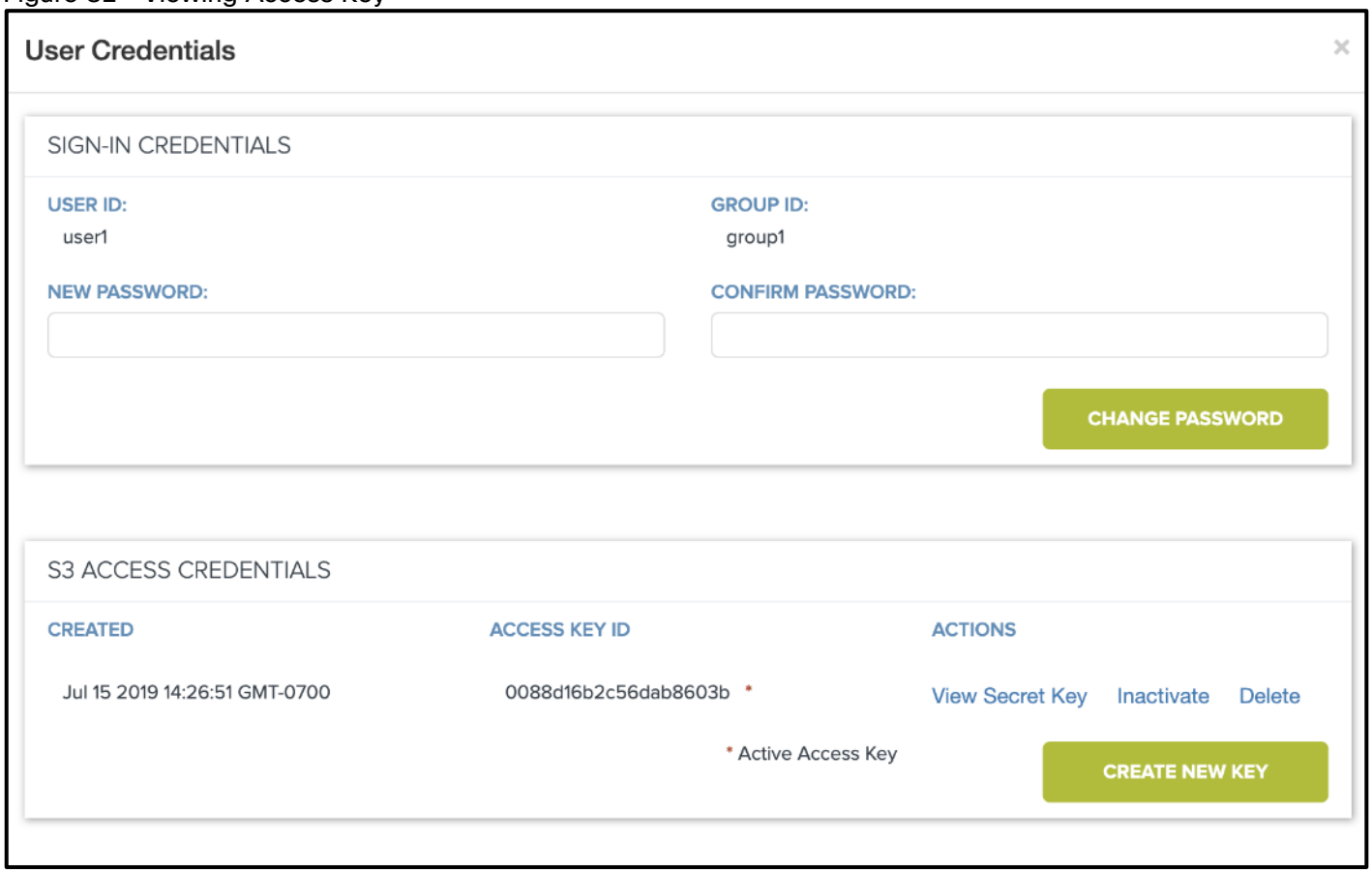
Figure 81 View Security Credentials



3. Copy the ACCESS KEY ID and SECRET KEY, this with the defined s3-endpoint name is what is needed to connect any S3 enabled applications with the user that was just created.

Alternatively, the username, password and group ID can be used to log into the CMC to retrieve this information. The s3-endpoint name(s) can be found under CLUSTER tab and then ClusterConfig.

Figure 82 Viewing Access Key



Users, Groups and credentials can also be created or retrieved through the admin API, please see the Admin API section of th HyperStore Admin Guide.

Create Buckets

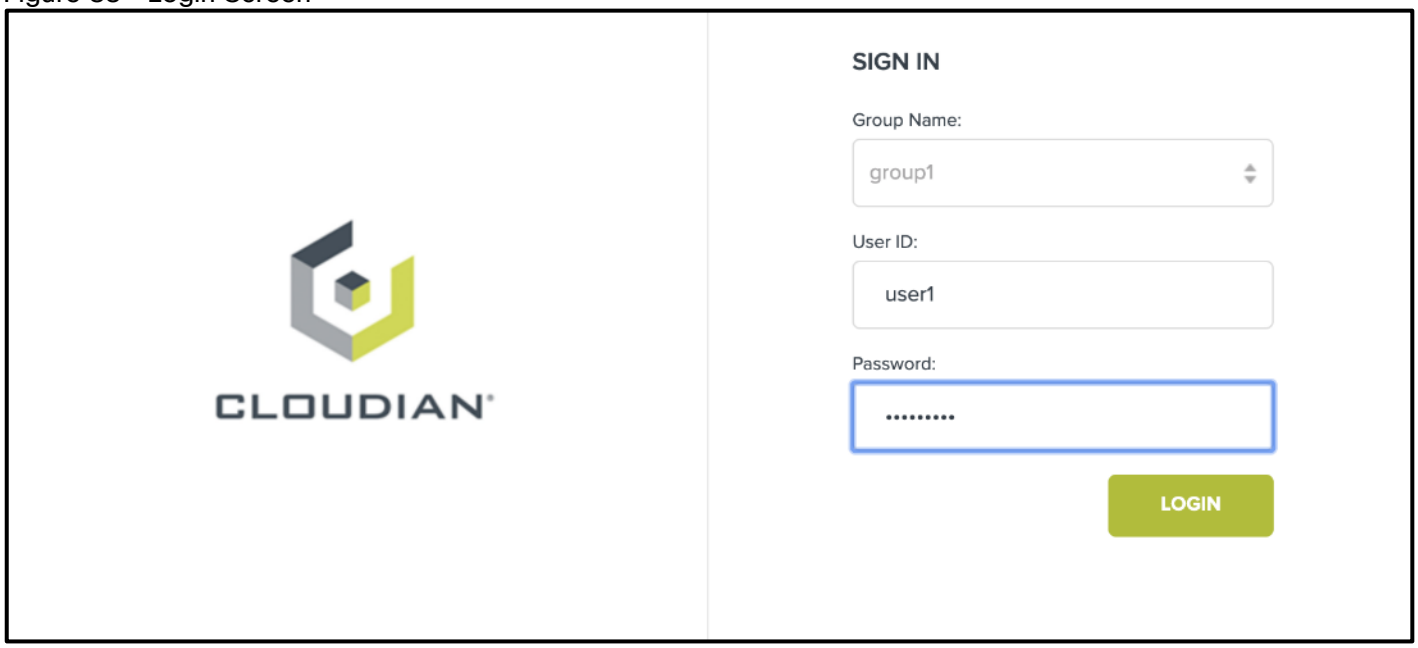
When the user has been created, that user can create buckets and place date into those buckets using any s3 enabled application. When creating a bucket, by default the default Storage Policy will be used as the protection scheme for that bucket.

When multiple Storage Policies are available to the user, the user can choose to assign different Storage Policies for different buckets by creating the buckets through the CMC.

To create buckets through the CMC as a user, follow these steps:

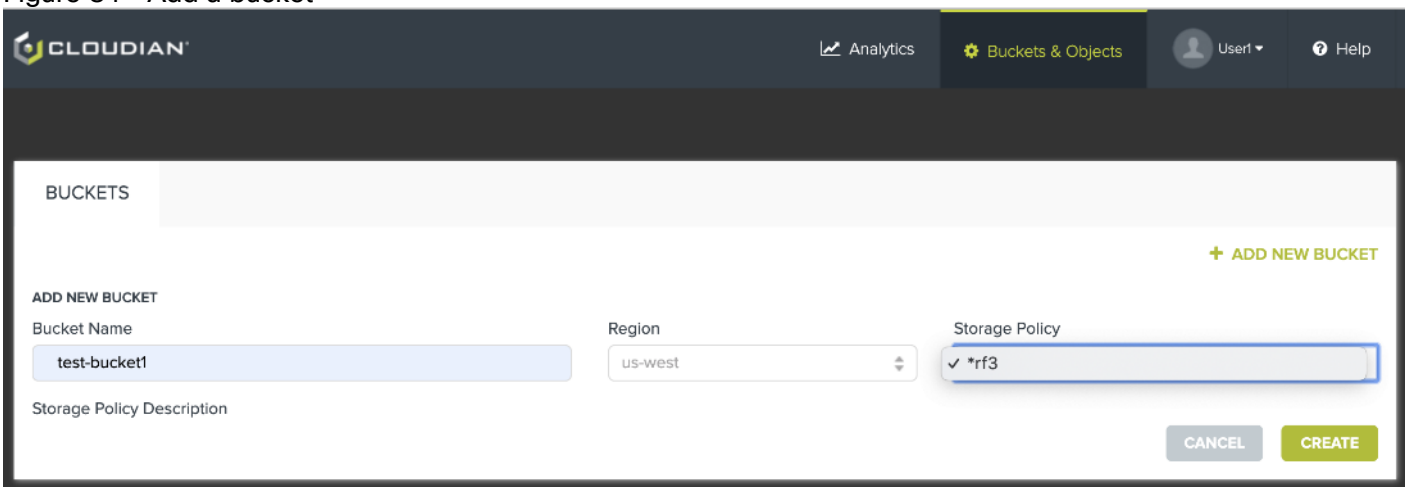
1. Log into the CMC with the associated username, password and GroupID.

Figure 83 Login Screen



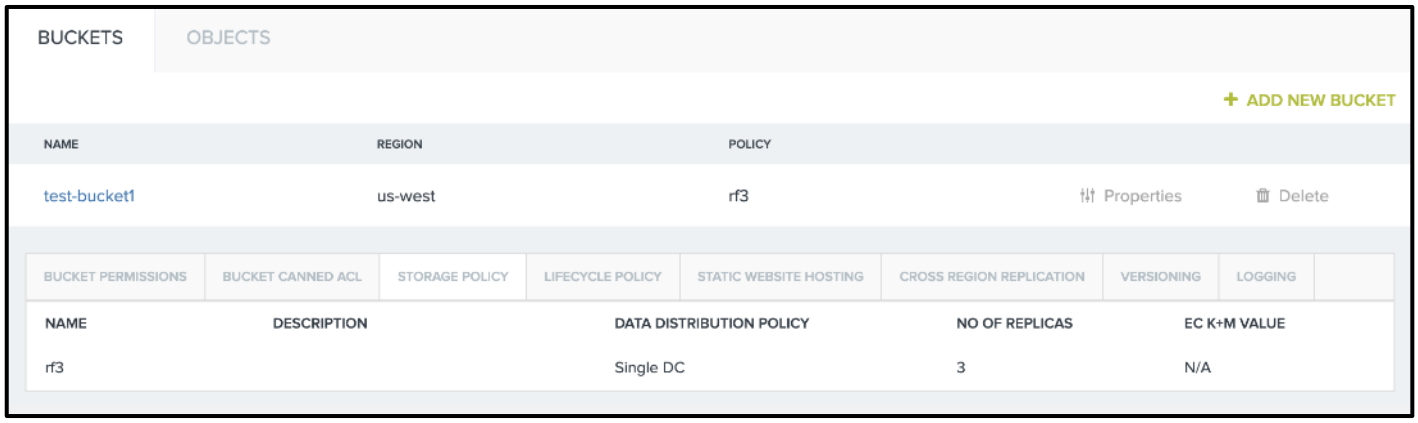
2. To add a new bucket, provide a unique bucket name and select the desired Storage Policy.

Figure 84 Add a bucket



As a System or Group Admin, managed user's data can be viewed by searching for the user in Users and Groups and clicking the View User Data link for the selected user.

3. When a bucket has been created, that bucket can be configured for permissions, life cycle policies, static webhosting, CCR, versioning and logging by clicking the Properties for that bucket.



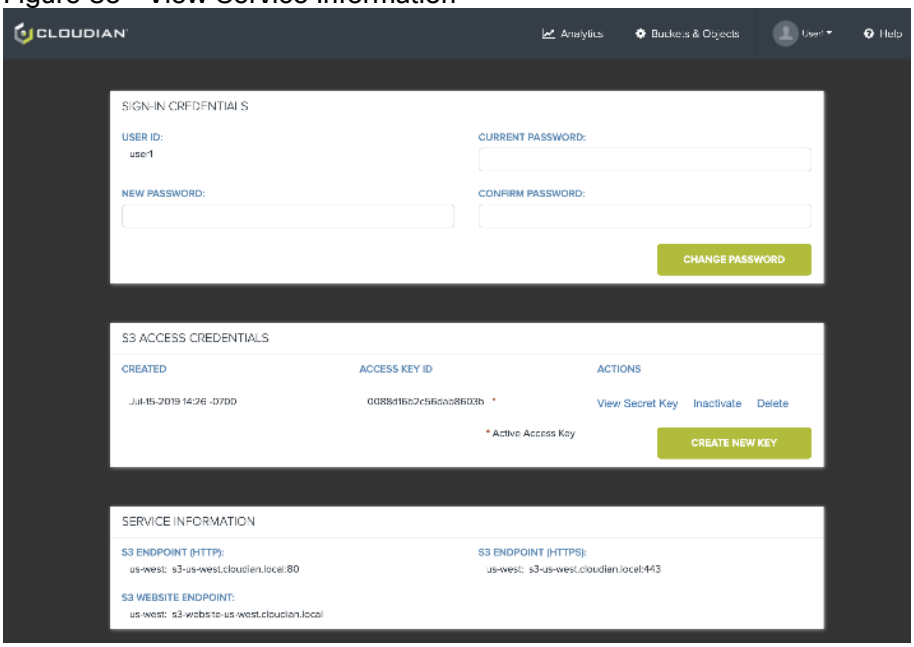
More information on bucket properties can be found in the HyperStore Admin Guide.

Verify Credentials and Service Endpoints as a User

While logged in as the user, the s3 credentials can be found under the tab Username and then Security Credentials Additional keys can be created and the CMC password can be changed.

The Security Credentials Page also shows the SERVICE INFORMATION that next to the s3 credentials is required to setup a connection to the Object Store.

Figure 85 View Service Information



Cloudian Hyperstore Installation verification

Verify HyperStore S3 Connectivity

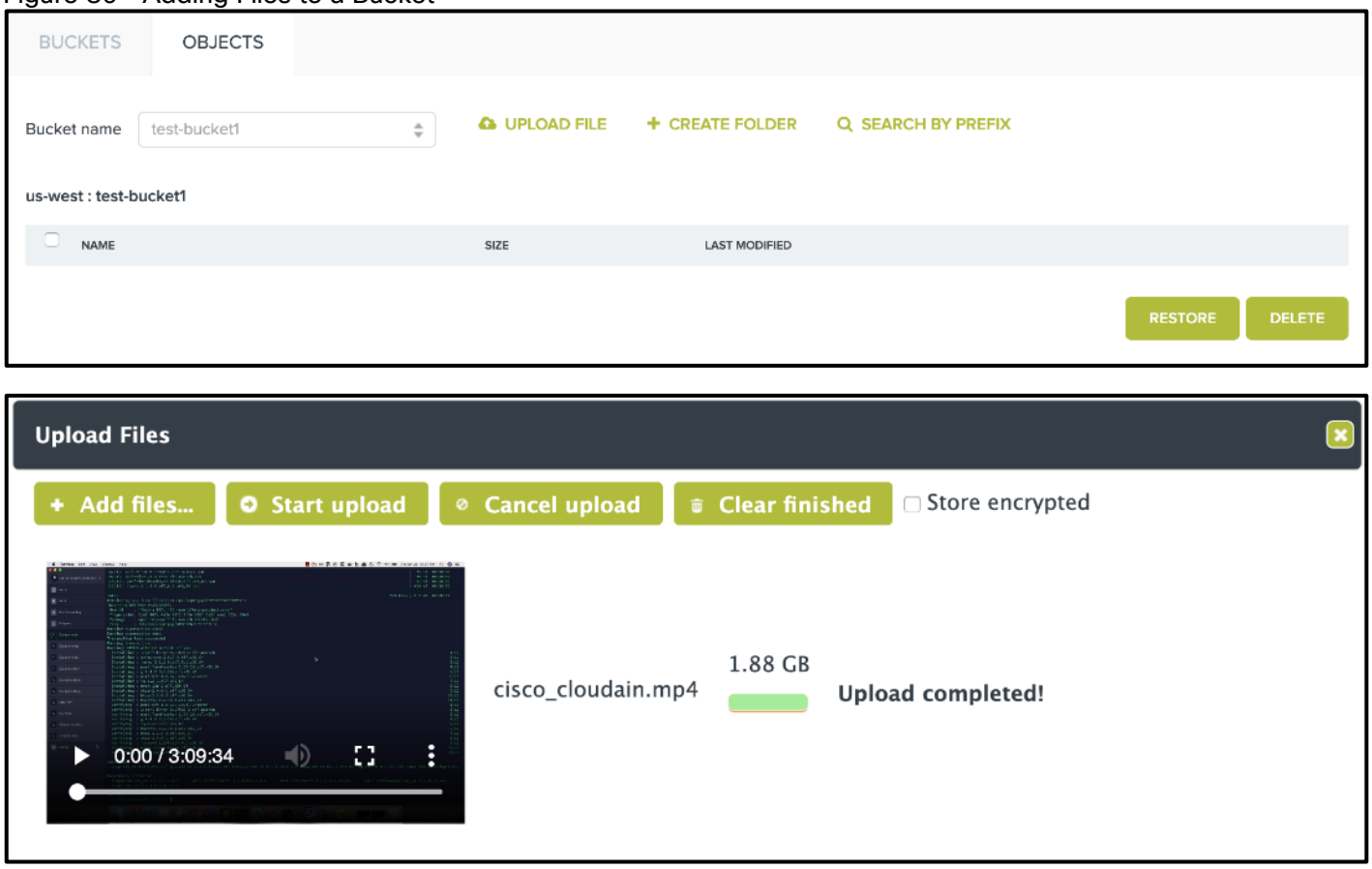
At this point, Cloudian HyperStore is installed, configured and ready to receive data. To verify Cloudian HyperStore is functioning properly and the connected environment is configured correctly, you need to run some tests.

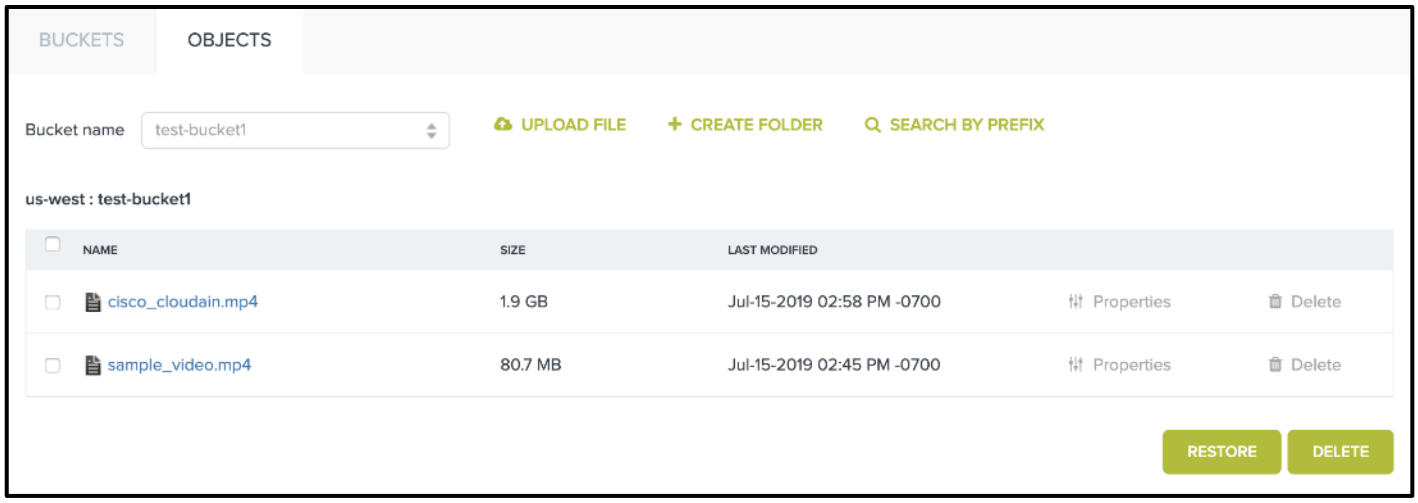
To verify the HyperStore S3 connectivity, follow these steps:

1. Verify if objects can be uploaded through the CMC. Login as a user or use the View User Data link to go to the bucket of the user that was created earlier and select UPLOAD FILE.

2. Select Add files… in the popup window followed by start upload. Congratulations, you just uploaded the first object into the Cloudian HyperStore Object Store!

Figure 86 Adding Files to a Bucket





The default maximum size for an Object that can be uploaded through the CMC is 5GB. More information on uploading objects through the CMC can be found in the HyperStore Admin Guide.

3. A client-side test should be run to ensure that the rest of the connecting infrastructure is correctly configured to support Cloudian HyperStore.

4. Connect to any Linux distribution client server and install s3cmd. When using Centos you will need to have epel-release installed.

```
yum install -y s3cmd
```

5. Configure s3cmd to use the s3 credentials of user1 and the s3 service endpoint used by Cloudian Hyperstore. To configure s3cmd follow the instructions below:

```
[root@storage-client1 ~]# s3cmd --configure

Enter new values or accept defaults in brackets with Enter.
Refer to user manual for detailed description of all options.

Access key and Secret key are your identifiers for Amazon S3. Leave them empty for
using the env variables.
Access Key: 0088d16b2c56dab8603b
Secret Key: prMKOstG47C1D9vd3KNh8RuO2t9ifKyrDgIv3ZC6
Default Region [US]: ucs-west

Use s3.amazonaws.com for S3 Endpoint and not modify it to the target Amazon S3.
S3 Endpoint [s3.amazonaws.com]: s3-us-west.cloudian.local

Use %(bucket)s.s3.amazonaws.com to the target Amazon S3. %(bucket)s and %(location)s
vars can be used
if the target S3 system supports dns based buckets.
DNS-style bucket+hostname:port template for accessing a bucket
[%(bucket)s.s3.amazonaws.com]: %(bucket)s.s3-us-west.cloudian.local

Encryption password is used to protect your files from reading
by unauthorized persons while in transfer to S3
Encryption password: P@ssw0rd!
```

```
Path to GPG program [/usr/bin/gpg]:

When using secure HTTPS protocol all communication with Amazon S3
servers is protected from 3rd party eavesdropping. This method is
slower than plain HTTP, and can only be proxied with Python 2.7 or newer
Use HTTPS protocol [Yes]: No

On some networks all internet access must go through a HTTP proxy.
Try setting it here if you can't connect to S3 directly
HTTP Proxy server name:

New settings:
  Access Key: 0088d16b2c56dab8603b
  Secret Key: prMKOstG47C1D9vd3KNh8RuO2t9ifKyrDgIv3ZC6
  Default Region: ucs-west
  S3 Endpoint: s3-us-west.cloudian.local
  DNS-style bucket+hostname:port template for accessing a bucket: %(bucket)s.s3-us-
west.cloudian.local
  Encryption password: P@ssw0rd!
  Path to GPG program: /usr/bin/gpg
  Use HTTPS protocol: False
  HTTP Proxy server name:
  HTTP Proxy server port: 0

Test access with supplied credentials? [Y/n] Y
Please wait, attempting to list all buckets...
Success. Your access key and secret key worked fine :-)

Now verifying that encryption works...
Success. Encryption and decryption worked fine :-)

Save settings? [y/N] y
Configuration saved to '/root/.s3cfg'
[root@storage-client1 ~]#
```

When DNS is NOT available, the client should have the service endpoints defined in /etc/hosts. As the usage of wildcards is not allowed in /etc/hosts, the buckets that will be used by the client should also be defined in /etc/hosts.

6. When s3cmd has been successfully configured with s3 credentials and the s3 service endpoint name, you can list the buckets that have been created with the following command:

```
 [root@storage-client1 ~]# s3cmd ls
2019-07-15 21:33  s3://test-bucket1
```

7. The objects in the bucket can be listed with the following command:

```
 [root@storage-client1 ~]# s3cmd la
2019-07-15 21:58 2023138702   s3://test-bucket1/cisco_cloudain.mp4
2019-07-15 21:45  84650499   s3://test-bucket1/sample_video.mp4
```

8. To download an object from Cloudian HyperStore to the local home directory, run the following command:

```
 [root@storage-client1 ~]# s3cmd get s3://test-bucket1/sample_video.mp4
download: 's3://test-bucket1/sample_video.mp4' -> './sample_video.mp4'  [1 of 1]
 84650499 of 84650499   100% in    0s   330.49 MB/s  done
```

9. To upload a file to Cloudian HyperStore, run the following:

```
[root@storage-client1 ~]# s3cmd put node-list.txt s3://test-bucket1/
upload: 'node-list.txt' -> 's3://test-bucket1/node-list.txt'  [1 of 1]
 96 of 96   100% in    0s     7.00 kB/s  done
```

10. To verify the file was successfully uploaded, run the following:

```
 [root@storage-client1 ~]# s3cmd la
2019-07-15 21:58 2023138702   s3://test-bucket1/cisco_cloudain.mp4
2019-07-15 22:34         96   s3://test-bucket1/node-list.txt
2019-07-15 21:45  84650499   s3://test-bucket1/sample_video.mp4
```

11. When all tests are successful, your environment has been setup correctly and is ready for client access.

Add Datacenter and Nodes

Adding nodes and datacenters to Cloudian HyperStore is easy and can be done through the CMC. However, the candidate cluster nodes need to be properly prepared, similar to the nodes that were used for the initial installation.

Ensure the OS is installed and basic network configuration has been setup for the additional nodes that will be added to Cloudian HyperStore.

Multiple datacenters can also be installed during the initial installation, when doing so make sure to correctly specify the Datacenter Name in the fourth tab of the survey.csv.

Prepare the new nodes

From the master node, run the system_setup.sh script and select option 9 Prep New Node to add to Cluster. This remotely connects to the new candidate server and copy over the required binaries to prepare the system.

```
    System Setup

   9) Prep New Node to Add to Cluster

Choice: 9
System Setup » Run On Cluster

   1) Prep New Node to Add to Cluster

   P) Return to the Previous Menu

Choice: 1
System Setup » Prep New Node to Add to Cluster

IP Address of new node: 192.168.10.24


Attempting to install SSH key on 192.168.10.24
If your root password is the same on all (or most) nodes in the cluster, you can
supply it as a cluster password
If you do not want to supply a password, each server will prompt for one when
connecting.
Cluster Password:

==> On Server: 192.168.10.24 ... Done
Adding SSH Key to '/root/.ssh/authorized_keys'

Attempting to copy self extract installer to 192.168.10.24
Checking and creating remote directory path before transferring
'/root/CloudianPackages/selfextract_prereq_el7.bin'

==> On Server: 192.168.100.246 ... Done
==> Transferring to Server: 192.168.10.24 ... Done
    Attempting to copy System Setup script to 192.168.100.24
Checking and creating remote directory path before transferring
'/root/CloudianPackages/system_setup.sh'
```

```
    ==> On Server: 192.168.10.24 ... Done
==> Transferring to Server: 192.168.100.24 ... Done
    ==> On Server: 192.168.10.24 ... <==
```

The script is now remotely running on the new candidate cluster node and should be properly prepared as described in section Prepare Cluster Nodes. Complete this step for all nodes that will be added to the cluster.

```
System Setup (storage-node4)

   1) Configure Networking
   2) Change Timezone
   3) Install & Configure Prerequisites
   4) Setup Disks
   5) Script Settings

   X) Return to Master Node

Choice:
```

After running system_setup.sh on all new nodes and completing preparations, run the following command to verify all data drives have been successfully mounted.

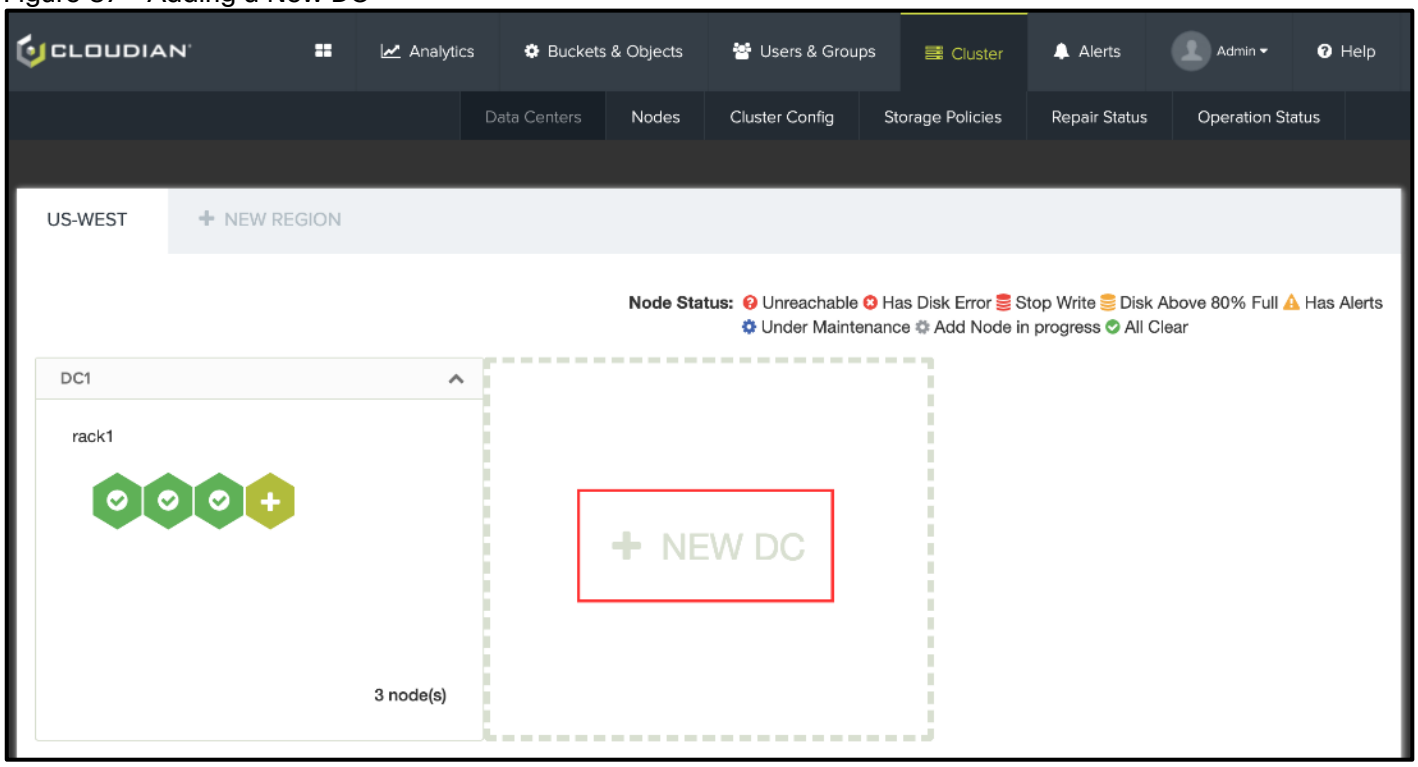
```
ssh -t -i /root/CloudianPackages/cloudian-installation-key storage-node4 df -h |grep
-c cloudian

Connection to storage-node4 closed.
10
```

Add a New DC

When it's confirmed that all drives are mounted correctly, connect to the CMC. To add a new DC, follow these steps:

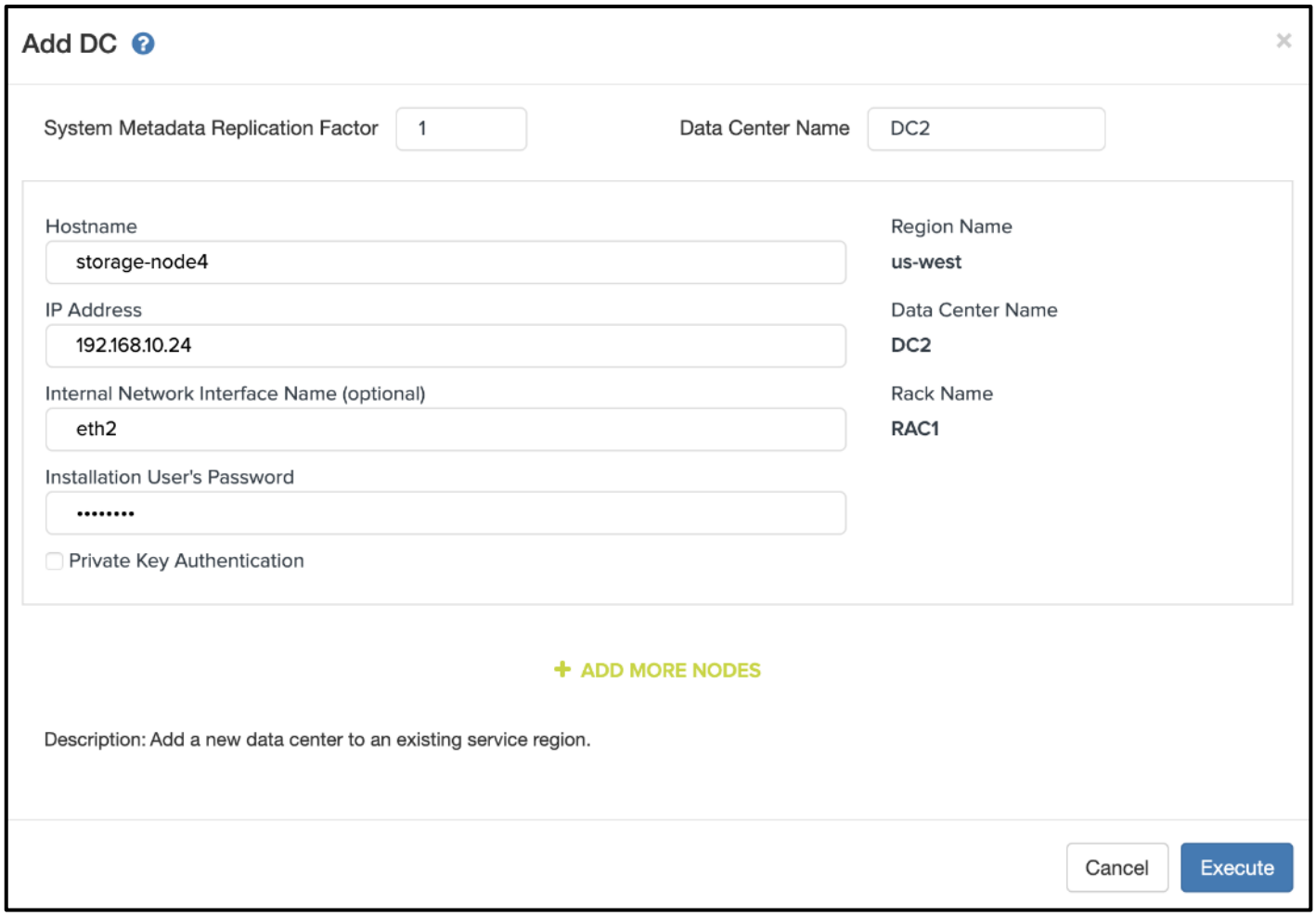
1. Go to Cluster, click Data Centers and then click the + NEW DC next to the initial installed data center.

Figure 87 Adding a New DC



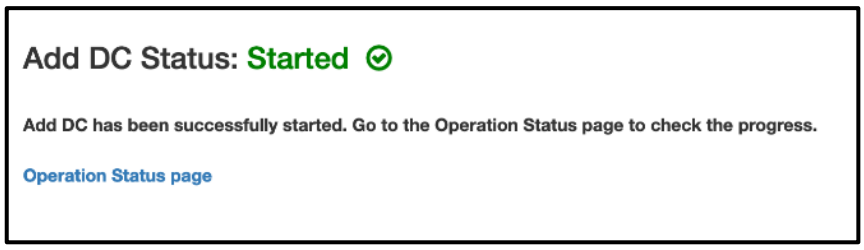
2. When adding a DC, set the System Metadata Replication Factor to 3 and complete the required fields to add a host and click ADD MORE NODES to add details for all nodes that will be added to the additional data center. Once the information for all nodes has been provided and verified click execute to start the add DC progress.

Make sure to provide the correct Interface name for the internal cluster communication network since it might be different from the initial DC when using VLAN interfaces.

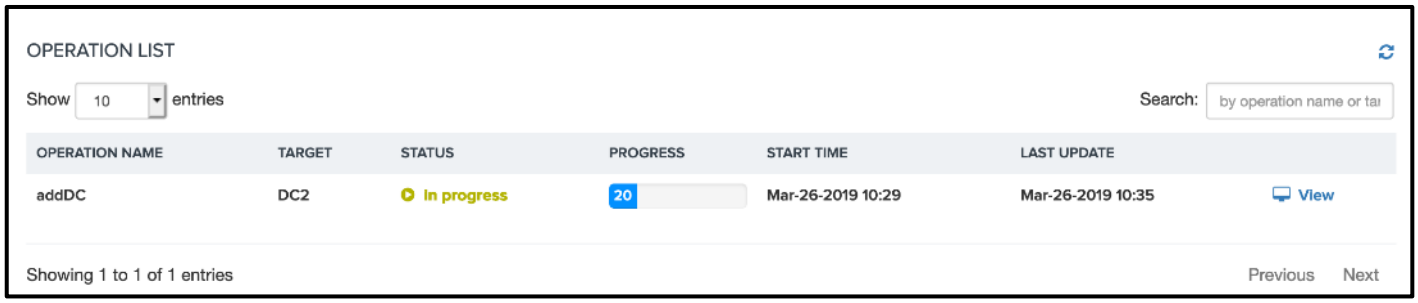


Do not use more than one Rack Name unless you are using replication and fully understand the concept.

3. To follow the progress of the DC add operation, go to the Operation Status Page by clicking the Operation Status Page link or by going to Cluster followed by Operation Status.

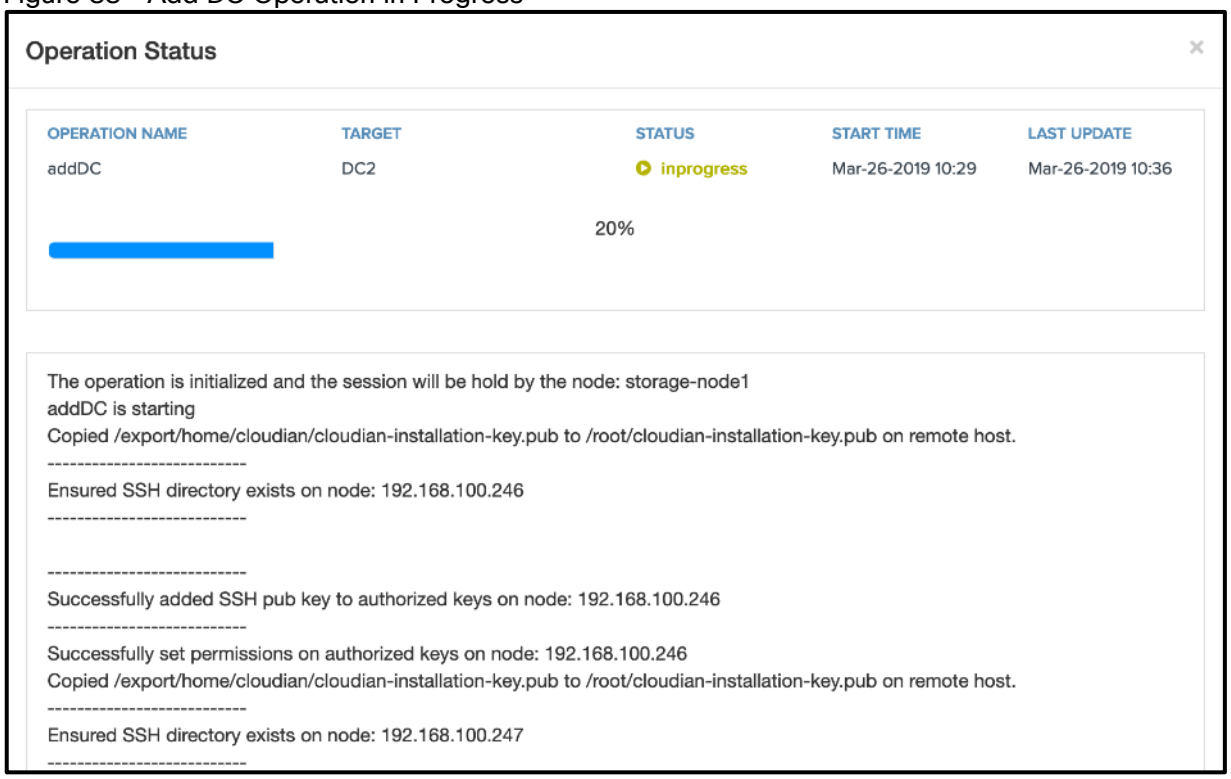


4. Click View to see more details of the addDC operation.



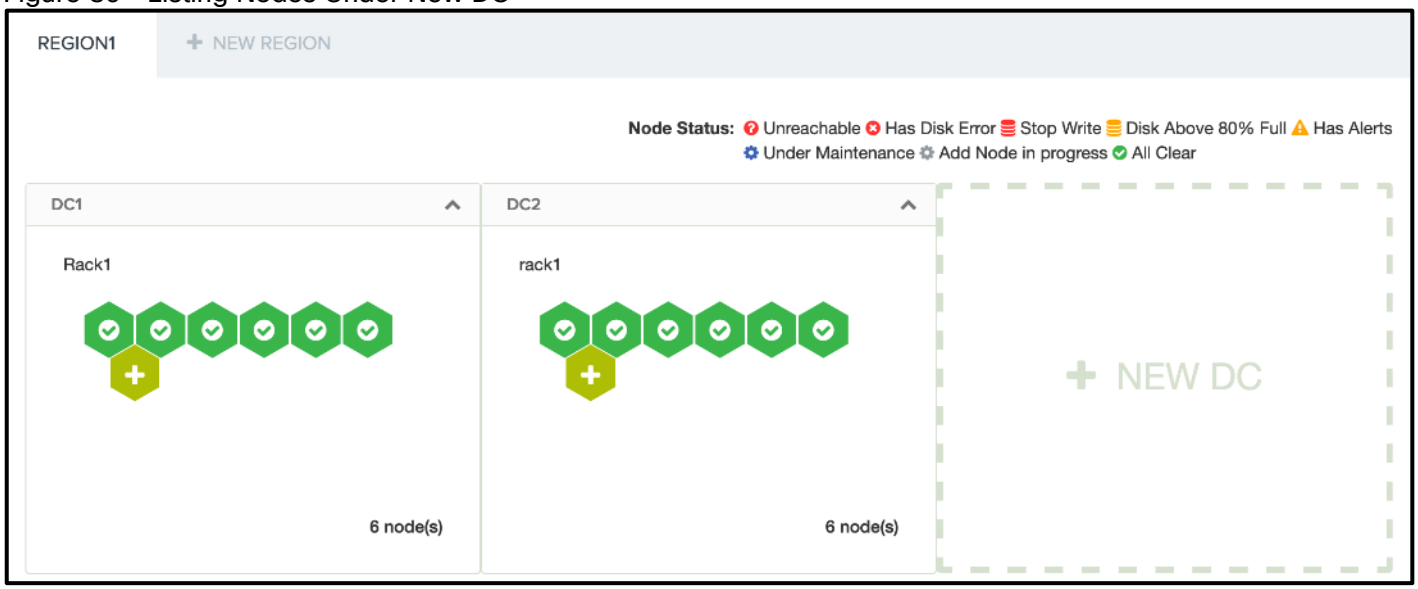
5. The addDC operation will create SSH keys, update the survey.csv and run all of the pre-installation checks before adding the nodes to the additional datacenter.

Figure 88 Add DC Operation in Progress



6. When the addDC operation has completed successfully, the new datacenter and nodes should now be listed under Datacenters and are ready to be configured with a storage policy.

Figure 89 Listing Nodes Under New DC



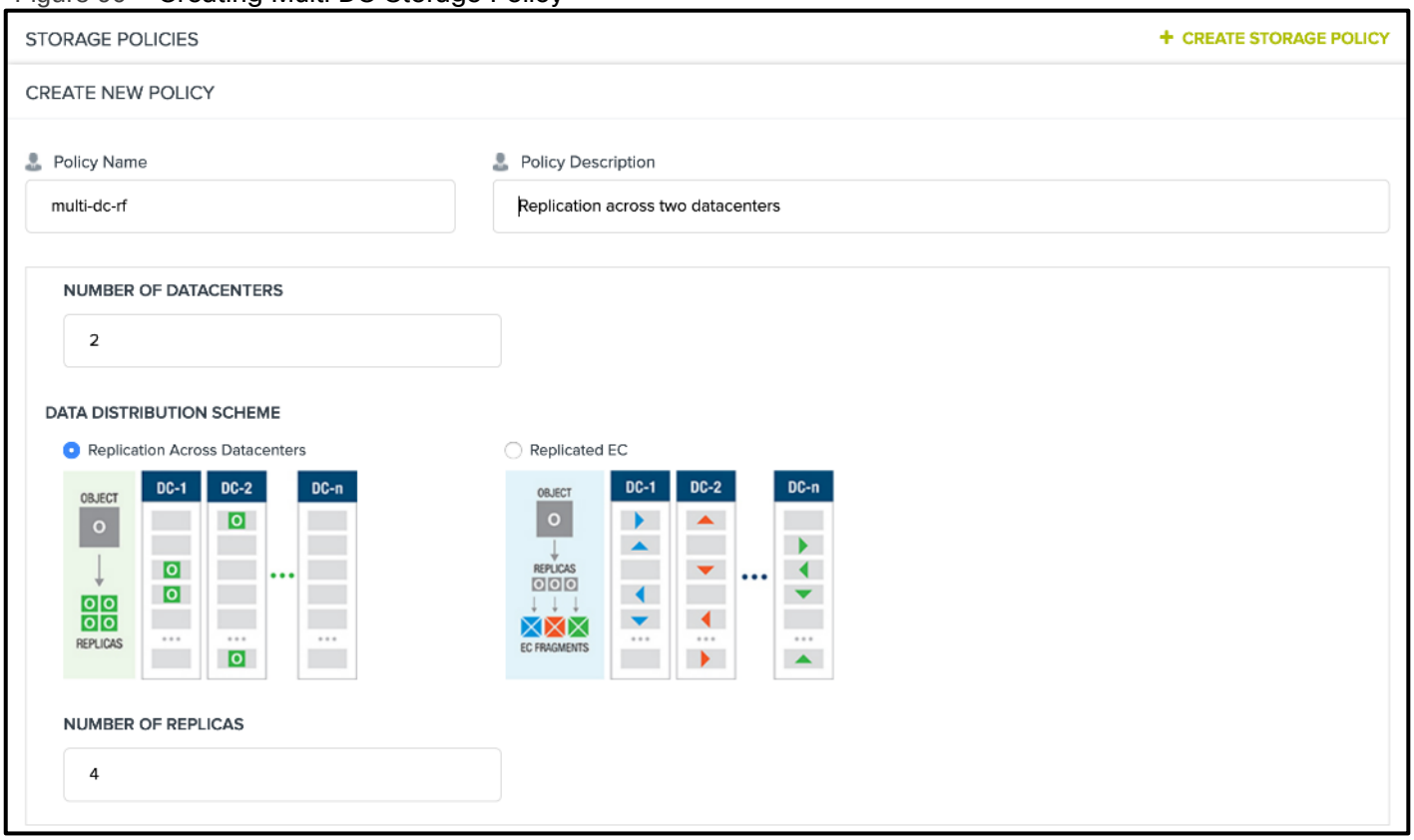
To add a new node to an existing data center, follow the same node preparation steps and simply click the + symbol in the DC you want to add the node to be added and complete the node information. For more information on adding datacenters, refer to the HyperStore Admin Guide.

Create a Multi DC Storage Policy

When the additional datacenter and nodes have been successfully added, a storage policy has to be created to utilize the new DC. To create a multi DC storage policy, follow these steps:

1. Within the CMC, go to Cluster, click Storage Policies and then click \ + CREATE STORAGE POLICY.
2. Specify a Policy Name and select the desired data distribution scheme which could be Replication Across datacenters or 'Replicated EC with different EC scheme options.
3. For Replication Across datacenters, select the total number of copies that need to be replicated.

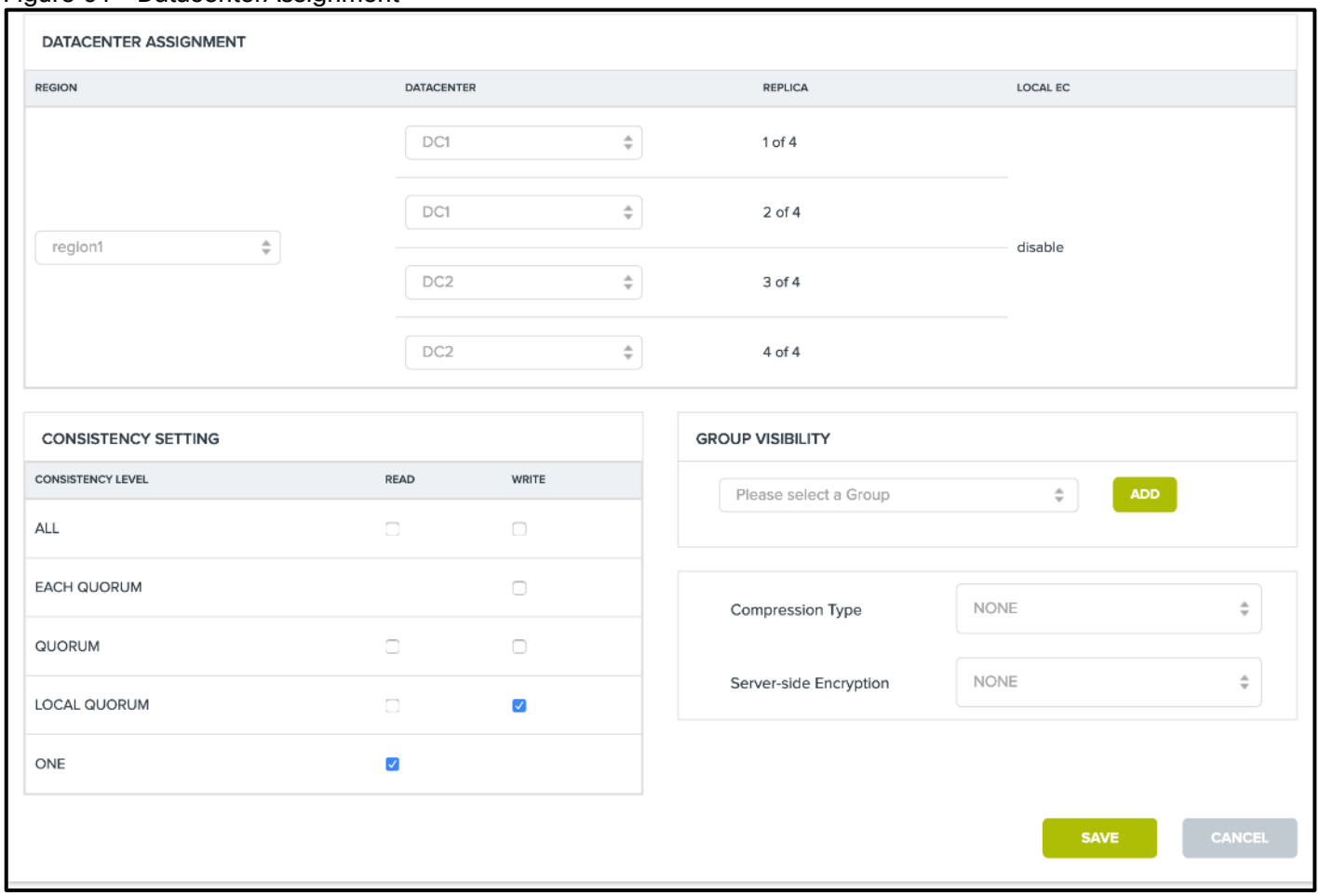
Figure 90 Creating Multi DC Storage Policy



4. Select the datacenter assignment for each replica and set the desired consistency level. Since there are multiple DC's there are a lot more consistency level options. To ensure strong consistency in the local DC but eventual consistency for replication to the remote DC, make sure to use LOCAL QUORUM.

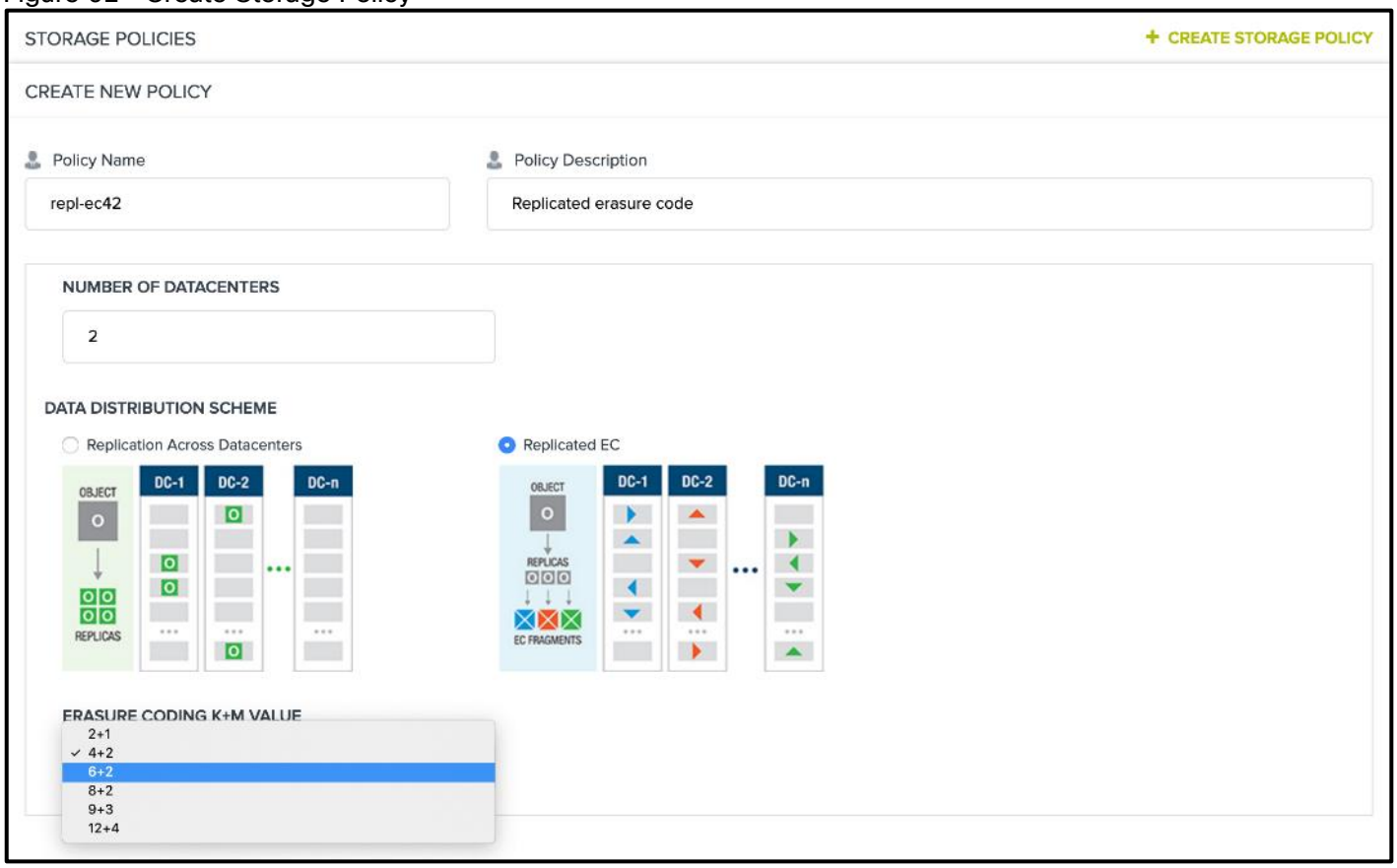
For more information about consistency levels, refer to the HyperStore Admin Guide .

Figure 91 DatacenterAssignment



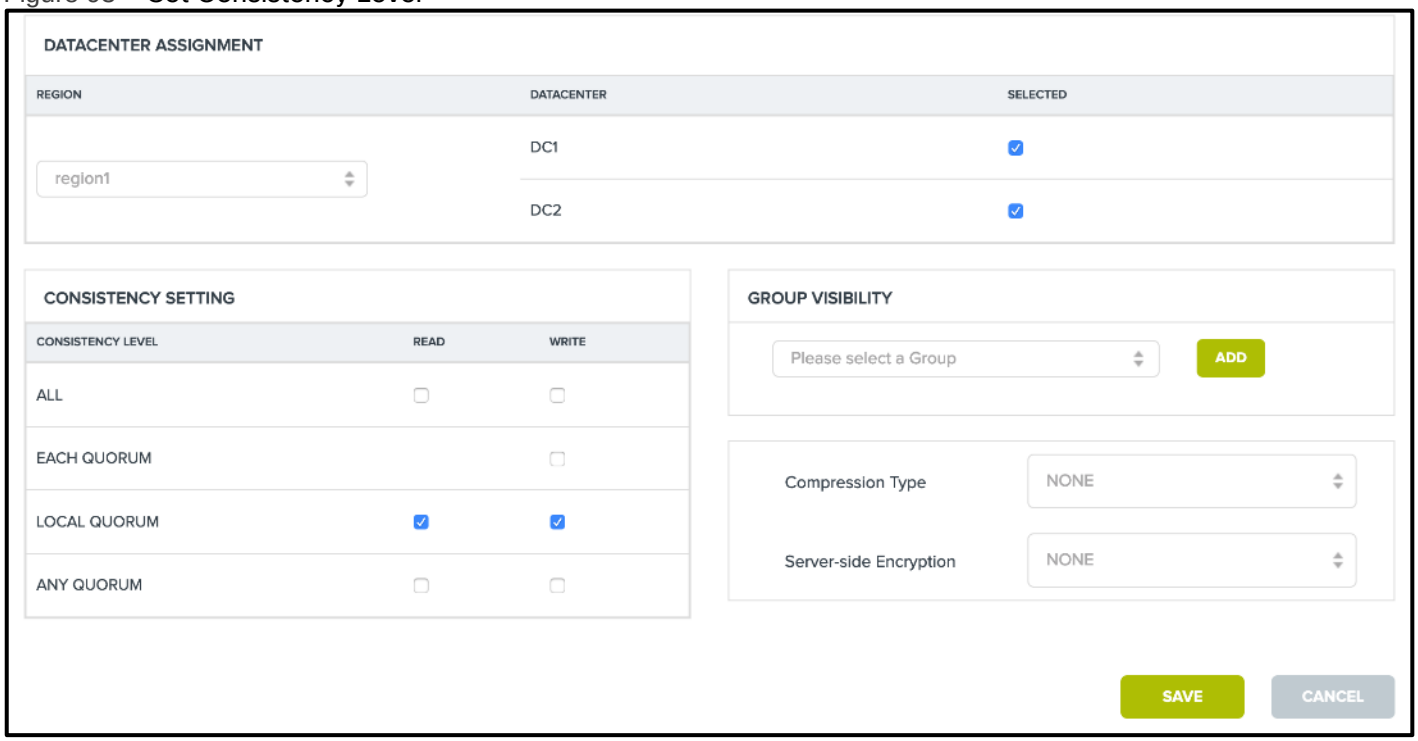
5. For Replicated EC select the desired Erasure Coding scheme.

Figure 92 Create Storage Policy

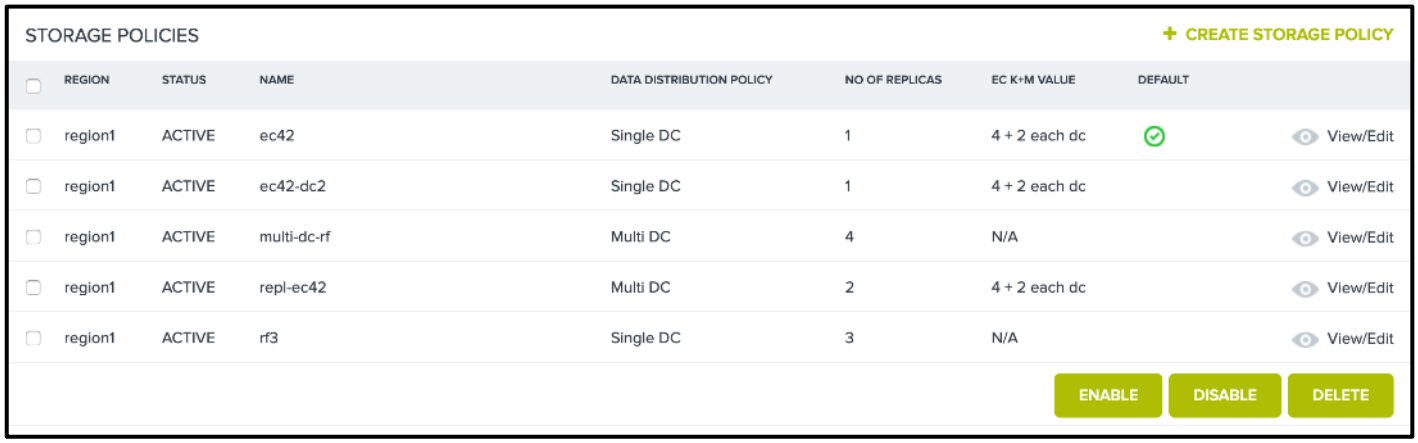


6. Select target DC's and set the desired consistency level and click Save to exit.

Figure 93 Set Consistency Level



7. A local storage policy for the new DC must be created. When more than one storage policy exists, the default policy can be set by editing a storage policy and configure the set default setting.



For more information on installing and configuring Cloudian HyperStore, refer to the HyperStore Admin Guide.

Performance

S3 Performance was evaluated on the 3-node Cloudian HyperStore system running on Cisco UCS C240 M5 hardware. The evaluation was done with both Intel Xeon scalable family CPUs and 2nd Generation Intel® Xeon® scalable family CPUs. A 3-way replication storage policy was considered for tests. The goal of the performance testing was to evaluate peak object performance under ideal conditions.

The performance tests were done using Intel's Cosbench with the following range of object sizes and worker threads:

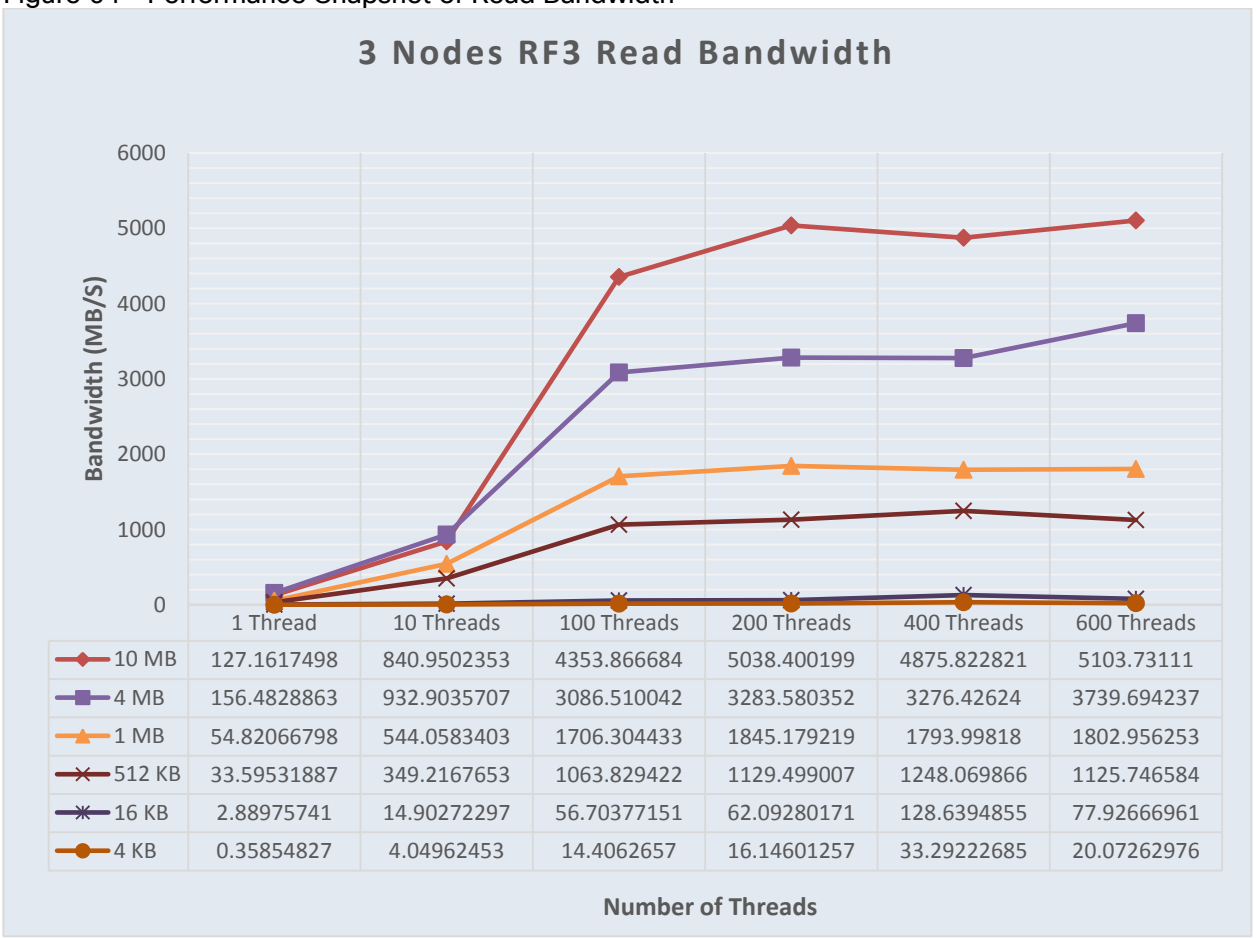
```
Object sizes: 4KB, 16KB, 512KB, 1MB, 4MB, 10MB
Threads: 1, 10, 100, 200, 400, 600
```

To run the Cosbench workload, 3 client nodes with 40Gb ethernet were used as Cosbench drivers. The same Cosbench workload was used to run load on buckets with different 3-way replication storage policy.

Performance with 2nd Generation Intel Xeon Scalable Family CPUs (Cascade Lake)

3-Way Replication – Read Performance

Figure 94 Performance Snapshot of Read Bandwidth

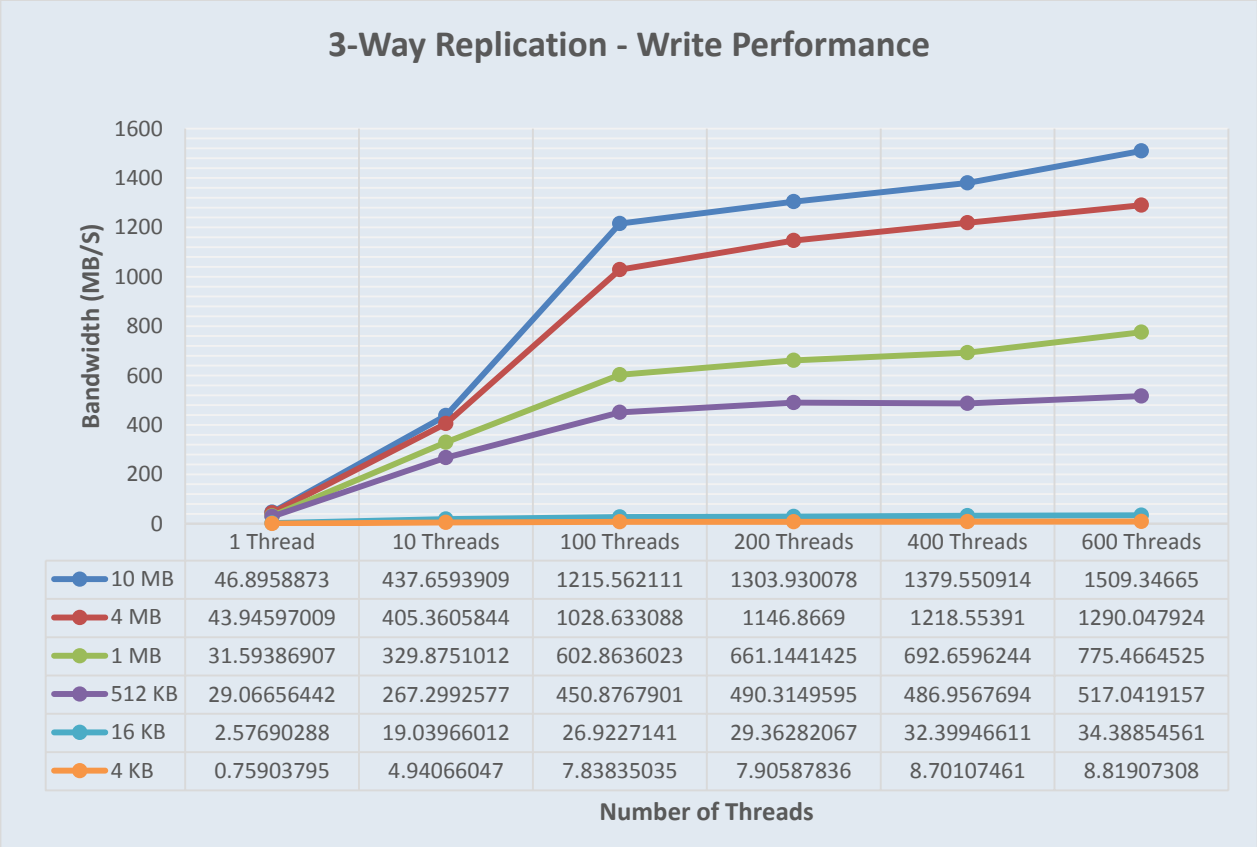


	1 Thread	10 Threads	100 Threads	200 Threads	400 Threads	600 Threads
10 MB	127.1617498	840.9502353	4353.866684	5038.400199	4875.822821	5103.73111
4 MB	156.4828863	932.9035707	3086.510042	3283.580352	3276.42624	3739.694237
1 MB	54.82066798	544.0583403	1706.304433	1845.179219	1793.99818	1802.956253
512 KB	33.59531887	349.2167653	1063.829422	1129.499007	1248.069866	1125.746584
16 KB	2.88975741	14.90272297	56.70377151	62.09280171	128.6394855	77.92666961
4 KB	0.35854827	4.04962453	14.4062657	16.14601257	33.29222685	20.07262976

Notice that the bandwidth peaks at 5.1 GB/s attempting to read 10 MB objects with 600 threads running in parallel.

3-Way Replication – Write Performance

Figure 95 Performance Snapshot of Write Bandwidth

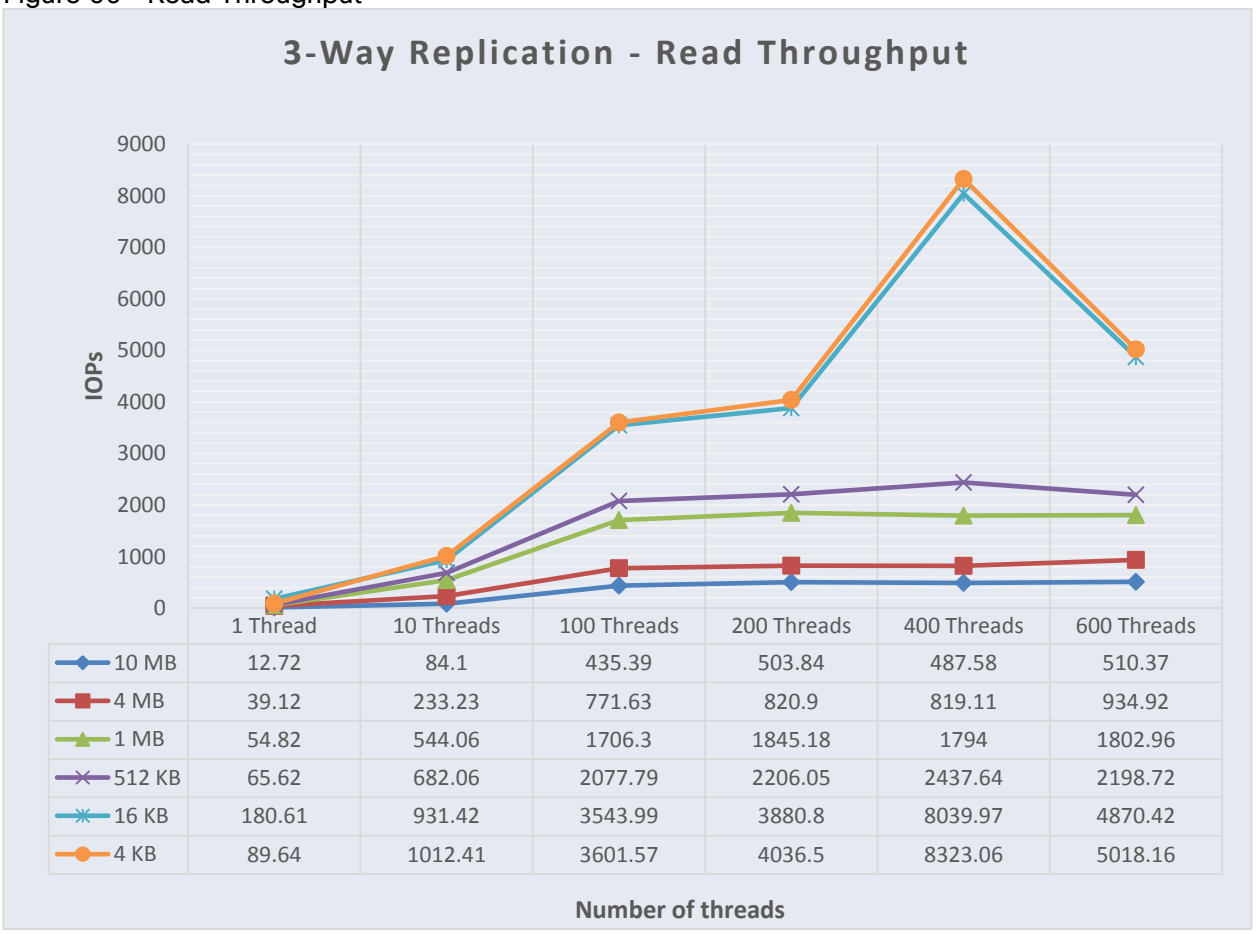


	1 Thread	10 Threads	100 Threads	200 Threads	400 Threads	600 Threads
10 MB	46.8958873	437.6593909	1215.562111	1303.930078	1379.550914	1509.34665
4 MB	43.94597009	405.3605844	1028.633088	1146.8669	1218.55391	1290.047924
1 MB	31.59386907	329.8751012	602.8636023	661.1441425	692.6596244	775.4664525
512 KB	29.06656442	267.2992577	450.8767901	490.3149595	486.9567694	517.0419157
16 KB	2.57690288	19.03966012	26.9227141	29.36282067	32.39946611	34.38854561
4 KB	0.75903795	4.94066047	7.83835035	7.90587836	8.70107461	8.81907308

Notice that bandwidth peaks at 1.5 GB/s attempting to write 10 MB objects with 600 threads running in parallel.

3-Way Replication – Read Throughput

Figure 96 Read Throughput

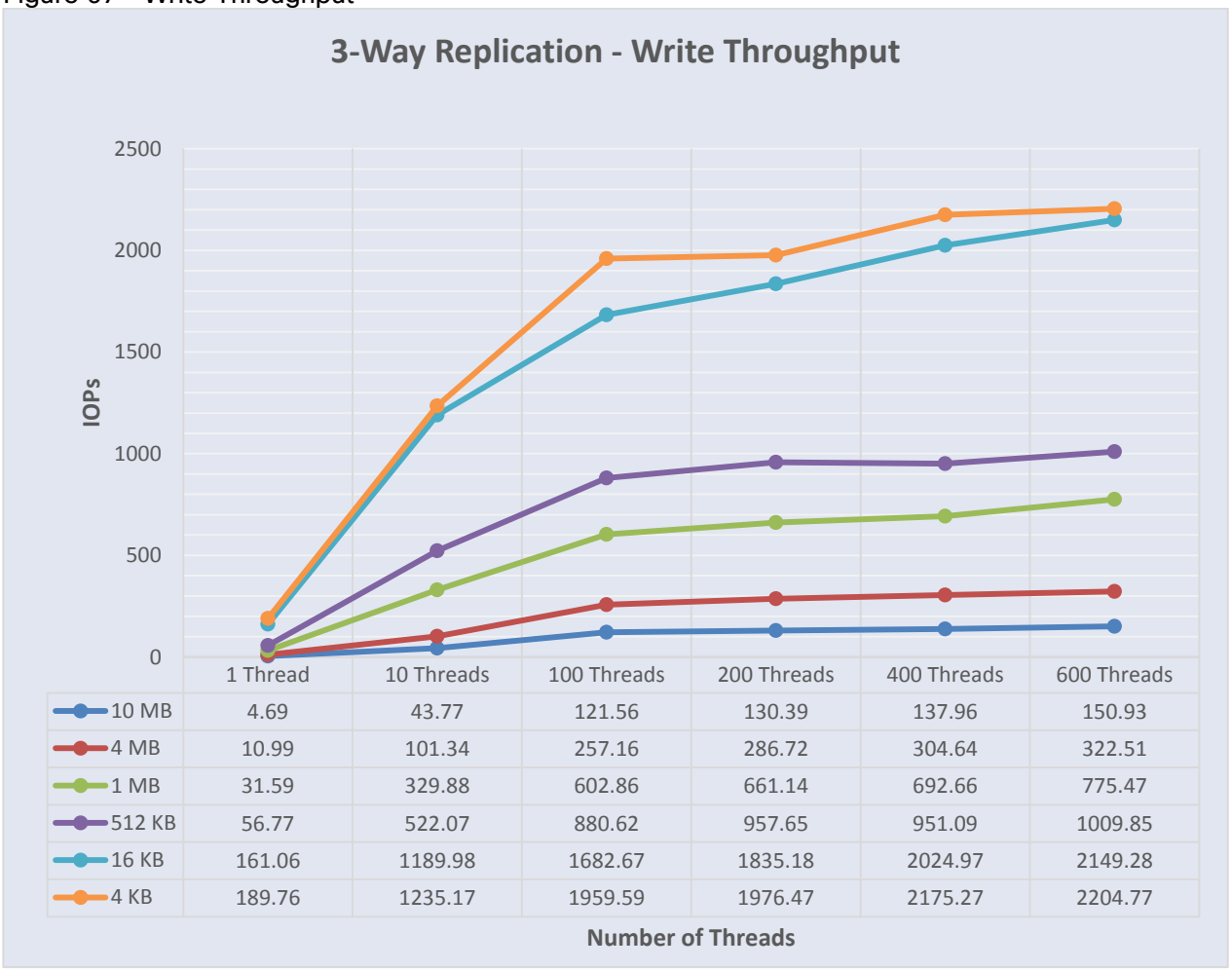


	1 Thread	10 Threads	100 Threads	200 Threads	400 Threads	600 Threads
10 MB	12.72	84.1	435.39	503.84	487.58	510.37
4 MB	39.12	233.23	771.63	820.9	819.11	934.92
1 MB	54.82	544.06	1706.3	1845.18	1794	1802.96
512 KB	65.62	682.06	2077.79	2206.05	2437.64	2198.72
16 KB	180.61	931.42	3543.99	3880.8	8039.97	4870.42
4 KB	89.64	1012.41	3601.57	4036.5	8323.06	5018.16

Read throughput peaks at around 8323 Operation per second with 400 parallel threads for an oject size of 4 KB.

3-Way Replication – Write Throughput

Figure 97 Write Throughput

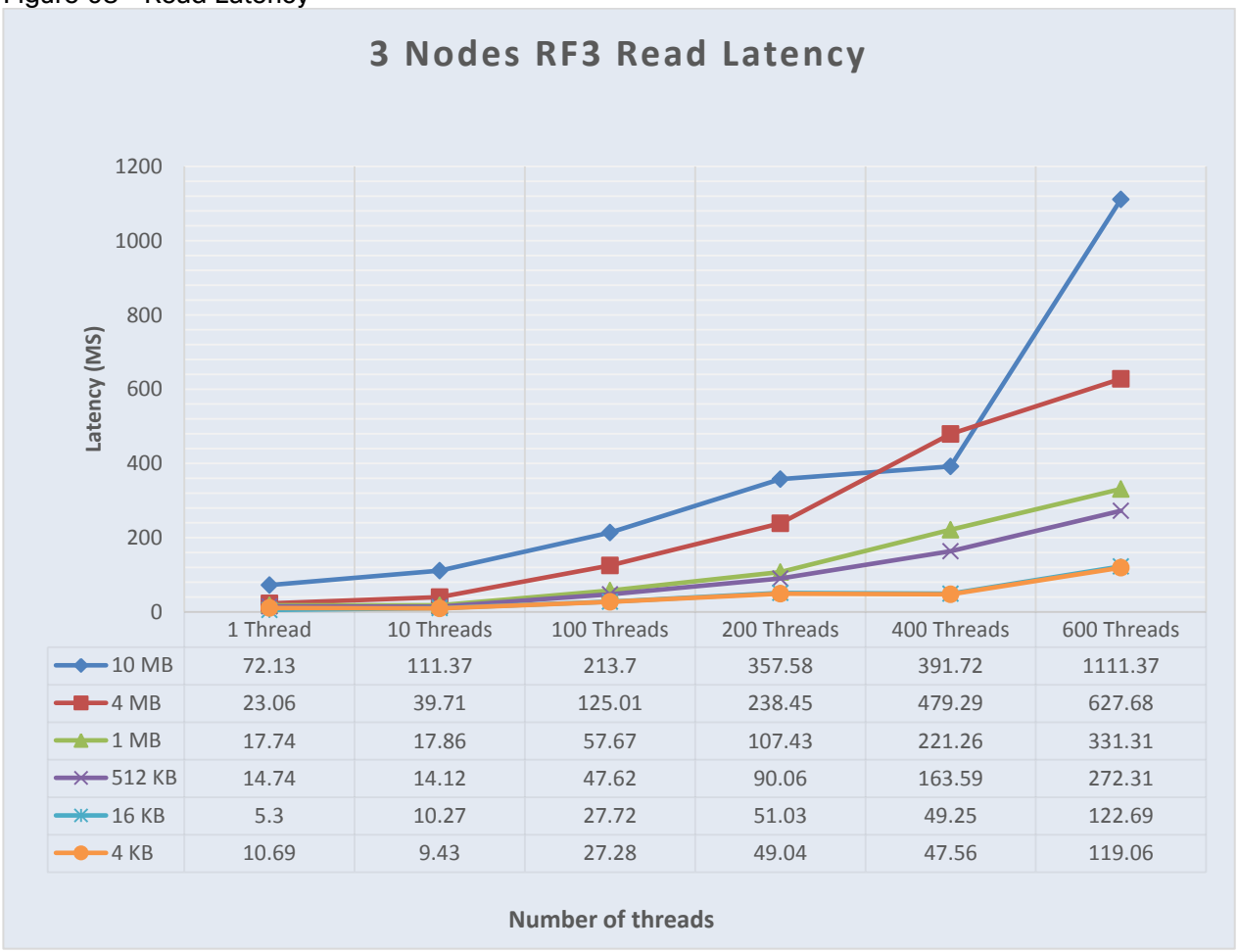


	1 Thread	10 Threads	100 Threads	200 Threads	400 Threads	600 Threads
10 MB	4.69	43.77	121.56	130.39	137.96	150.93
4 MB	10.99	101.34	257.16	286.72	304.64	322.51
1 MB	31.59	329.88	602.86	661.14	692.66	775.47
512 KB	56.77	522.07	880.62	957.65	951.09	1009.85
16 KB	161.06	1189.98	1682.67	1835.18	2024.97	2149.28
4 KB	189.76	1235.17	1959.59	1976.47	2175.27	2204.77

Write throughput peaks at around 2204 Operation per second with 600 parallel threads for an oject size of 4 KB.

3-Way Replication - Read Latency

Figure 98 Read Latency

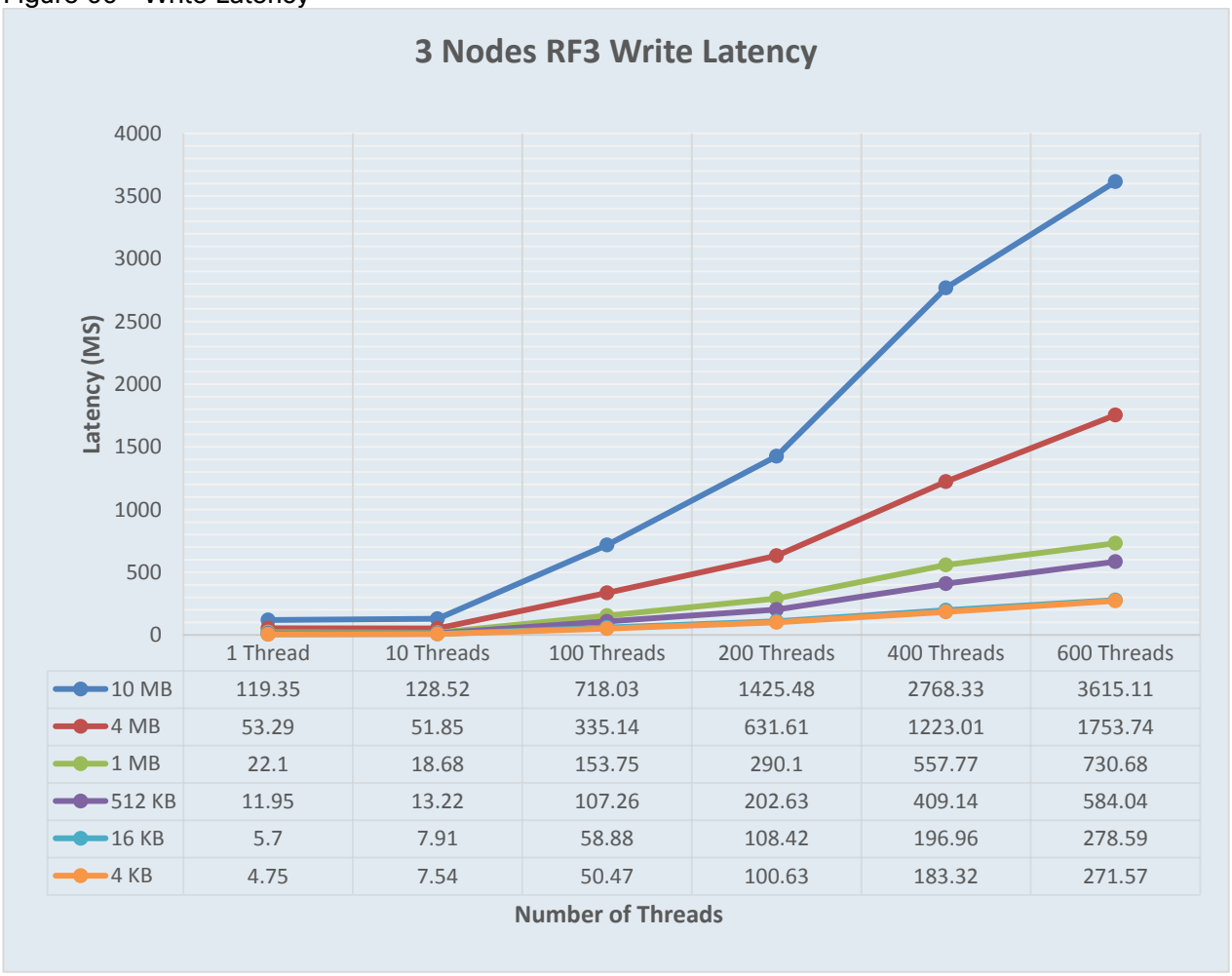


	1 Thread	10 Threads	100 Threads	200 Threads	400 Threads	600 Threads
10 MB	72.13	111.37	213.7	357.58	391.72	1111.37
4 MB	23.06	39.71	125.01	238.45	479.29	627.68
1 MB	17.74	17.86	57.67	107.43	221.26	331.31
512 KB	14.74	14.12	47.62	90.06	163.59	272.31
16 KB	5.3	10.27	27.72	51.03	49.25	122.69
4 KB	10.69	9.43	27.28	49.04	47.56	119.06

Latency of 5.3 Milliseconds was observed with read operation for an object size of 16 KB with one thread. Latency of 1111 Milliseconds was observed with read operation for an object size of 10 MB with 600 parallel threads.

3-Way Replication - Write Latency

Figure 99 Write Latency

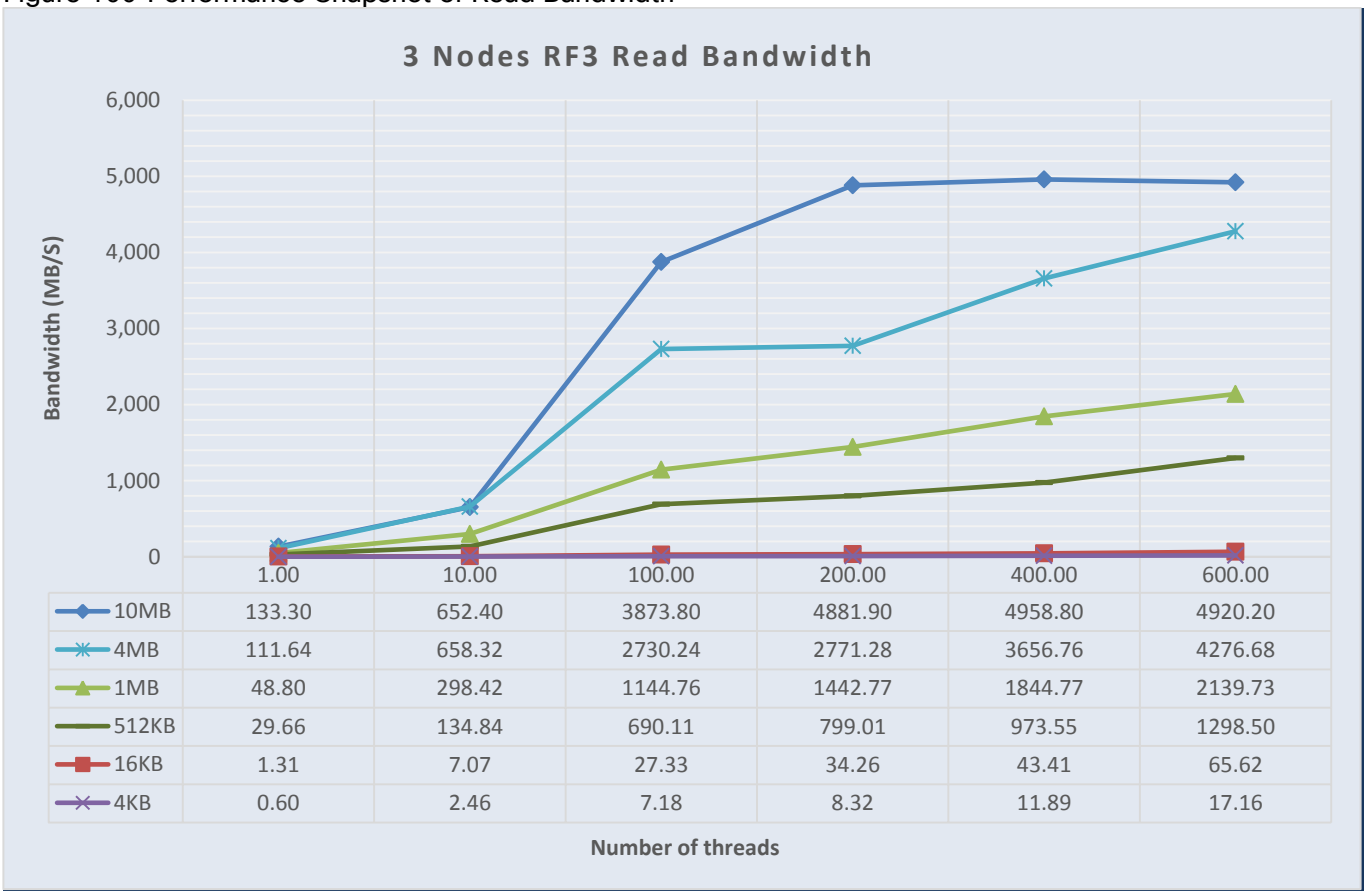


Latency of 4.75 Milliseconds was observed with write operation for an object size of 4 KB with one thread. Latency of 3.6 Seconds was observed for object size of 10 MB with 600 parallel threads.

Performance with Intel Xeon Scalable Family CPUs(Sklylake)

3-Way Replication – Read Performance

Figure 100 Performance Snapshot of Read Bandwidth

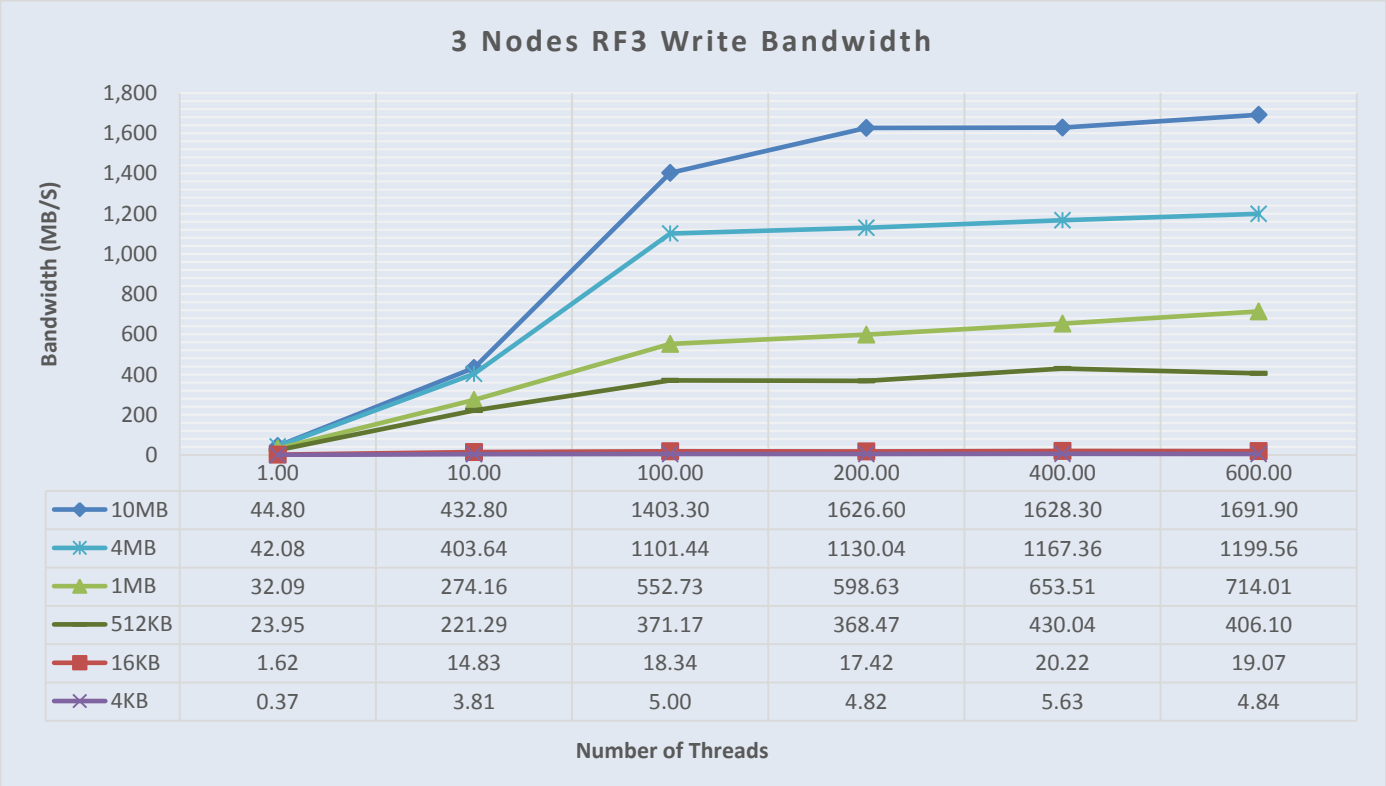


	1.00	10.00	100.00	200.00	400.00	600.00
10MB	133.30	652.40	3873.80	4881.90	4958.80	4920.20
4MB	111.64	658.32	2730.24	2771.28	3656.76	4276.68
1MB	48.80	298.42	1144.76	1442.77	1844.77	2139.73
512KB	29.66	134.84	690.11	799.01	973.55	1298.50
16KB	1.31	7.07	27.33	34.26	43.41	65.62
4KB	0.60	2.46	7.18	8.32	11.89	17.16

Notice that bandwidth peaks at 4.96 GB/s attempting to read 10 MB objects with 400 threads running in parallel.

3-Way Replication – Write Performance

Figure 101 Performance Snapshot of Write Bandwidth

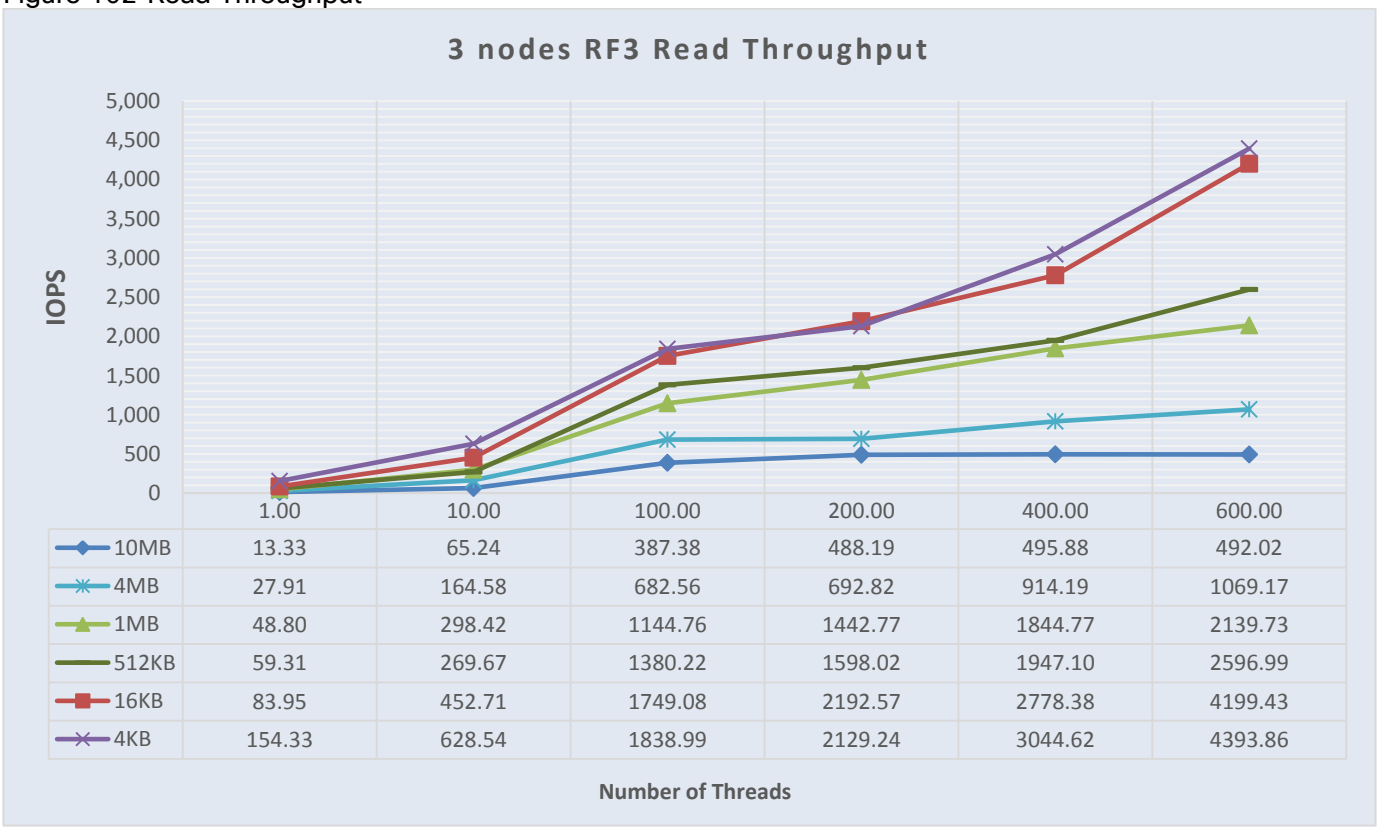


	1.00	10.00	100.00	200.00	400.00	600.00
10MB	44.80	432.80	1403.30	1626.60	1628.30	1691.90
4MB	42.08	403.64	1101.44	1130.04	1167.36	1199.56
1MB	32.09	274.16	552.73	598.63	653.51	714.01
512KB	23.95	221.29	371.17	368.47	430.04	406.10
16KB	1.62	14.83	18.34	17.42	20.22	19.07
4KB	0.37	3.81	5.00	4.82	5.63	4.84

Notice that bandwidth peaks at 1.7 GB/s attempting to write 10 MB objects with 600 threads running in parallel.

3-Way Replication – Read Throughput

Figure 102 Read Throughput

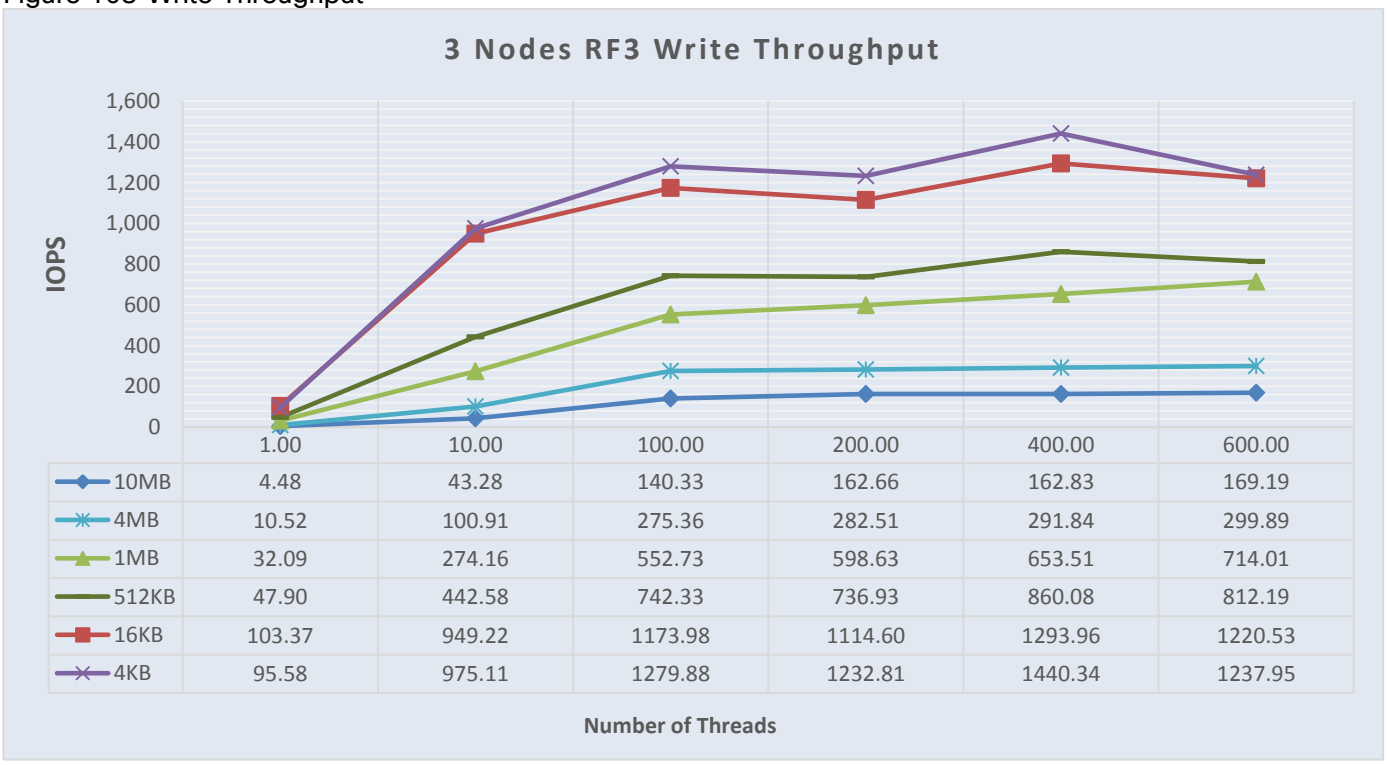


	1.00	10.00	100.00	200.00	400.00	600.00
10MB	13.33	65.24	387.38	488.19	495.88	492.02
4MB	27.91	164.58	682.56	692.82	914.19	1069.17
1MB	48.80	298.42	1144.76	1442.77	1844.77	2139.73
512KB	59.31	269.67	1380.22	1598.02	1947.10	2596.99
16KB	83.95	452.71	1749.08	2192.57	2778.38	4199.43
4KB	154.33	628.54	1838.99	2129.24	3044.62	4393.86

Read throughput peaks at around 4394 Operation per second with 600 parallel threads for an oject size of 4 KB.

3-Way Replication – Write Throughput

Figure 103 Write Throughput

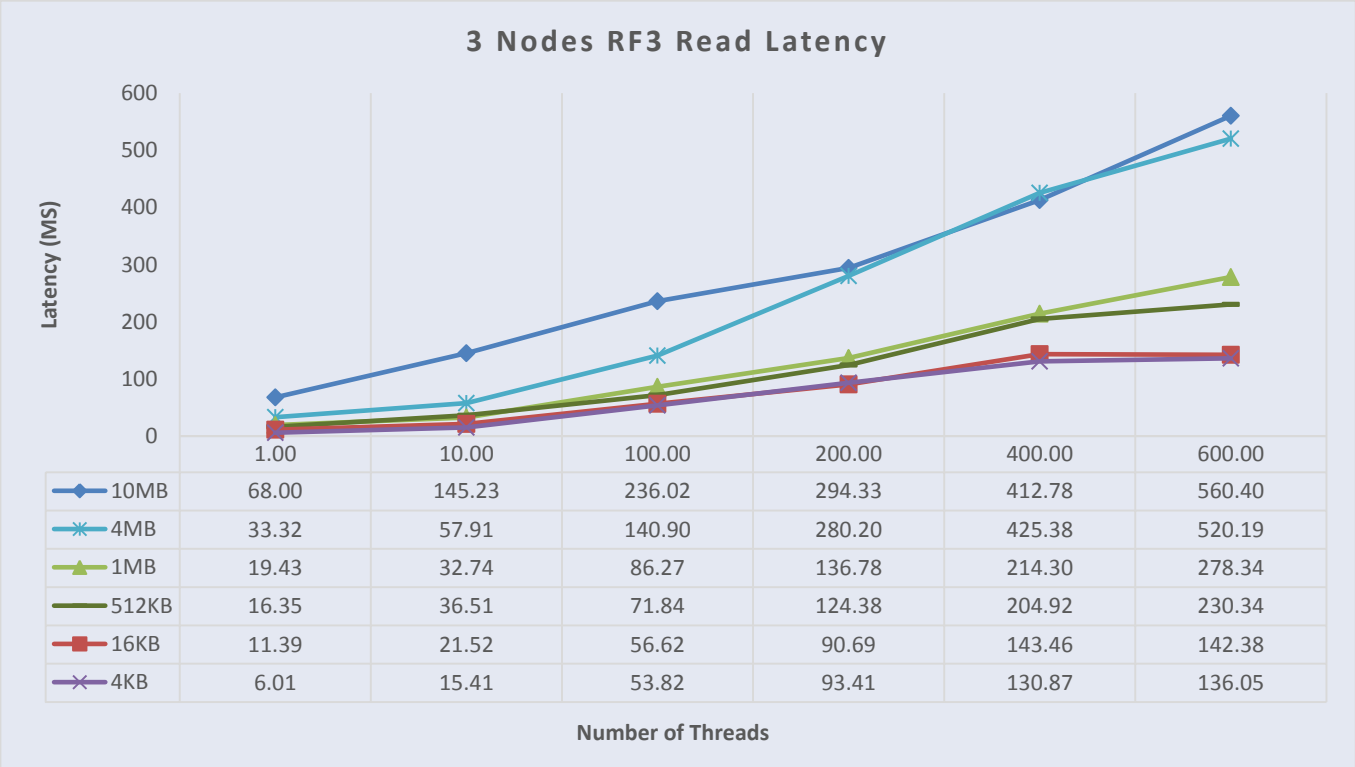


	1.00	10.00	100.00	200.00	400.00	600.00
10MB	4.48	43.28	140.33	162.66	162.83	169.19
4MB	10.52	100.91	275.36	282.51	291.84	299.89
1MB	32.09	274.16	552.73	598.63	653.51	714.01
512KB	47.90	442.58	742.33	736.93	860.08	812.19
16KB	103.37	949.22	1173.98	1114.60	1293.96	1220.53
4KB	95.58	975.11	1279.88	1232.81	1440.34	1237.95

Write throughput peaks at around 1440 Operation per second with 400 parallel threads for an oject size of 4 KB.

3-Way Replication - Read Latency

Figure 104 Read Latency

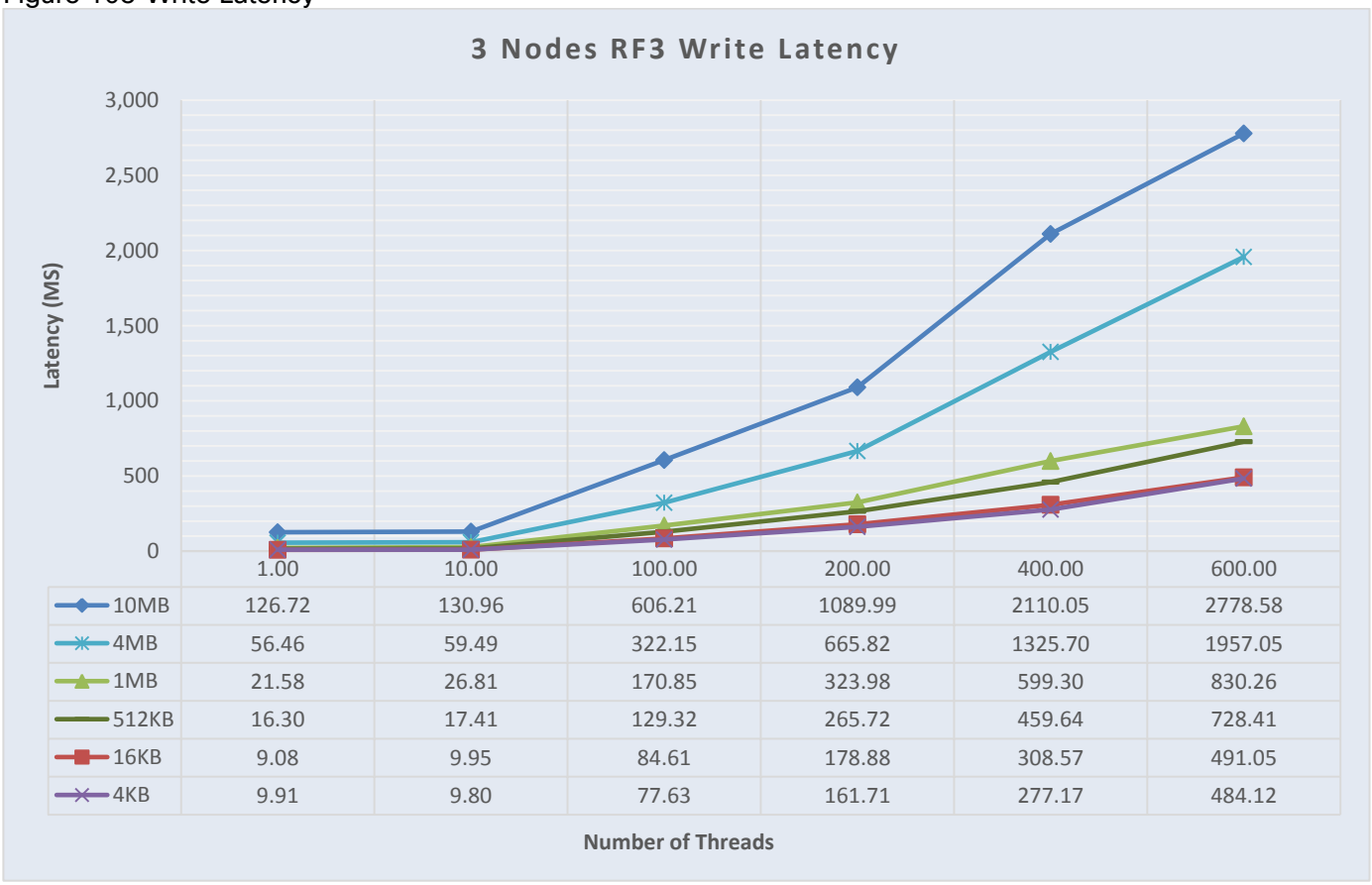


	1.00	10.00	100.00	200.00	400.00	600.00
10MB	68.00	145.23	236.02	294.33	412.78	560.40
4MB	33.32	57.91	140.90	280.20	425.38	520.19
1MB	19.43	32.74	86.27	136.78	214.30	278.34
512KB	16.35	36.51	71.84	124.38	204.92	230.34
16KB	11.39	21.52	56.62	90.69	143.46	142.38
4KB	6.01	15.41	53.82	93.41	130.87	136.05

Latency of 6.01 Milliseconds was observed with read operation for an object size of 4 KB with one thread. Latency of 560 Milliseconds was observed with read operation for an object size of 10 MB with 600 parallel threads.

3-Way Replication – Write Latency

Figure 105 Write Latency



	1.00	10.00	100.00	200.00	400.00	600.00
10MB	126.72	130.96	606.21	1089.99	2110.05	2778.58
4MB	56.46	59.49	322.15	665.82	1325.70	1957.05
1MB	21.58	26.81	170.85	323.98	599.30	830.26
512KB	16.30	17.41	129.32	265.72	459.64	728.41
16KB	9.08	9.95	84.61	178.88	308.57	491.05
4KB	9.91	9.80	77.63	161.71	277.17	484.12

Latency of 9 Milliseconds was observed with write operation for an object size of 16 KB with one thread. Latency of 2.28 Seconds was observed for object size of 10 MB with 600 parallel threads.

High Availability Tests

The high availability of this solution was validated by failing out one of the components of the infrastructure.

The purpose of the high availability tests is to ensure Business Continuity when the underlying hardware components fail and study the behavior of the system during fault injections. The following points were considered while doing the high availability tests:

- As part of the high availability testing, a random read and write load test with objects of 10MB in size was run during the failure injections. The outputs like bandwidth and operations was collected before and after the failure events.
- Only one fault is injected at any point of time. No double failures are considered.
- Performance degradation is acceptable but there should not be any business interruption. The underlying infrastructure components should continue to operate with the remaining components.

A few of the high availability tests conducted were:

- Fabric Interconnect Failures
- Nexus 9000 Failures
- S3 Service failure
- Disk Failures

Fabric Interconnect Failures

To check the business continuity of the system during fabric interconnect failures, one of the fabric interconnects was rebooted after ramping up load through COSBench. The sequence of events for fault injection and checking the health of the cluster is provided below:

1. Log into one of the Fabric Interconnects.
2. Check the cluster status.
3. Start COSBench traffic.
4. Reboot fabric interconnect.
5. Check the cluster health and the performance:

```
Cloudian-FI-6332-B# show cluster extended-state
Cluster Id: 0x54ed1b34952f11e9-0x9132b08bcfa3f04d

Start time: Thu Jul 18 21:08:02 2019
Last election time: Thu Jul 18 21:20:45 2019

B: UP, PRIMARY
A: UP, SUBORDINATE

B: memb state UP, lead state PRIMARY, mgmt services state: UP
A: memb state UP, lead state SUBORDINATE, mgmt services state: UP
   heartbeat state PRIMARY_OK
```

```
INTERNAL NETWORK INTERFACES:
eth1, UP
eth2, UP

HA READY
Detailed state of the device selected for HA storage:
Chassis 1, serial: FOX2235P4CV, state: active
Chassis 2, serial: FOX2233P26C, state: active
Chassis 3, serial: FOX2235P4DJ, state: active
Cloudian-FI-6332-B#
```

S3 COSBench test started for 10MB object size with 600 threads and with mix of read and write.

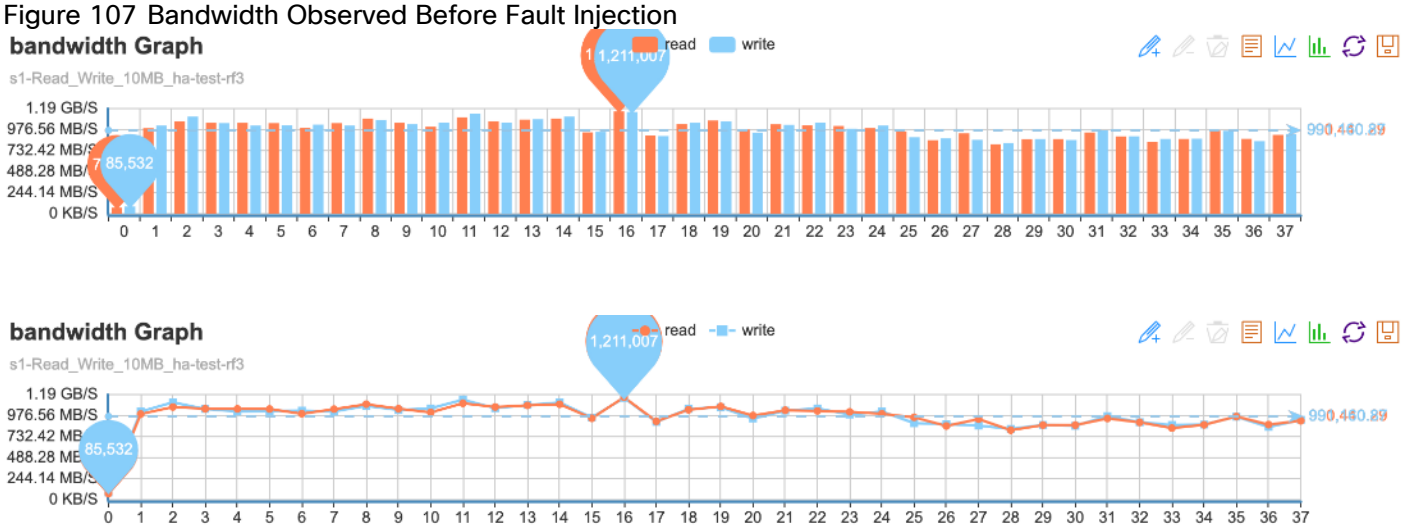
The following data was gathered after ramping up the load before fault injection:

Figure 106 COSBench Statistics Before Fault Injection

Op-Type	Op-Count	Byte-Count	Avg-ResTime	Avg-ProcTime	Throughput	Bandwidth	Succ-Ratio
op1:read	2.26 kops	22.6 GB	4742.53 ms	4727.33 ms	116.26 op/s	1.16 GB/S	100%
op2:write	2.27 kops	22.67 GB	732.86 ms	613.06 ms	114.14 op/s	1.14 GB/S	100%

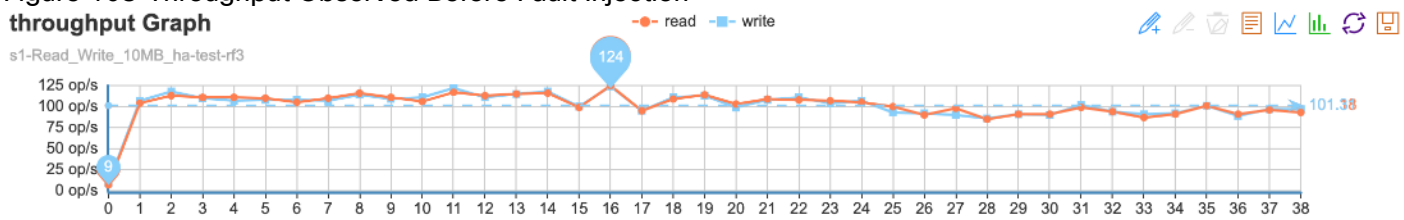
The bandwidth observed prior to fault injection is shown below:

Figure 107 Bandwidth Observed Before Fault Injection



The visualization of the number of read/write mix operations pers second:

Figure 108 Throughput Observed Before Fault Injection



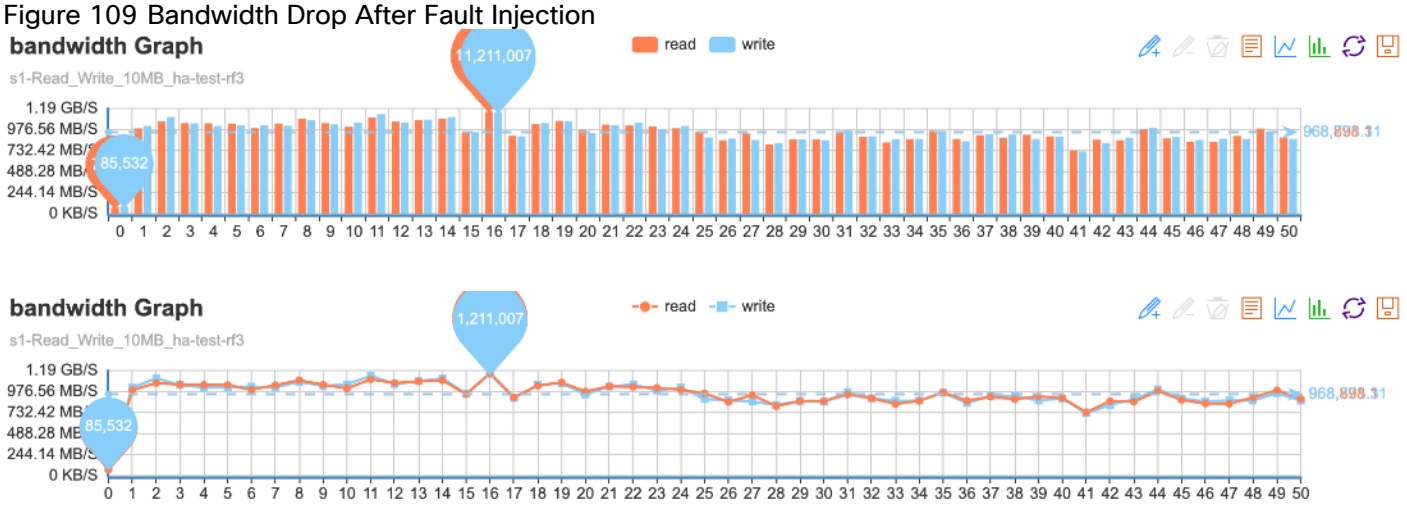
Now reboot the fabric interconnect that carries the cluster traffic:

```
Cloudian-FI-6332-B# connect local-mgmt
Cisco Nexus Operating System (NX-OS) Software
TAC support: http://www.cisco.com/tac
Copyright (c) 2009, Cisco Systems, Inc. All rights reserved.
The copyrights to certain works contained in this software are
owned by other third parties and used and distributed under
license. Certain components of this software are licensed under
the GNU General Public License (GPL) version 2.0 or the GNU
Lesser General Public License (LGPL) Version 2.1. A copy of each
such license is available at
http://www.opensource.org/licenses/gpl-2.0.php and
http://www.opensource.org/licenses/lgpl-2.1.php

Cloudian-FI-6332-B(local-mgmt)# reboot
Before rebooting, please take a configuration backup.
Do you still want to reboot? (yes/no):yes
```

The FI was rebooted between 40 to 42. The lowpoint in the graphs show the activity when the FI was rebooted.

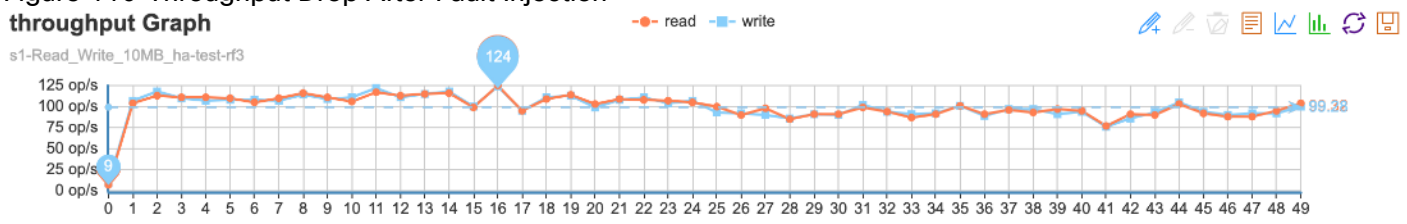
When one of the FI's is down, COSBench continues to send the requests. However, the bandwidth reduces to 734 MB/sec for read and write operations.

Figure 109 Bandwidth Drop After Fault Injection



The throguhput drop can be seen below between points 40 and 42:

Figure 110 Throughput Drop After Fault Injection

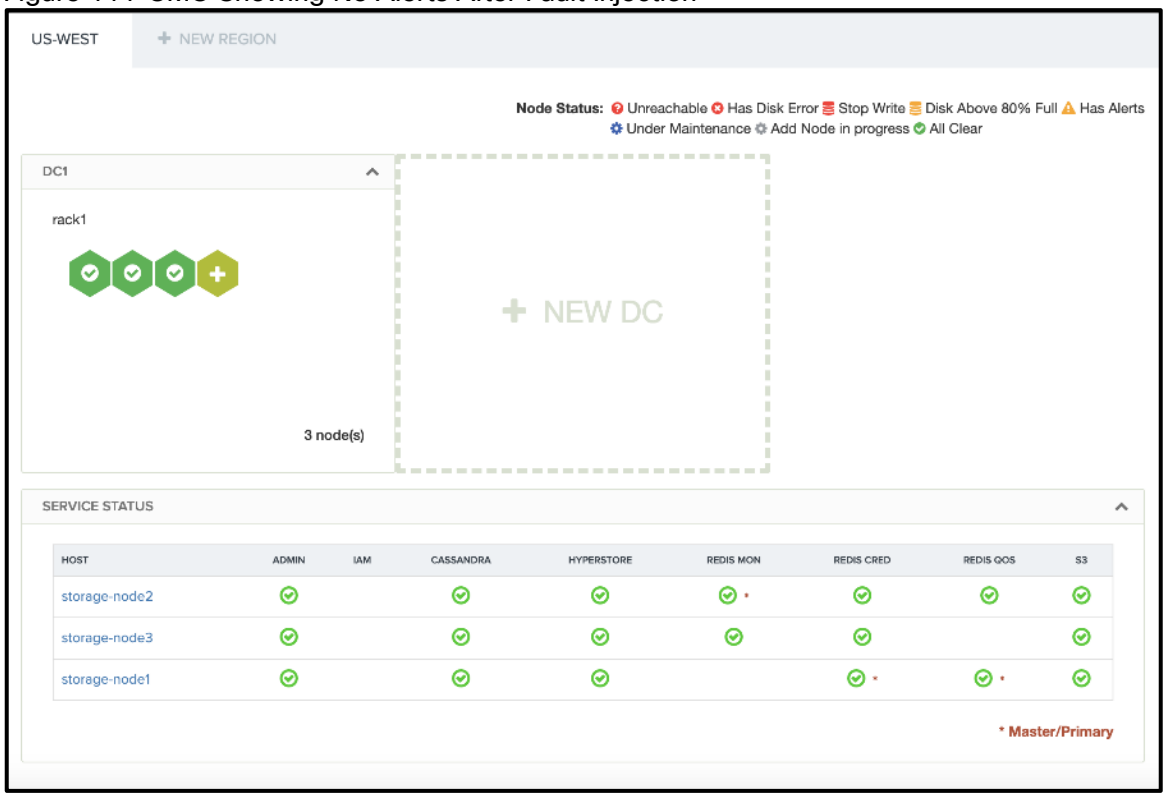


The output confirms that the FI is down and the cluster is running on single FI in a degraded mode:

```
Cloudian-FI-6332-A# show cluster extended-state
Cluster Id: 0x54ed1b34952f11e9-0x9132b08bcfa3f04d

Start time: Thu Jul 18 21:18:57 2019
Last election time: Thu Jul 18 21:55:48 2019

A: UP, PRIMARY
B: UNRESPONSIVE, SUBORDINATE

A: memb state UP, lead state PRIMARY, mgmt services state: UP
B: memb state UNRESPONSIVE, lead state SUBORDINATE, mgmt services state: INVALID
   heartbeat state SECONDARY_REQUESTED

INTERNAL NETWORK INTERFACES:
eth1, DOWN
eth2, DOWN

HA NOT READY
Peer Fabric Interconnect is down
Detailed state of the device selected for HA storage:
Chassis 1, serial: FOX2235P4CV, state: active
Chassis 2, serial: FOX2233P26C, state: active
Chassis 3, serial: FOX2235P4DJ, state: active
Cloudian-FI-6332-A#
```

The FI failure does not impact the HyperStore software. The CMC dashboard does not show any faults regarding the FI failure.

Figure 111 CMC Showing No Alerts After Fault Injection



The system recovers after the fabric joins the cluster and when HA READY.

```
Cloudian-FI-6332-A# show cluster extended-state
Cluster Id: 0x54ed1b34952f11e9-0x9132b08bcfa3f04d

Start time: Thu Jul 18 21:18:57 2019
Last election time: Thu Jul 18 22:43:13 2019

A: UP, PRIMARY
B: UP, SUBORDINATE

A: memb state UP, lead state PRIMARY, mgmt services state: UP
B: memb state UP, lead state SUBORDINATE, mgmt services state: UP
   heartbeat state PRIMARY_OK

INTERNAL NETWORK INTERFACES:
eth1, UP
eth2, UP

HA READY
Detailed state of the device selected for HA storage:
Chassis 1, serial: FOX2235P4CV, state: active
Chassis 2, serial: FOX2233P26C, state: active
Chassis 3, serial: FOX2235P4DJ, state: active
Cloudian-FI-6332-A#
```

Nexus 9000 Switch Failures

Like FI failures, one of the upstream Nexus switches was reloaded to make sure that there is business continuity. Since both the FI's are connected to the switches and with VPC, the requests from the Nexus are still forwarded to the FI's.

The S3 COSBench test started for 10MB object size with 600 threads and with mix of read and write.

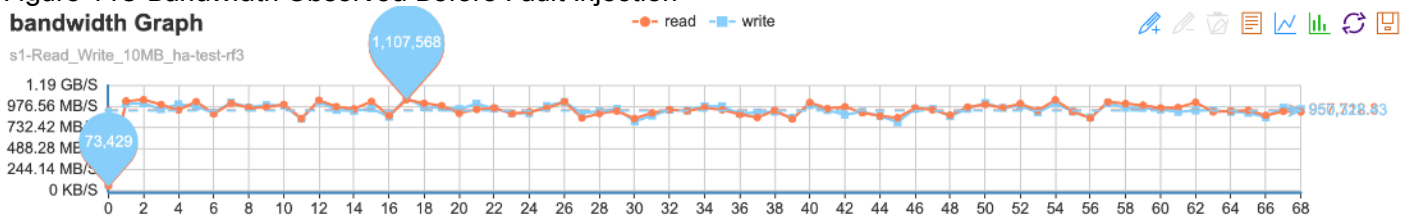
The following data was gathered after ramping up the load before fault injection:

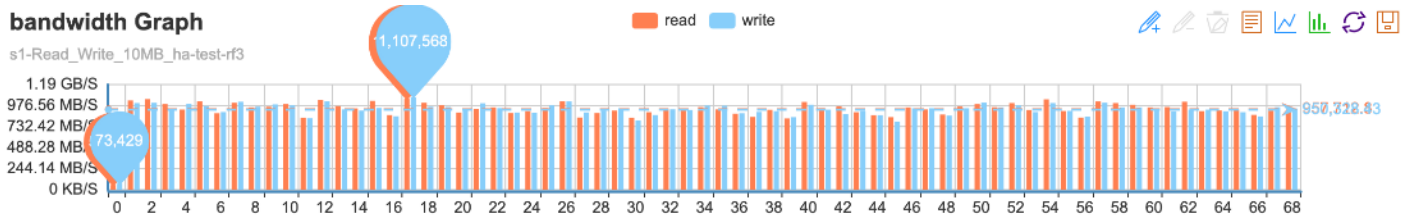
Figure 112 COSBench Statistics Before Fault Injection

Op-Type	Op-Count	Byte-Count	Avg-ResTime	Avg-ProcTime	Throughput	Bandwidth	Succ-Ratio
op1:read	2.47 kops	24.68 GB	4826.2 ms	4811.92 ms	110.73 op/s	1.11 GB/S	100%
op2:write	2.36 kops	23.62 GB	670.2 ms	557.36 ms	106.36 op/s	1.06 GB/S	100%

The bandwidth observed prior to the fault injection is shown below:

Figure 113 Bandwidth Observed Before Fault Injection





The visualization of the number of read/write mix operations pers second is shown below:

Figure 114 Throughput Observed Before Fault Injection

The switch was reloaded to check the impact on the application.

The N9K switch was reloaded.

```
N9K-Cloudian-Fab-B(config)# show version |grep uptime
Kernel uptime is 23 day(s), 19 hour(s), 10 minute(s), 57 second(s)
N9K-Cloudian-Fab-B(config)#

N9K-Cloudian-Fab-B(config)# reload
This command will reboot the system. (y/n)?  [n] y
```

Cosbench performance graphs:

Figure 115 Bandwidth Drop After Fault Injection

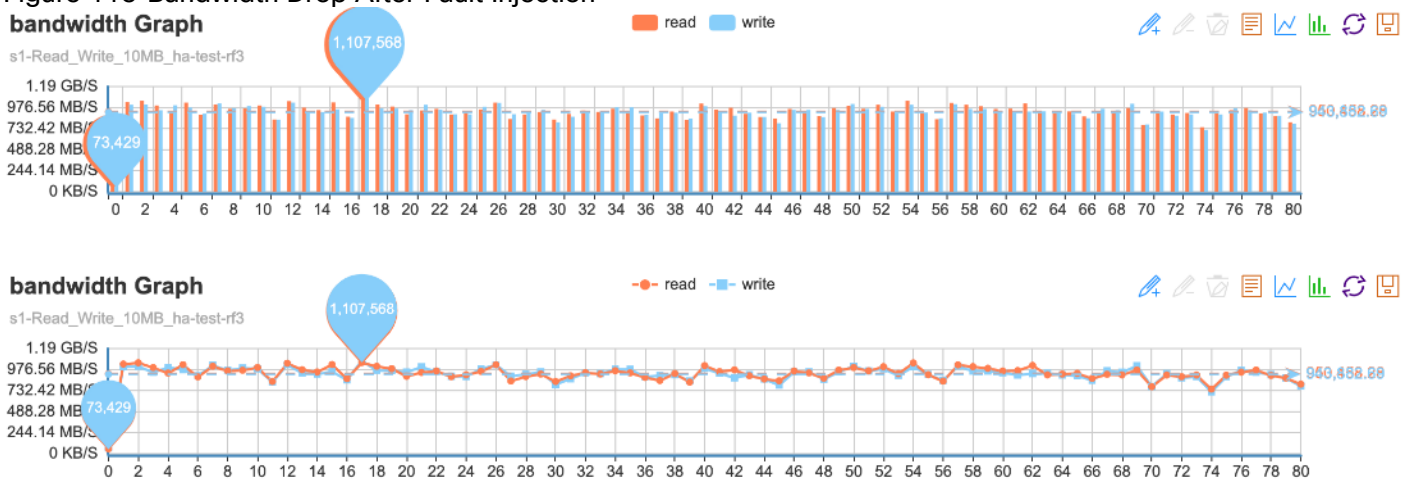


Figure 116 Throughput Drop After Fault Injection

The system was performing a read/write mix of around 734 MB/s when the Nexus switch was reloaded [Between 72 and 76].

The system continues to operate without any interruption. The Nexus switch eventually came back up.

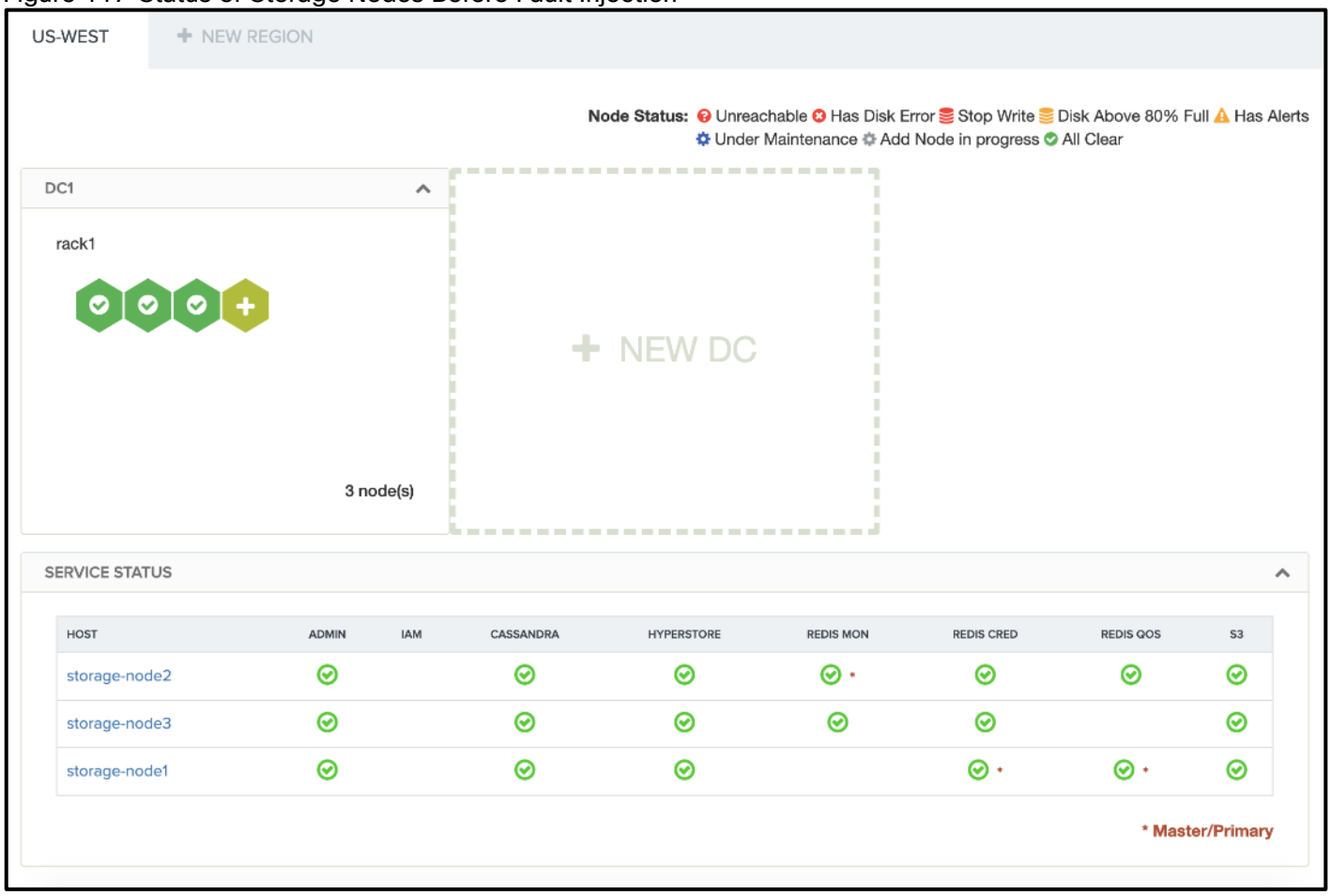
```
N9K-Cloudian-Fab-B# show version |grep uptime
Kernel uptime is 0 day(s), 0 hour(s), 14 minute(s), 29 second(s)
N9K-Cloudian-Fab-B#
```

S3 Service Failures

The S3 COSBench test started for a 10MB object size with 600 threads and with mix of read and write.

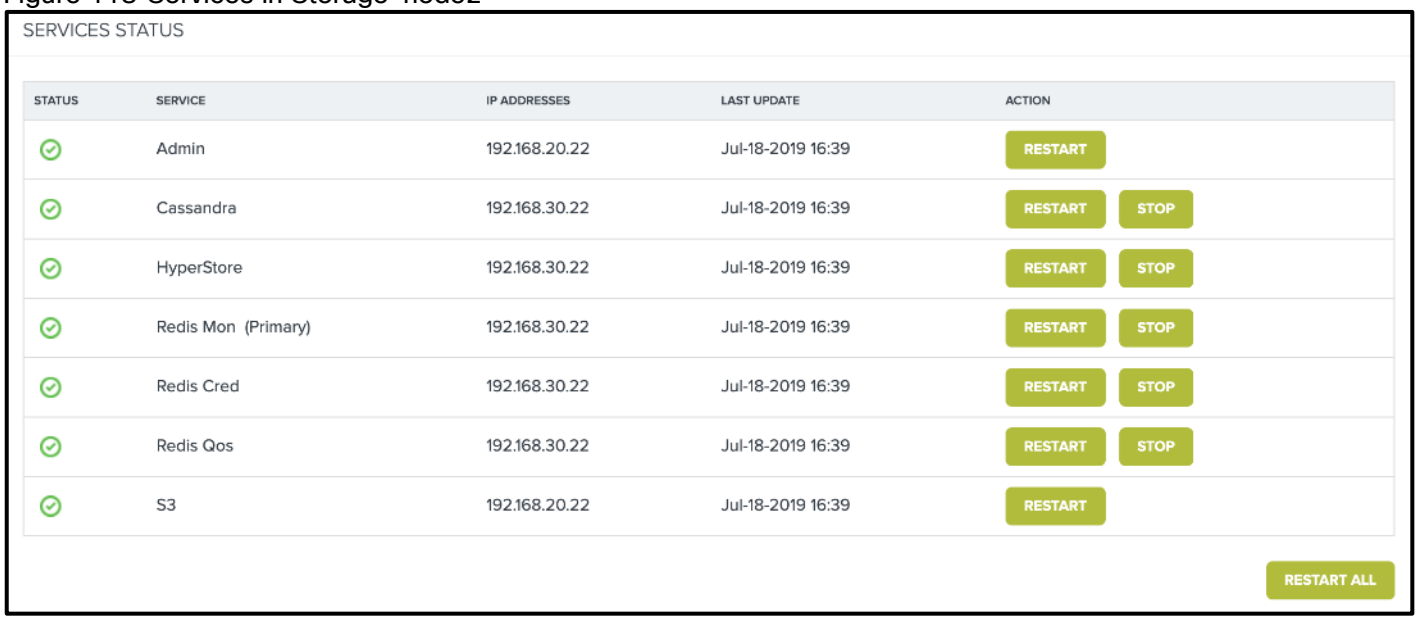
The storage node 2 was rebooted. The status of the storage nodes before fault injection is shown below:

Figure 117 Status of Storage Nodes Before Fault Injection



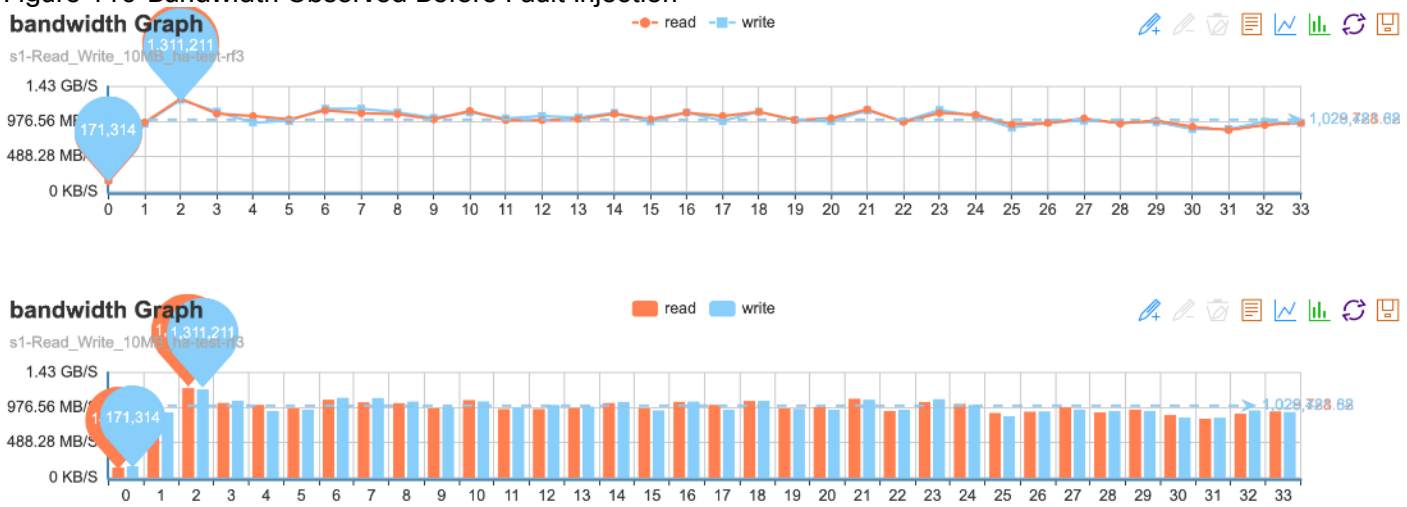
All the processes and services were running in node-2.

Figure 118 Services in Storage-node2



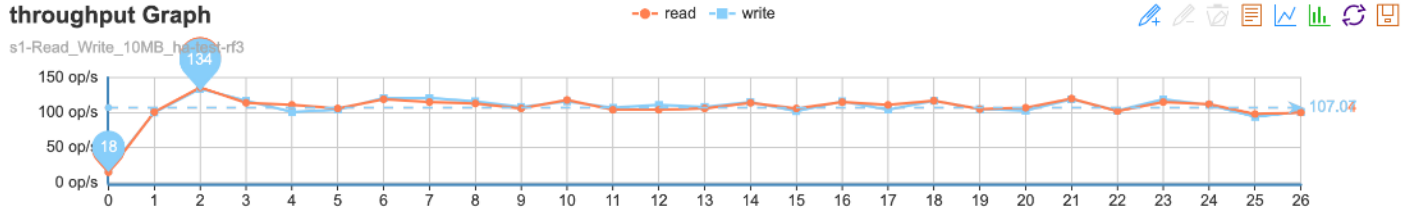
The bandwidth graph before the reboot:

Figure 119 Bandwidth Observed Before Fault Injection



The visualization of number of read/write mix operations pers second is shown below:

Figure 120 Throughput Observed Before Fault Injection



The success ratio was 100% for both read and write operations.

Figure 121 COSBench Statistics Before Fault Injection

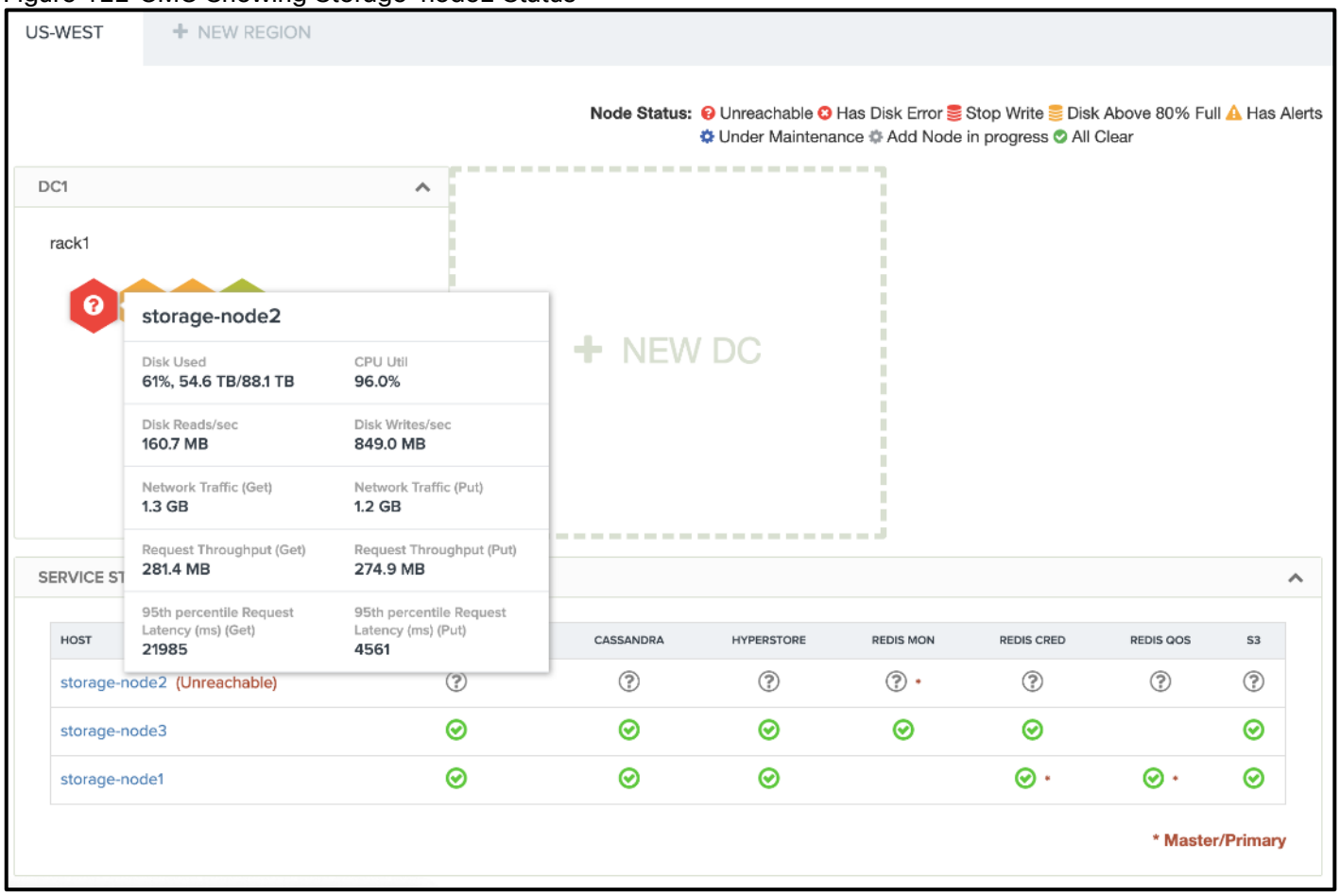
Op-Type	Op-Count	Byte-Count	Avg-ResTime	Avg-ProcTime	Throughput	Bandwidth	Succ-Ratio
op1:read	2.15 kops	21.51 GB	4143.74 ms	4127.65 ms	116.7 op/s	1.17 GB/S	100%
op2:write	2.19 kops	21.89 GB	1120.23 ms	995.6 ms	116.85 op/s	1.17 GB/S	100%

Server uptime report:

```
[root@storage-node2 ~]# uptime
 16:45:37 up 16 days,  1:20,  1 user,  load average: 5.59, 29.58, 127.46
[root@storage-node2 ~]#
```

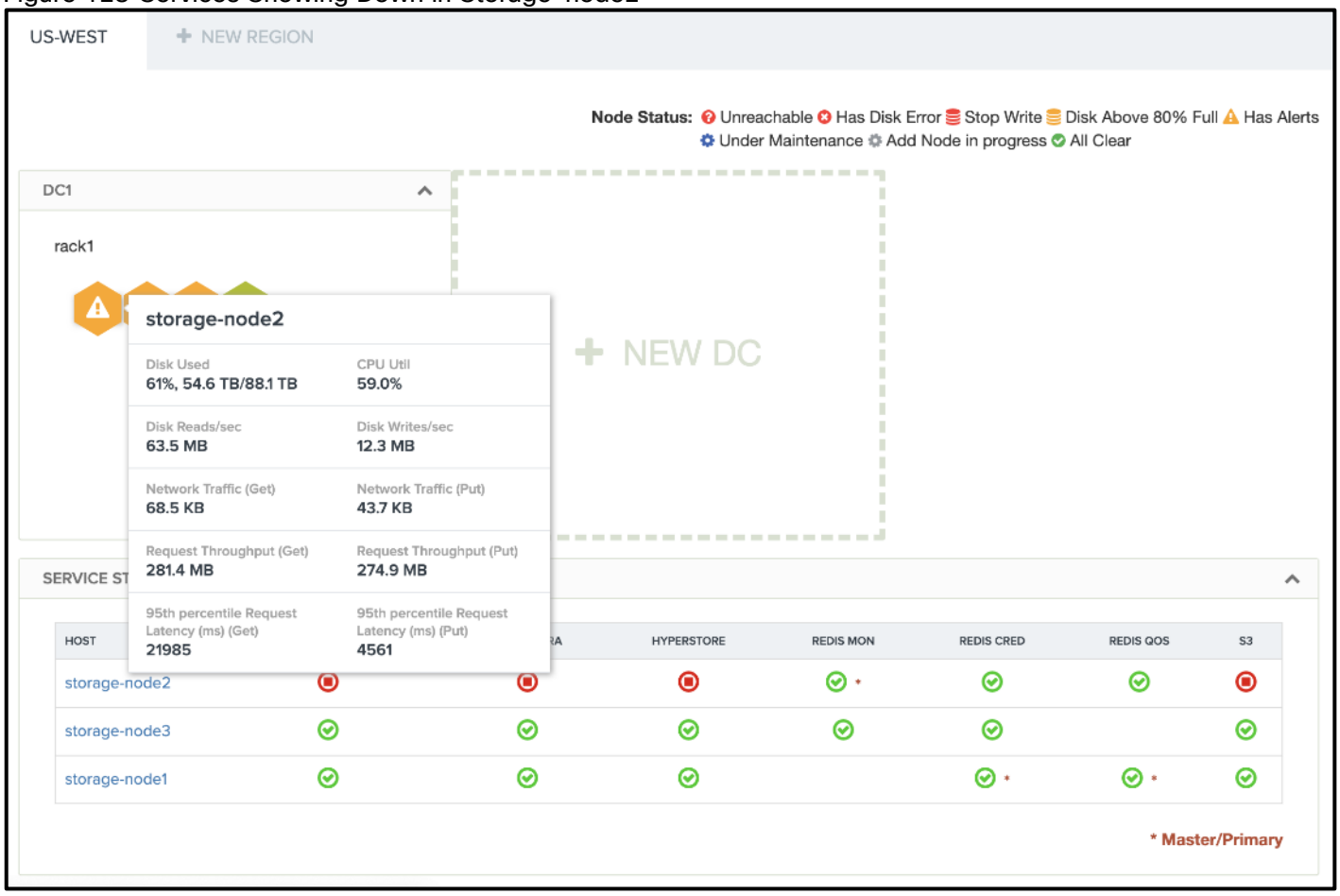
The server was brought down. The node reported an error in CMC.

Figure 122 CMC Showing Storage-node2 Status



The services in storage-node2 were down.

Figure 123 Services Showing Down in Storage-node2



Alerts were seen for the storage node as shown below:

Figure 124 Alerts in Storage-node2

High	Cron Mon	2019-07-18 12:06:06 ERROR [7756] [CronDown] Cron service on cron host down. Failing over after 9 minutes	Jul-18-2019 17:17	1
High	Cron Mon	2019-07-18 12:06:06 ERROR [7756] [SSHFail] Cannot SSH to cron job host: storage-node3	Jul-18-2019 17:17	1
High	HyperStore	[Service Down or Unreachable]	Jul-18-2019 17:17	2
High	Admin	[Service Down or Unreachable]	Jul-18-2019 17:17	6
High	S3	[Service Down or Unreachable]	Jul-18-2019 17:17	6
Critical	Node Unreachable	Node is unreachable from monitoring host (data collector)	Jul-18-2019 17:15	4

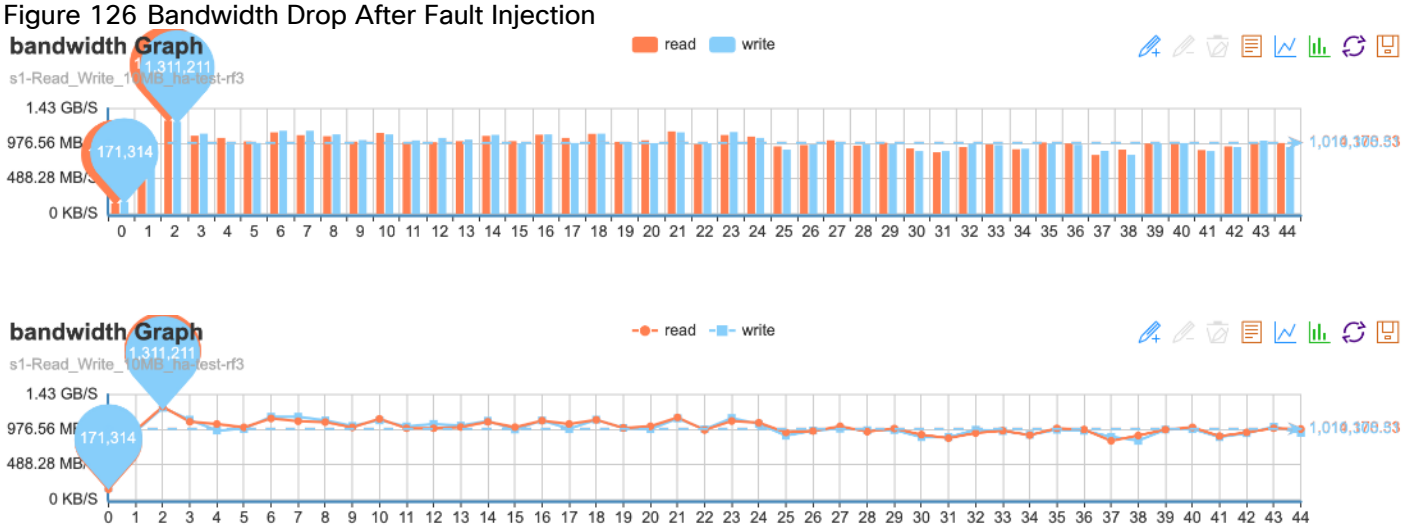
There were some write miss due to node failure as shown below:

Figure 125 COSBench Showing Write Miss

Op-Type	Op-Count	Byte-Count	Avg-ResTime	Avg-ProcTime	Throughput	Bandwidth	Succ-Ratio
op1:read	1.66 kops	16.61 GB	4844.85 ms	4832.74 ms	79.74 op/s	797.4 MB/S	100%
op2:write	1.68 kops	16.79 GB	1821.18 ms	1710.32 ms	81.7 op/s	817.02 MB/S	99.88%

The read bandwidth was temporarily down to 797 MB/s and write to 817 MB/s [Between 36 and 39]

Figure 126 Bandwidth Drop After Fault Injection



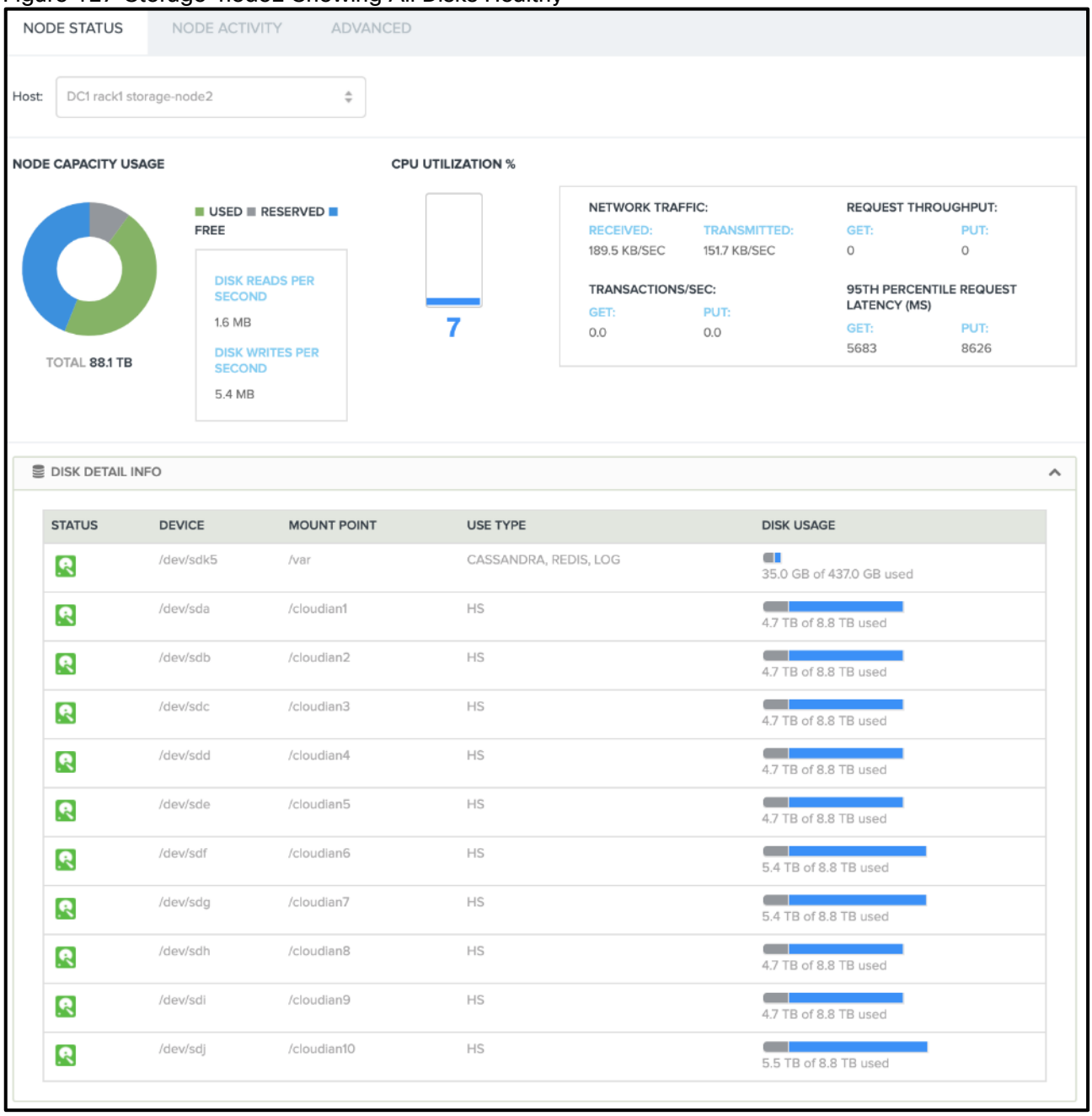
Everything was performing normally when the storage node came back up . [After 42 server was up]

Disk Failure Tests

The disk was removed to understand the impact on HyperStore.

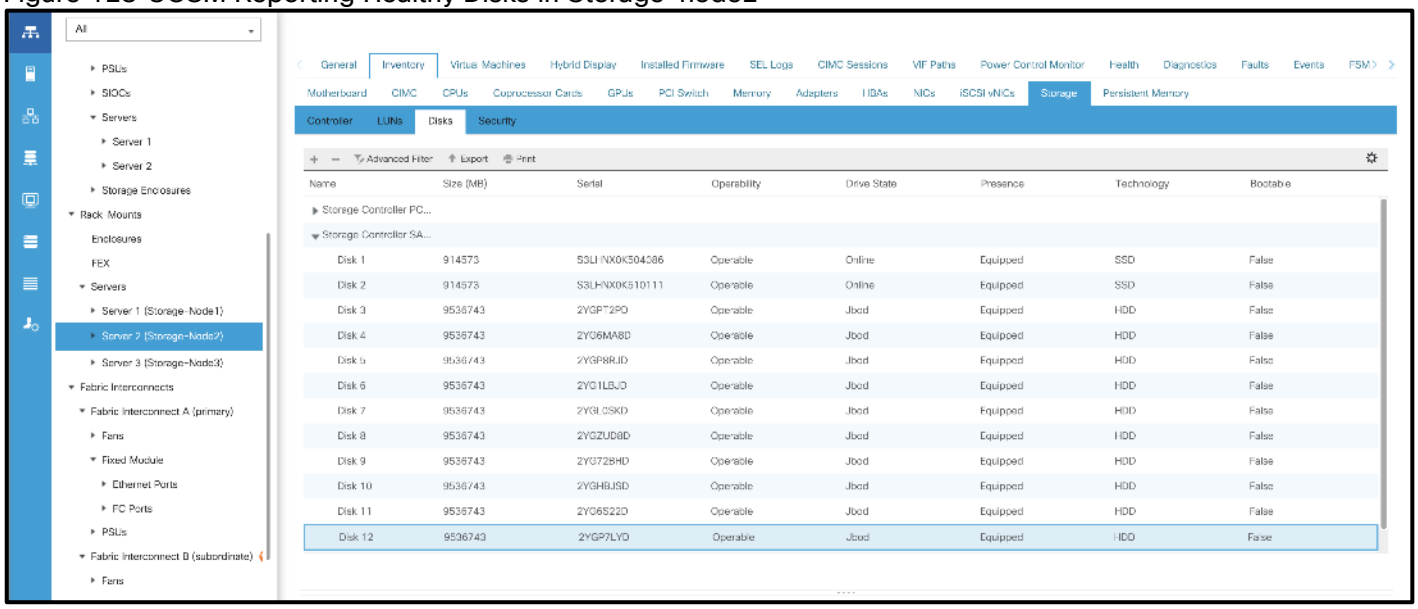
Figure 127 shows the healthy disk-7 [/dev/sde] on storage node 2 as reported CMC.

Figure 127 Storage-node2 Showing All Disks Healthy



UCSM also showed all the disks are healthy.

Figure 128 UCSM Reporting Healthy Disks in Storage-node2



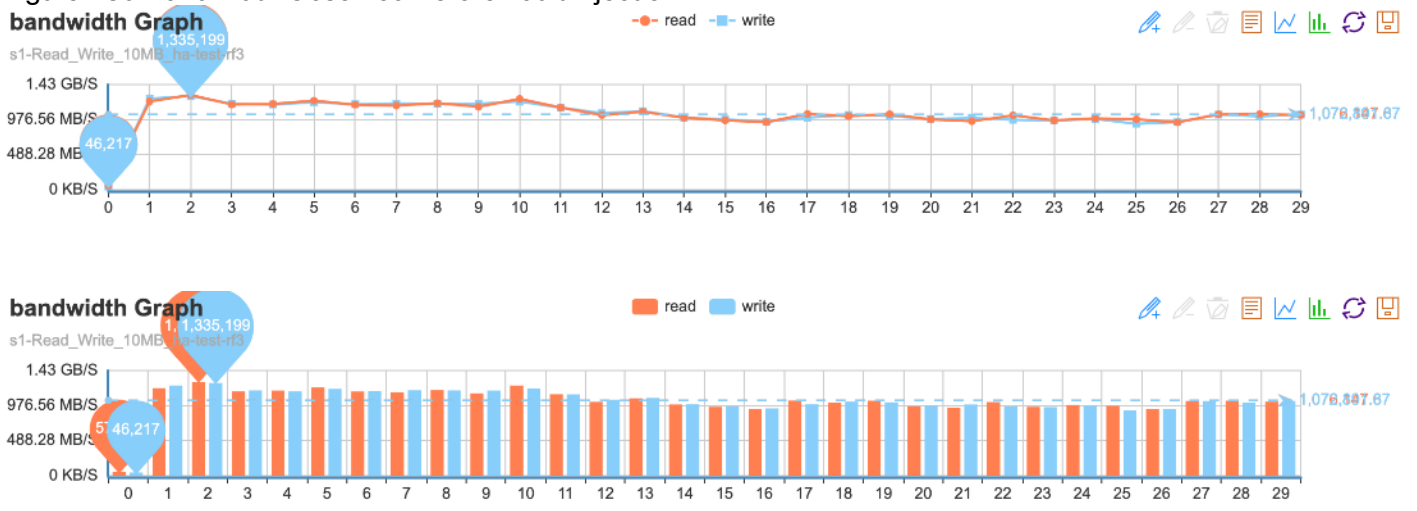
The success ratio was 100% for both read and write.

Figure 129 COSBench Statistics Before Fault Injection

Op-Type	Op-Count	Byte-Count	Avg-ResTime	Avg-ProcTime	Throughput	Bandwidth	Succ-Ratio
op1:read	2.65 kops	26.47 GB	3677.49 ms	3651.51 ms	125.35 op/s	1.25 GB/S	100%
op2:write	2.63 kops	26.26 GB	1027.82 ms	862.35 ms	124.33 op/s	1.24 GB/S	100%

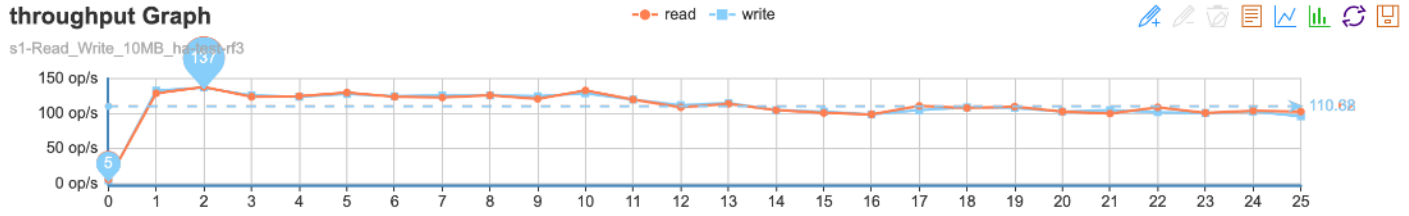
The bandwidth graph before the reboot is shown below:

Figure 130 Bandwidth Observed Before Fault Injection



The visualization of the number of read/write mix operations pers second is shown below:

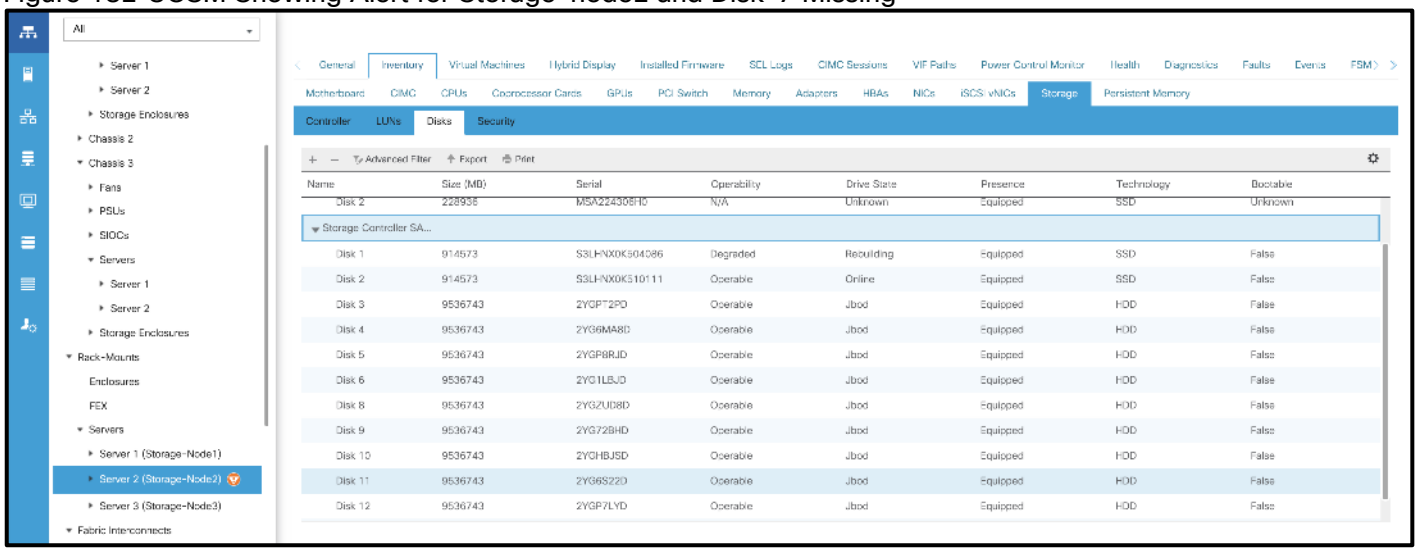
Figure 131 Throughput Observed Before Fault Injection



Disk-7 was removed.

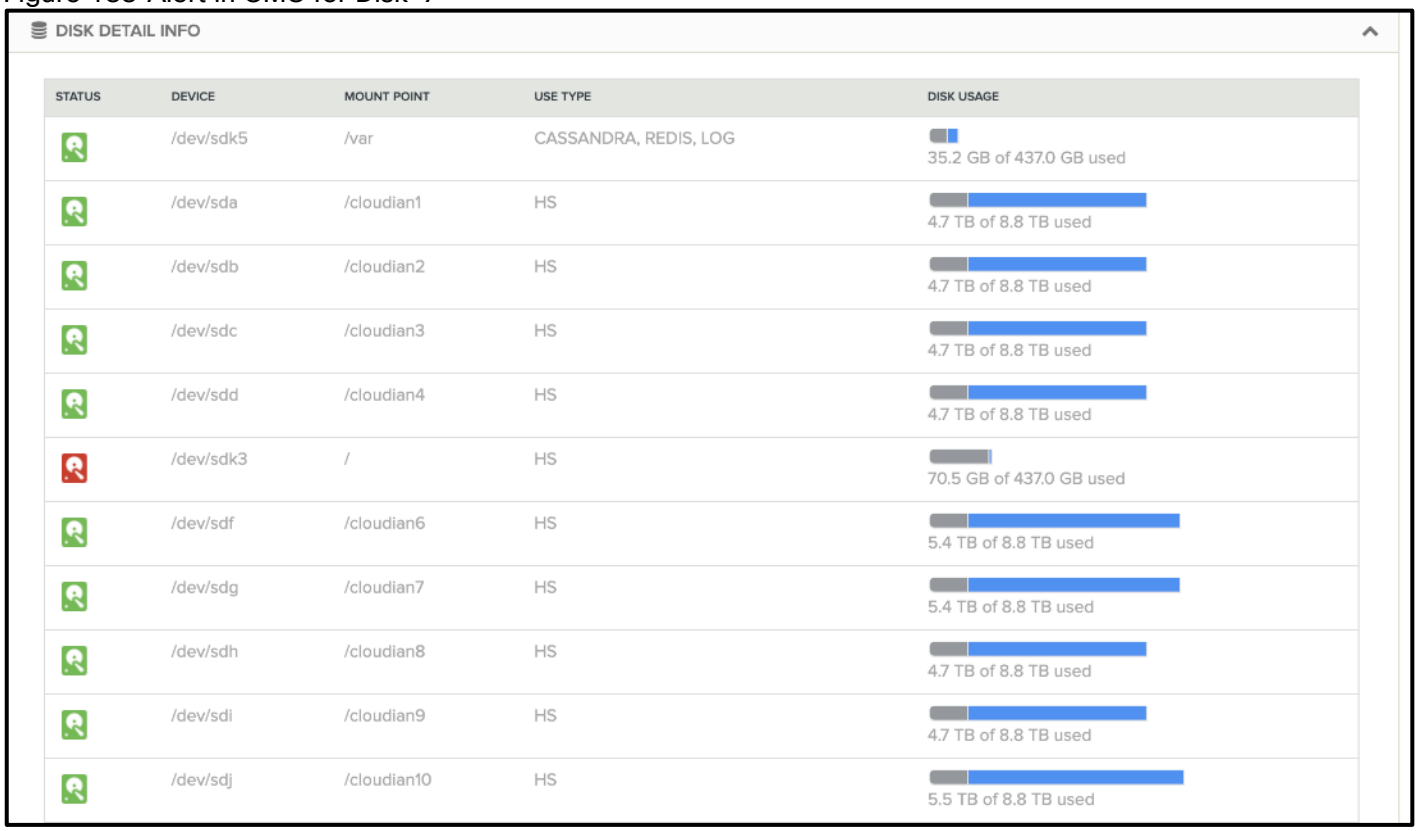
After the disk was removed, UCSM didn't show Disk 7 and reports an alert.

Figure 132 UCSM Showing Alert for Storage-node2 and Disk-7 Missing



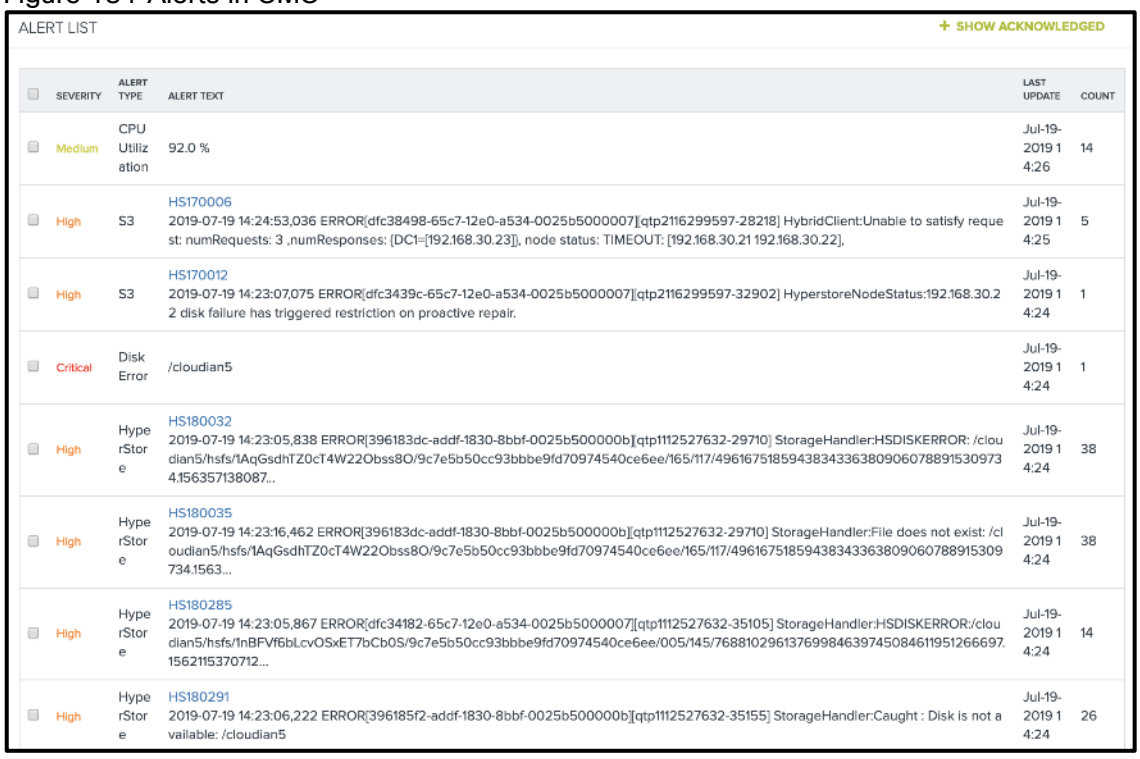
The disk is reported as unavailable in CMC.

Figure 133 Alert in CMC for Disk-7



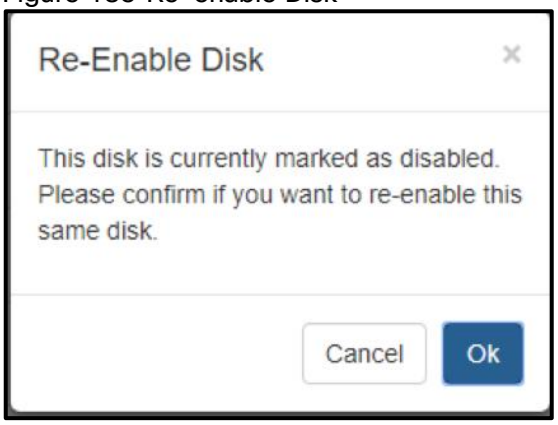
CMC reported an alert as shown below:

Figure 134 Alerts in CMC



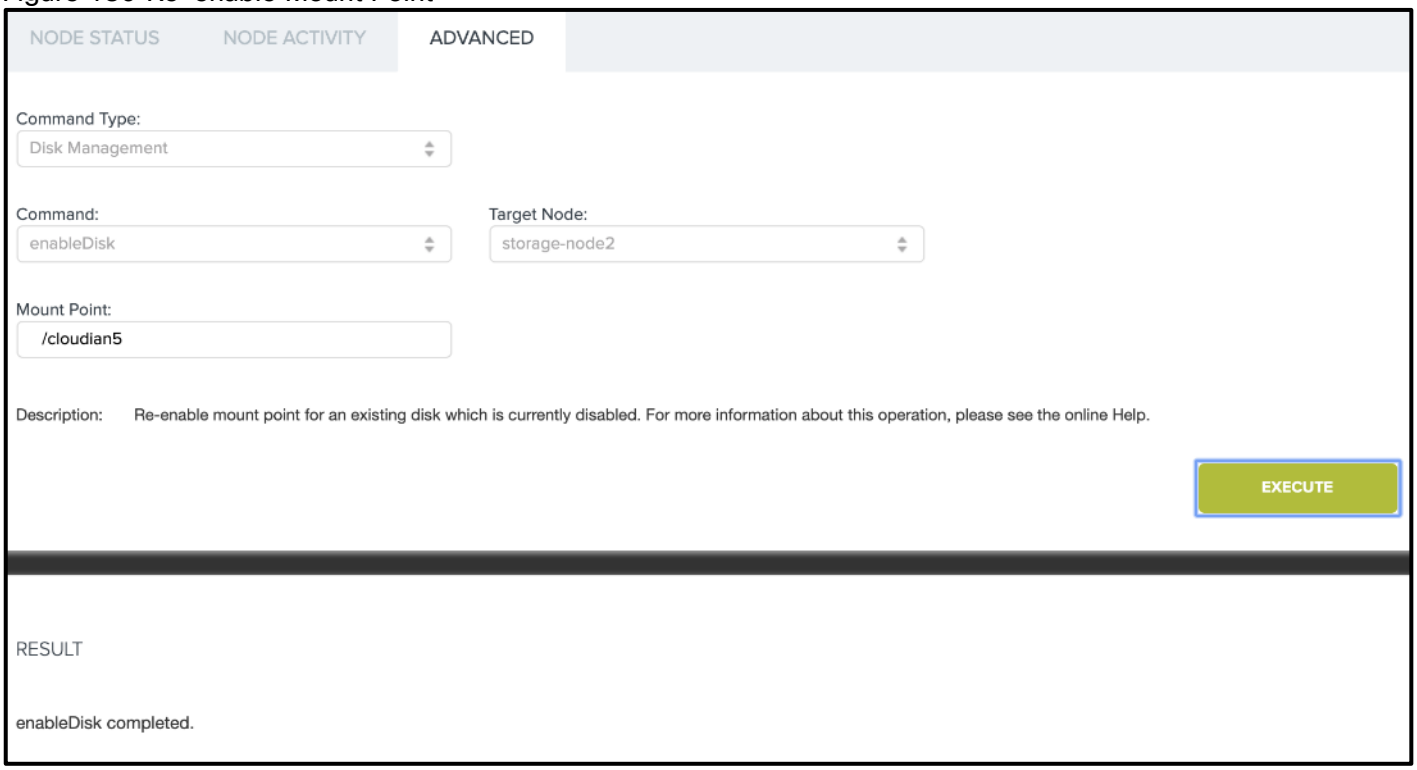
After reinserting the disk, the disk was re-enabled in CMC:

Figure 135 Re-enable Disk



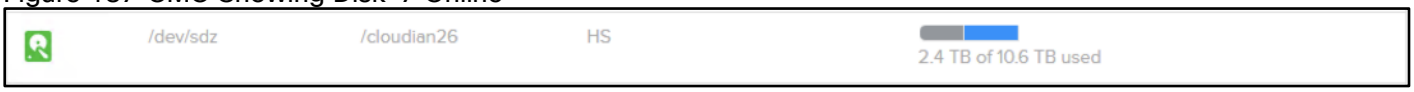
The mount points were re-enabled.

Figure 136 Re-enable Mount Point



The disk was back online.

Figure 137 CMC Showing Disk-7 Online



There was a small drop in performance [At point 32] After that it was back to normal.

Bandwidth:

Figure 138 Bandwidth Drop After Fault Injection

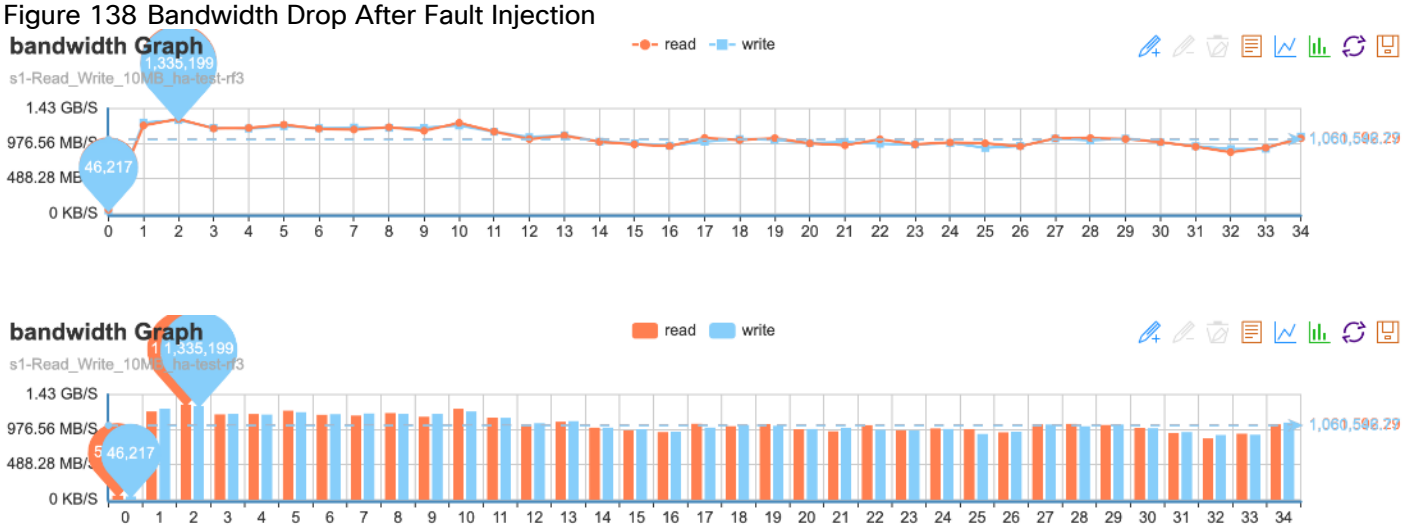
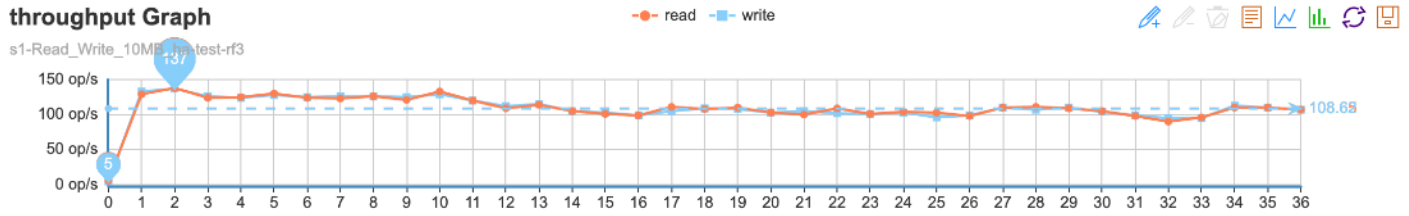


Figure 139 Throughput Drop After Fault Injection



Frequently Asked Questions

The following are the Frequently Asked Questions:

- What Cisco UCS Manager version is supported with Cloudian HyperStore on Cisco UCS?

  Cloudian HyperStore has been validated with UCSM version 4.0(4b) and with 40Gb Fabric Interconnects and Switches. It is strongly recommended to run the infrastructure on versions higher than the validation done in this CVD.

- How many minimum nodes are recommended?

  This is as per the SDS requirements. The validation was done with 6 Chassis 12 nodes. This can be scaled up.

- Can Cloudian HyperStore work without a load balancer?

  Yes. Cloudian can run without a load balancer but high availability will be compromised unless using a virtual IP manager like CTDB.

- Can Cloudian HyperStore support multiple storage policies?

  Yes. Cloudian supports 20 storage policies by default within same environment but this number can be increased, it can be a combination of erasure code and replication.

- Does Cloudian support both eventual and strong consistency?

  Yes. Cloudian support both eventual, strong and dynamic consistency to be able accommodate every requirement.

- How many DC's can Cloudian HyperStore support?

  Cloudian HyperStore supports up to 200 DC's within 20 regions.

- Does Cloudian HyperStore support a single global namespace?

  Yes. Cloudian supports one ore more single global namespace, and supports one or more individual name spaces as well.

- Does Cloudian HyperStore support compression?

  Yes. Cloudian supports snappy, lib and lz4 compression.

- Does Cloudian HyperStore support deduplication?

  No. Cloudian HyperStore does not support deduplication.

- Can Cloudian HyperStore tier to another object store?

  Yes. Cloudian HyperStore can tier to any other s3 compatible objectstore such as AWS, s3, glacier, Google GCP and Azure.

- Does Cloudian HyperStore support heterogeneous nodes?

  Yes. Cloudian HyperStore supports nodes from different vendors with different hardware components, performance characteristics and capacities.

- Can I use Cloudian HyperStore as a backup target?

Yes. when using any of the major backup vendors that has an s3 connector, Cloudian HyperStore will be an excellent choice to store your backups.

- Can I use Cloudian HyperStore as a media target for MAM?

  Yes. Any MAM software that support s3 as a repository target will be a good use case for Cloudian.

Troubleshooting

Troubleshooting issues and remedies are detailed below:

- Where to find the HyperStore the log files?

  The log files can be found under /var/log/cloudian. The CMC also provides a log page that can be reviewed before acknowledging the errors

- How can generate a log bundle to send to support?

  To create log bundles to send to support for further analyses please run:

  `/opt/cloudian/tools/smartsup_systeminfo.sh on each node.`

- Read requests are failing specifically with larger objects

  Please ensure NTP is properly configured and time is in sync on all nodes and clients.

- A service has failed on my cluster

  The failed service can be restarted from the CMC under that node or from the installer menu under the maintenance option.

- S3 service does not successfully restart

  When the s3 fails after restarting the service, stop the S3 service first and then start the s3 service again.

Appendix

Appendix A – Kickstart File for High Availability Proxy Node for Cisco UCS C220 M5

```
#version=DEVEL

#from the linux installation menu, hit tab and append this:

#biosdevname=0 net.ifnames=0 ip=eth1:dhcp

#ks=ftp://192.168.10.2/{hostname}.cfg

# System authorization information

auth --enableshadow --passalgo=sha512

# Use CDROM installation media

cdrom

# Use text install

text

# Run the Setup Agent on first boot

firstboot --disable

selinux --disable

firewall --disable

# Keyboard layouts

keyboard --vckeymap=us --xlayouts='us'

# System language

lang en_US.UTF-8

# Network information

network  --bootproto=static --device=eth0 --ip=173.36.220.241 --netmask=255.255.255.0 --onboot=on --
gateway=173.36.220.1 --nameserver=171.70.168.183 --ipv6=auto --activate

network  --bootproto=static --device=eth1 --ip=192.168.10.19 --netmask=255.255.255.0 --onboot=on --
ipv6=auto --activate

network  --bootproto=static --device=eth2 --ip=192.168.20.19 --netmask=255.255.255.0 --onboot=on --
ipv6=auto -activate

network  --bootproto=static --device=eth3 --ip=192.168.30.19 --netmask=255.255.255.0 --onboot=on --
ipv6=auto --activate
```

```
network  --hostname=ha-proxy

# Root password
rootpw --iscrypted $6$yfE2jHtdy.OSmO8g$InneiVXQl9Kc9m4w2cEiS8/og6BKUlu5HSR0eCYgh5dVaeCV54Q6piS7k10lalXignLCBvAZPqmw4dvYgy66V1
# System services
services --disabled=chronyd
# System timezone
timezone America/Los_Angeles --isUtc --nontp
# System bootloader configuration
bootloader --append= crashkernel=auto --location=mbr --boot-drive=sda
# Partition clearing information
clearpart --drives=sda --all --initlabel
# Disk partitioning information
part /boot --fstype=ext4 --ondisk=sda --size=8192
part swap --fstype=swap --ondisk=sda --size=32767
part /var --fstype=ext4 --ondisk=sda --grow
part / --fstype=ext4 --ondisk=sda --size=40960

reboot  --eject

%packages
@^minimal
@core
kexec-tools

%end

%addon com_redhat_kdump --enable --reserve-mb='auto'
```

```
%end

%anaconda
pwpolicy root --minlen=6 --minquality=50 --notstrict --nochanges --notempty
pwpolicy user --minlen=6 --minquality=50 --notstrict --nochanges --notempty
pwpolicy luks --minlen=6 --minquality=50 --notstrict --nochanges --notempty
%end

################
#POST SCRIPT
################
%post --log=/root/ks-post.log
################
#GPT Labels for HDDs
################
for i in a b {d..z} aa ab ac; do parted -s /dev/sd$i mklabel gpt; done;
################
#Turn off Transparent Hugepages and ensure that hyperthreading
#is turned off.
################
grubby --update-kernel=ALL --args=transparent_hugepage=never numa=off;
tuned-adm profile latency-performance;
systemctl enable ntpd;
################
#Preconfigure /etc/hosts
################
cat >> /etc/hosts <<EOF4
192.168.20.20       ha-proxy
192.168.20.21       storage-node1
192.168.20.22       storage-node2
```

```
192.168.20.23       storage-node3
EOF4
###############
#Setup ssh keys
###############
mkdir /root/.ssh;
cat > /root/.ssh/id_rsa <<EOF5
-----BEGIN RSA PRIVATE KEY-----
MIIEpAIBAAKCAQEAsYGqxWxQdGUsiUzafYLuX6MVD3mjq3r6KaL0QcNSuZ8F3Xfw
7WJWjmhuu/rurLVoA90fjZDQY6aEAdHSH+o27mH6hfkMVqyunwQ6u3MtUqqkwRK2
NtEJqJBiHZw9+bmgofyFYI5wBSWPGlig0kb8m+cBm0uRoE5SFFuAGc7usHkfIFIO
QQd9vz9h6OX8ba3c6yUAZDzWSnt2udyLOTqV4SPpQY4O2NvYgm1VpbIHvUvmP7Yu
5yl8hxn0in+RmferTq8WwyZihMV0EyN4q5HfT+gdbSY6xPMM9UHF89+IYNNxdZ4/
VuBcbBskey3UbQ332KqA7wS+Sra2DXmnfysWbwIDAQABAoIBAQCbeRFUXiyR5IP9
5Iyw9k9HYRX/OfGLLumSMnJyb1wzzP9cHcPeh/V8QihLadxHVZTHXZRXcHG19pFE
7rx2y7RVU2gUIDCkchd4nEG9EYKvF1u66GLE3I7zH5Nwj/sQkfAKMZ26rTC8sUsG
mBUUWKzE+K7Fklj6ud7WidZHxKH32ok1IEcFOsH/nK1BXR29XmQ/O/Kg2h0V/KiM
1Y9CJngpghnybcDzIvpV6LS8bEiRieHJGT5RTyDk+ad0uSv+f2YtIpvSUly7NAft
e1feAq3RWT82ZGyKTHWGTFNbfltcUjzPI/dcyS8AurYf+oQjJVAKhAI+yln7IUrL
V6xKsdYBAoGBANwNb96gJHZUeSoOP/JCnTps+MeOhT1vyrhRRZf1IaFnEmX7hXmE
RKXaQUvGcOSPumZMkKYyqRN22B2PLM7n1D0ypKshRmk1eq6tZ/W9gkYfldn0+QAx
AAVfUA8vJm9XLgkCAE4o2BHvtQ1w63CfygoF4V3OAsQv677F6ItROeiBAoGBAM6A
9quEOrPiRDiF25HnXXFUeRUXM4H77QB6WRV3AKggJjVlBXkhNt34g8Jr6/MfW4WO
SebQEwwBYH6NN7IG1Q0PeDRzrcv2voqzM7bV7I1rpc2E2BQhplcSyGr/aA6lW0OA
LI/HZldqb6OXXR8lmcP0rfxuqUJ8e6SHskG6qAbvAoGAlrw4QXMT7I3NNndDXtFn
EjbrWkzD+XuxC0FA9Aisw1aKz/BRFGptj6SRFA4B+gI6ETXay3FJwRnMaXYVQ5/S
n8pjteOtwqO/dt1GgMLmUn1NkaMavw39C9wMvijaL08apC9drvjBiqtE8Bc4Avlm
KUjeVzlStHdABkAlQgCTXlECgYEAur6BU4YWmAnsa7kRYRZ7uDsN7Ha4y7mJED+U
RAcD/wZjxzF+C5ZvybgtXyq9i3U2DMcqKaLNNrQgERGf5kyrak4tBDIAX0zZ7xAz
mgplrw7kN8EErt/nTyLbP3eNIlGE0LwgM9lbHeKw5p3BRok+lKi2Imt0gX2VSqq0
```

FyC3Rt0CgYADqOJ53sV7NEXfd/NG5D9bzS5yCKW+KNH4fzxAoAYhMBo3nAkgpa/1

rdjPH4f5bAMX6dKZCh5Sy9BFxgqbl0tdjVGZBUPK8tb0xbcnJ2F3+aLq02fCfyr+

TfYW1tZ7g7gZJ+To42h4Tv9wj8iWGe+pnR4Moh3WqM1TttuaCJf1nQ==

-----END RSA PRIVATE KEY-----

EOF5

cat > /root/.ssh/id_rsa.pub <<EOF6

ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAABAQCxgarFbFB0ZSyJTNp9gu5foxUPeaOrevopovRBw1K5nwXdd/DtYlaOaG
67+u6stWgD3R+NkNBjpoQB0dlf6jbuYfqF+QxWrK6fBDq7cy1SqqTBErY20QmokGldnD35uaCh/lViXnAFJY8YiKDSR
vyb5wGbS5GgTllUW4AZzu6weR8gWU5BB32/P2Ho5fxtrdzrJQBkPNZKe3a53ls5OpXhl+lBjg7Y29iCbVWluUe9S+Y
/ti7nKXyHGfSKf5GZ96tOrxbDJmKExXQTl3irkd9P6B1tJjrE8wz1QcXz36Vg03F1nj9W4FxsGyR7LdRtDffYqoDvBL5Kt
rYNead/KxZv root@storage-node7

EOF6

cat > /root/.ssh/authorized_keys <<EOF7

ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAABAQCxgarFbFB0ZSyJTNp9gu5foxUPeaOrevopovRBw1K5nwXdd/DtYlaOaG
67+u6stWgD3R+NkNBjpoQB0dlf6jbuYfqF+QxWrK6fBDq7cy1SqqTBErY20QmokGldnD35uaCh/lViXnAFJY8YiKDSR
vyb5wGbS5GgTllUW4AZzu6weR8gWU5BB32/P2Ho5fxtrdzrJQBkPNZKe3a53ls5OpXhl+lBjg7Y29iCbVWluUe9S+Y
/ti7nKXyHGfSKf5GZ96tOrxbDJmKExXQTl3irkd9P6B1tJjrE8wz1QcXz36Vg03F1nj9W4FxsGyR7LdRtDffYqoDvBL5Kt
rYNead/KxZv root@storage-node7

EOF7

chmod 700 /root/.ssh;

chmod 600 /root/.ssh/authorized_keys;

chmod 600 /root/.ssh/id_rsa;

chmod 644 /root/.ssh/id_rsa.pub;

################

Remove NetworkManager, a core package which is not needed.

yum -y remove NetworkManager;

%end

Appendix B – Kickstart File for Storage Nodes for Cisco UCS C240 M5 Server

#For storage-node-1

#version=DEVEL

#from the linux installation menu, hit tab and append this:

#biosdevname=0 net.ifnames=0 ip=eth1:dhcp

```
#ks=ftp://192.168.10.2/{hostname}.cfg

# System authorization information

auth --enableshadow --passalgo=sha512

# Use CDROM installation media

cdrom

# Use text install

text

# Run the Setup Agent on first boot

firstboot --disable

selinux --disable

firewall --disable

# Keyboard layouts

keyboard --vckeymap=us --xlayouts='us'

# System language

lang en_US.UTF-8

# Network information

network  --bootproto=static --device=eth0 --ip=173.36.220.240 --netmask=255.255.255.0 --onboot=on --
gateway=173.36.220.1 --nameserver=171.70.168.183 --ipv6=auto --activate

network  --bootproto=static --device=eth1 --ip=192.168.10.21 --netmask=255.255.255.0 --onboot=on --
ipv6=auto --activate

network  --bootproto=static --device=eth2 --ip=192.168.30.21 --netmask=255.255.255.0 --onboot=on --
ipv6=auto --activate

network  --bootproto=static --device=eth3 --ip=192.168.20.21 --netmask=255.255.255.0 --onboot=on --
ipv6=auto --activate

network  --hostname=storage-node1


# Root password

rootpw --iscrypted
$6$yfE2jHtdy.OSmO8g$InneiVXQI9Kc9m4w2cEiS8/og6BKUlu5HSR0eCYgh5dVaeCV54Q6piS7k10IalXignLCBvAZ
Pqmw4dvYgy66V1

# System services

services --disabled=chronyd

# System timezone
```

```
timezone America/Los_Angeles --isUtc --nontp
# System bootloader configuration
bootloader --append= crashkernel=auto --location=mbr --boot-drive=sda
# Partition clearing information
clearpart --drives=sdk --all --initlabel
# Disk partitioning information
part /boot --fstype=ext4 --ondisk=sdk --size=1024
part swap --fstype=swap --ondisk=sdk --size=4096
part /var --fstype=ext4 --ondisk=sdk --grow
part / --fstype=ext4 --ondisk=sdk --grow

reboot  --eject

%packages
@^minimal
@core
kexec-tools

%end

%addon com_redhat_kdump --enable --reserve-mb='auto'

%end

%anaconda
pwpolicy root --minlen=6 --minquality=50 --notstrict --nochanges --notempty
pwpolicy user --minlen=6 --minquality=50 --notstrict --nochanges --notempty
pwpolicy luks --minlen=6 --minquality=50 --notstrict --nochanges --notempty
%end
```

```
###############
#POST SCRIPT
###############
%post --log=/root/ks-post.log
###############
#GPT Labels for HDDs
###############
for i in a b {d..z} aa ab ac; do parted -s /dev/sd$i mklabel gpt; done;
###############
#Turn off Transparent Hugepages and ensure that hyperthreading
#is turned off.
###############
grubby --update-kernel=ALL --args=transparent_hugepage=never numa=off nr_cpus=24;
tuned-adm profile latency-performance;
systemctl enable ntpd;
###############
#Preconfigure /etc/hosts
###############
cat >> /etc/hosts <<EOF4
192.168.10.21       storage-node1
192.168.10.22       storage-node2
192.168.100.23      storage-node3
EOF4
###############
#Setup ssh keys
###############
mkdir /root/.ssh;
cat > /root/.ssh/id_rsa <<EOF5
-----BEGIN RSA PRIVATE KEY-----
MIIEpAIBAAKCAQEAsYGqxWxQdGUsiUzafYLuX6MVD3mjq3r6KaL0QcNSuZ8F3Xfw
```

7WJWjmhuu/rurLVoA90fjZDQY6aEAdHSH+o27mH6hfkMVqyunwQ6u3MtUqqkwRK2

NtEJqJBiHZw9+bmgofyFYl5wBSWPGlig0kb8m+cBm0uRoE5SFFuAGc7usHkflFlO

QQd9vz9h6OX8ba3c6yUAZDzWSnt2udyLOTqV4SPpQY4O2NvYgm1VpblHvUvmP7Yu

5yl8hxn0in+RmferTq8WwyZihMV0EyN4q5HfT+gdbSY6xPMM9UHF89+lYNNxdZ4/

VuBcbBskey3UbQ332KqA7wS+Sra2DXmnfysWbwlDAQABAolBAQCbeRFUXiyR5lP9

5lyw9k9HYRX/OfGLLumSMnJyb1wzzP9cHcPeh/V8QihLadxHVZTHXZRXcHG19pFE

7rx2y7RVU2gUlDCkchd4nEG9EYKvF1u66GLE3l7zH5Nwj/sQkfAKMZ26rTC8sUsG

mBUUWKzE+K7Fklj6ud7WidZHxKH32ok1lEcFOsH/nK1BXR29XmQ/O/Kg2h0V/KiM

1Y9CJngpghnybcDzlvpV6LS8bEiRieHJGT5RTyDk+ad0uSv+f2YtlpvSUly7NAft

e1feAq3RWT82ZGyKTHWGTFNbfltcUjzPI/dcyS8AurYf+oQjJVAKhAI+yln7lUrL

V6xKsdYBAoGBANwNb96gJHZUeSoOP/JCnTps+MeOhT1vyrhRRZf1laFnEmX7hXmE

RKXaQUvGcOSPumZMkKYyqRN22B2PLM7n1D0ypKshRmk1eq6tZ/W9gkYfldn0+QAx

AAVfUA8vJm9XLgkCAE4o2BHvtQ1w63CfygoF4V3OAsQv677F6ltROeiBAoGBAM6A

9quEOrPiRDiF25HnXXFUeRUXM4H77QB6WRV3AKggJjVlBXkhNt34g8Jr6/MfW4WO

SebQEwwBYH6NN7lG1Q0PeDRzrcv2voqzM7bV7l1rpc2E2BQhplcSyGr/aA6lW0OA

Ll/HZldqb6OXXR8lmcP0rfxuqUJ8e6SHskG6qAbvAoGAlrw4QXMT7l3NNndDXtFn

EjbrWkzD+XuxC0FA9Aisw1aKz/BRFGptj6SRFA4B+gl6ETXay3FJwRnMaXYVQ5/S

n8pjteOtwqO/dt1GgMLmUn1NkaMavw39C9wMvijaL08apC9drvjBiqtE8Bc4Avlm

KUjeVzlStHdABkAlQgCTXlECgYEAur6BU4YWmAnsa7kRYRZ7uDsN7Ha4y7mJED+U

RAcD/wZjxzF+C5ZvybgtXyq9i3U2DMcqKaLNNrQgERGf5kyrak4tBDlAX0zZ7xAz

mgplrw7kN8EErt/nTyLbP3eNllGE0LwgM9lbHeKw5p3BRok+lKi2lmt0gX2VSqq0

FyC3Rt0CgYADqOJ53sV7NEXfd/NG5D9bzS5yCKW+KNH4fzxAoAYhMBo3nAkgpa/1

rdjPH4f5bAMX6dKZCh5Sy9BFxgqbl0tdjVGZBUPK8tb0xbcnJ2F3+aLq02fCfyr+

TfYW1tZ7g7gZJ+To42h4Tv9wj8iWGe+pnR4Moh3WqM1TttuaCJf1nQ==

-----END RSA PRIVATE KEY-----

EOF5

cat > /root/.ssh/id_rsa.pub <<EOF6

ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAABAQCxgarFbFB0ZSyJTNp9gu5foxUPeaOrevopovRBw1K5nwXdd/DtYlaOaG
67+u6stWgD3R+NkNBjpoQB0dlf6jbuYfqF+QxWrK6fBDq7cy1SqqTBErY20QmokGldnD35uaCh/lViXnAFJY8YiKDSR
vyb5wGbS5GgTllUW4AZzu6weR8gWU5BB32/P2Ho5fxtrdzrJQBkPNZKe3a53ls5OpXhl+lBjg7Y29iCbVWluUe9S+Y

```
/ti7nKXyHGfSKf5GZ96tOrxbDJmKExXQTI3irkd9P6B1tJjrE8wz1QcXz36Vg03F1nj9W4FxsGyR7LdRtDffYqoDvBL5Kt
rYNead/KxZv root@storage-node7

EOF6

cat > /root/.ssh/authorized_keys <<EOF7

ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAABAQCxgarFbFB0ZSyJTNp9gu5foxUPeaOrevopovRBw1K5nwXdd/DtYlaOaG
67+u6stWgD3R+NkNBjpoQB0dlf6jbuYfqF+QxWrK6fBDq7cy1SqqTBErY20QmokGldnD35uaCh/lViXnAFJY8YiKDSR
vyb5wGbS5GgTllUW4AZzu6weR8gWU5BB32/P2Ho5fxtrdzrJQBkPNZKe3a53ls5OpXhl+lBjg7Y29iCbVWluUe9S+Y
/ti7nKXyHGfSKf5GZ96tOrxbDJmKExXQTI3irkd9P6B1tJjrE8wz1QcXz36Vg03F1nj9W4FxsGyR7LdRtDffYqoDvBL5Kt
rYNead/KxZv root@storage-node7

EOF7

chmod 700 /root/.ssh;

chmod 600 /root/.ssh/authorized_keys;

chmod 600 /root/.ssh/id_rsa;

chmod 644 /root/.ssh/id_rsa.pub;

################

# Remove NetworkManger, a core package which is not needed.

yum -y remove NetworkManager;

%end
```

Summary

Cisco UCS Infrastructure for Cloudian Software Defined Storage is an integrated solution to deploy Cloudian HyperStore and combines the value of Intel Xeon architecture, Cisco data center hardware and software, along with Red Hat Linux. This solution increases the speed of deployment and reduces the risk of scaling from proof-of-concept to full-enterprise production, and is validated and supported by Cisco and Cloudian.

Cisco UCS hardware with Cisco UCS Manager Software brings an integrated, scalable, multi-chassis platform in which all resources participate in a unified management domain. Creating and cloning service profiles from its templates and maintaining the hardware from a single pane of glass not only provides rapid provisioning of hardware but also makes management and firmware upgrades simpler.

Cloudian HyperStore software makes it easy to build fully featured, Amazon S3-compliant cloud storage, on-premise. Cloudian HyperStore software ensures unlimited scale, multi-data center storage, fully automated data tiering, and support for all S3 applications—all behind your firewall.

Cloudian HyperStore software deployed on UCS S-Series servers, combines robust availability with system management control, monitoring capabilities and reporting. A host of features, including hybrid cloud streaming, virtual nodes, configurable erasure coding, and data compression and encryption sets Cloudian apart with highly efficient storage and seamless data management that lets you store and access your data where you want it, when you want it. Built on a robust object storage platform for effortless data sharing, cloud service providers around the world use Cloudian HyperStore to deploy and manage both public and private clouds, while enterprises rely on it to maintain their private and hybrid clouds.

This Cisco Validated Design is a partnership of Cisco Systems and Cloudian. Combining these technologies, expertise and experience in the field, we are able to provide an enterprise-ready hardware and software solution.

About the Authors

Paniraja Koppa, Cisco Systems, Inc.

Paniraja Koppa is a Technical Marketing Engineer for UCS Solutions. He has more than 13 years of experience with a primary focus on data center technologies such as Cisco UCS, Storage, Operating systems, Automation, Virtualization and Cloud. In his current role at Cisco Systems, he works on best practices, optimization,automation and performance tuning of software defined storage on Cisco UCS platforms. Prior to this, he has led QA efforts for 4 new virtual adapter card's firmware and software features for Cisco UCS. He also worked as customer support engineer and advocate in the DatacenterVirtualization space.

Eddo Jansen, Cloudian, Inc

Eddo Jansen is Principal Architect at Cloudian. He has over 15 years of experience in IT Infrastructure, Storage, Virtualization and automation. His current role is building performant, scalable, highly available, and durable object store solutions with specialties in Performance testing, analyzing, troubleshooting and tuning.

Acknowledgements

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

- Chris O'Brien, Cisco Systems, Inc.
- Samuel Nagalingam, Cisco Systems, Inc.
- Oliver Walsdorf, Cisco Systems, Inc.
- Jawwad Memon, Cisco Systems, Inc.
- Sanjay Jagad, Cloudian