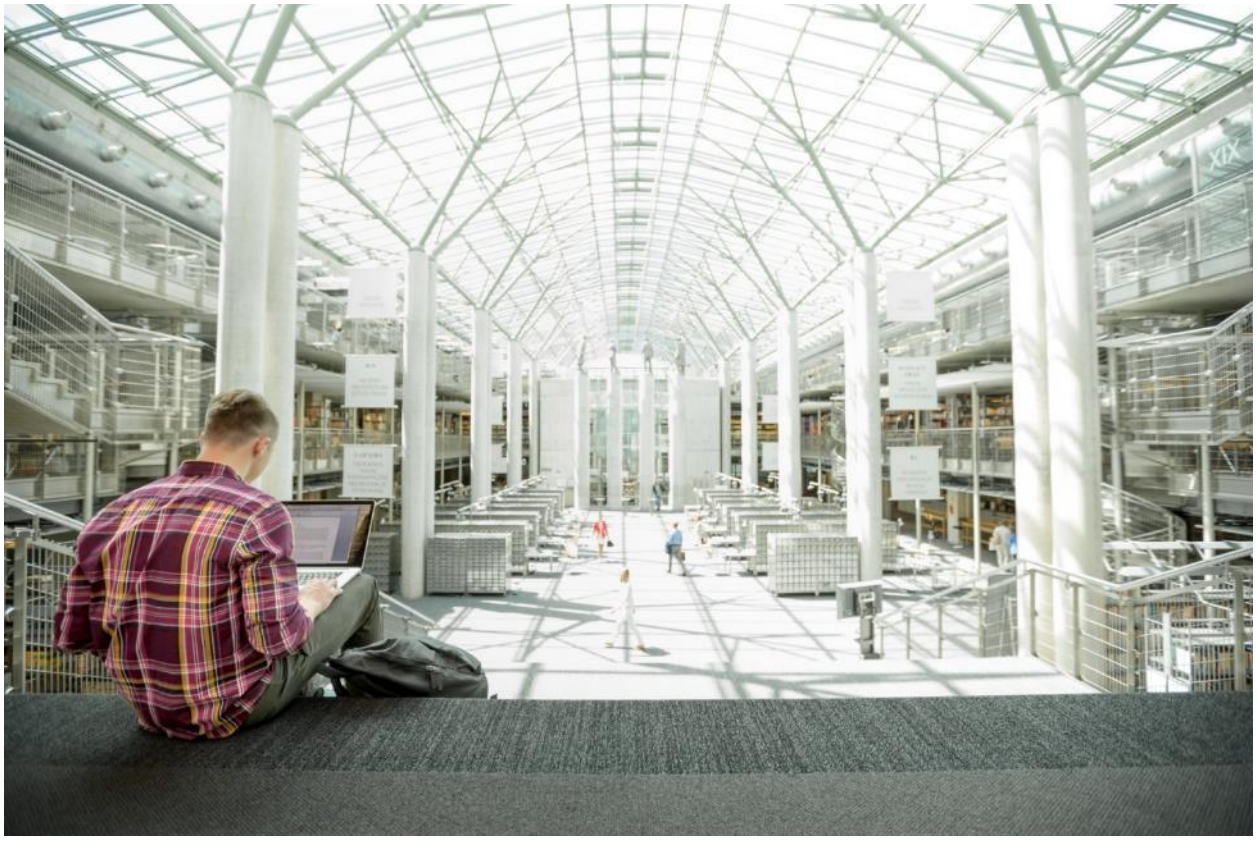


Cisco UCS S3260 M5 Storage Server with Scality RING

Design and Deployment of Scality Object Storage on Cisco UCS S3260 M5 Storage Server

Last Updated: November 26, 2018

About the Cisco Validated Design Program

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

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

Table of Contents

Executive Summary

Cisco Validated Designs 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.

The purpose of this document is to describe the design and deployment of Scality Ring on Red Hat Enterprise Linux with the latest generation of Cisco UCS S3260 M5 Servers. This CVD provides the framework of designing and deploying Scality SDS software on Cisco UCS S3260 storage servers. Cisco Unified Computing System provides the storage, network, and storage access components for the Scality Ring, deployed as a single cohesive system.

This Cisco Validated Design describes how Cisco Unified Computing System (Cisco UCS) can be used in conjunction with Scality Ring 7.4. With the continuous evolution of SDS there has been increased demand to have Scality Ring validated on Cisco UCS servers. The Cisco UCS S3260 Storage Server, originally designed for the data center, together with Scality RING is optimized for object storage solutions, making it an excellent fit for unstructured data workloads such as backup, archive, and cloud data. The Cisco UCS S3260 delivers a complete infrastructure with exceptional scalability for computing and storage resources together with 40 Gigabit Ethernet networking.

Cisco and Scality are collaborating to offer customers a scalable object storage solution for unstructured data that is integrated with Scality RING Storage. With the power of the Cisco UCS management framework, the solution is cost effective to deploy and manage and enables the next-generation cloud deployments that drive business agility, lower operational costs, and avoids vendor lock-in.

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 is performed using RESTful API's, which uses HTTP verbs such as GETs and PUTs. Most of the newly generated data is unstructured today. With about 80 percent of data being unstructured, new approaches using x86 servers are proving to be more cost effective, providing storage that can be expanded as easily as your data grows. Scale-out Object storage is the newest approach for handling massive amounts of data.

The Scality RING is a Software-Defined Storage that is designed to create unbounded scale-out storage systems that converge the storage of Petabyte scale data from multiple applications and use-cases, including both object and file based applications

Together with Cisco UCS, Scality Ring can deliver a fully enterprise-ready solution that can manage different workloads and still remain flexible. The Cisco UCS S3260 Storage Server is an excellent platform to use with the main types of Object and File workloads, such as capacity-optimized and performance-optimized workloads. It is best suited for sequential access, as opposed to random, to unstructured data, and to whatever the data size. It is essentially designed for applications, not direct end-users.

This document describes the architecture, design and deployment procedures of Scality storage on Cisco UCS S3260 M5 servers along with Cisco UCS C220 M5 rack-mounted servers.

Solution

This Cisco Validated Design is a simple and linearly scalable architecture that provides an object storage solution on Scality RING and Cisco UCS S3260 Storage Server. The solution includes the following features:

- Infrastructure for large scale object storage
- Design of a Scality Object Storage solution together with Cisco UCS S3260 Storage Servers
- Simplified infrastructure management with Cisco UCS Manager
- Architectural scalability – linear scaling based on network, storage, and compute requirements

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 Scality Ring 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.

Solution Summary

This solution is focused on Scality storage on Red Hat Enterprise Linux 7.5 on Cisco Unified Computing System UCS S3260 storage server. The advantages of Cisco UCS and Scality combine to deliver an object storage solution that is simple to install, scalable and performant. The configuration uses the following components for the deployment:

- Cisco Unified Computing System
  - Cisco UCS 6332 Series Fabric Interconnects
  - Cisco UCS S3260 M5 storage servers.
  - Cisco S3260 system IO controller with VIC 1380
  - Cisco C220M5 servers with VIC 1387
- Cisco Nexus C9332PQ Series Switches
- Scality storage 7.4
- Red Hat Enterprise Linux 7.5

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. Cisco UCS servers offer the patented Cisco Extended Memory Technology to support applications with large datasets and allow more virtual machines (VM) 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 the 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 S3260 Storage Server

The Cisco UCS Storage Server is a modular, high-density, high-availability, dual node rack server well-suited for service providers, enterprises, and industry-specific environments. It addresses the need for dense cost effective storage for the ever-growing data needs. Designed for a new class of cloud-scale applications, it is simple to deploy and excellent for big data applications, software-defined storage environments, and other unstructured data repositories, media streaming, and content distribution.

Figure 1 The Cisco UCS® S3260 Storage Server

Extending the capability of the Cisco UCS C3000 portfolio, the Cisco UCS S3260 helps you achieve the highest levels of data availability. With dual-node capability that is based on the Intel® Xeon® scalable processor, it features up to 720 TB of local storage in a compact 4-rack-unit (4RU) form factor. All hard-disk drives can be asymmetrically split between the dual-nodes and are individually hot-swappable. The drives can be built-in in an enterprise-class Redundant Array of Independent Disks (RAID) redundancy or be in a pass-through mode.

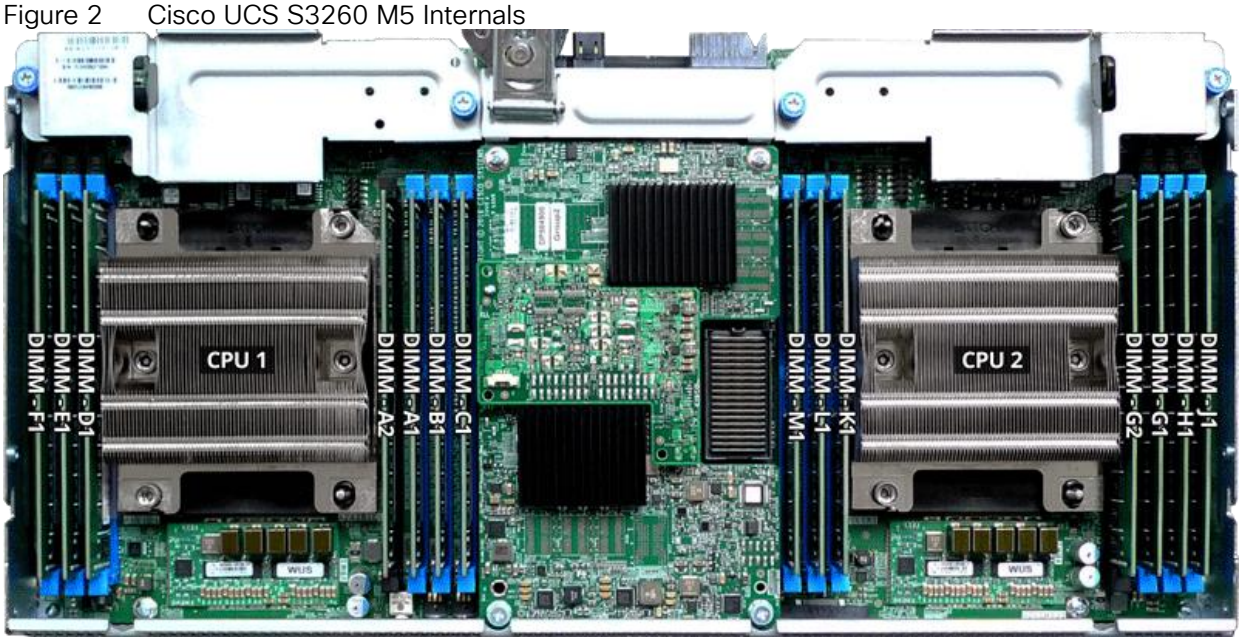
This high-density rack server comfortably fits in a standard 32-inch depth rack, such as the Cisco® R42610 Rack.

The Cisco UCS S3260 is deployed as a standalone server in both bare-metal or virtualized environments. Its modular architecture reduces TCO by allowing you to upgrade individual components over time and as use cases evolve, without having to replace the entire system.

The Cisco UCS S3260 uses a modular server architecture that, using Cisco's blade technology expertise, allows you to upgrade the computing or network nodes in the system without the need to migrate data migration from one system to another. It delivers the following:

- Dual server nodes
- Up to 44 computing cores per server node
- Up to 60 drives mixing a large form factor (LFF) with up to 28 solid-state disk (SSD) drives plus 2 SSD SATA boot drives per server node
- Up to 1.5 TB of memory per server node (3 TB Total) with 128GB DIMMs
- Support for 12-Gbps serial-attached SCSI (SAS) drives
- A system I/O Controller either with HBA Passthrough or RAID controller, with DUAL LSI 3316 Chip
- Cisco VIC 1300 Series Embedded Chip supporting Dual-port 40Gbps
- High reliability, availability, and serviceability (RAS) features with tool-free server nodes, system I/O controller, easy-to-use latching lid, and hot-swappable and hot-pluggable components
- Dual 7mm NVMe - Capacity points: 512G, 1TB and 2TB
- 1G Host Management Port

Figure 2 Cisco UCS S3260 M5 Internals



Cisco UCS C220 M5 Rack-Mounted Server

The Cisco UCS C220 M5 Rack-Mounted 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-Mounted Servers can be deployed as standalone servers or as part of the Cisco Unified Computing System™ 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 3 Cisco UCS C220M5 Rack-Mounted 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.

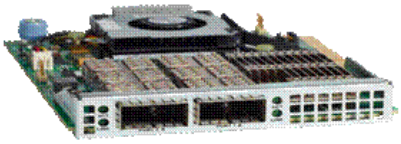
The Cisco UCS C220 M5 server can be used standalone, or as part of the Cisco Unified Computing System, which unifies computing, networking, management, virtualization, and storage access into a single integrated

architecture enabling end-to-end server visibility, management, and control in both bare metal and virtualized environments.

Cisco UCS Virtual Interface Card 1387

The Cisco UCS Virtual Interface Card (VIC) 1387 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) modular LAN-on-motherboard (mLOM) adapter is designed exclusively for Cisco UCS C-Series and 3260 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 Data Center Virtual Machine Fabric Extender (VM-FEX) technology. This technology extends the Cisco UCS fabric interconnect ports to virtual machines, simplifying server virtualization deployment.

Figure 4 Cisco UCS Virtual Interface Card 1387



The Cisco UCS VIC 1387 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

Cisco UCS 6300 Series Fabric Interconnect

The Cisco UCS 6300 Series Fabric Interconnects are a core part of Cisco UCS, providing both network connectivity and management capabilities for the system (Figure 5). 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 5 Cisco UCS 6300 Series Fabric Interconnect

The Cisco UCS 6300 Series provides the management and communication backbone for the Cisco UCS B-Series Blade Servers, 5100 Series Blade Server Chassis, and 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 Nexus 9332PQ Switch

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 6 Cisco 9332PQ

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 non-blocking 10 and 40 Gigabit Ethernet switches with up to 2.56 terabits per second (Tbps) of internal bandwidth.

The Cisco Nexus 9332PQ Switch is a 1-rack-unit (1RU) switch that supports 2.56 Tbps of bandwidth and over 720 million packets per second (mpps) across thirty-two 40-Gbps Enhanced QSFP+ 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 Manager

Cisco UCS® Manager (UCSM) provides 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, C-Series Rack 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 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 7 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 command-line interface (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. 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).

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.

Red Hat Enterprise Linux 7.5

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 world's 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:

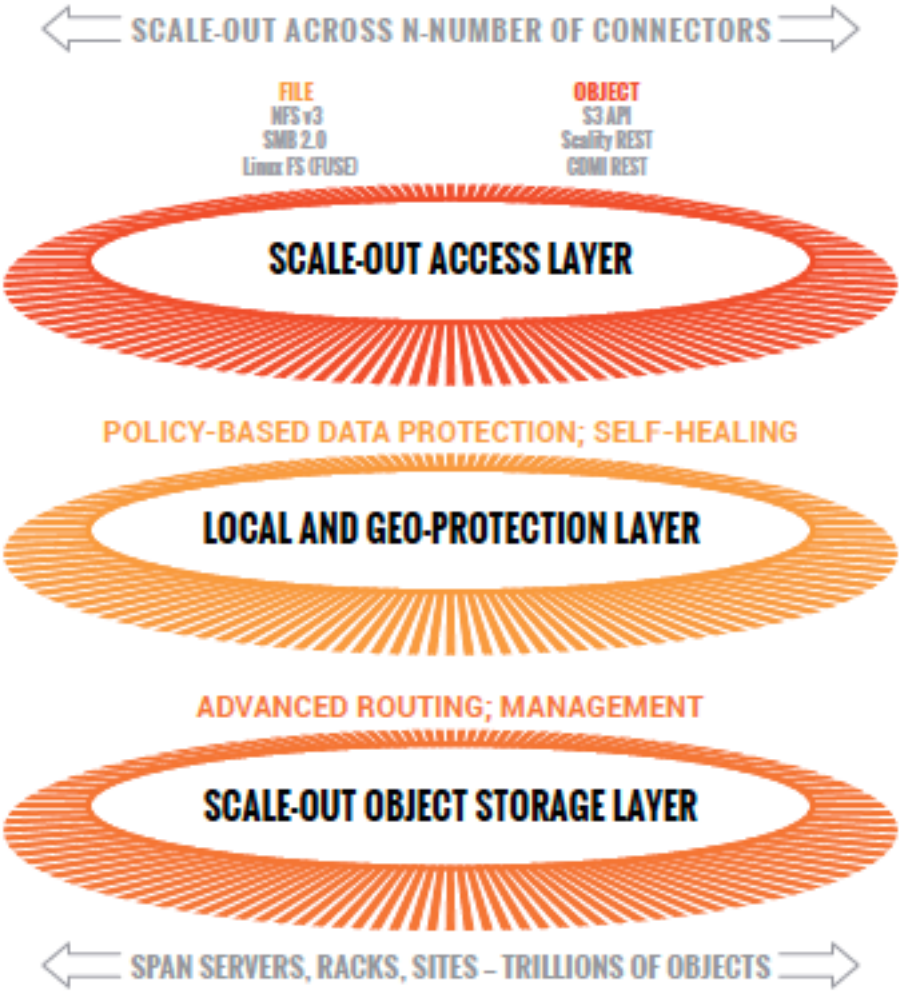
- Delivers high performance, reliability, and security
- Is certified by the leading hardware and software vendors
- Scales from workstations, to servers, to mainframes
- Provides a consistent application environment across physical, virtual, and cloud deployments

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

Scality RING Overview

RING is a cloud-scale, distributed software solution for petabyte-scale unstructured data storage. It is designed to create unbounded scale-out storage systems for the many petabyte-scale applications and use cases, both object and file, that are deployed in today's enterprise data centers. RING is a fully distributed system deployed on industry standard hardware, starting with a minimum of three (3) storage servers. The system can be seamlessly scaled-out to thousands of servers with 100's of petabytes of storage capacity. RING has no single points of failure, and requires no downtime during any upgrades, scaling, planned maintenance or unplanned system events. With self-healing capabilities, it continues operating normally throughout these events. To match performance to increasing capacity, RING can also independently scale-out its access layer of protocol "Connectors", to enable an even match of aggregate performance to the application load. RING provides data protection and resiliency through local or geo-distributed erasure-coding and replication, with services for continuous self-healing to resolve expected failures in platform components such as servers and disk drives. RING is fundamentally built on a scale-out object-storage layer that employs a second-generation peer-to-peer architecture. This approach uniquely distributes both the user data and the associated metadata across the underlying nodes to eliminate the typical central metadata database bottleneck. To enable file and object data in the same system, the RING integrates a virtual file system layer through an internal NoSQL scale-out database system, which provides POSIX-based access semantics using standard NFS, SMB and FUSE protocols with shared access to the files as objects using the REST protocol.

Figure 8 Scality RING Diagram



Scality has designed RING along the design criteria spearheaded by the leading cloud-scale service providers, such as Google, Facebook, and Amazon. RING leverages loosely-coupled, distributed systems' designs that leverage commodity, mainstream hardware along the following key tenets:

- 100 percent parallel design for metadata and data - to enable scaling of capacity and performance to unbounded numbers of objects, with no single points of failures, service disruptions, or forklift upgrades as the system grows.
- Multi-protocol data access – to enable the widest variety of object, file and host-based applications to leverage RING storage.
- Flexible data protection mechanisms - to efficiently and durably protect a wide range of data types and sizes.
- Self-healing from component failures – to provide high-levels of data durability, the system expects and tolerates failures and automatically resolves them.
- Hardware freedom – to provide optimal platform flexibility, eliminate lock-in and reduce TCO.

RING incorporates these design principles at multiple levels, to deliver the highest levels of data durability, at the highest levels of scale, for most optimal economics.

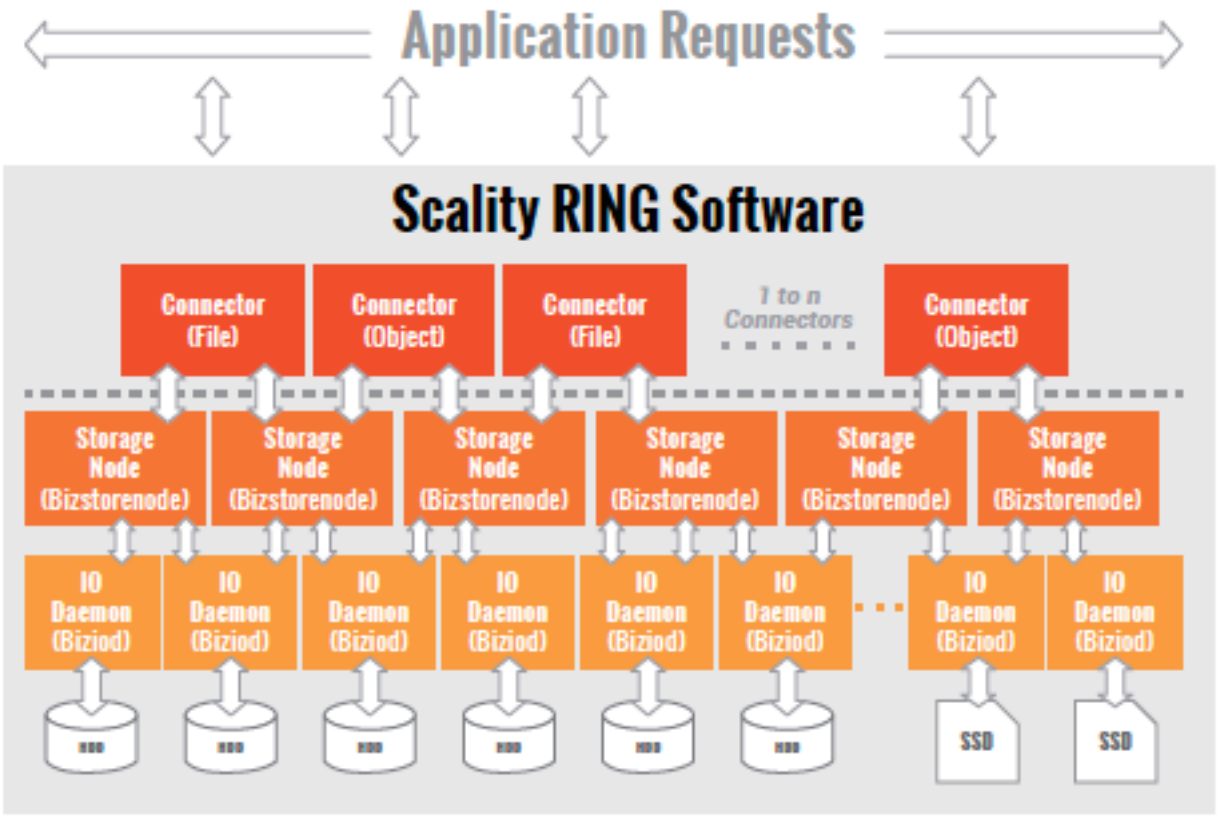
Scality RING Architecture

To scale both storage capacity and performance to massive levels, the Scality RING software is designed as a distributed, parallel, scale out architecture with a set of intelligent services for data access and presentation, data protection and systems management. To implement these capabilities, RING provides a set of fully abstracted software services including a top-layer of scalable access services (Connectors) that provide storage protocols for applications. The middle layers are comprised of a distributed virtual file system layer, a set of data protection mechanisms to ensure data durability and integrity, self-healing processes and a set of systems management and monitoring services. At the bottom of the stack, the system is built on a distributed storage layer comprised of virtual storage nodes and underlying IO daemons that abstract the physical storage servers and disk drive interfaces.

At the heart of the storage layer is a scalable, distributed object key/value store based on a second-generation peer-to-peer routing protocol. This routing protocol helps ensure that store and lookup operations scale efficiently to very high numbers of nodes.

RING software is comprised of the following main components: RING Connectors, a distributed internal NoSQL database called MESA, RING Storage Nodes and IO daemons, and the Supervisor web-based management portal. The MESA database is used to provide object indexing, as well as the integral Scale-Out-File-System (SOFS) file system abstraction layer, and the underlying core routing protocol and Keyspace mechanisms are described later in this paper.

Figure 9 Scality Scale-out Architecture



RING Connectors

The Connectors provide the data access endpoints and protocol services for applications that use RING for data storage. As a scale-out system, RING supports any number of Connectors and endpoints to support large and growing application workloads. The RING 7 release provides a family of object and file interfaces:

- AWS S3 API – a comprehensive implementation of the AWS S3 REST API, with support for the Bucket and Object data model, AWS style Signature v4/v2 authentication, and the AWS model of Identity and Access Management (IAM)
- http/REST (sproxyd) – the RING's native key/value REST API, provides a flat object storage namespace with direct access to RING objects
- NFS v3 – SOFS volumes presented as a standard NFSv3 mount points
- SMB 2.0 – SOFS volumes presented as SMB Shares to Microsoft Windows clients. Several SMB 3.0 features are currently supported, a later release will provide full SMB 3.0 support
- FUSE – SOFS volumes presented as a local Linux file system
- CDMI/REST – support for the SNIA CDMI REST interface, with full compatibility to SOFS file data
- S3 on SOFS – SOFS volumes may be accessed in read-only mode over the S3 protocol, for namespace and data sharing between objects and files

Connectors provide storage services for read, write, delete and lookup for objects or files stored into the RING based on either object or POSIX (file) semantics. Applications can make use of multiple connectors in parallel to scale out the number of operations per second, or the aggregate throughput of the RING. A RING deployment may be designed to provide a mix of file access and object access (over NFS and S3 for example), simultaneously—to support multiple application use cases.

Storage Nodes and IO Daemons

The heart of the ring are the Storage Nodes, that are virtual processes that own and store a range of objects associated with its portion of the RING's keyspace. A complete description of RING's keyspace mechanism is provided in the next section. Each physical storage server (host) is typically configured with six (6) storage nodes (termed bizstorenode). Under the storage nodes are the storage daemons (termed biziod), which are responsible for persistence of the data on disk, in an underlying local standard disk file system. Each biziod instance is a low-level software process that manages the IO operations to a particular physical disk drive and maintains the mapping of object keys to the actual object locations on disk. Biziod processes are local to a given server, managing only local, direct-attached storage and communicating only with Storage Nodes on the same server. The typical configuration is one biziod per physical disk drive, with support for up to hundreds of daemons2 per server, so the system can support very large, high-density storage servers.

Each biziod stores object payloads and metadata in a set of fixed size container files on the disk it is managing. With such containerization the system can maintain high-performance access even to small files, without any storage overhead. The bizoid deamons typically leverage low-latency flash (SSD or NVMe) devices to store the index files for faster lookup performance. The system provides data integrity assurance and validation through the use of stored checksums on the index and data container files, which are validated upon read access to the data. The use of a standard file system underneath biziod ensures that administrators can use normal operating system utilities and tools to copy, migrate, repair and maintain the disk files if required.

The recommended deployment for systems that have both HDD and SSD media on the storage servers is to deploy a data RING on HDD, and the associated metadata in a separate RING on SSD. Typically, the requirements for metadata are approximately 2 percent of the storage capacity of the actual data, so the sizing of SSD should follow that percentage for best effect. Scality can provide specific sizing recommendations based on the expected average file sizes, and number of files for a given application.

RING Systems Management

Managing and monitoring the RING is enabled through a cohesive suite of user interfaces, built on top of a family of RESTful interfaces termed the Supervisor API ("SupAPI"). The SupAPI provides an API based method that may be accessed from scripts, tools and frameworks for gathering statistics, metrics, health check probes and alerts, and for provisioning new services on the RING. The SupAPI is also enabled with Role Based Access Control (RBAC), by supporting an administrator identity to provide access control privileges for Super-Admin and Monitor admin user Roles.

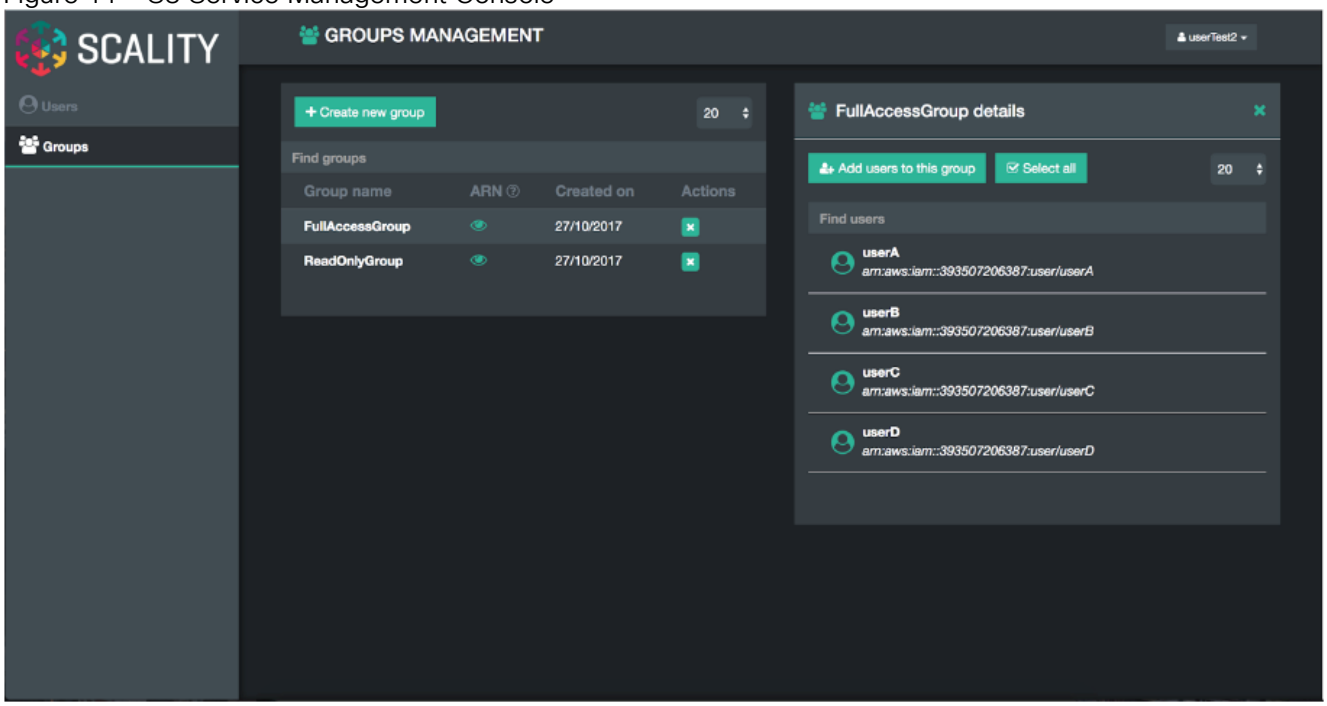
RING provides a family of tools that use the SupAPI for accessing the same information and services. RING 7 includes the new "Scality Supervisor", a browser-based portal for both systems monitoring and management of Scality components. In RING 7, the Supervisor now provides capabilities across object (S3) and file (NFS, SMB, FUSE) Connectors including integrated dashboards including Key Performance Indicators (KPIs) with trending information such as "Global Health", "Performance", "Availability" and "Forecast." The Supervisor also includes provisioning capabilities to add new servers in the system and a new zone management module to handle customer failure domains for multi-site deployments.

Figure 10 Supervisor Web GUI

RING Supervisor also includes an “Advanced Monitoring” dashboard where all collected metrics can be graphed and analyzed component per-component and per-server. This is based on a very powerful graphing engine that has access to thousands of metrics.

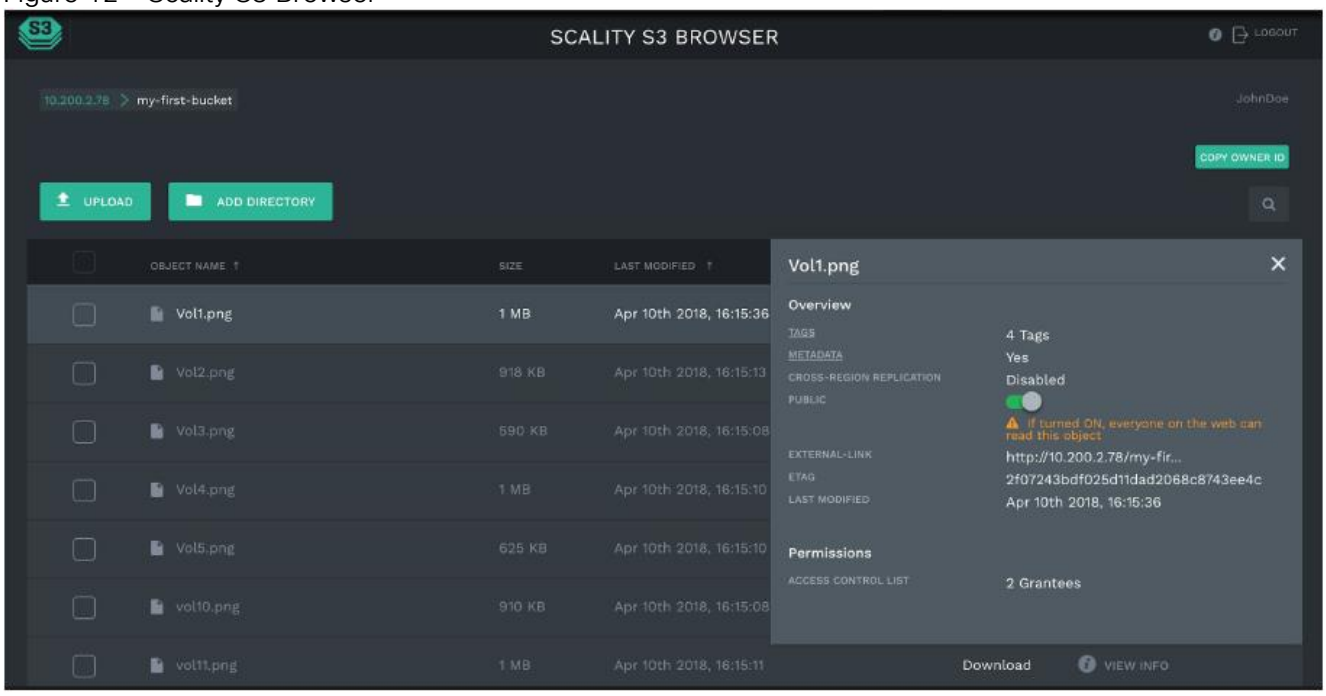
A new “S3 Service Management console” portal is provided to manage the integrated AWS Identity and Access Management (IAM) model of S3 multi-tenancy in the RING. This provides two-level management of Accounts, Users/Groups and IAM access control policies. The S3 Console may also be easily customized for white-labeling purposes.

Figure 11 S3 Service Management Console



A new “Scality S3 Browser” is also provided to browse S3 buckets, upload and download object data, and for managing key S3 features such as bucket versioning, CORS, editing of metadata attributes and tagging. The S3 Browser is an S3 API client that runs on the S3 user browser and is accessible to both the Storage administrator and also to the S3 end-user.

Figure 12 Scality S3 Browser



A scriptable Command Line Interface (CLI) called RingSH is also provided, as well as an SNMP compatible MIB and traps interface for monitoring from standard SNMP consoles. RING is designed to be self-managing and autonomous to free administrators to work on other value-added tasks, and not worry about the component level management tasks common with traditional array based storage solutions.

S3 Connector: AWS S3 Storage with Identity and Access Management (IAM)

The Scality S3 Connector provides a modern S3 compatible application interface to the Scality RING. The AWS S3 API has now become the industry's default cloud storage API and has furthermore emerged as the standard RESTful dialect for object storage as NFS was for the NAS generation. The S3 Connector is built on a distributed scale-out architecture to support very high levels of application workloads and concurrent user access. This is based on a highly-available, high-performance metadata engine that can also be scaled-out for increased performance. Deployments can be geo-replicated deployments to enable highly-available disaster recovery solutions, for both Metro-Area Network environments ("stretched" deployments), as well as Cross Region Replication (CRR) asynchronous replication of individual S3 buckets or a full site.

The Scality S3 Connector also provides a full implementation of the AWS multi-tenancy and identity management (AWS IAM) model with federated authentication to any Enterprise Directories via SAML 2.0. In addition to the RING Supervisor management UI, the S3 Service Provider UI is a web-based user interface to manage multi-tenancy accounts, users, group and policies. To support enterprise security, development and operational methodologies, the S3 Connector on RING supports:

- Integration with Enterprise directory/security servers: most commonly Microsoft Active Directory or LDAP servers. Federated authentication integration is supported through a SAML 2.0-compatible Identity Provider such as Microsoft ADFS, and many other SAML compatible products, to enable a complete Single Sign-On (SSO) solution.
- Secure Multi-tenancy support: through IAM Accounts, secure access keys, Users, Groups, access control policies and v4 authentication per-tenant, bucket encryption (integrated with corporate KMS solutions) and auditing.

- Utilization reporting to enable chargeback: the S3 Connector Utilization API provides an extended API for reporting on comprehensive usage metrics including capacity, #objects, bandwidth and S3 operations (per unit time). This provides all of the metrics required for consumption into corporate tools for chargeback calculations.
- Cloud-based S3 monitoring: integration through Scality's upcoming Cloud Monitoring console, and load-balancer driven health checks.
- High-performance, scale-out access: to support many corporate applications and workloads simultaneously reading and writing to the S3 service.
- Highly-available disaster-recovery solutions: enabled deployments through multi-data center deployments to provide availability in the event of site failure.

In RING 7, the feature set for the S3 Connector now supports Bucket Versioning via the S3 API, and for Cross Region Replication (CRR) of Buckets via the S3 API, this provides bucket-level asynchronous replication to another S3/RING deployment.

Scale-Out-File-System (SOFS)

RING supports native file system access to RING storage through the integrated Scale-Out-File-System (SOFS) with NFS, SMB and FUSE Connectors for access over these well-known file protocols. SOFS is a POSIX compatible, parallel file system that provides file storage services on the RING without the need for external gateways.

SOFS is more precisely a virtual file system, which is based on an internal distributed database termed MESA ("table" in Spanish) on top of the RING's storage services. MESA is a distributed, semi-structured database that is used to store the file system directories and file inode structures. This provides the virtual file system hierarchical view, with the consistency required for file system data, by ensuring that file system updates are always atomic. This means that updates are either committed or rolled back entirely–which guarantees the file system is never left in an intermediate or inconsistent state. A key advantage for scaling is that MESA is itself is distributed as a set of objects across all of the RING's storage node in a shared nothing manner to eliminate any bottlenecks or limitations.

File system lookups are performed using the RING's standard peer-to-peer routing protocol. For fast access performance, SOFS metadata is recommended to be stored on flash storage, typically on its own dedicated SSD drives in the storage servers, with the SOFS file payloads stored in the "data RING" on hard disk drives (HDDs). SOFS works directly with the data protection and durability mechanisms present in the RING, including replication and configurable Erasure Coding schemas.

SOFS can be provisioned into one or more "volumes," and can be scaled in capacity as needed to support application requirements. Each volume can be accessed by any number of Connectors to support the incoming workload, even with mixed protocols (NFS, SMB or FUSE). RING can support an enormous number of volumes (up to 232) and can grow to billions of files per volume. There is no need to pre-configure volumes for capacity (the RING effectively supports thin-provisioning of volumes). Volumes will utilize the RING's storage pool to expand as needed when files are created and updated. For efficient storage of very large files, the RING supports the concept of "sparse files", effectively files combined from multiple individual data-stripes.

While multiple Connectors may be used to simultaneously access a volume, the RING currently supports scale-out access for multiple concurrent readers, and a new "File Access Coordination" mode that allows multiple readers on a file while it is being written from another Connector. This is useful in use-cases such as video streaming where very large video files are written over the course of minutes or hours, but the file must be accessed for content distribution before the write is complete. Multiple Connectors attempt to write to the same directory or one per file within a directory, SOFS maintains view consistency across multiple connectors. By

supporting scale-out across any number of Connectors, SOFS throughput can be scaled out to support increasing workload demands. When performance saturation is reached, it is always possible to add more connectors or storage nodes (and disk spindles) to the RING to further increase throughput into the system. The system can achieve 10's of Gigabytes per second of aggregate throughput for parallel workloads through this architecture.

SOFS provides volume-level utilization metering and quota support, in addition to User and Group (uid/gid) quotas. This enables administrators to effectively use the concept of volumes to meter, report and limit space (capacity) usage at the volume level. This is useful in a multi-tenant environment where multiple applications or use cases are sharing the same RING, but accessing data stored in their own volume.

SOFS also provides integrated failover and load balancer services for the NFS and SMB Connectors. The load balancer uses an integrated DNS service to expose one or more service names (for example, sofs1.companyname. com) on Virtual IP addresses (VIPs), which can be mounted as NFS mount points or SMB shares. The load balancer can be configured with multiple underlying NFS or SMB connector real IP addresses, and provides load balancing of file traffic across these SOFS connectors. In combination with the RING 6.0 Folder Scale-out feature, this also provides transparent multi-connector access to a single folder, as well as enabling failover. In the event one of the underlying NFS or SMB Connectors becomes non-responsive, the load balancer can select another Connector IP address as the access point for the request.

Intelligent Data Durability and Self-Healing

RING is designed to expect and manage a wide range of component failures including disks, server networks and even across multiple data centers, while ensuring that data remains durable and available during these conditions. RING provides data durability through a set of flexible data protection mechanisms optimized for distributed systems, including replication, erasure coding and geo-replication capabilities that allow applications to select the best data protection strategies for their data. These flexible data protection mechanisms implement Scality's design principle to address a wide spectrum (80 percent) of storage workloads and data sizes. A full description of multi-site data protection is provided in the next section, Multi-Site Geo-Distribution.

Replication Class of Service (COS)

To optimize data durability in a distributed system, the RING employs local replication, or the storage of multiple copies of an object within the RING. RING will attempt to spread these replicas across multiple storage nodes, and across multiple disk drives, in order to separate them from common failures (assuming sufficient numbers of servers and disks are available). RING supports six Class-of-Service levels for replication (0-5), indicating that the system can maintain between 0 to 5 replicas (or 1-6 copies) of an object. This allows the system to tolerate up to 5 simultaneous disk failures, while still preserving access and storage of the original object. Note that any failure will cause the system to self-heal the lost replica, to automatically bring the object back up to its original Class-of-Service, as fast as possible.

While replication is optimal for many use cases where the objects are small, and access performance is critical, it does impose a high storage overhead penalty compared to the original data. For example, a 100KB object being stored with a Class-of-Service=2 (2 extra copies so 3 total), will therefore consume 3 x 100KB = 300KB of actual physical capacity on the RING, in order to maintain its 3 replicas. This overhead is acceptable in many cases for small objects but can become a costly burden for megabyte or gigabyte level video and image objects. In this case, paying a penalty of 200 percent to store a 1GB object since it will require 3GB of underlying raw storage capacity for its 3 replicas. When measured across petabytes of objects, this becomes a significant cost burden for many businesses, requiring a more efficient data protection mechanism.

Flexible Erasure Coding

Scality's erasure coding (EC) provides an alternative data protection mechanism to replication that is optimized for large objects and files. RING implements Reed-Solomon erasure coding6 techniques, to store large objects with an extended set of parity "chunks", instead of multiple copies of the original object. The basic idea with erasure

coding is to break an object into multiple chunks (m) and apply a mathematical encoding to produce an additional set of parity chunks (k). A description of the mathematical encoding is beyond the scope of this paper, but they can be simply understood as an extension of the XOR parity calculations used in traditional RAID. The resulting set of chunks, (m+k) are then distributed across the RING nodes, providing the ability to access the original object as long as any subset of m data or parity chunks are available. Stated another way, this provides a way to store an object with protection against k failures, with only k/m overhead in storage space.

Many commercial storage solutions impose a performance penalty on reading objects stored through erasure coding, due to the fact that all of the chunks, including the original data, are encoded before they are stored. This requires mandatory decoding on all access to the objects, even when there are no failure conditions on the main data chunks. With Scality's EC, the data chunks are stored in the clear, without any encoding, so that this performance penalty is not present during normal read accesses. This means that EC data can be accessed as fast as other data, unless a data chunk is missing which would require a parity chunk to be accessed and decoded. In summary, for single-site data protection, Scality's replication and EC data protection mechanisms can provide very high-levels of data durability, with the ability to trade-off performance and space characteristics for different data types.

Replication and EC may be combined, even on a single Connector, by configuring a policy for the connector to store objects below a certain size threshold with a replication CoS, but files above the file size limit with a specific EC schema. This allows the application to simply store objects without worrying about the optimal storage strategy per object, with the system managing that automatically.

RING does not employ traditional RAID based data protection techniques. While RAID has served the industry well in legacy NAS and SAN systems, industry experts have written at large about the inadequacies of classical RAID technologies when employed on high-density disk drives, in capacity-optimized and distributed storage systems. These deficiencies include higher probabilities of data loss due to long RAID rebuild times, and the ability to protect against only a limited set of failure conditions (for example, only two simultaneous disk failures per RAID6 group). Further information and reading on the limitations of RAID as a data protection mechanism on high-capacity disk drives is widely available

Self-Healing

RING provides self-healing processes that monitor and automatically resolve component failures. This includes the ability to rebuild missing data chunks due to disk drive or server failures, rebalance data when nodes leave and join the RING, and to proxy requests around component failures. In the event a disk drive or even a full server fails, background rebuild operations are spawned to restore the missing object data from its surviving replicas or EC chunks. The rebuild process completes when it has restored the original Class of Service – either the full number of replicas or the original number of EC data and parity chunks. A local disk failure can also be repaired quickly on a node (distinct from a full distributed rebuild), through the use of an in-memory key map maintained on each node. Nodes are also responsible for automatically detecting mismatches in their own Keyspace, rebalancing keys and for establishing and removing proxies during node addition and departure operations. Self-healing provides the RING with the resiliency required to maintain data availability and durability in the face of the expected wide set of failure conditions, including multiple simultaneous component failures at the hardware and software process levels.

To optimize rebuilds as well as mainline IO performance during rebuilds, RING utilizes the distributed power of the entire storage pool. The parallelism of the underlying architecture pays dividends by eliminating any central bottlenecks that might otherwise limit performance or cause contention between servicing application requests, and normal background operations such as rebuilds, especially when the system is under load. To further optimize rebuild operations, the system will only repair the affected object data, not the entire set of disk blocks,

as is commonly the case in RAID arrays. Rebuilds are distributed across multiple servers and disks in the system, to utilize the aggregate processing power and available IO of multiple resources in parallel, rather than serializing the rebuilds onto a single disk drive.

By leveraging the entire pool, the impact of rebuilding data stored either with replication or EC is minimized since there will be relatively small degrees of overlap between disks involved in servicing data requests, and those involved in the rebuilds.

Scality RING Multi-Site Deployments

To support multi datacenter deployments with site protection and complete data consistency between all sites, the RING natively supports a stretched (synchronous) deployment mode across sites. In this mode, a single logical RING is deployed across multiple data centers, with all nodes participating in the standard RING protocols as if they were local to one site.

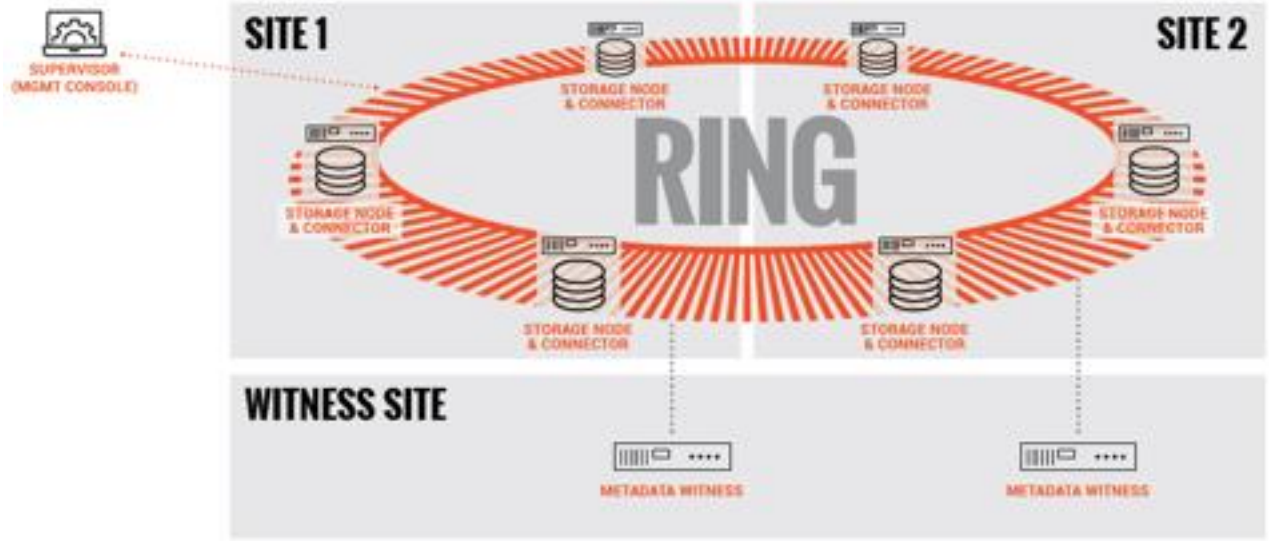
When a stretched RING is deployed with EC, it provides multiple benefits including full site-level failure protection, active/active access from both data centers, and dramatically reduced storage overhead compared to mirrored RINGs. An EC schema for a three-site stretched RING of EC (7,5) would provide protection against one complete site failure, plus one additional disk/server failure in another site, with approximately 70 percent space overhead. This compares favorably to a replication policy that might require 300-400 percent space overhead, for similar levels of protection across these sites.

File System (SOFS) Multi-Site Geo-Distribution

The Scality RING can be stretched across 2 to 3 sites within a Metro-Area Network (MAN) to provide full site failover, should the latency between the several sites go above 10ms. The stretched architecture provides zero RTO and RPO since the failover is automatized. This is the same for the failback procedure since when the lost site is recovered, the system will recover automatically the data. For the two-site stretched architecture only and to manage the mitigation between the 2 sites, 2 witness servers will be needed.

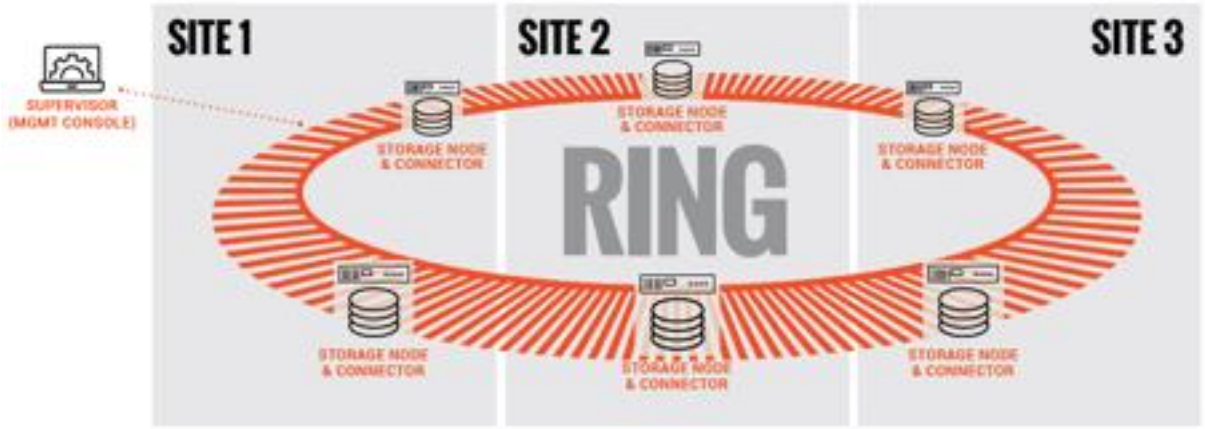
The 2 stretched sites + witness is an Active / Active replication system based on a synchronous replication.

Figure 13 SOFS – Two-site Stretched



The 3 stretched sites is an Active / Active replication system based on a synchronous replication.

Figure 14 SOFS – Three-Site Stretched

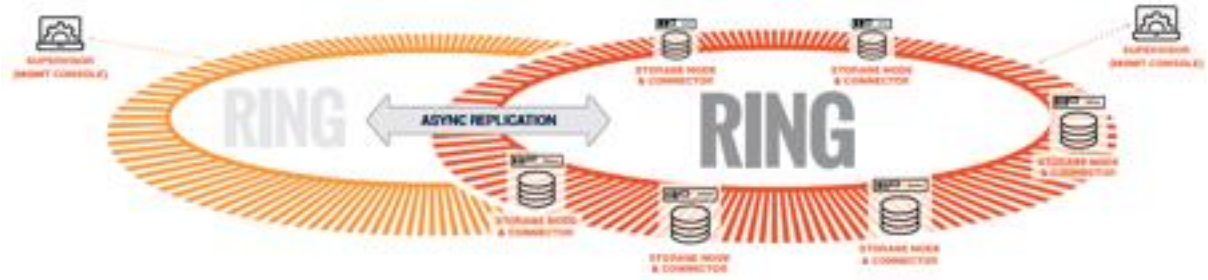


For high latency between sites, Scality supports SOFS 2 Sites Full Asynchronous replication mechanism at Scale to enable the replication of massive amount of file data across the 2 sites. Scality also supports a Full Diff mechanism that can compare at scale the content of the 2 sites to make sure the data are effectively replicated. Should one site be fully lost, Scality provides a mechanism to fully reconstruct the lost site.

To manage replication burst, Scality integrates a back-pressure system to be sure your production network link won't be overloaded by the replication itself and in the same time will respect the RPO defined during the setup of the installation. This feature enables the Disaster Recovery (DR) feature by providing Failover and Failback system to recover in case of partial or full loss.

The 2 sites with high latency between them is an Active / Passive replication system based on an asynchronous replication.

Figure 15 SOFS – Two-Site Asynchronous Replication

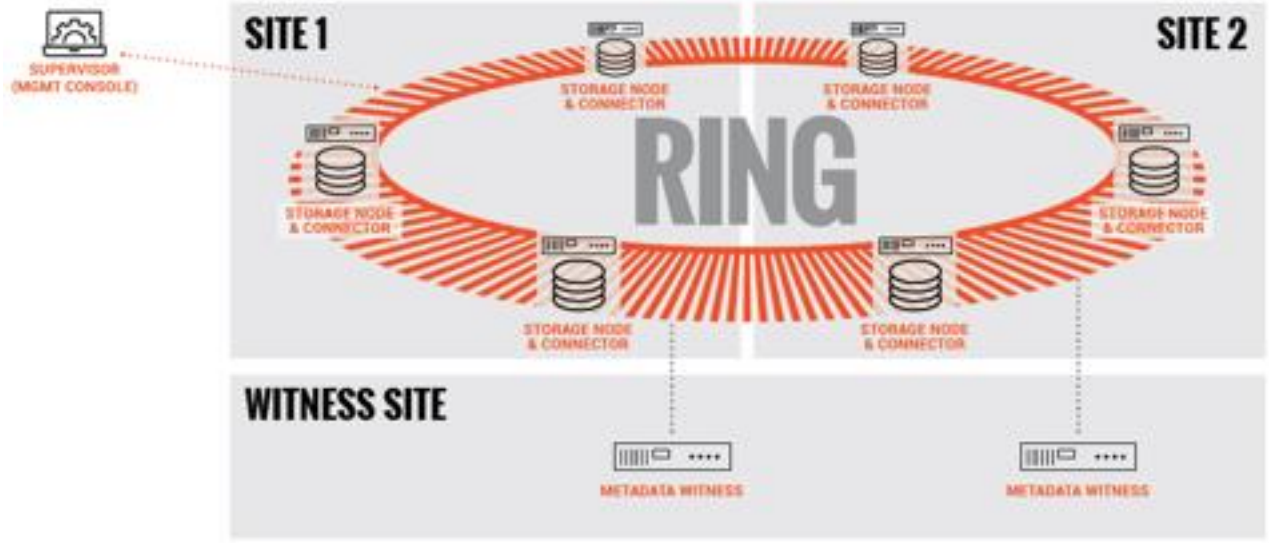


S3 Object Multi-Site Geo-Distribution

The same multi-site architectures are supported for S3 as with SOFS, both synchronous and asynchronous. The first one with a stretched solution on two and three sites with no RPO and no RTO. As for SOFS, a stretched architecture is available within a MAN to provide full site failover, should the latency between the several sites goes above 10ms. For the two-site stretched architecture only and to manage the mitigation between the 2 sites, 2 witness servers will be needed.

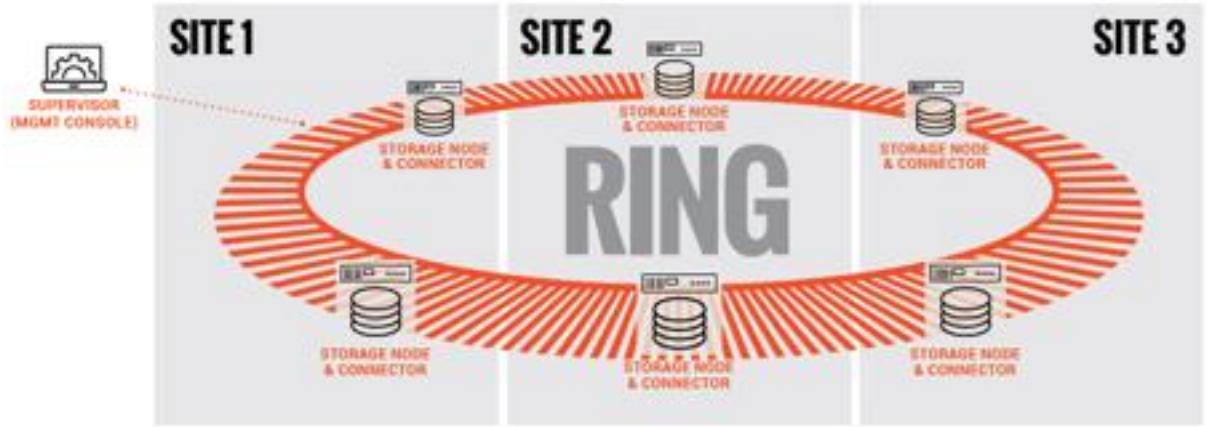
The 2 stretched sites + witness is an Active / Active replication system based on a synchronous replication.

Figure 16 S3 – Two-Site Stretched



The 3 stretched sites is an Active / Active replication system based on a synchronous replication.

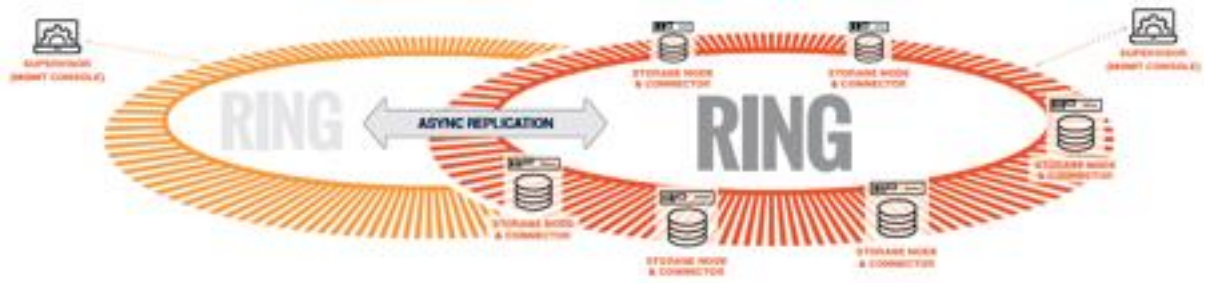
Figure 17 S3 – Three-Site Stretched



For high latency between sites (such as on a Wide Area Network - WAN), Scality supports the S3 2 Sites Full Asynchronous replication mechanism at Scale to enable the replication of massive amount of data across the 2 sites. This system is based on the S3 CRR (Cross-Region Replication) design to replicate a bucket between 2 sites. For site replication, Scality developed its own system to support site replication instead of just bucket. This feature enables the Disaster Recovery (DR) feature by providing Failover and Failback system to recover in case of partial or fully (flooding, fire) lost.

The 2 sites with high latency between them is an Active / Passive replication system based on an asynchronous replication.

Figure 18 S3 – Two-Site Asynchronous Replication



Solution Design

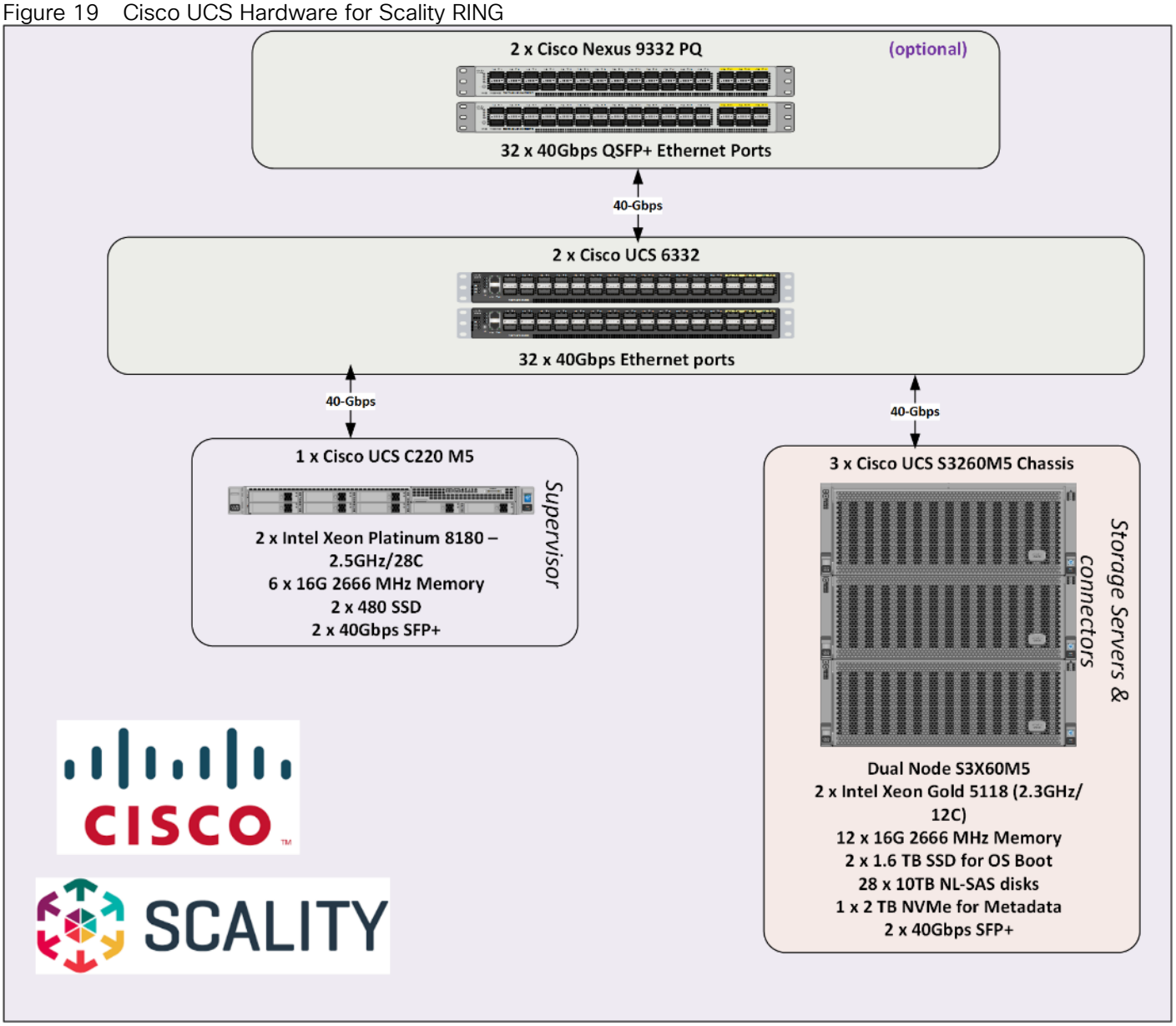
Deployment Architecture

The reference architecture use case provides a comprehensive, end-to-end example of designing and deploying the Scality RING on Cisco UCS S3260. This CVD describes the architecture and design of a Scality Scale-out object Storage and file system solution on three Cisco UCS S3260 Storage Server Chassis', each with two Cisco UCS S3260 M5 nodes configured as Storage servers and one Cisco UCS C220 M5 Rack Server as Supervisor node. The whole solution is connected to a pair of Cisco UCS 6332 Fabric Interconnects and to a pair of upstream network switches Cisco Nexus 9332PQ.

The detailed configuration is as follows:

- 2 x Cisco Nexus 9332PQ Switches
- 2 x Cisco UCS 6332 Fabric Interconnects
- 3 x Cisco UCS S3260 Storage Servers with 2 x Cisco UCS C3260 M5 server nodes each
- 1 x Cisco UCS C220 M5 Rack Servers

Figure 19 Cisco UCS Hardware for Scality RING



Solution Overview

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

Software Distributions and Versions

The required software distribution versions are listed below in Error! Reference source not found.

Table 1 Software Versions

Layer	Component	Version or Release
Storage (Chassis) UCS S3260	Chassis Management Controller	4.0(1x)
	Shared Adapter	4.3(1b)

Layer	Component	Version or Release
Compute (Server Nodes) UCS S3X60 M5	BIOS	S3260M5.4.0.1b
	CIMC Controller	4.0(1x)
Compute (Rack Server) C220 M5S	BIOS	C220M5.4.0.1c
	CIMC Controller	4.0(1x)
Network 6332 Fabric Interconnect	UCS Manager	4.0(1x)
	Kernel	5.0(3)N2(4.01x)
	System	5.0(3)N2(4.01x)
Network Nexus 9332PQ	BIOS	07.49
	NXOS	7.0(3)I3(1)
Software	Red Hat Enterprise Linux Server	7.5 (x86_64)
	Scality RING	7.4

Please contact your Cisco representative for country specific information.

Hardware Requirement and Bill of Materials

This section contains the hardware components in the lab for CVD testing.

Table 2 Bill of Materials

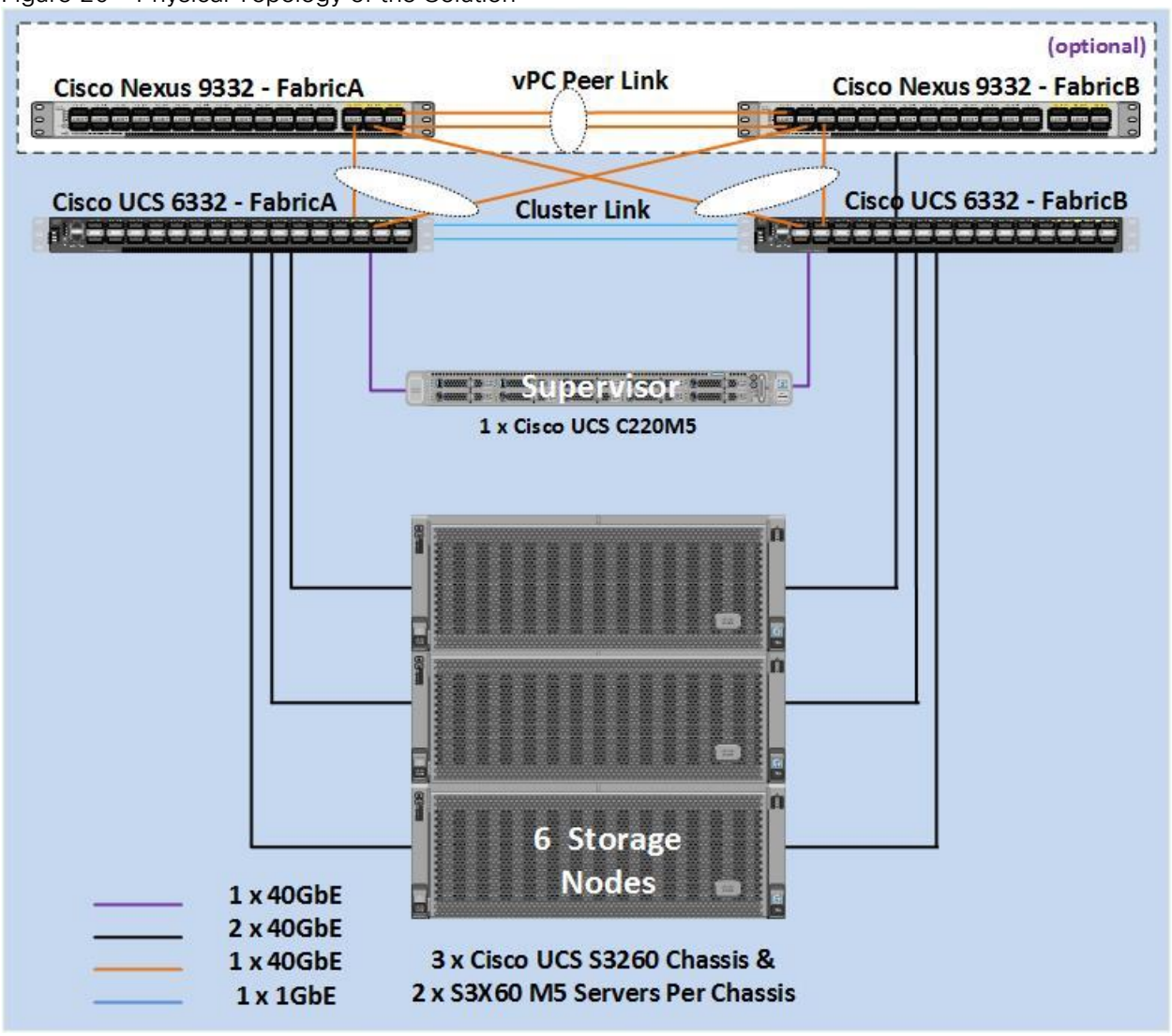
Component	Model	Quantity	Comments
Scality Storage Node (also the Connector Node)	Cisco UCS S3260 M5 Chassis	3	2 x UCS S3X60 M5 Server Nodes per Chassis (Total = 6nodes) Per Server Node 2 x Intel Xeon Gold 5118 (2.3GHz/12cores), 192 GB RAM Cisco 12G SAS RAID Controller 2 x 1.6TB SSD for OS 28 x 10TB HDDs for Data, 1 x 2 TB of NVMe's. Dual-port 40 Gbps VIC
Scality Supervisor	Cisco UCS C220 M5 Rack server	1	2 x Intel Xeon Platinum 8180 (2.5GHz/28 Cores), 96GB RAM Cisco 12G SAS RAID Controller 2 x 480GB SSD for OS Dual-port 40 Gbps VIC

Component	Model	Quantity	Comments
UCS Fabric Interconnects	Cisco UCS 6332 Fabric Interconnects	2	
Switches	Cisco Nexus 9332PQ Switches	2	

Physical Topology and Configuration

The following sections describe the physical design of the solution and the configuration of each component.

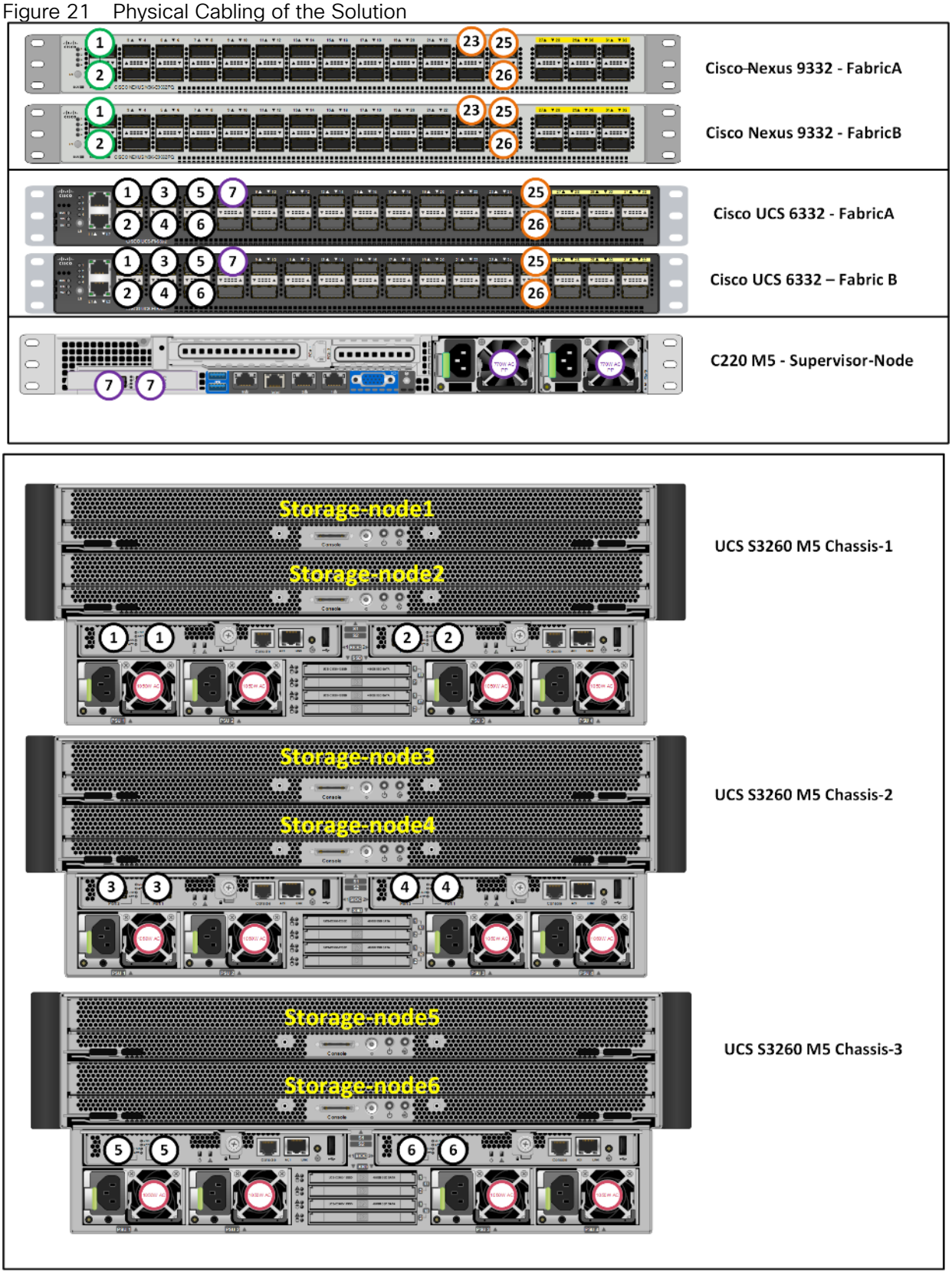
Figure 20 Physical Topology of the Solution



The connectivity of the solution is based on 40 Gbit. All components are connected together via 40 QSFP cables. Between both Cisco Nexus 9332PQ switches are 2 x 40 Gbit cabling. Each Cisco UCS 6332 Fabric Interconnect is connected via 2 x 40 Gbit to each Cisco UCS 9332PQ switch, and each Cisco UCS C220 M5 is connected via 1 x 40 Gbit and each Cisco UCS S3260 M5 server is connected with 2 x 40 Gbit cable 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 will be covered during validation and testing.

Figure 21 Physical Cabling of the Solution



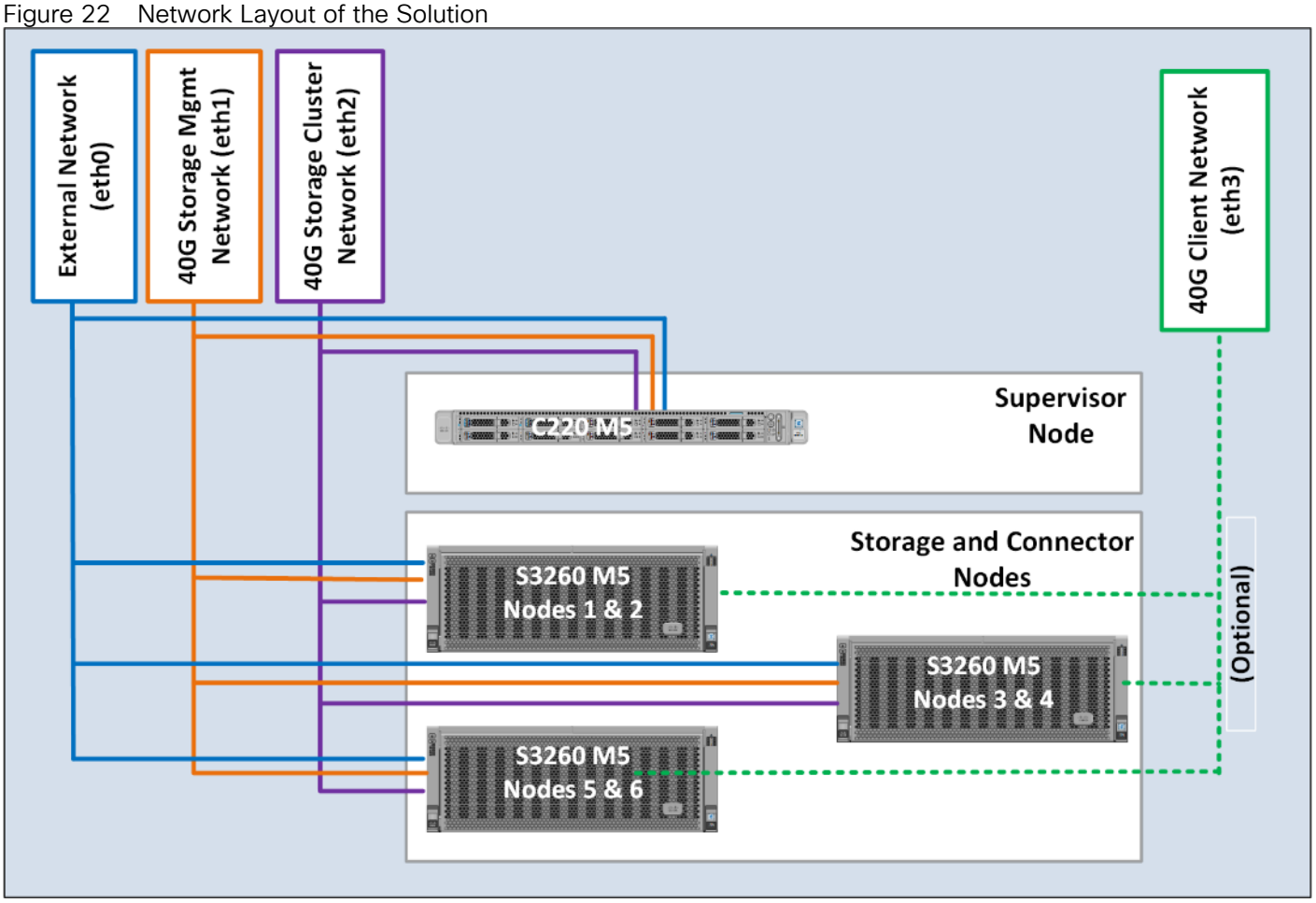
The exact cabling for the Cisco UCS S3260 Storage Server, Cisco UCS C220 M5, and the Cisco UCS 6332 Fabric Interconnect is illustrated in Table 3 .

Table 3 Cabling Information

Local Device	Local Port	Connection	Remote Device	Remote Port	Cable
Cisco Nexus 9332 Switch A	Eth1/1	40GbE	Cisco Nexus 9372 Switch B	Eth1/1	QSFP-H40G-CU1M
	Eth1/2	40GbE	Cisco Nexus 9372 Switch B	Eth1/2	QSFP-H40G-CU1M
	Eth1/25	40GbE	Cisco UCS Fabric Interconnect A	Eth1/25	QSFP-H40G-CU1M
	Eth1/26	40GbE	Cisco UCS Fabric Interconnect B	Eth1/25	QSFP-H40G-CU1M
	Eth1/23	40GbE	Top of Rack (Upstream Network)	Any	QSFP+ 4SFP10G
	MGMT0	1GbE	Top of Rack (Management)	Any	1G RJ45
Cisco Nexus 9332 Switch B	Eth1/1	40GbE	Cisco Nexus 9372 Switch B	Eth1/1	QSFP-H40G-CU1M
	Eth1/2	40GbE	Cisco Nexus 9372 Switch B	Eth1/2	QSFP-H40G-CU1M
	Eth1/25	40GbE	Cisco UCS Fabric Interconnect A	Eth1/26	QSFP-H40G-CU1M
	Eth1/26	40GbE	Cisco UCS Fabric Interconnect B	Eth1/26	QSFP-H40G-CU1M
	Eth1/23	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	S3260 Chassis 1 - SIOC 1 (right)	port 1	QSFP-H40G-CU3M
	Eth1/2	40GbE	S3260 Chassis 1 - SIOC 2 (left)	port 1	QSFP-H40G-CU3M
	Eth1/3	40GbE	S3260 Chassis 2 - SIOC 1 (right)	port 1	QSFP-H40G-CU3M
	Eth1/4	40GbE	S3260 Chassis 2 - SIOC 2 (left)	port 1	QSFP-H40G-CU3M
	Eth1/5	40GbE	S3260 Chassis 3 - SIOC 1 (right)	port 1	QSFP-H40G-CU3M
	Eth1/6	40GbE	S3260 Chassis 3 - SIOC 2 (left)	port 1	QSFP-H40G-CU3M

Local Device	Local Port	Connection	Remote Device	Remote Port	Cable
	Eth1/7	40GbE	C220 M5 - Server1 - VIC1387	VIC - Port 1	QSFP-H40G-CU1M
	Eth1/25	40GbE	Nexus 9332 A	Eth 1/25	QSFP-H40G-CU1M
	Eth1/26	40GbE	Nexus 9332 B	Eth 1/25	QSFP-H40G-CU1M
	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 Interconnect B	Eth1/1	40GbE	S3260 Chassis 1 - SIOC 1 (right)	port 2	QSFP-H40G-CU3M
	Eth1/2	40GbE	S3260 Chassis 1 - SIOC 2 (left)	port 2	QSFP-H40G-CU3M
	Eth1/3	40GbE	S3260 Chassis 2 - SIOC 1 (right)	port 2	QSFP-H40G-CU3M
	Eth1/4	40GbE	S3260 Chassis 2 - SIOC 2 (left)	port 2	QSFP-H40G-CU3M
	Eth1/5	40GbE	S3260 Chassis 3 - SIOC 1 (right)	port 2	QSFP-H40G-CU3M
	Eth1/6	40GbE	S3260 Chassis 3 - SIOC 2 (left)	port 2	QSFP-H40G-CU3M
	Eth1/7	40GbE	C220 M5 - Server1 - VIC1387	VIC -Port2	QSFP-H40G-CU1M
	Eth1/25	40GbE	Nexus 9332 A	Eth 1/26	QSFP-H40G-CU1M
	Eth1/26	40GbE	Nexus 9332 B	Eth 1/26	QSFP-H40G-CU1M
	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

Figure 22 Network Layout of the Solution

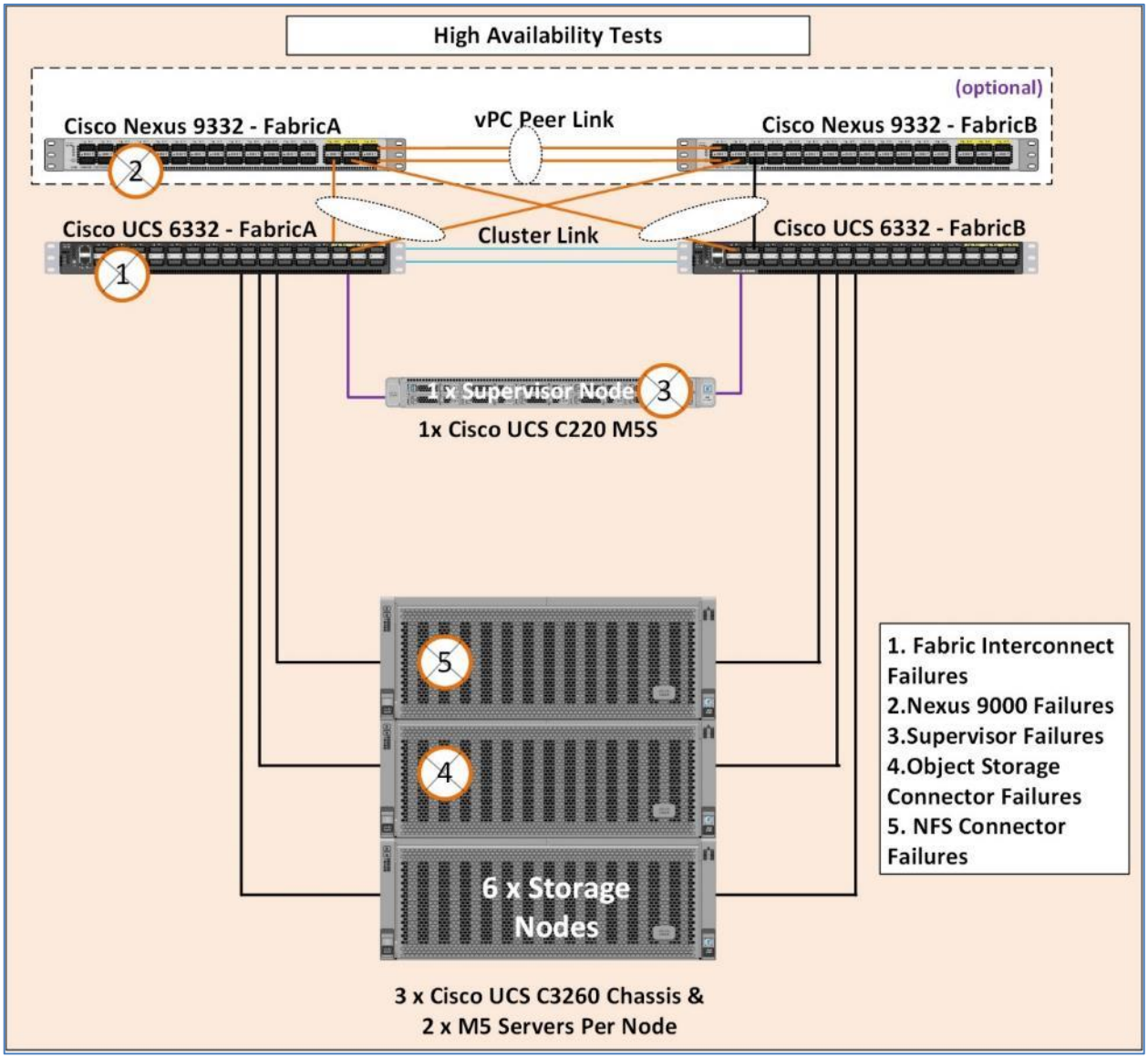


High Availability

As part of hardware and software resiliency, the following tests are being conducted on the test bed. The results of the tests will be included in the deployment guide.

1. Fabric Interconnect failures
2. Nexus 9000 failures
3. Supervisor failures
4. S3 connector failures
5. NFS connector failures

Figure 23 High Availability



Deployment Hardware and Software

Configuration of Nexus 9332PQ Switch A and B

Both Cisco UCS Fabric Interconnect A and B are connected to two Cisco Nexus 9332PQ switches for connectivity to Upstream Network. The following sections describe the setup of both Cisco Nexus 9332PQ switches.

Initial Setup of Nexus 9332PQ Switch A and B

To configure Switch A, connect a Console to the Console port of each switch, power on the switch and complete the following 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]:
  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-Fab-A
  Continue with Out-of-band (mgmt0) management configuration? (yes/no) [y]: yes
    Mgmt0 IPv4 address : 192.168.10.103
    Mgmt0 IPv4 netmask : 255.255.255.0
  Configure the default gateway? (yes/no) [y]: yes
    IPv4 address of the default gateway : 192.168.10.1
  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 : 192.168.10.2
  Configure default interface layer (L3/L2) [L3]: L2
  Configure default switchport interface state (shut/noshut) [shut]: shut
  Configure CoPP system profile (strict/moderate/lenient/dense) [strict]:
The following configuration will be applied:
  password strength-check
  switchname N9k-Fab-A
vrf context management
ip route 0.0.0.0/0 192.168.10.1
exit
  no feature telnet
  ssh key rsa 1024 force
  feature ssh
  ntp server 192.168.10.2
  no system default switchport
  system default switchport shutdown
  copp profile strict
interface mgmt0
ip address 192.168.10.103 255.255.255.0
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.

User Access Verification
```

```
N9k-Fab-A login:
```

Repeat these steps for the Nexus 9332PQ Switch B with the exception of configuring a different IPv4 management address 192.168.10.104 as described in step 7.

Enable Features on Nexus 9332PQ 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 complete the following steps on both Switch A and B:

Switch A

```
N9k-Fab-A# configure terminal

Enter configuration commands, one per line. End with CNTL/Z.

N9k-Fab-A(config)# feature udld

N9k-Fab-A(config)# feature interface-vlan

N9k-Fab-A(config)# feature hsrp

N9k-Fab-A(config)# feature lacp

N9k-Fab-A(config)# feature vpc

N9k-Fab-A(config)# system jumbomtu 9216

N9k-Fab-A(config)# exit

N9k-Fab-A(config)# copy running-config startup-config
```

Switch B

```
N9k-Fab-B# configure terminal

Enter configuration commands, one per line. End with CNTL/Z.

N9k-Fab-B(config)# feature udld

N9k-Fab-B(config)# feature interface-vlan

N9k-Fab-B(config)# feature hsrp

N9k-Fab-B(config)# feature lacp

N9k-Fab-B(config)# feature vpc

N9k-Fab-B(config)# system jumbomtu 9216

N9k-Fab-B(config)# exit

N9k-Fab-B(config)# copy running-config startup-config
```

Configuring VLANs on Nexus 9332PQ 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, complete the following steps on Switch A and Switch B:

Switch A

```
N9k-Fab-A# config terminal

Enter configuration commands, one per line. End with CNTL/Z.
```

```
N9k-Fab-A(config)# vlan 10
N9k-Fab-A(config-vlan)# name Storage-Management
N9k-Fab-A(config-vlan)# no shut
N9k-Fab-A(config-vlan)# exit
N9k-Fab-A(config)# vlan 20
N9k-Fab-A(config-vlan)# name Storage-Cluster
N9k-Fab-A(config-vlan)# no shut
N9k-Fab-A(config-vlan)# exit
N9k-Fab-A(config)# vlan 79
N9k-Fab-A(config-vlan)# name External-Mgmt
N9k-Fab-A(config-vlan)# no shut
N9k-Fab-A(config-vlan)# exit
N9k-Fab-A(config)# vlan 30
N9k-Fab-A(config-vlan)# name Client-Network
N9k-Fab-A(config-vlan)# no shut
N9k-Fab-A(config-vlan)# exit

N9k-Fab-A(config)# interface vlan10
N9k-Fab-A(config-if)# description Storage-Mgmt
N9k-Fab-A(config-if)# no shutdown
N9k-Fab-A(config-if)# no ip redirects
N9k-Fab-A(config-if)# ip address 192.168.10.253/24
N9k-Fab-A(config-if)# no ipv6 redirects
N9k-Fab-A(config-if)# hsrp version 2
N9k-Fab-A(config-if)# hsrp 10
N9k-Fab-A(config-if-hsrp)# preempt
N9k-Fab-A(config-if-hsrp)# priority 10
N9k-Fab-A(config-if-hsrp)# ip 192.168.10.1
N9k-Fab-A(config-if-hsrp)# exit

N9k-Fab-A(config-if)# exit
```

```
N9k-Fab-A(config)# interface vlan20
N9k-Fab-A(config-if)# description Storage-Cluster
N9k-Fab-A(config-if)# no shutdown
N9k-Fab-A(config-if)# no ip redirects
N9k-Fab-A(config-if)# ip address 192.168.20.253/24
N9k-Fab-A(config-if)# no ipv6 redirects
N9k-Fab-A(config-if)# hsrp version 2
N9k-Fab-A(config-if)# hsrp 20
N9k-Fab-A(config-if-hsrp)# preempt
N9k-Fab-A(config-if-hsrp)# priority 10
N9k-Fab-A(config-if-hsrp)# ip 192.168.20.1
N9k-Fab-A(config-if-hsrp)# exit
N9k-Fab-A(config)# interface vlan30
N9k-Fab-A(config-if)# description Client-Network
N9k-Fab-A(config-if)# no shutdown
N9k-Fab-A(config-if)# no ip redirects
N9k-Fab-A(config-if)# ip address 192.168.30.253/24
N9k-Fab-A(config-if)# no ipv6 redirects
N9k-Fab-A(config-if)# hsrp version 2
N9k-Fab-A(config-if)# hsrp 30
N9k-Fab-A(config-if-hsrp)# preempt
N9k-Fab-A(config-if-hsrp)# priority 10
N9k-Fab-A(config-if-hsrp)# ip 192.168.30.1
N9k-Fab-A(config-if-hsrp)# exit
N9k-Fab-A(config-if)# exit
```

Switch B

```
N9k-Fab-B# config terminal
Enter configuration commands, one per line. End with CNTL/Z.
N9k-Fab-B(config)# vlan 10
N9k-Fab-B(config-vlan)# name Storage-Management
N9k-Fab-B(config-vlan)# no shut
N9k-Fab-B(config-vlan)# exit
N9k-Fab-B(config)# vlan 20
```

```
N9k-Fab-B(config-vlan)# name Storage-Cluster
N9k-Fab-B(config-vlan)# no shut
N9k-Fab-B(config-vlan)# exit
N9k-Fab-B(config)# vlan 79
N9k-Fab-B(config-vlan)# name External-Mgmt
N9k-Fab-B(config-vlan)# no shut
N9k-Fab-B(config-vlan)# exit
N9k-Fab-B(config)# vlan 30
N9k-Fab-B(config-vlan)# name Client-Network
N9k-Fab-B(config-vlan)# no shut
N9k-Fab-B(config-vlan)# exit
N9k-Fab-B(config)# interface vlan10
N9k-Fab-B(config-if)# description Storage-Mgmt
N9k-Fab-B(config-if)# no ip redirects
N9k-Fab-B(config-if)# ip address 192.168.10.254/24
N9k-Fab-B(config-if)# no ipv6 redirects
N9k-Fab-B(config-if)# hsrp version 2
N9k-Fab-B(config-if)# hsrp 10
N9k-Fab-B(config-if-hsrp)# preempt
N9k-Fab-B(config-if-hsrp)# priority 5
N9k-Fab-B(config-if-hsrp)# ip 192.168.10.1
N9k-Fab-B(config-if-hsrp)# exit
N9k-Fab-B(config-if)# exit

N9k-Fab-B(config)# interface vlan20
N9k-Fab-B(config-if)# description Storage-Cluster
N9k-Fab-B(config-if)# no ip redirects
N9k-Fab-B(config-if)# ip address 192.168.20.254/24
N9k-Fab-B(config-if)# no ipv6 redirects
N9k-Fab-B(config-if)# hsrp version 2
N9k-Fab-B(config-if)# hsrp 20
N9k-Fab-B(config-if-hsrp)# preempt
```

```
N9k-Fab-B(config-if-hsrp)# priority 5
N9k-Fab-B(config-if-hsrp)# ip 192.168.20.1
N9k-Fab-B(config-if-hsrp)# exit
N9k-Fab-B(config)# interface vlan30
N9k-Fab-B(config-if)# description Storage-Cluster
N9k-Fab-B(config-if)# no ip redirects
N9k-Fab-B(config-if)# ip address 192.168.30.254/24
N9k-Fab-B(config-if)# no ipv6 redirects
N9k-Fab-B(config-if)# hsrp version 2
N9k-Fab-B(config-if)# hsrp 30
N9k-Fab-B(config-if-hsrp)# preempt
N9k-Fab-B(config-if-hsrp)# priority 5
N9k-Fab-B(config-if-hsrp)# ip 192.168.30.1
N9k-Fab-B(config-if-hsrp)# exit
N9k-Fab-B(config-if)# exit
N9k-Fab-B(config)# copy running-config startup-config
```

Configure vPC and Port Channels on Nexus C9332PQ Switch A and B

To enable vPC and Port Channels on both Switch A and B, complete the following steps:

vPC and Port Channels for Peerlink on Switch A

```
N9k-Fab-B# config terminal
Enter configuration commands, one per line. End with CNTL/Z.
N9k-Fab-A(config)# vpc domain 2
N9k-Fab-A(config-vpc-domain)# peer-keepalive destination 192.168.10.104
Note:
 --------:: Management VRF will be used as the default VRF ::--------
N9k-Fab-A(config-vpc-domain)# peer-gateway
N9k-Fab-A(config-vpc-domain)# exit

N9k-Fab-A(config)# interface port-channel 1
N9k-Fab-A(config-if)# description vPC peerlink for N9k-Fab-A and N9k-Fab-B
N9k-Fab-A(config-if)# switchport
N9k-Fab-A(config-if)# switchport mode trunk
```

```
N9k-Fab-A(config-if)# spanning-tree port type network

N9k-Fab-A(config-if)# speed 40000

N9k-Fab-A(config-if)# vpc peer-link

Please note that spanning tree port type is changed to "network" port type on
vPC peer-link.

This will enable spanning tree Bridge Assurance on vPC peer-link provided the
STP Bridge Assurance

(which is enabled by default) is not disabled.

N9k-Fab-A(config-if)# exit


N9k-Fab-A(config)# interface ethernet 1/1

N9k-Fab-A(config-if)# description connected to peer N9k-Fab-B port 1

N9k-Fab-A(config-if)# switchport

N9k-Fab-A(config-if)# switchport mode trunk

N9k-Fab-A(config-if)# speed 40000

N9k-Fab-A(config-if)# channel-group 1 mode active

N9k-Fab-A(config-if)# exit


N9k-Fab-A(config)# interface ethernet 1/2

N9k-Fab-A(config-if)# description connected to peer N9k-Fab-B port 2

N9k-Fab-A(config-if)# switchport

N9k-Fab-A(config-if)# switchport mode trunk

N9k-Fab-A(config-if)# speed 40000

N9k-Fab-A(config-if)# channel-group 1 mode active

N9k-Fab-A(config-if)# exit

N9k-Fab-A(config)# copy running-config startup-config
```

vPC and Port Channels for Peerlink on Switch B

```
N9k-Fab-B# config terminal

Enter configuration commands, one per line. End with CNTL/Z.

N9k-Fab-B(config)# vpc domain 2

N9k-Fab-B(config-vpc-domain)# peer-keepalive destination 192.168.10.103

Note:
```

```
 --------:: Management VRF will be used as the default VRF ::--------
N9k-Fab-B(config-vpc-domain)# peer-gateway
N9k-Fab-B(config-vpc-domain)# exit

N9k-Fab-B(config)# interface port-channel 1
N9k-Fab-B(config-if)# description vPC peerlink for N9k-Fab-A and N9k-Fab-B
N9k-Fab-B(config-if)# switchport
N9k-Fab-B(config-if)# switchport mode trunk
N9k-Fab-B(config-if)# spanning-tree port type network
N9k-Fab-B(config-if)# speed 40000
N9k-Fab-B(config-if)# vpc peer-link
Please note that spanning tree port type is changed to "network" port type on
vPC peer-link.
This will enable spanning tree Bridge Assurance on vPC peer-link provided the
STP Bridge Assurance
(which is enabled by default) is not disabled.
N9k-Fab-B(config-if)# exit

N9k-Fab-B(config)# interface ethernet 1/1
N9k-Fab-B(config-if)# description connected to peer N9k-Fab-A port 1
N9k-Fab-B(config-if)# switchport
N9k-Fab-B(config-if)# switchport mode trunk
N9k-Fab-B(config-if)# speed 40000
N9k-Fab-B(config-if)# channel-group 1 mode active
N9k-Fab-B(config-if)# exit

N9k-Fab-B(config)# interface ethernet 1/2
N9k-Fab-B(config-if)# description connected to peer N9k-Fab-A port 2
N9k-Fab-B(config-if)# switchport
N9k-Fab-B(config-if)# switchport mode trunk
N9k-Fab-B(config-if)# speed 40000
N9k-Fab-B(config-if)# channel-group 1 mode active
N9k-Fab-B(config-if)# exit
```

```
N9k-Fab-B(config)# copy running-config startup-config
```

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

```
N9k-Fab-B# config terminal
Enter configuration commands, one per line. End with CNTL/Z.
N9k-Fab-A(config)# interface port-channel 25
N9k-Fab-A(config-if)# description vPC for UCS FI-A ports 25 to 26
N9k-Fab-A(config-if)# vpc 25
N9k-Fab-A(config-if)# switchport
N9k-Fab-A(config-if)# switchport mode trunk
N9k-Fab-A(config-if)# switchport trunk allowed vlan 10,20,30,79
N9k-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-Fab-A(config-if)# mtu 9216
N9k-Fab-A(config-if)# exit

N9k-Fab-A(config)# interface port-channel 26
N9k-Fab-A(config-if)# description vPC for UCS FI-B ports 25 to 26
N9k-Fab-A(config-if)# vpc 26
N9k-Fab-A(config-if)# switchport
N9k-Fab-A(config-if)# switchport mode trunk
N9k-Fab-A(config-if)# switchport trunk allowed vlan 10,20,30,79
N9k-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-Fab-A(config-if)# mtu 9216
```

```
N9k-Fab-A(config-if)# exit

N9k-Fab-A(config)# interface ethernet 1/25
N9k-Fab-A(config-if-range)# switchport
N9k-Fab-A(config-if-range)# switchport mode trunk
N9k-Fab-A(config-if-range)# description Uplink from UCS FI-A ports 25
N9k-Fab-A(config-if-range)# channel-group 25 mode active
N9k-Fab-A(config-if)# exit

N9k-Fab-A(config)# interface ethernet 1/26
N9k-Fab-A(config-if-range)# switchport
N9k-Fab-A(config-if-range)# switchport mode trunk
N9k-Fab-A(config-if-range)# description Uplink from UCS FI-B ports 26
N9k-Fab-A(config-if-range)# channel-group 26 mode active
N9k-Fab-A(config-if)# exit
N9k-Fab-A(config)# copy running-config startup-config
```

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

```
N9k-Fab-B# config terminal
Enter configuration commands, one per line. End with CNTL/Z.
N9k-Fab-B(config)# interface port-channel 25
N9k-Fab-B(config-if)# description vPC for UCS FI-A ports 25 to 26
N9k-Fab-B(config-if)# switchport
N9k-Fab-B(config-if)# switchport mode trunk
N9k-Fab-B(config-if)# switchport trunk allowed vlan 10,20,30,79
N9k-Fab-B(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-Fab-B(config-if)# vpc 25
N9k-Fab-B(config-if)# mtu 9216
```

```
N9k-Fab-B(config-if)# exit

N9k-Fab-B(config)# interface port-channel 26
N9k-Fab-B(config-if)# description vPC for UCS FI-B ports 25 to 26
N9k-Fab-B(config-if)# switchport
N9k-Fab-B(config-if)# switchport mode trunk
N9k-Fab-B(config-if)# switchport trunk allowed vlan 10,20,30,79
N9k-Fab-B(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-Fab-B(config-if)# vpc 26
N9k-Fab-B(config-if)# mtu 9216
N9k-Fab-B(config-if)# exit

N9k-Fab-B(config)# interface ethernet 1/25
N9k-Fab-B(config-if-range)# switchport
N9k-Fab-B(config-if-range)# switchport mode trunk
N9k-Fab-B(config-if-range)# description Uplink from UCS FI-A ports 25 to 26
N9k-Fab-B(config-if-range)# channel-group 25 mode active
N9k-Fab-B(config-if)# exit

N9k-Fab-B(config)# interface ethernet 1/26
N9k-Fab-B(config-if-range)# switchport
N9k-Fab-B(config-if-range)# switchport mode trunk
N9k-Fab-B(config-if-range)# description Uplink from UCS FI-B ports 25 to 26
N9k-Fab-B(config-if-range)# channel-group 26 mode active
N9k-Fab-B(config-if)# exit
N9k-Fab-B(config)# copy running-config startup-config
```

Verification Check of Nexus C9332PQ Configuration for Switch A and B

Switch A

```
N9k-Fab-B# config terminal
Enter configuration commands, one per line. End with CNTL/Z.
N9k-Fab-A(config)# 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         : 4
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)

vPC Peer-link status
---------------------------------------------------------------------
id   Port   Status Active vlans
--   ----   ------ --------------------------------------------------
1    Po1    up     1,10,20,30,79

vPC status
----------------------------------------------------------------------
id   Port   Status Consistency Reason                     Active vlans
```

```
--   ----   ------ ---------- ------                    ------------
25   Po25   up     success    success                   10,20,30,79

26   Po26   up     success    success                   10,20,30,79

N9k-Fab-A(config)#
N9k-Fab-A(config)# 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
        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
--------------------------------------------------------------------------------
1     Po1(SU)     Eth      LACP      Eth1/1(P)   Eth1/2(P)
25    Po25(SU)    Eth      LACP      Eth1/25(P)
26    Po26(SU)    Eth      LACP      Eth1/26(P)
N9k-Fab-A(config)#
```

Switch B

```
N9k-Fab-B# config terminal
Enter configuration commands, one per line. End with CNTL/Z.
N9k-Fab-B(config)# 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         : 4
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)

vPC Peer-link status
---------------------------------------------------------------------
id   Port   Status Active vlans
--   ----   ------ --------------------------------------------------
1    Po1    up     1,10,20,30,79

vPC status
----------------------------------------------------------------------------
id   Port   Status Consistency Reason                     Active vlans
--   ----   ------ ----------- ------                     ------------
25   Po25   up     success     success                    10,20,30,79

26   Po26   up     success     success                    10,20,30,79

N9k-Fab-B(config)#
N9k-Fab-B(config)# 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
        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
--------------------------------------------------------------------------------
1     Po1(SU)     Eth      LACP      Eth1/1(P)   Eth1/2(P)
25    Po25(SU)    Eth      LACP      Eth1/25(P)
26    Po26(SU)    Eth      LACP      Eth1/26(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, complete the following 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 UCS-**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:  UCS-FI-6332

  Physical Switch Mgmt0 IP address : 192.168.10.101

  Physical Switch Mgmt0 IPv4 netmask : 255.255.255.0

  IPv4 address of the default gateway : 192.168.10.1

  Cluster IPv4 address : 192.168.10.100

  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= UCS-FI-6332

    Enforced Strong Password=no

    Physical Switch Mgmt0 IP Address=192.168.10.101

    Physical Switch Mgmt0 IP Netmask=255.255.255.0

    Default Gateway=192.168.10.1

    Ipv6 value=0


    Cluster Enabled=yes

    Cluster IP Address=192.168.10.100

    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

UCS-FI-6332-A login:
```

Configure Fabric Interconnect B

To configure Fabric Interconnect B, complete the following 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: 192.168.10.101
    Peer Fabric interconnect Mgmt0 IPv4 Netmask: 255.255.255.0
    Cluster IPv4 address          : 192.168.10.100


    Peer FI is IPv4 Cluster enabled. Please Provide Local Fabric Interconnect
Mgmt0 IPv4 Address
  Physical Switch Mgmt0 IP address : 192.168.10.102


  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
UCS-FI-6332-B login:
```

Logging into Cisco UCS Manager

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

1. Open a Web browser and navigate to the Cisco UCS 6332 Fabric Interconnect cluster address.
2. Click the Launch link to download the Cisco UCS Manager software.
3. If prompted to accept security certificates, accept as necessary.
4. 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.

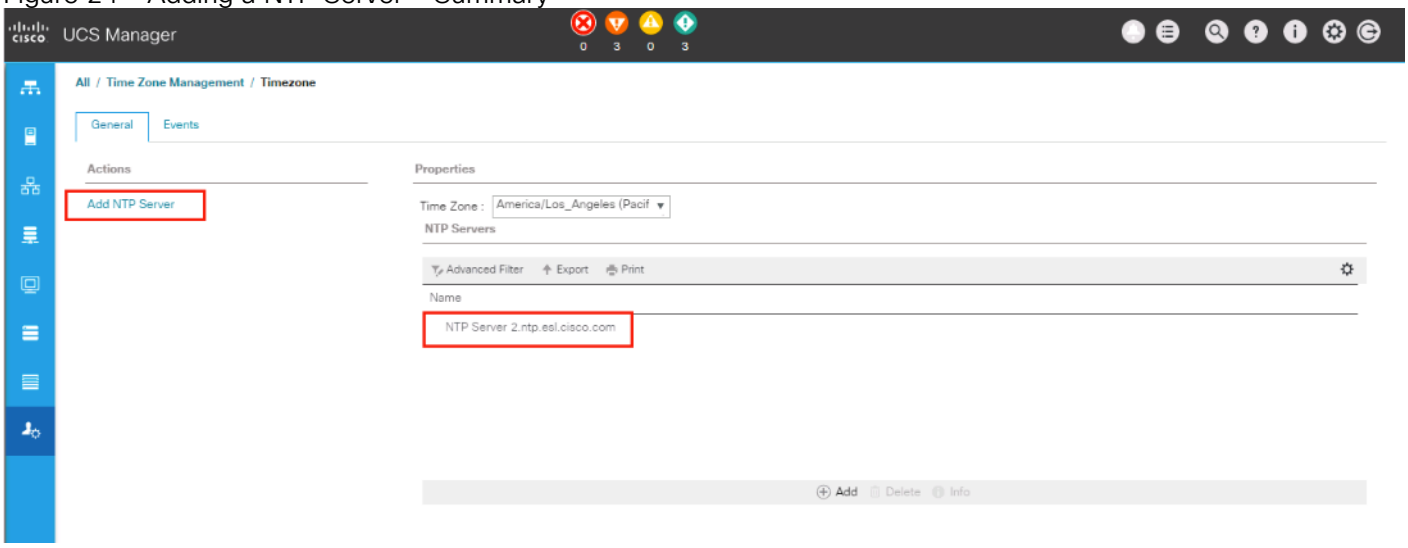
Configure NTP Server

To configure the NTP server for the Cisco UCS environment, complete the following steps:

1. Select Admin tab on the left site.

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 24 Adding a NTP Server – Summary



Initial Base Setup of the Environment

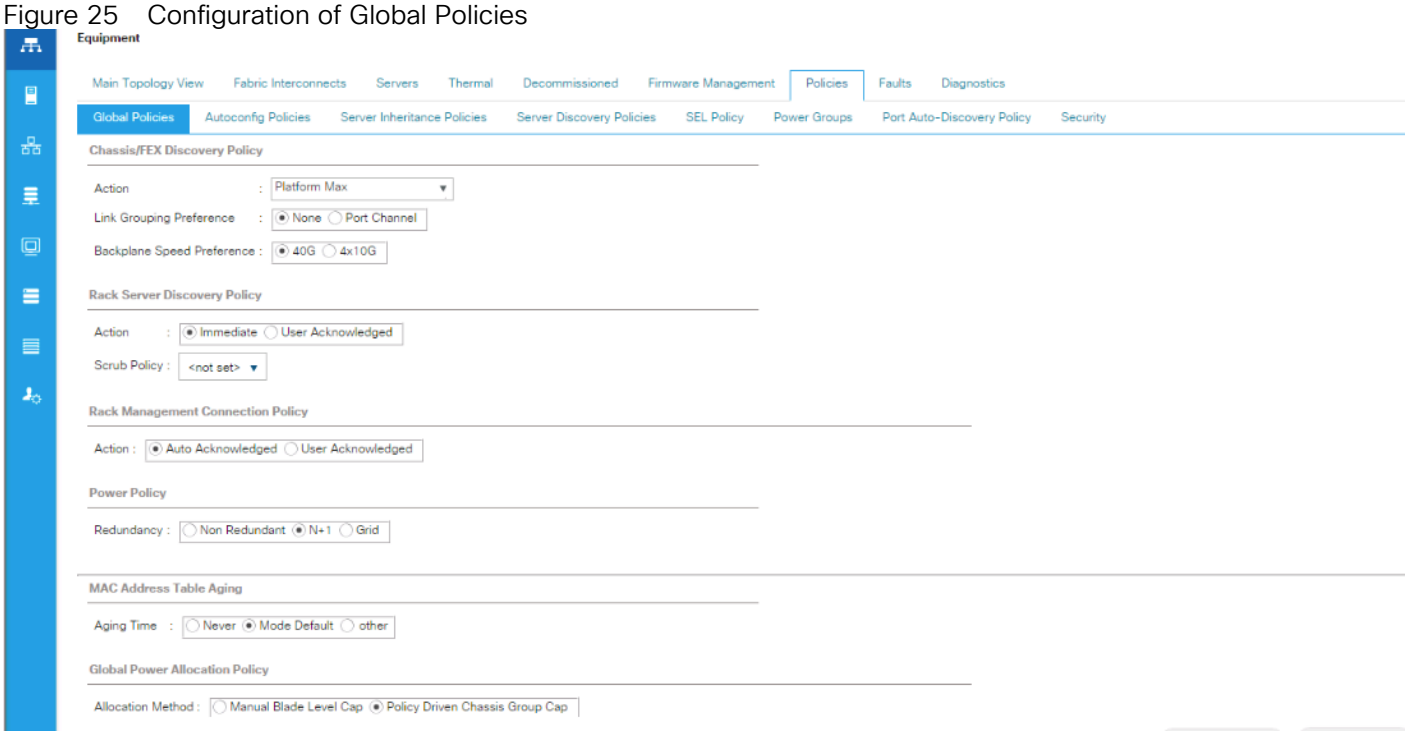
Configure Global Policies

To configure the global policies, complete the following steps:

1. Select the `Equipment` tab on the left site 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 25 Configuration of Global Policies

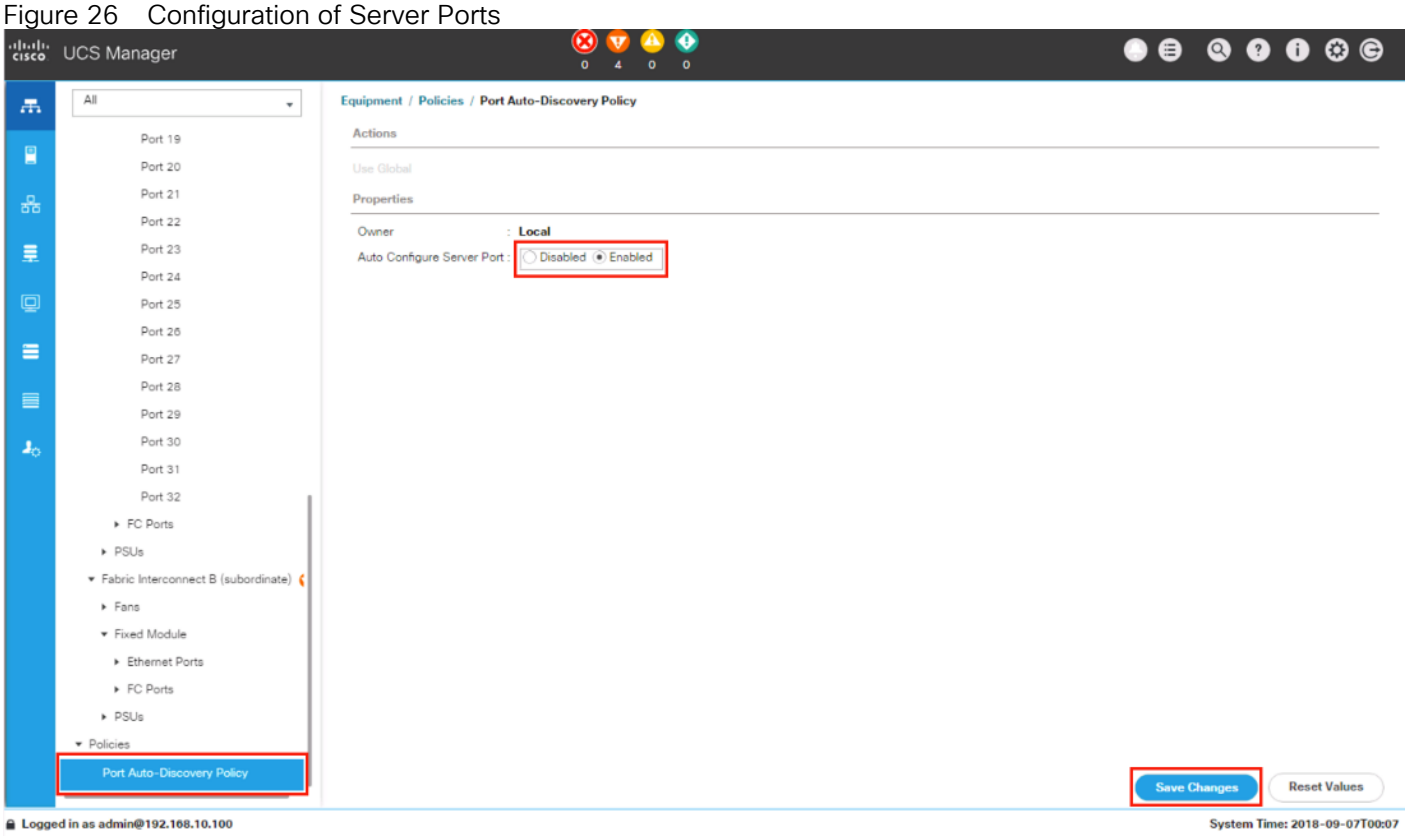


Enable Fabric Interconnect Server Ports

To enable server ports, complete the following steps:

1. Select the **Equipment** tab on the left site.
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 26 Configuration of Server Ports

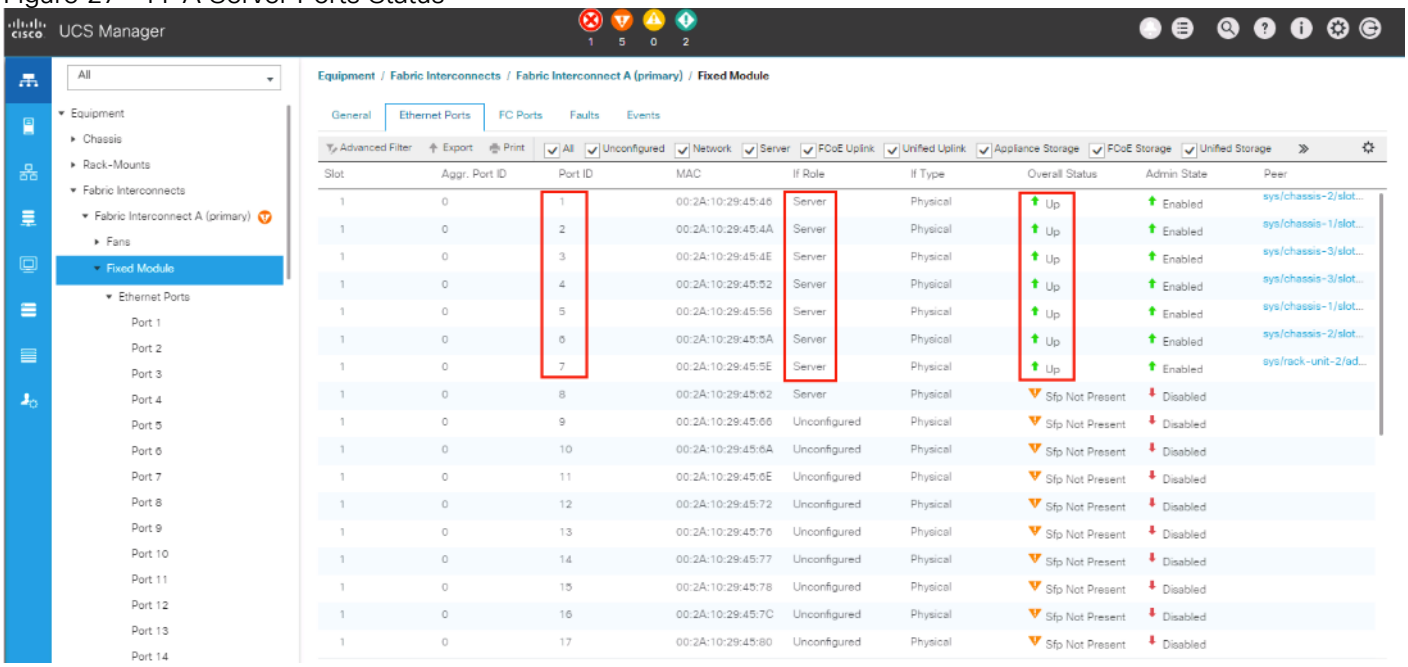


5. Verify the ports Server port on Fabric Interconnect A

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

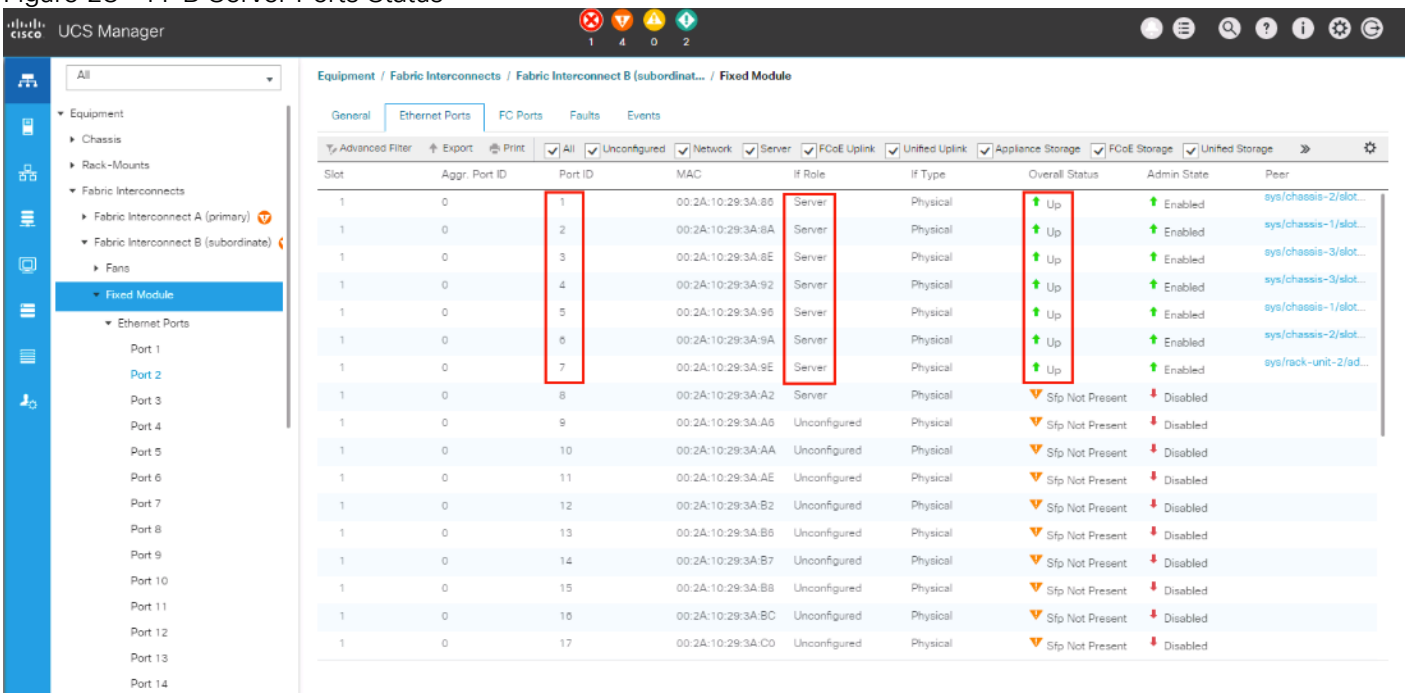
7. Click **`Ethernet Ports`** section.

Figure 27 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 28 FI-B Server Ports Status

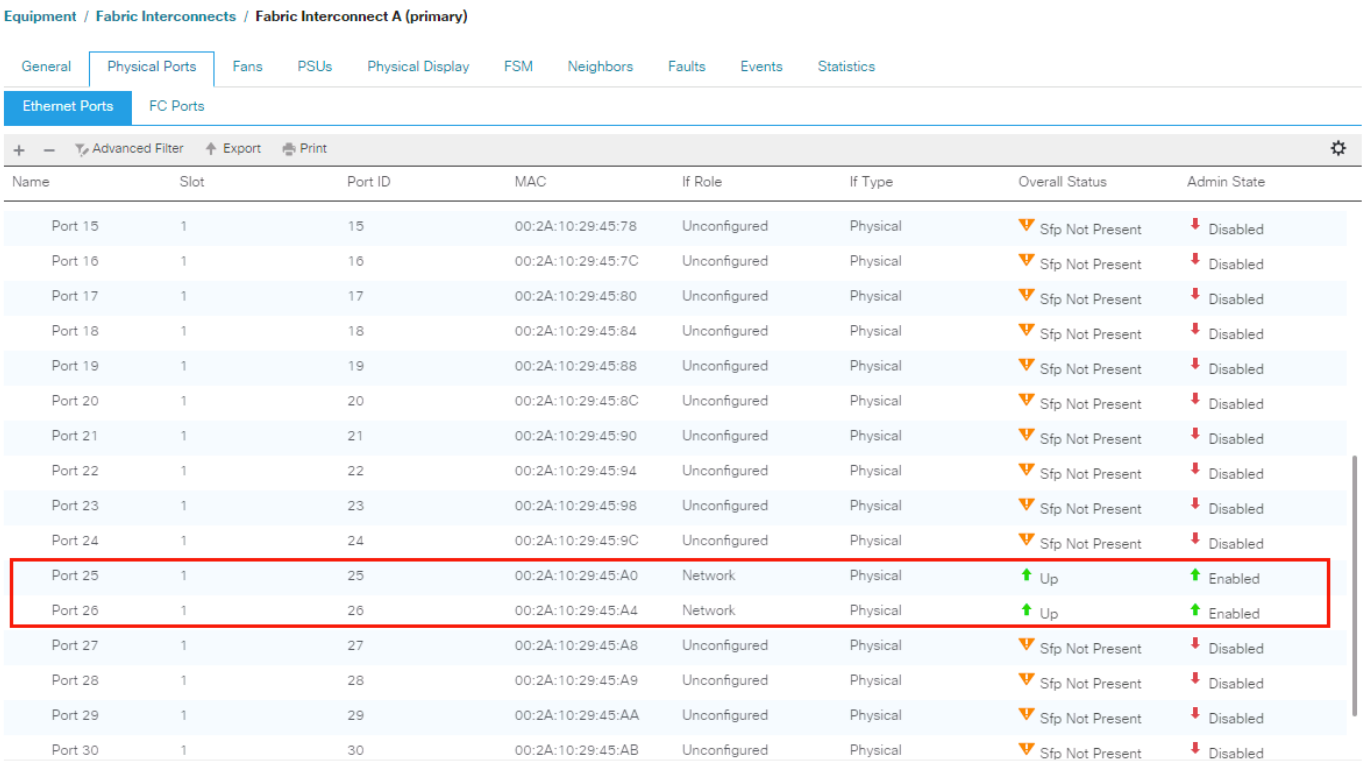


Enable Fabric Interconnect A Ports for Uplinks

To enable uplink ports, complete the following steps:

1. Select the **Equipment** tab on the left site.
2. Select Equipment > Fabric Interconnects > Fabric Interconnect A (primary) > Fixed Module.
3. Click **Ethernet Ports** section.
4. Select Ports 25-26, right-click and then select **Configure as Uplink Port**.
5. Click **Yes** and then **OK**.
6. Repeat the same steps for Fabric Interconnect B.

Figure 29 Configuring of Network Uplink Ports



Label Servers for Identification

For better identification, label each server by completing the following steps:

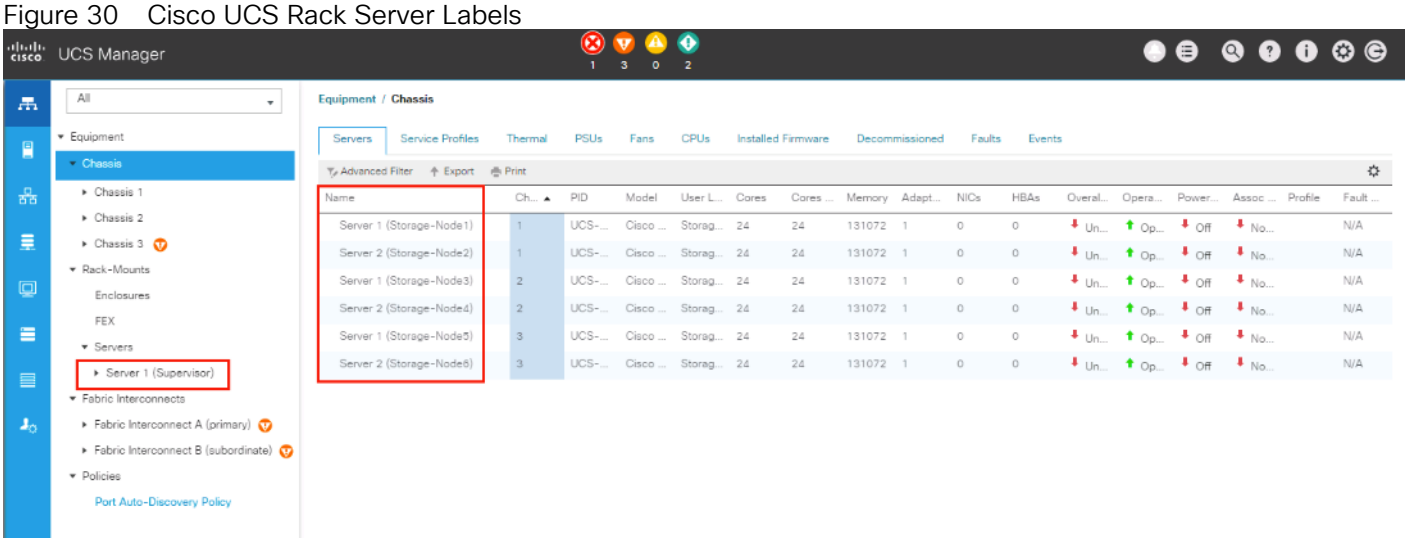
1. Select the **Equipment** tab on the left site.
2. Select Chassis > Chassis 1 > Server 1.
3. In the **Properties** section on the right go to **User Label** and add **Storage-Node1** to the field.
4. Repeat the previous steps for **Server 2** of **Chassis 1** and for all other servers of Chassis 2 – 6 according to Table 2.
5. Go then to **Servers > Rack-Mounts > Servers >** and repeat the step for all servers according to Table 4

Table 4 Server Label

Server	Name
Chassis 1 / Server 1	Storage-Node1
Chassis 1 / Server 2	Storage-Node2
Chassis 2/ Server 3	Storage-Node3
Chassis 2 / Server 4	Storage-Node4
Chassis 3 / Server 5	Storage-Node5
Chassis 3 / Server 6	Storage-Node6

Server	Name
Rack-Mount / Server 1	Supervisor

Figure 30 Cisco UCS Rack Server Labels

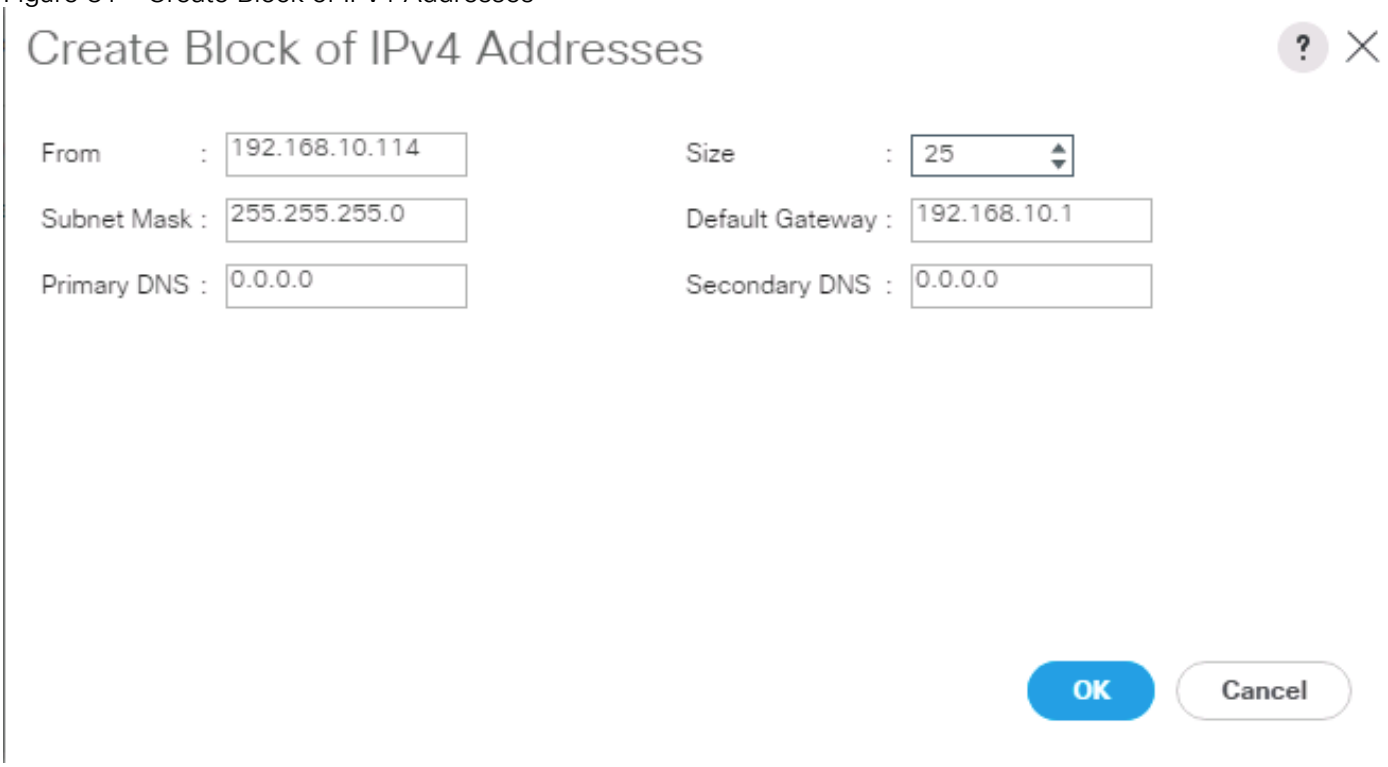


Create KVM IP Pool

To create a KVM IP Pool, complete the following steps:

1. Select the **LAN** tab on the left site.
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** 20.
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 31 Create Block of IPv4 Addresses

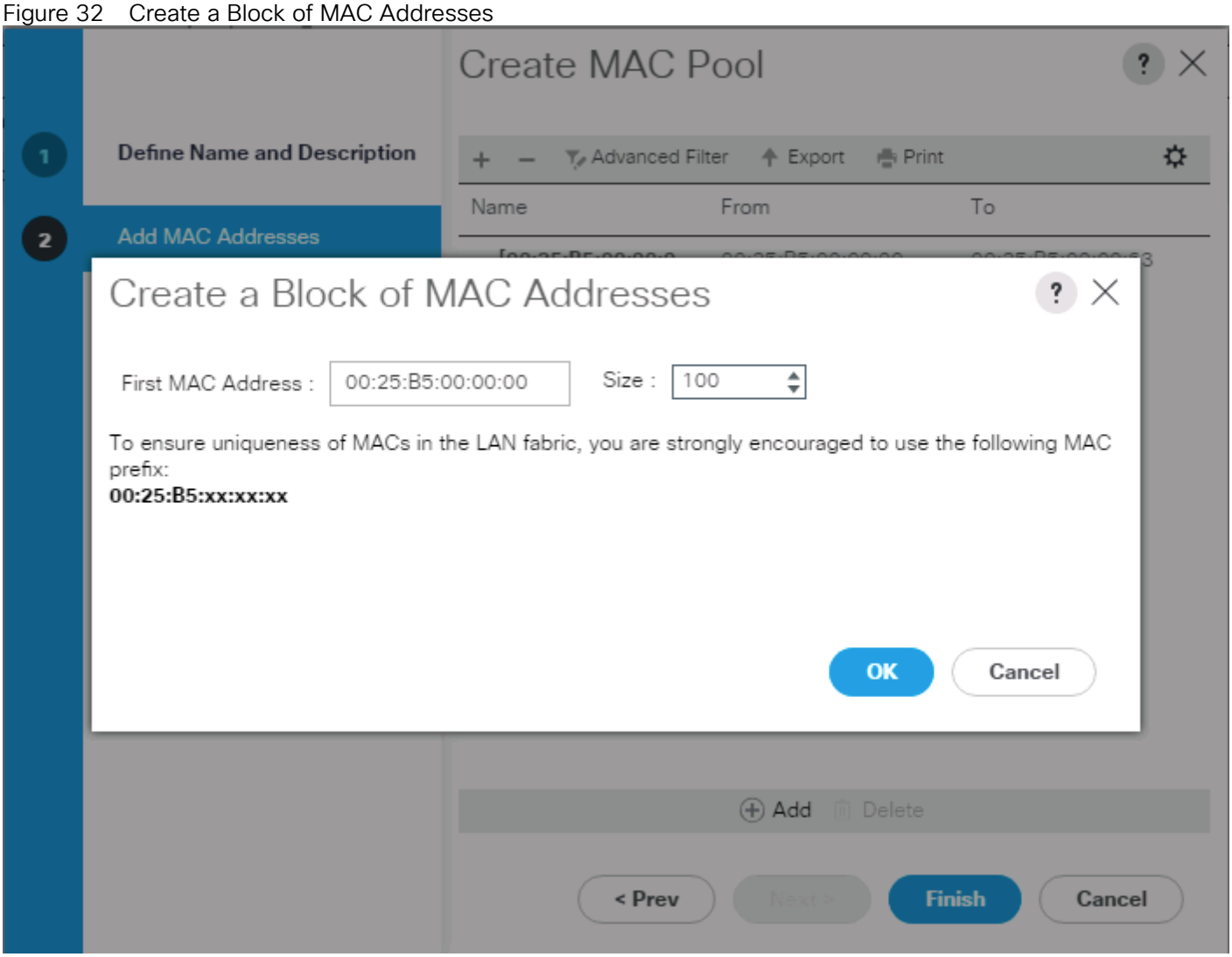


Create MAC Pool

To create a MAC Pool, complete the following steps:

1. Select the **LAN** tab on the left site.
2. Go to LAN > Pools > root > Mac Pools and right-click Create MAC Pool.
3. Type in “Scality-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 32 Create a Block of MAC Addresses



10. Click **OK**.

11. Click **Finish**.

Create UUID Pool

To create a UUID Pool, complete the following steps:

1. Select the **Servers** tab on the left site.

2. Go to Servers > Pools > root > UUID Suffix Pools and right-click Create UUID Suffix Pool.

3. Type in "**Scality-UUID-Pools**" for Name.

4. (Optional) Enter a **Description** of the MAC Pool.

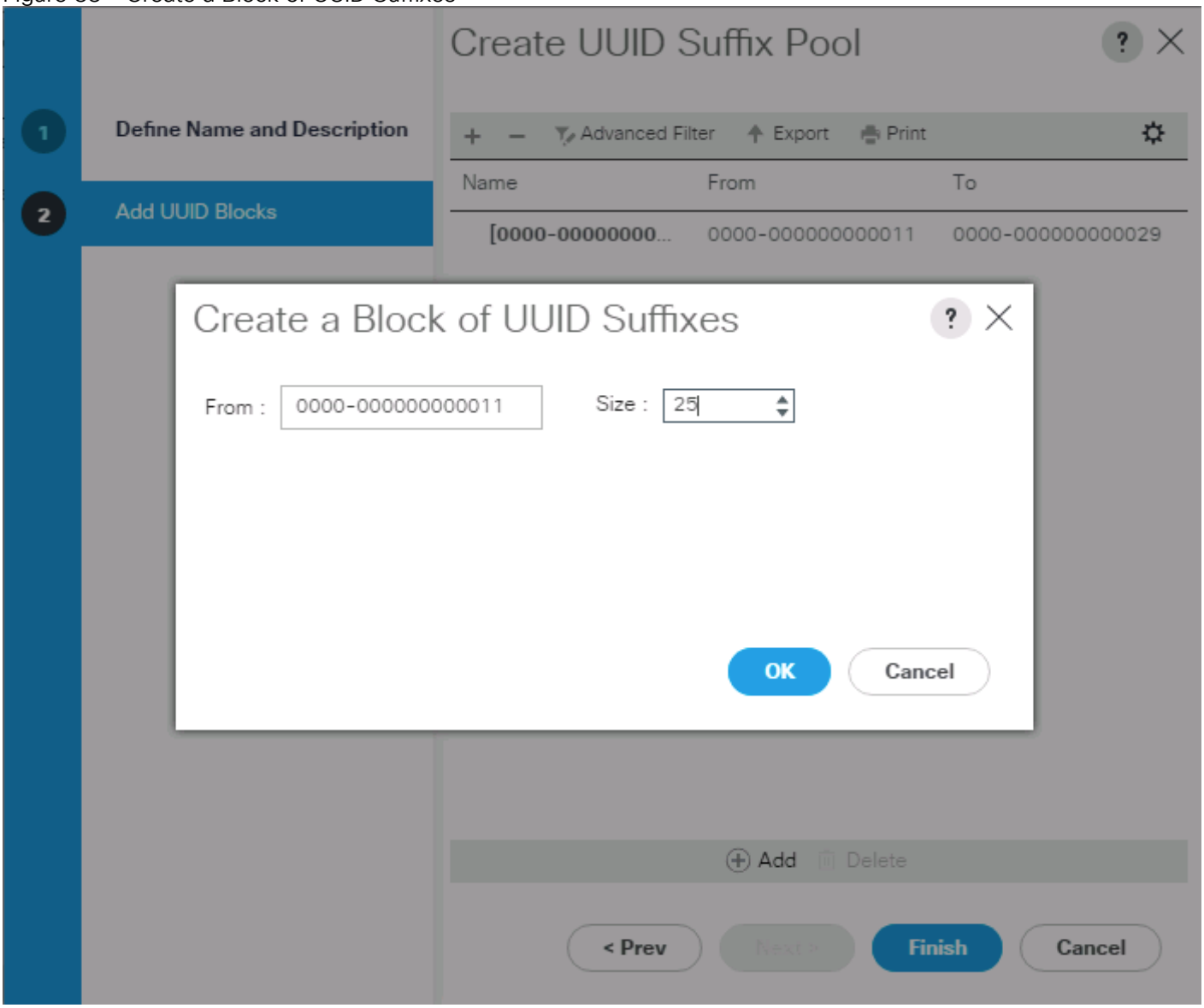
5. Set Assignment Order to Sequential and click Next.

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, 25.

Figure 33 Create a Block of UUID Suffixes



9. Click **OK**.

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

Create VLANs

As mentioned before 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 5 lists the configured VLANs.

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

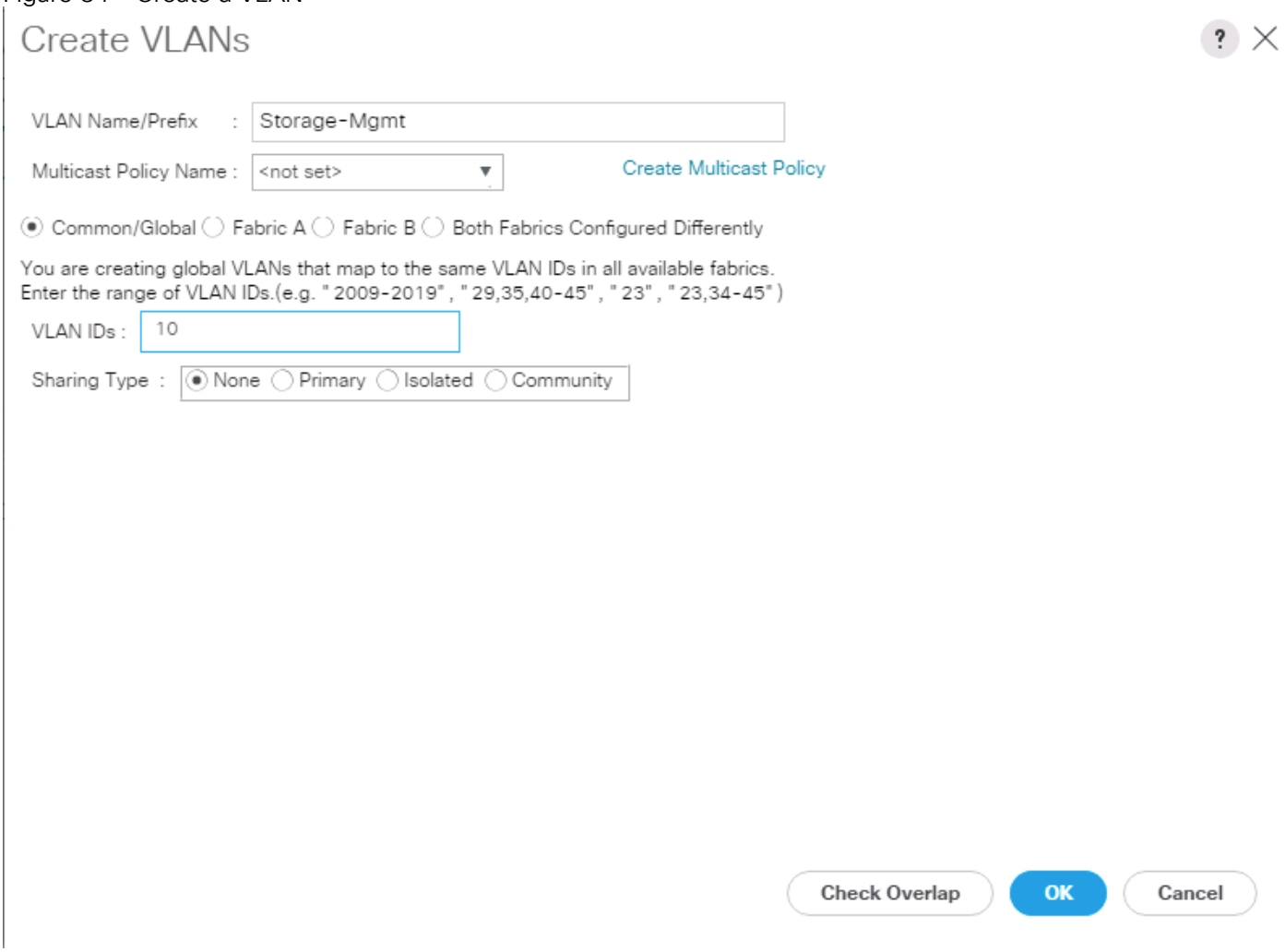
Table 5 VLAN Configurations

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

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

1. Select **LAN** in the left pane 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 34 Create a VLAN



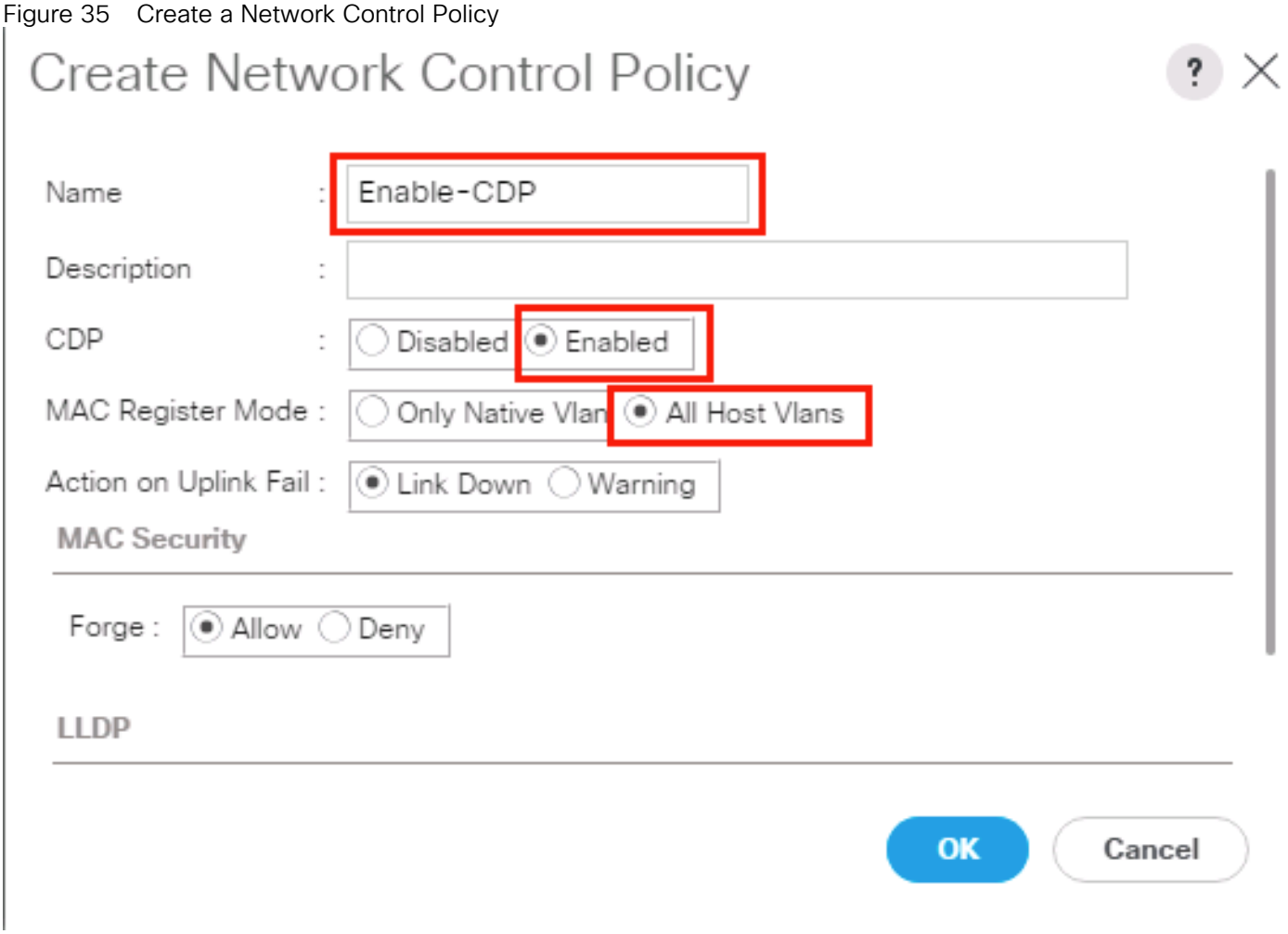
8. Repeat the steps for rest of the VLANs “Storage-Cluster” “External-Network and Client-Network.”

Enable CDP

To enable Network Control Policies, complete the following steps:

1. Select the **LAN** tab in the left pane of 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 35 Create a Network Control Policy

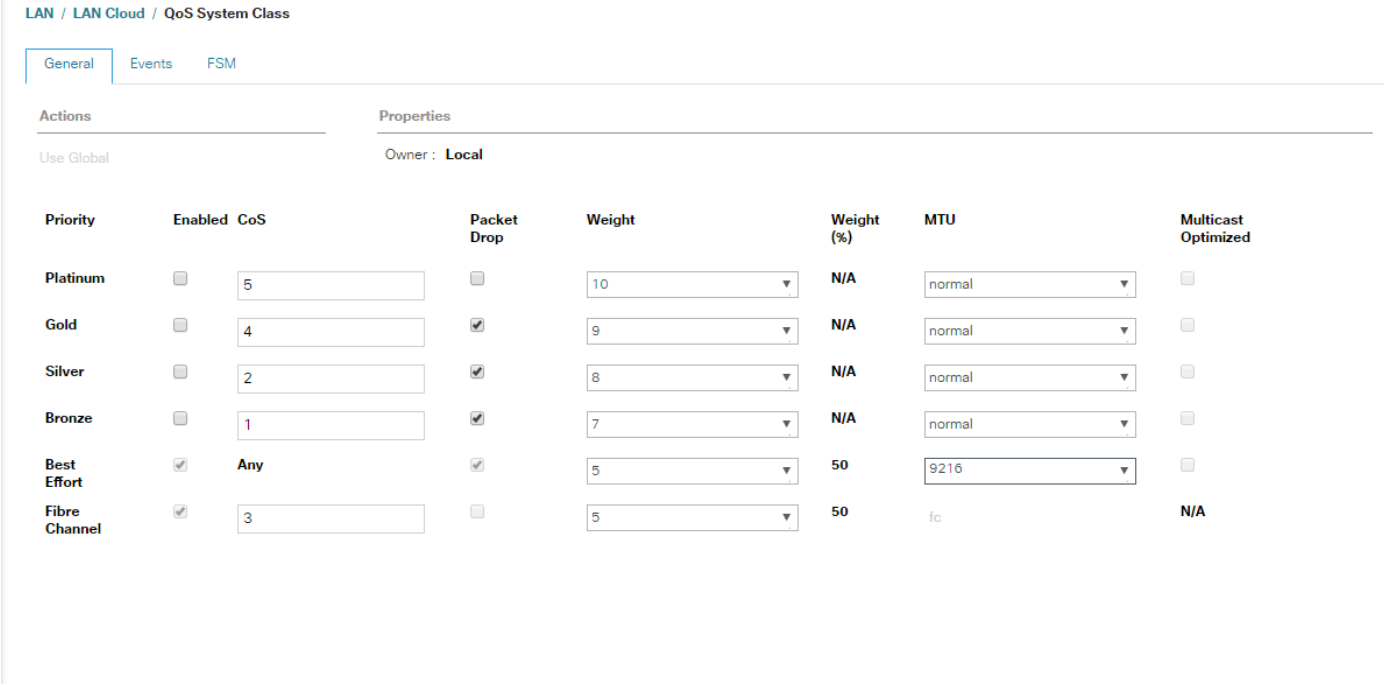


QoS System Class

To create a Quality of Service System Class, complete the following steps:

1. Select the **LAN** tab in the left pane of 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 36 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 Scality Storage we need to create four different vNICs, depending on the role of the server. Table 6 provides an overview of the configuration.

Table 6 vNIC Table

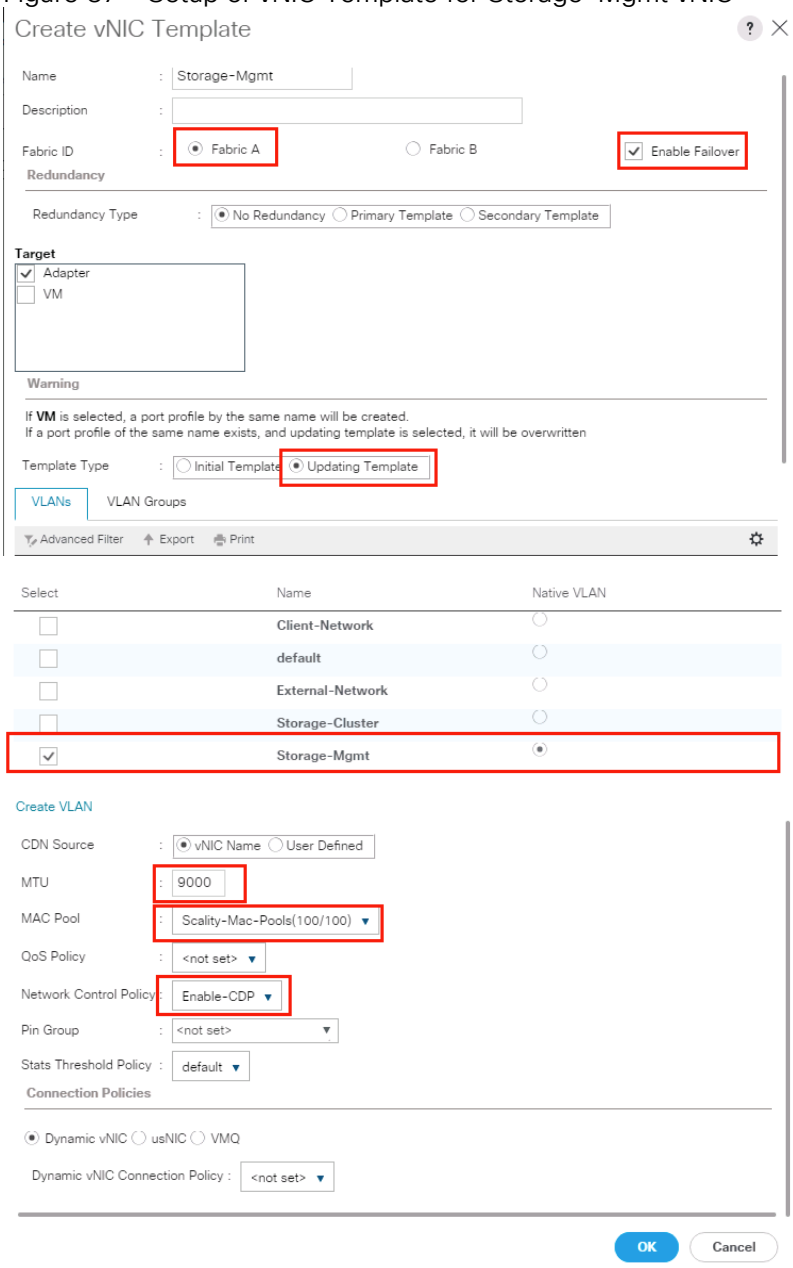
vNIC Name	Fabric	Failover	VLAN Name / ID	MTU Size	MAC Pool	Network Control Policy
Storage-Mgmt	A	Yes	Storage-Mgmt 10	9000	Scality-MAC-Pools	Enable-CDP
Storage-Cluster	B	Yes	Storage-Cluster 20	9000	Scality-MAC-Pools	Enable-CDP
External-Network	A	Yes	External-Network 79	1500	Scality-MAC-Pools	Enable-CDP
Client-Network	B	Yes	Client-Network 30	9000	Scality-MAC-Pools	Enable-CDP

To create the appropriate vNICs, complete the following steps:

1. Select the **LAN** tab in the left pane of 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 **Scality-MAC-Pools** as MAC Pool.
9. Select Enable-CDP as Network Control Policy.
10. Click **OK** and then **OK**.

Figure 37 Setup of vNIC Template for Storage-Mgmt vNIC



11. Repeat these steps 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 5 .

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, complete the following steps:

1. Select the **Server** tab in the left pane of 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

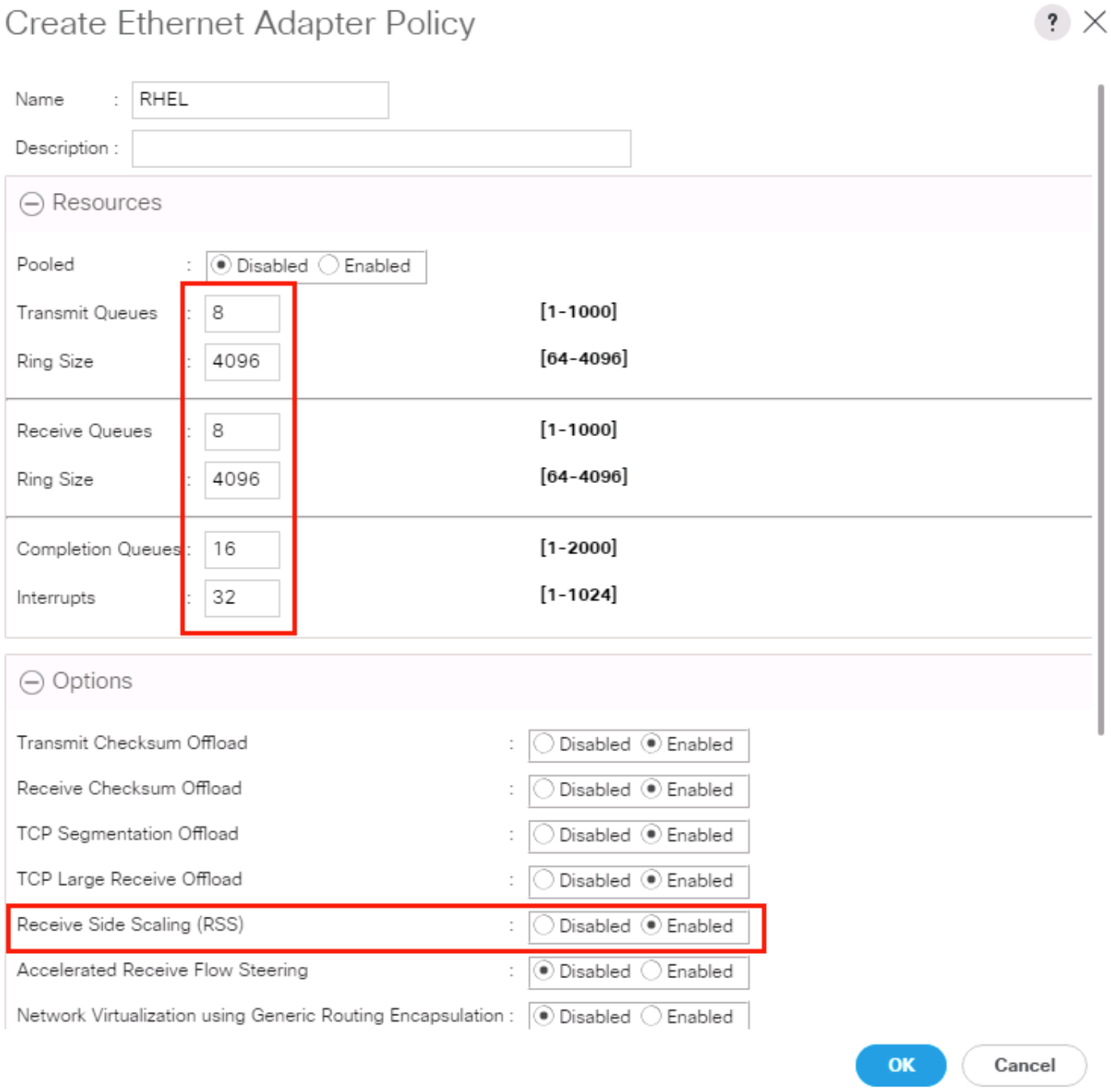
   Receive Queues: 8

   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 38 Adapter Policy for RHEL



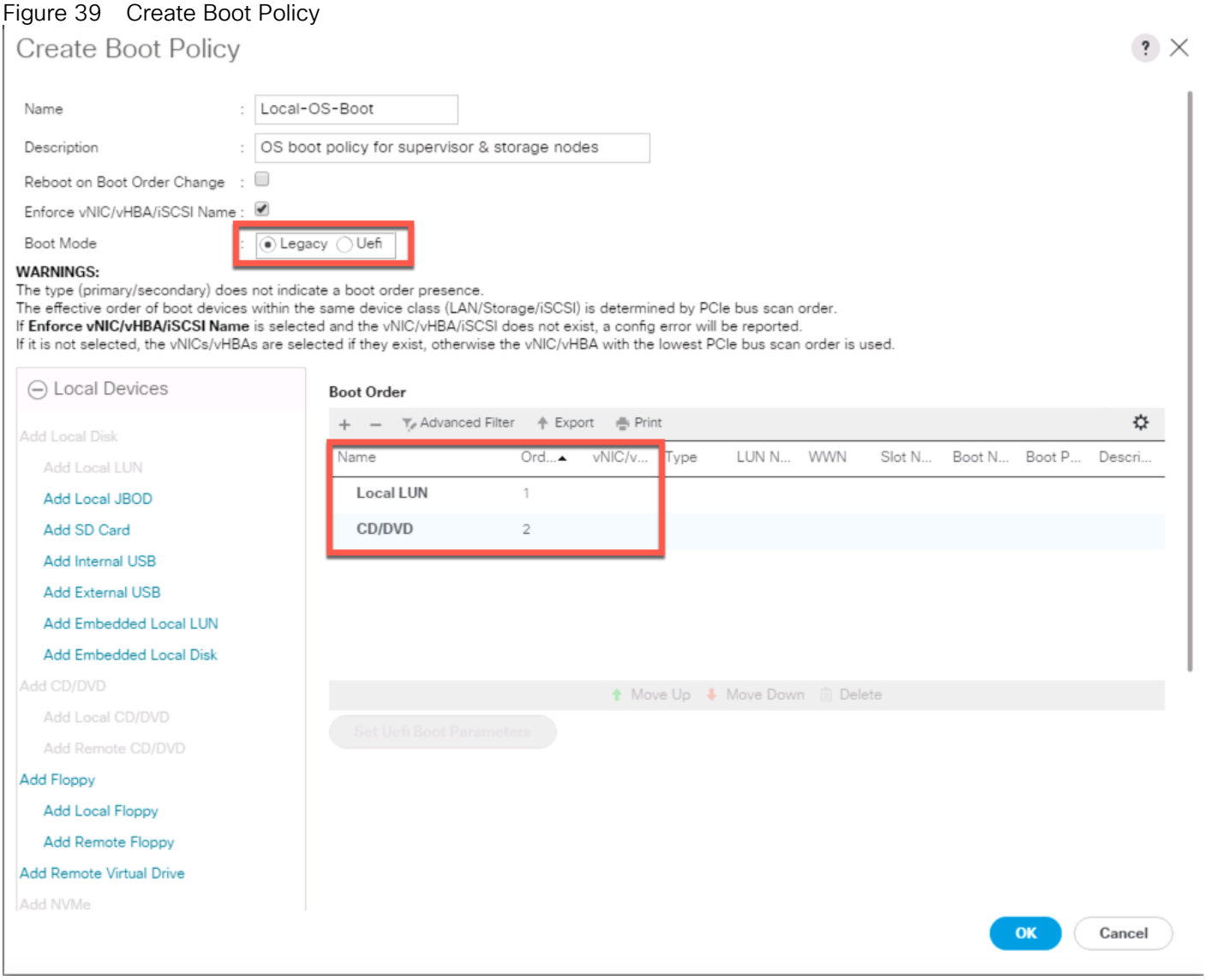
Boot Policy Setup

To create a Boot Policy, complete the following steps:

1. Select the **Servers** tab in the left pane.
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 39 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, complete the following steps:

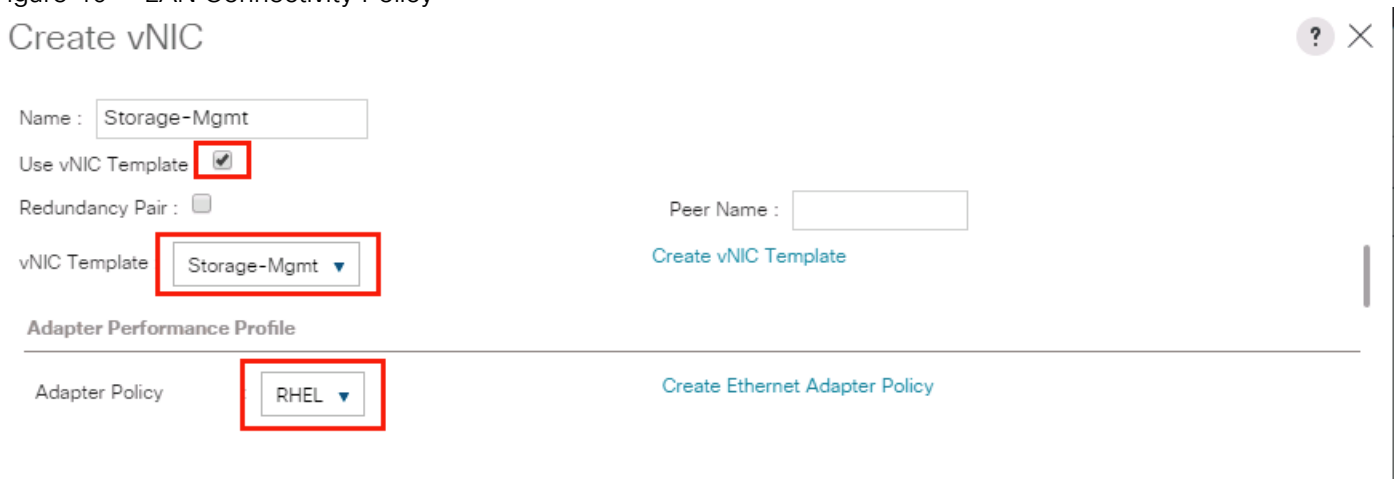
1. Select the **LAN** tab in the left pane.

2. Go to LAN > Policies > root > LAN Connectivity Policies and right-click Create LAN Connectivity Policy for Storage 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 40 LAN Connectivity Policy

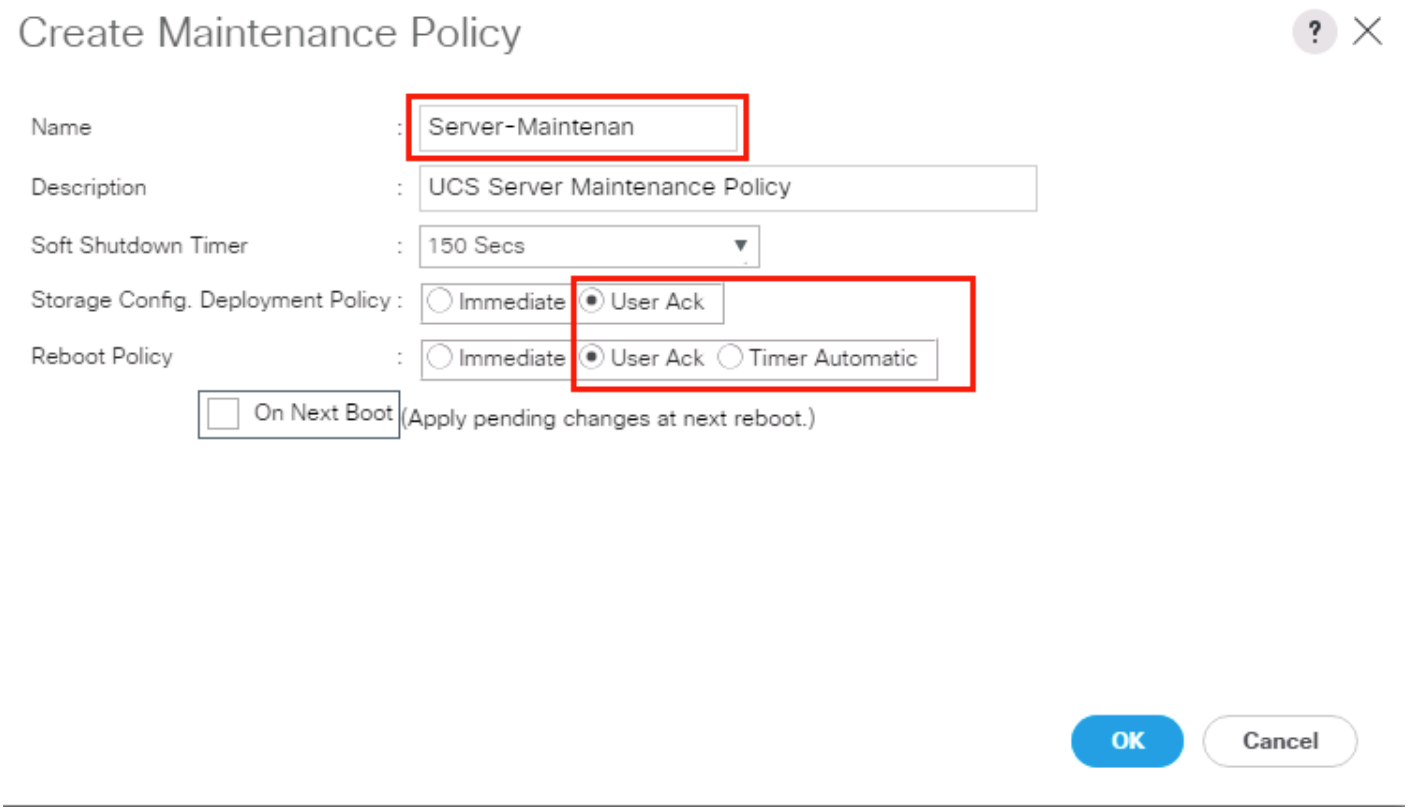


10. Repeat these steps for the remaining networks "Storage-Cluster", "External-Network", and "Client-Network" Make sure you choose Adapter Policy as "RHEL" for VNIC interface "Storage-Cluster."

Create Maintenance Policy Setup

To setup a Maintenance Policy, complete the following steps:

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



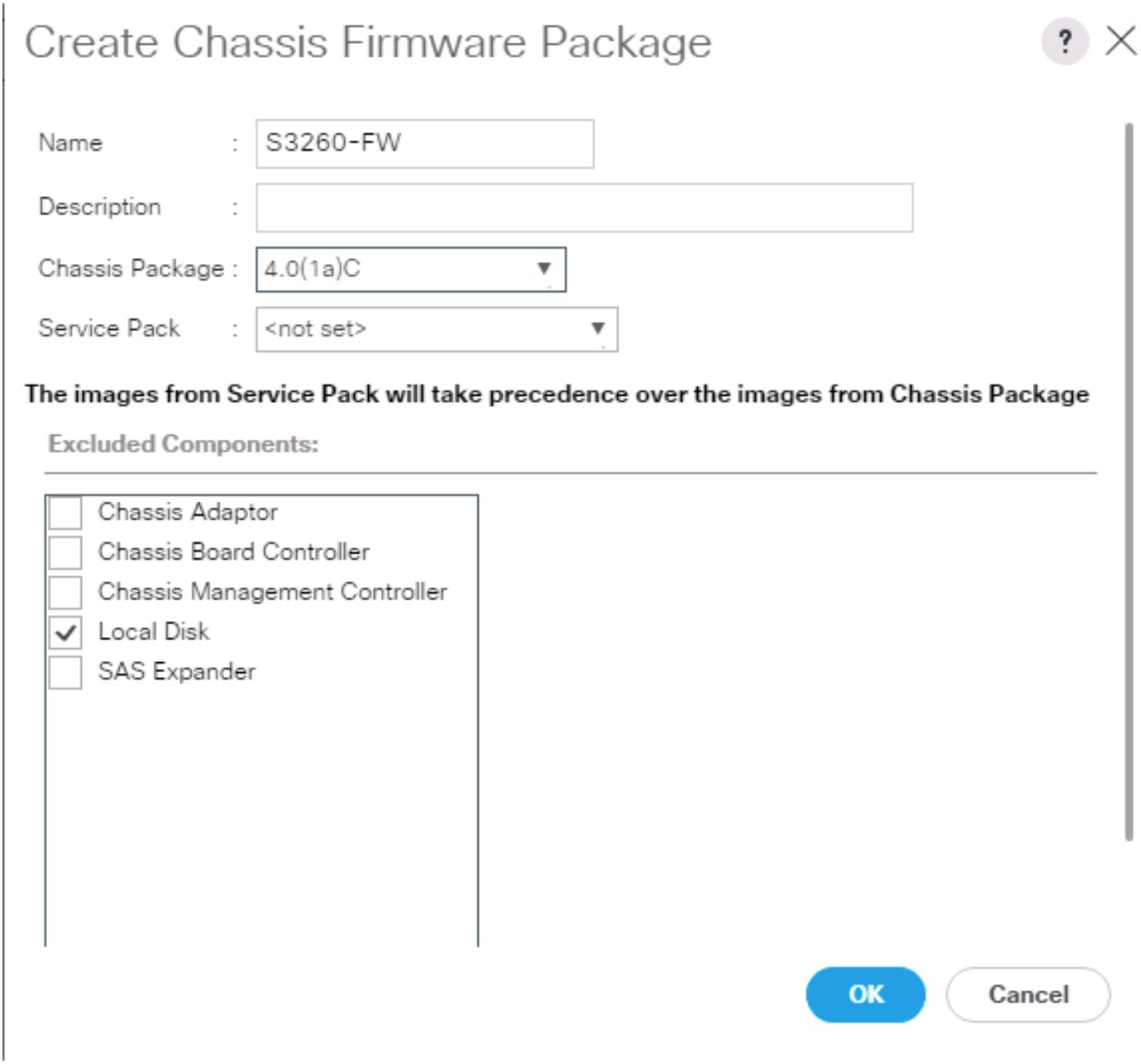
Creating Chassis Profile

The Chassis Profile is required to assign specific disks to a particular server node in a Cisco UCS S3260 Storage Server as well as upgrading to a specific chassis firmware package.

Create Chassis Firmware Package

To create a Chassis Firmware Package, complete the following steps:

1. Select the **Chassis** tab in the left pane of the Cisco UCS Manager GUI.
2. Go to Chassis > Policies > root > Chassis Firmware Package and right-click Create Chassis Firmware Package.
3. Type in **S3260-FW** in the Name field.
4. (Optional) Enter a description in the **Description** field.
5. Select **4.0(1a)C** form the drop-down list of **Chassis Package**.
6. Select **OK** and then click **OK** again.
7. Create Chassis Firmware Package.

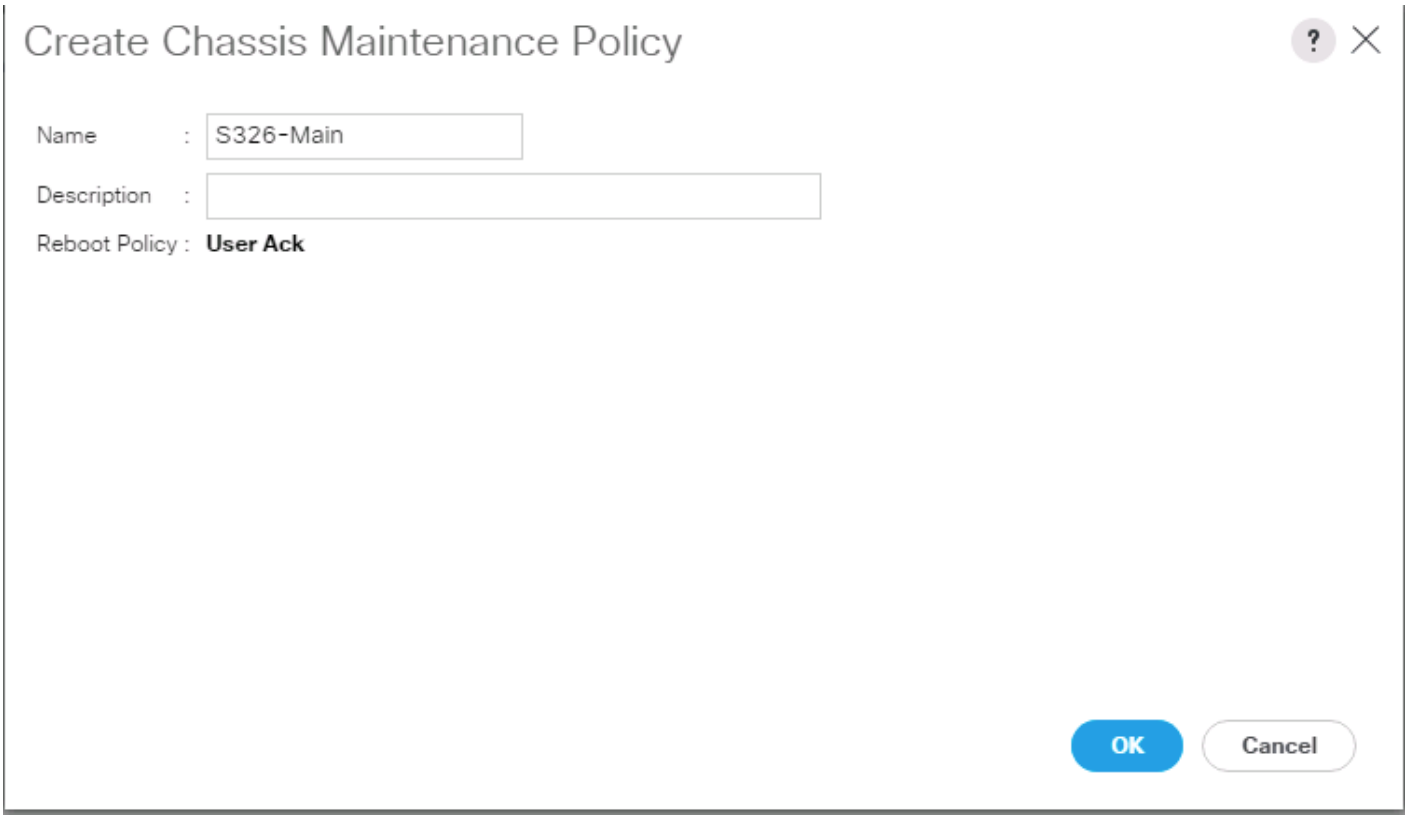


Create Chassis Maintenance Policy

To create a Chassis Maintenance Policy, complete the following steps:

1. Select the **Chassis** tab in the left pane of the Cisco UCS Manager GUI.
2. Go to Chassis > Policies > root > Chassis Maintenance Policies and right-click Create Chassis Maintenance Policy.
3. Type in **S3260-Main** in the **Name** field.
4. (Optional) Enter a description in the **Description** field.
5. Click **OK** and then **OK**.

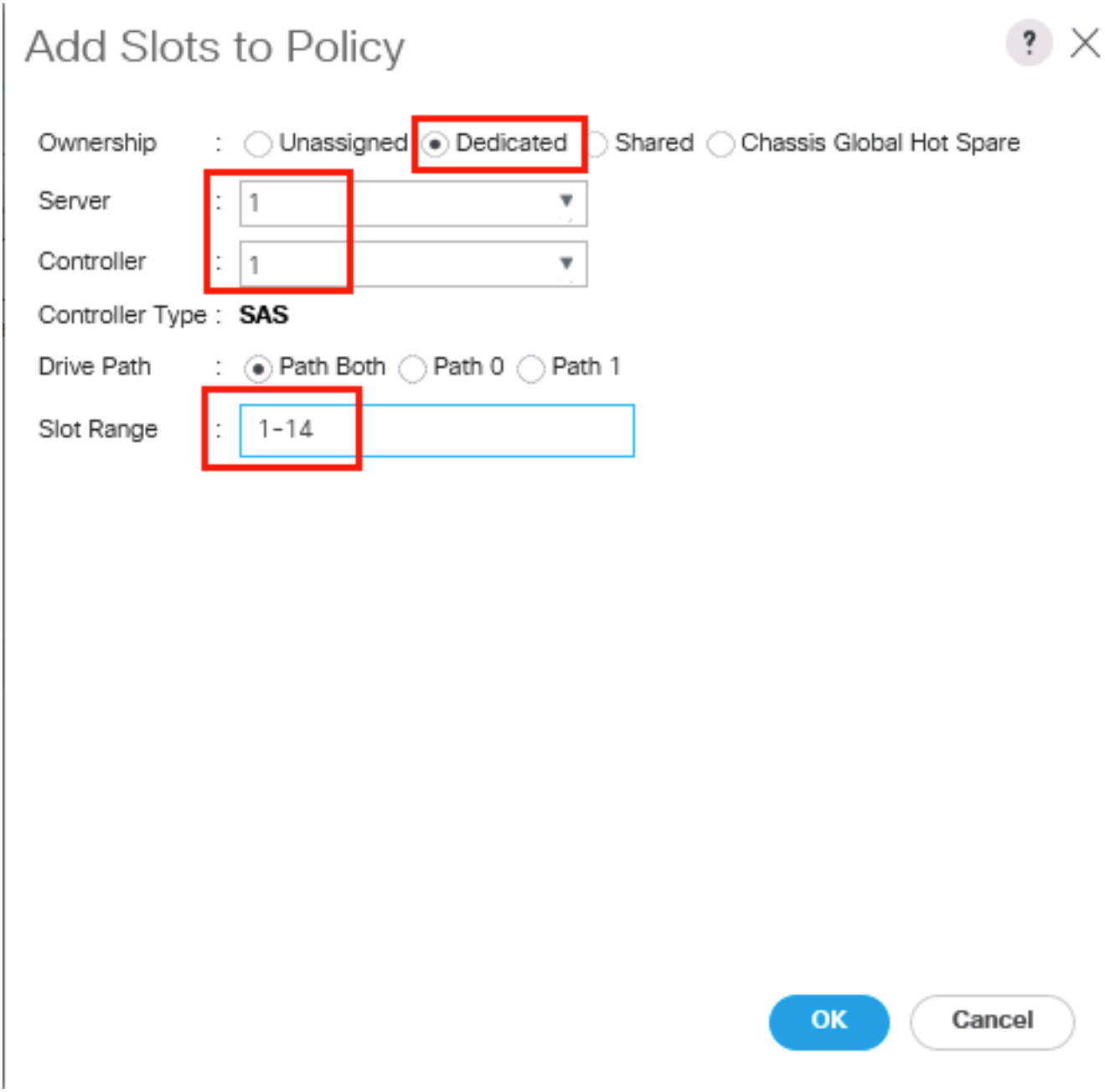
6. Create Chassis Maintenance Policy.



Create Disk Zoning Policy

To create a Disk Zoning Policy, complete the following steps:

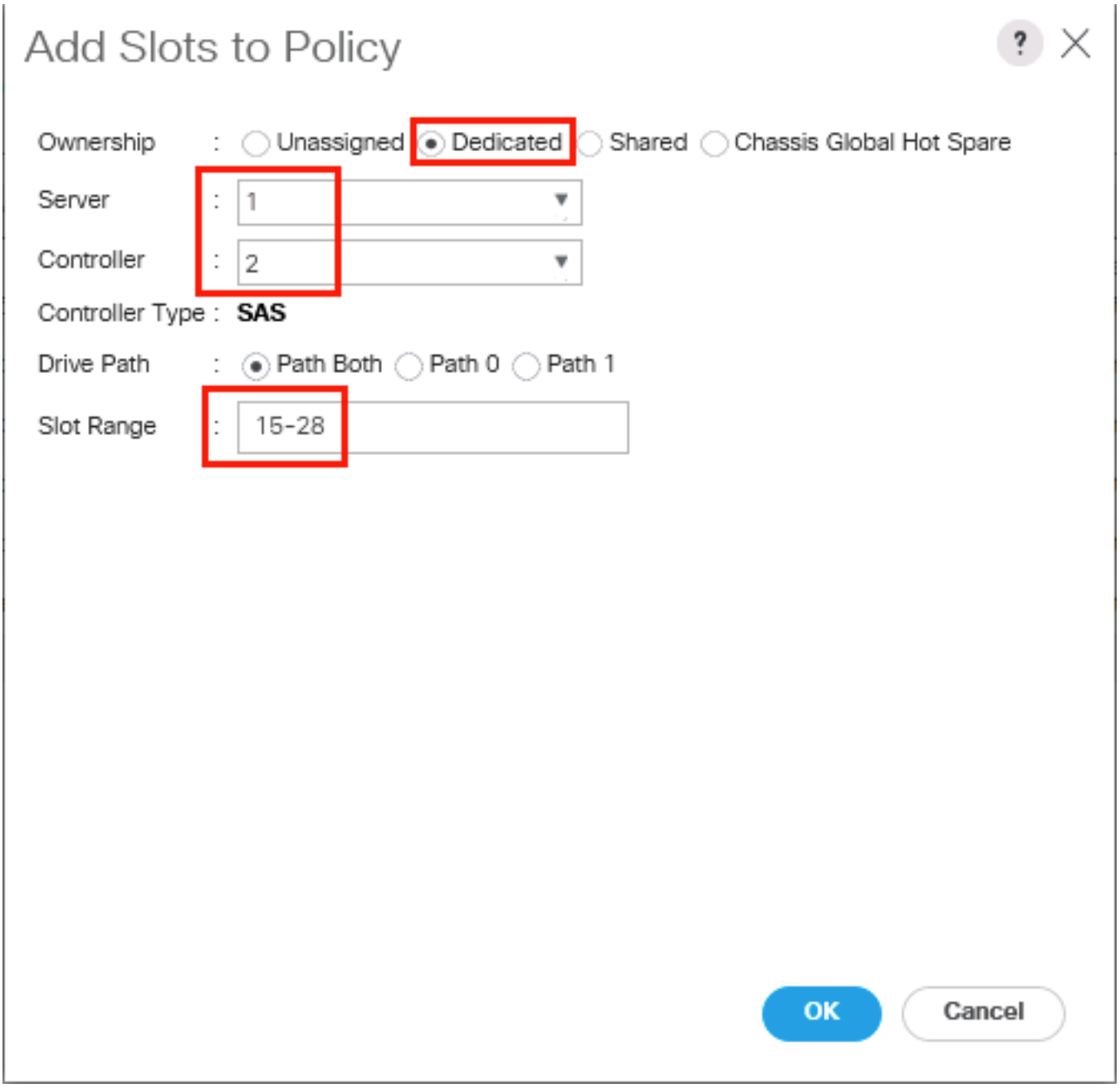
1. Select the **Chassis** tab in the left pane of the Cisco UCS Manager GUI.
2. Go to Chassis > Policies > root > Disk Zoning Policies and right-click Create Disk Zoning Policy.
3. Type in **S3260-DiskZoning** in the Name field.
4. (Optional) Enter a description in the **Description** field.
5. Create Disk Zoning Policy.
6. Click **Add**.
7. Select Dedicated under Ownership.
8. Select **Server** 1 and Select **Controller** 1.
9. Add **Slot Range 1-14** for the top node of the Cisco UCS S3260 Storage Server and click **OK**.



10. Select **`Server`** 1 and Select **`Controller`** 2.

11. Add **`Slot Range 15-28`** for the top node of the Cisco UCS S3260 Storage Server and click **`OK`**.

12. Add Slots to Top Node of Cisco UCS S3260.

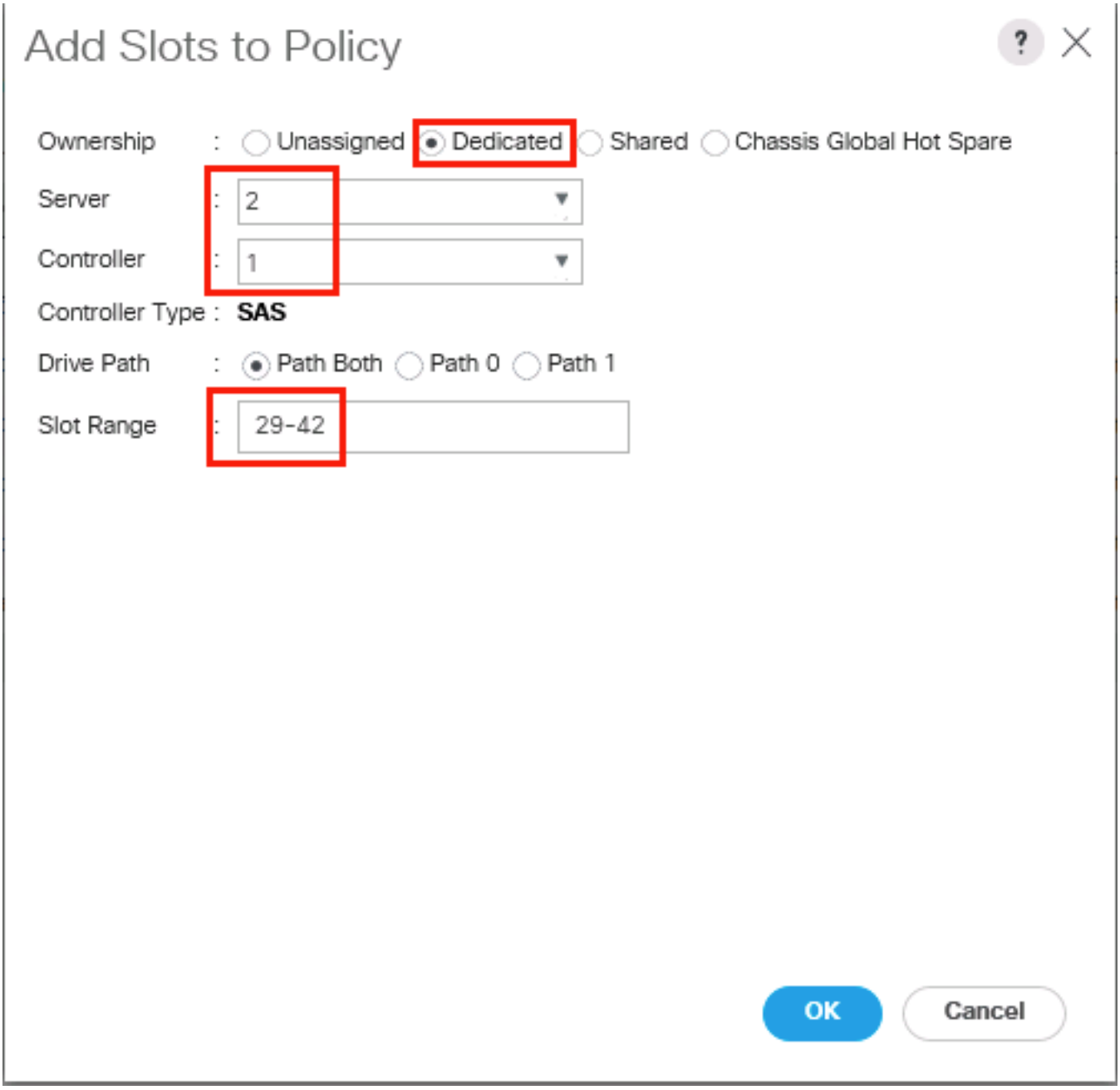


13. Click **Add**.

14. Select Dedicated under Ownership.

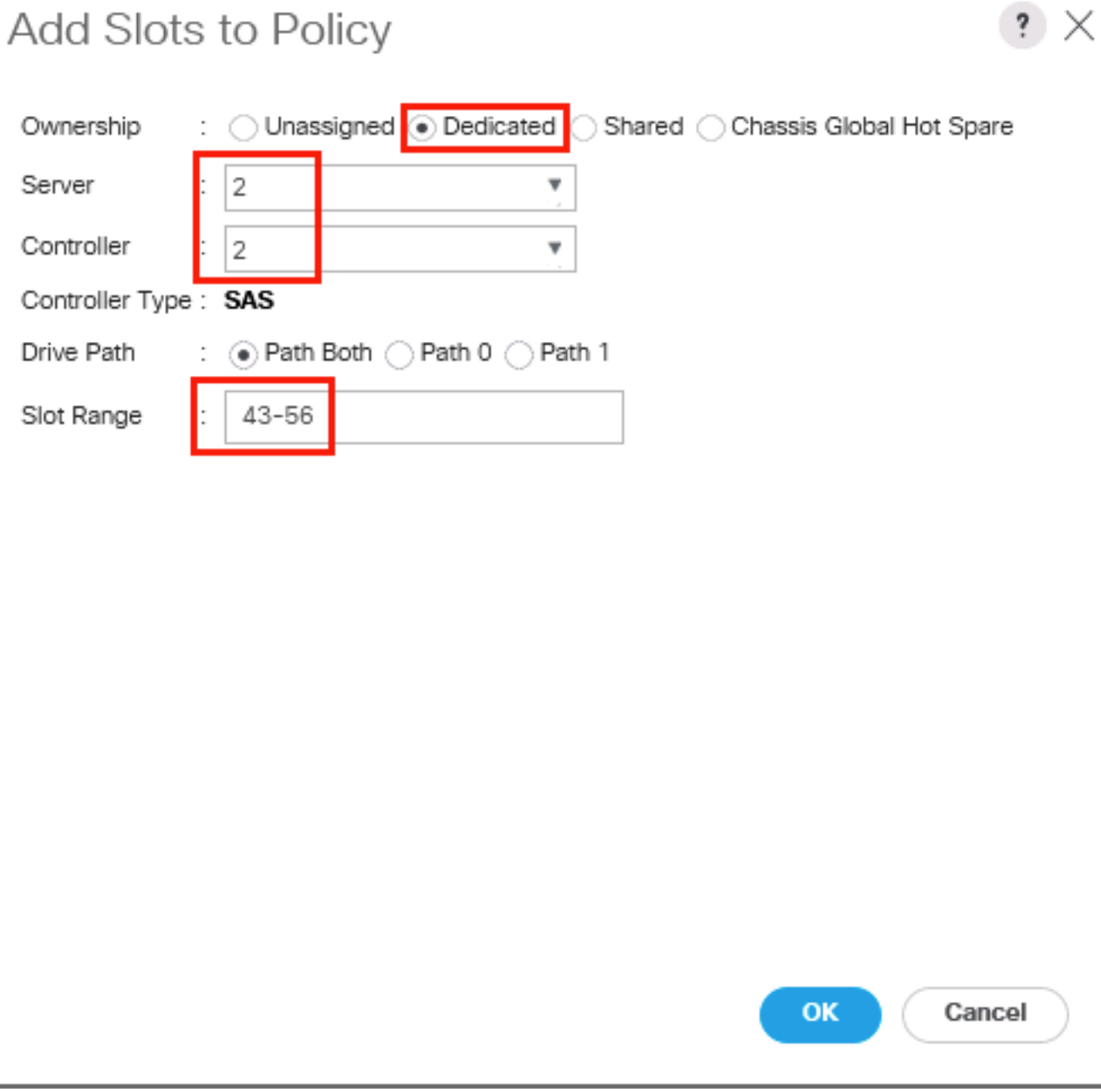
15. Select **Server 2** and Select Controller 1.

16. Add **Slot Range 29-42** for the bottom node of the Cisco UCS S3260 Storage Server and click OK.



17. Select **`Server 2`** and Select Controller 2.

18. Add **`Slot Range 43-56`** for the bottom node of the Cisco UCS S3260 Storage Server and click OK.



19. Add Slots to the Bottom Node of Cisco UCS S3260.

Create Chassis Profile Template

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

1. Select the **Chassis** tab in the left pane of the Cisco UCS Manager GUI.
2. Go to Chassis > Chassis Profile Templates and right-click Create Chassis Profile Template.
3. Type in S3260-Chassis in the Name field.
4. Under Type, select Updating Template.

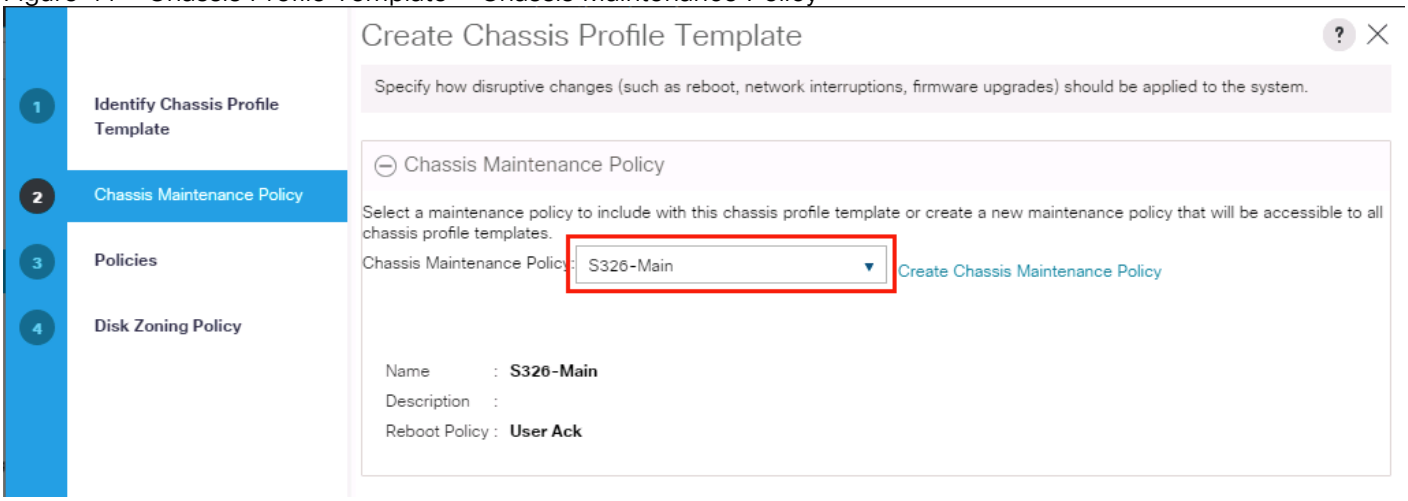
5. (Optional) Enter a description in the **Description** field.

6. Create Chassis Profile Template

7. Select **Next**.

8. Under the radio button **Chassis Maintenance Policy**, select your previously created Chassis Maintenance Policy.

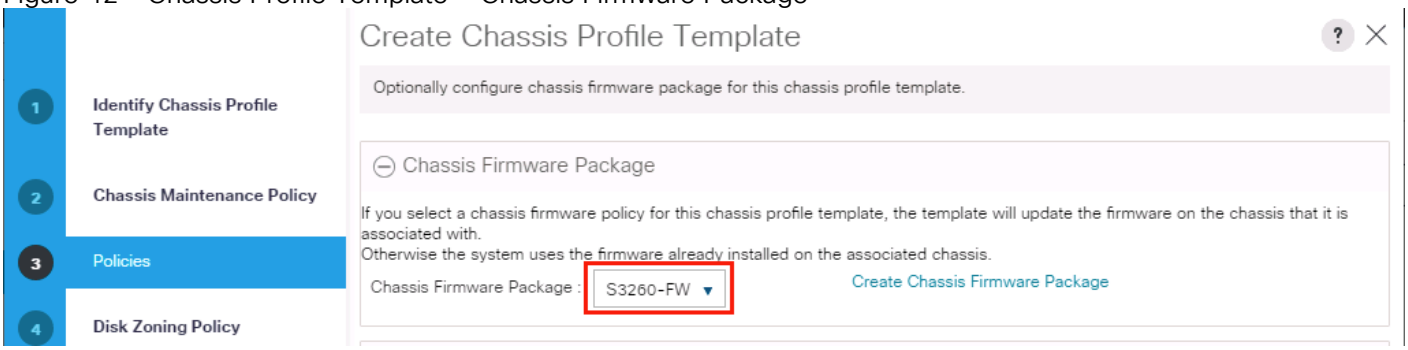
Figure 41 Chassis Profile Template – Chassis Maintenance Policy



9. Select **Next**.

10. Select the + button and select under **Chassis Firmware Package** your previously created Chassis Firmware Package Policy.

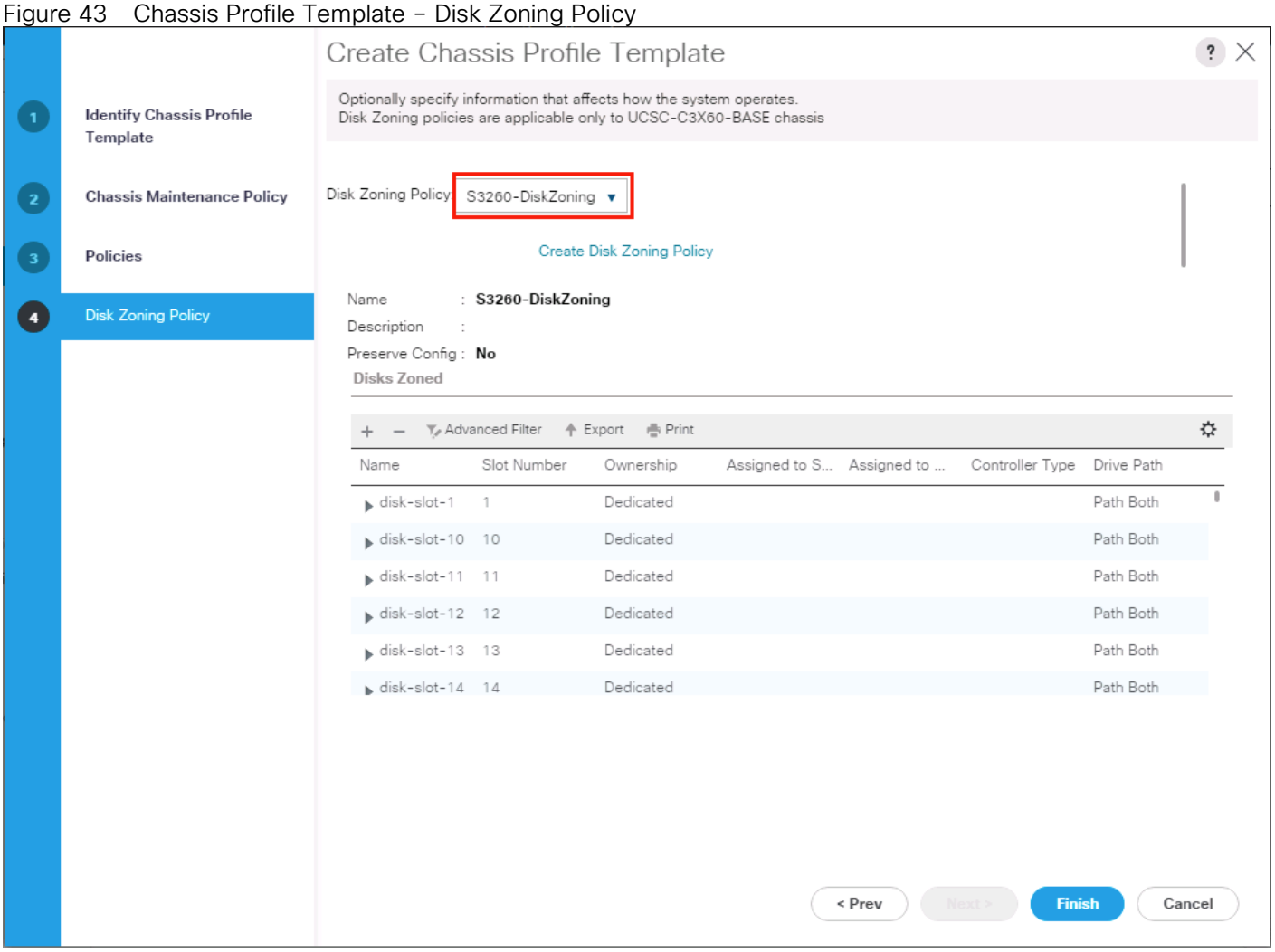
Figure 42 Chassis Profile Template – Chassis Firmware Package



11. Select Next.

12. Under **Disk Zoning Policy** select your previously created Disk Zoning Policy.

Figure 43 Chassis Profile Template – Disk Zoning Policy

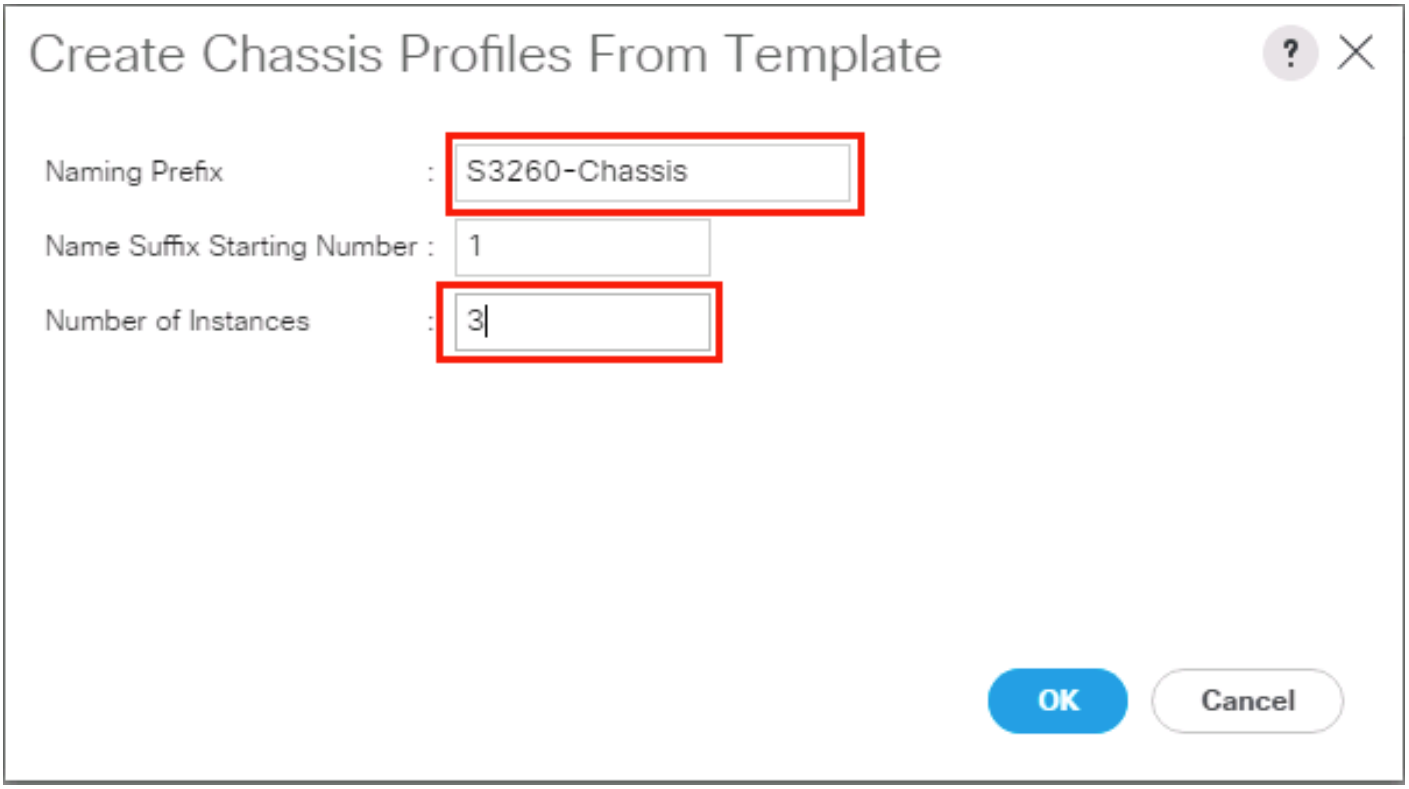


13. Click **Finish** and then click **OK** again.

Create Chassis Profile from Template

To create the Chassis Profiles from the previous created Chassis Profile Template, complete the following steps:

1. Select the **Chassis** tab in the left pane of the Cisco UCS Manager GUI.
2. Go to Chassis > Chassis Profile Templates and select “S3260-Chassis” you created previously.
3. Then right click to select “Create Chassis Profiles from Template."
4. Type in **S3260-Chassis** in the **Name** field.
5. Leave the Name Suffix Starting Number untouched.
6. Enter **3** for the **Number of Instances** for all connected Cisco UCS S3260 Storage Server.
7. Click **OK**.

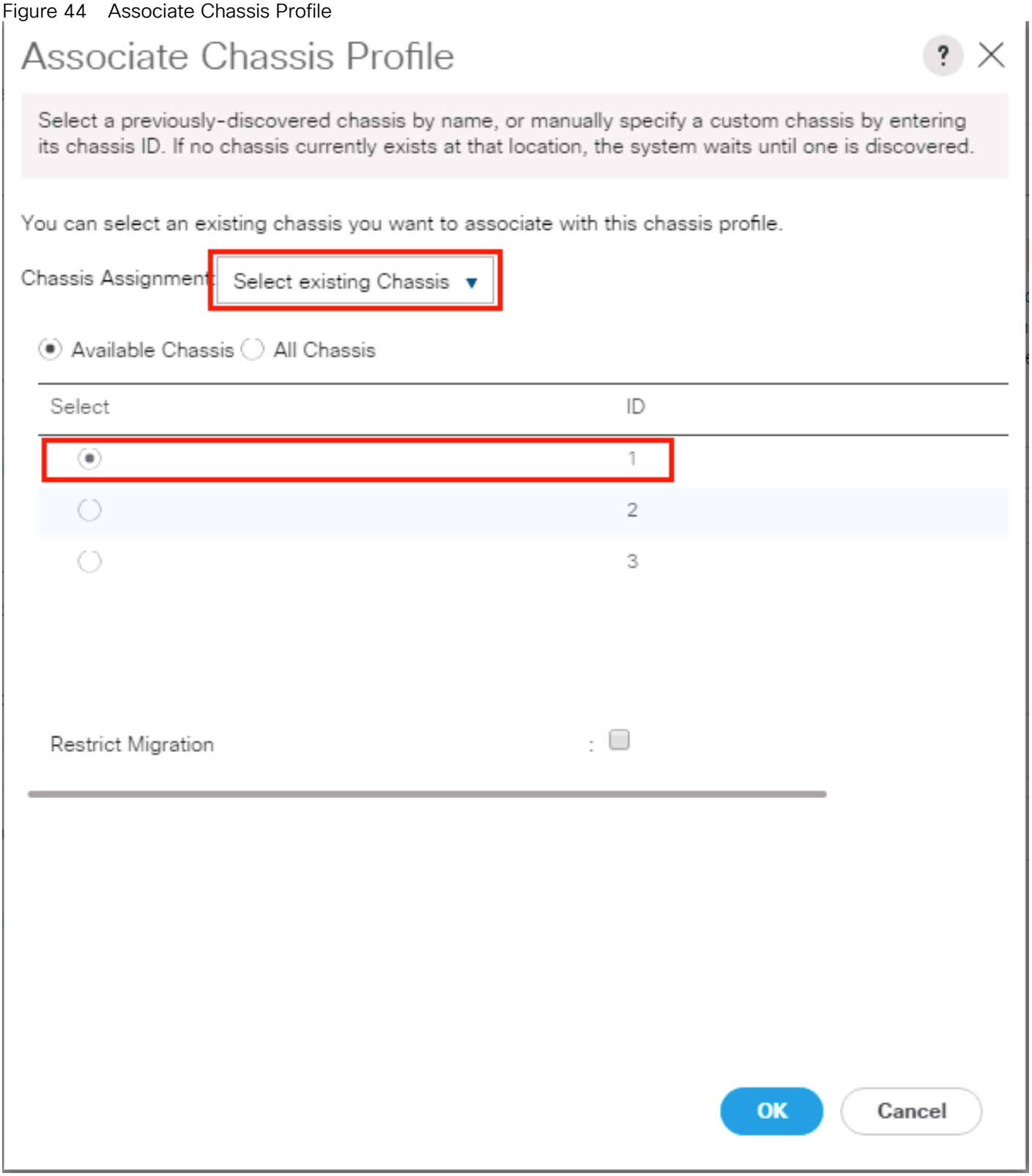


Associate Chassis Profile

To associate all previous created Chassis Profile, complete the following steps:

1. Select the **Chassis** tab in the left pane of the Cisco UCS Manager GUI.
2. Go to Chassis > Chassis Profiles and select "S3260-Chassis1."
3. Right-click Change Chassis Profile Association.
4. Under Chassis Assignment, choose Select existing Chassis from the drop-down list.
5. Under **Available Chassis**, select ID **1**.
6. Click **OK** and then click **OK** again.
7. Repeat the steps for the other two Chassis Profiles by selecting the IDs 2 – 3.

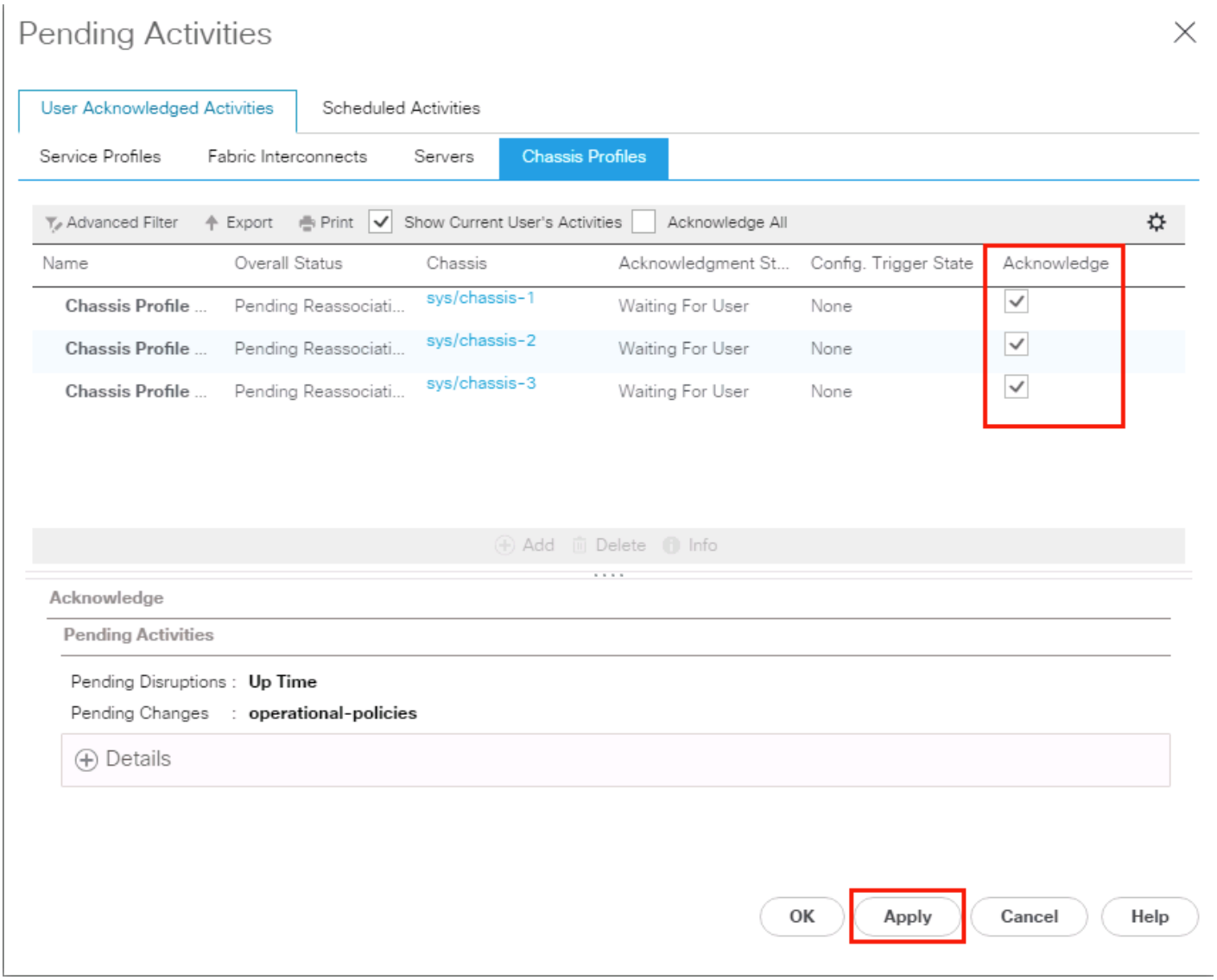
Figure 44 Associate Chassis Profile



8. A pop-up will appear on the top right side. Click Chassis Profiles and Acknowledge All Chassis profiles.

9. Click Apply.

10. Click OK



Creating Storage Profiles

Setting Disks for Cisco UCS S3260 M5 Servers to Unconfigured-Good

To prepare the OS drives reserved from the S3260 M5 servers for storage profiles, make sure the disks have to be converted from "JBOD" to "Unconfigured-Good". To convert the disks, complete the following steps:

1. Select the **`Equipment`** tab in the left pane of the Cisco UCS Manager GUI.
2. For S3260 M5 servers, Go to Equipment > Chassis > Chassis1 > Servers > Server1 > Inventory > Storage > Disks
3. Select both disks from slot "201 and 202" and right-click "**`Set JBOD to Unconfigured-Good`**".
4. Repeat the steps for the other S3260 M5 Servers.

Create Storage Profiles for Cisco UCS S3260 Storage Server

To create the Storage Profile for the top node of the Cisco UCS S3260 Storage Server, complete the following steps:

1. Select **Storage** in the left pane of the Cisco UCS Manager GUI.
2. Go to Storage > Storage Profiles and right-click Create Storage Profile.
3. Type in **S3260-TopNode** 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:

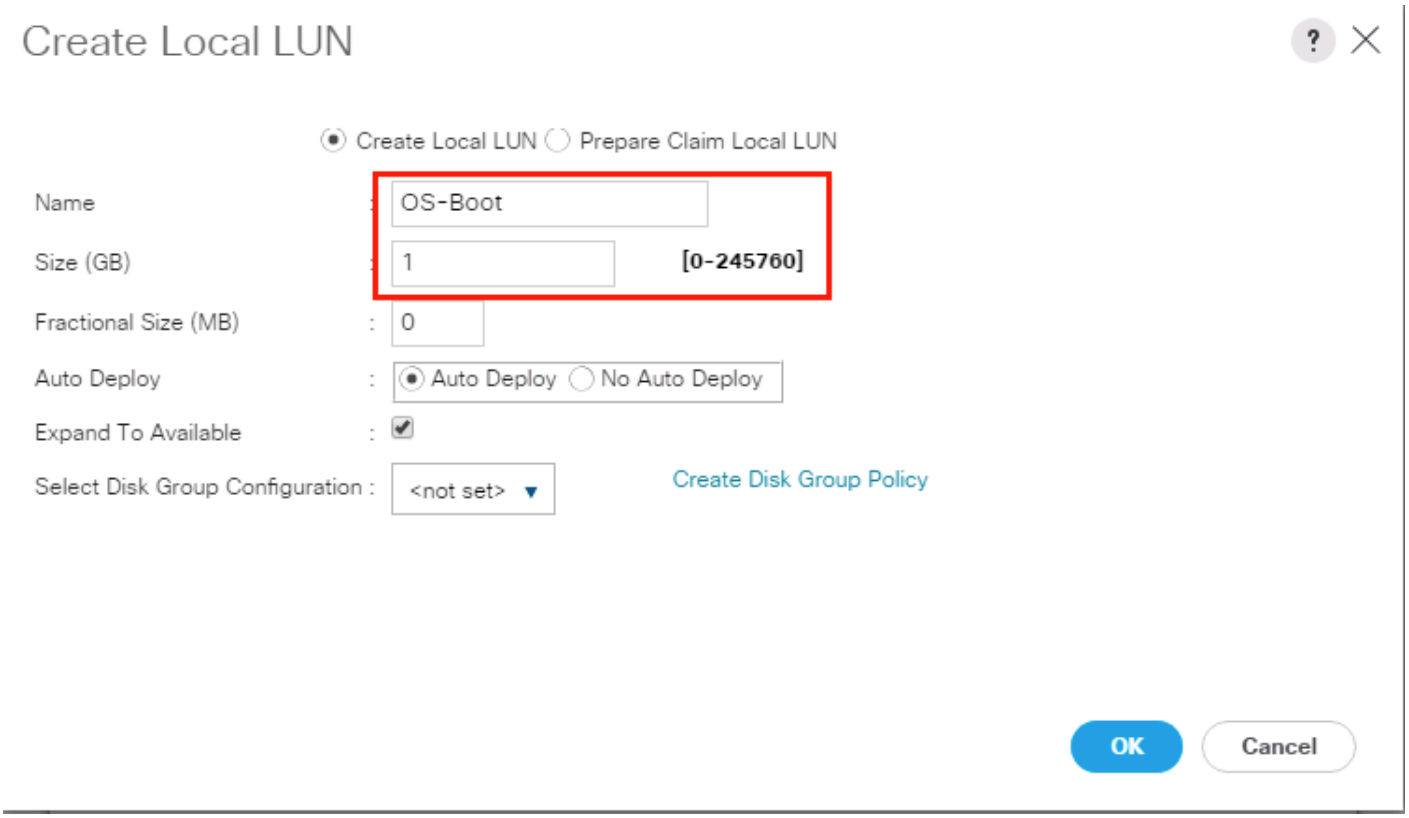
   Create Local LUN

   Size (GB) = 1

   Fractional Size (MB) = 0

   Auto Deploy

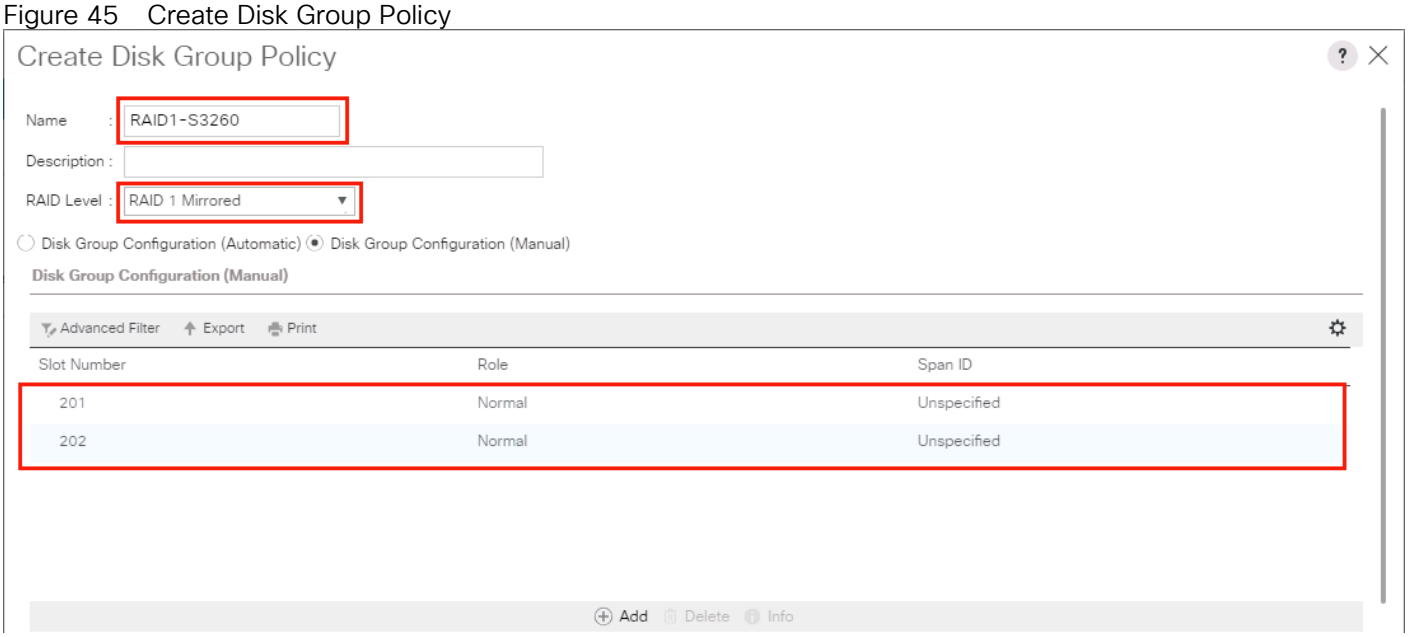
   Select Expand To Available



8. Click "Create Disk Group Policy" to Create RAID1 LUN.

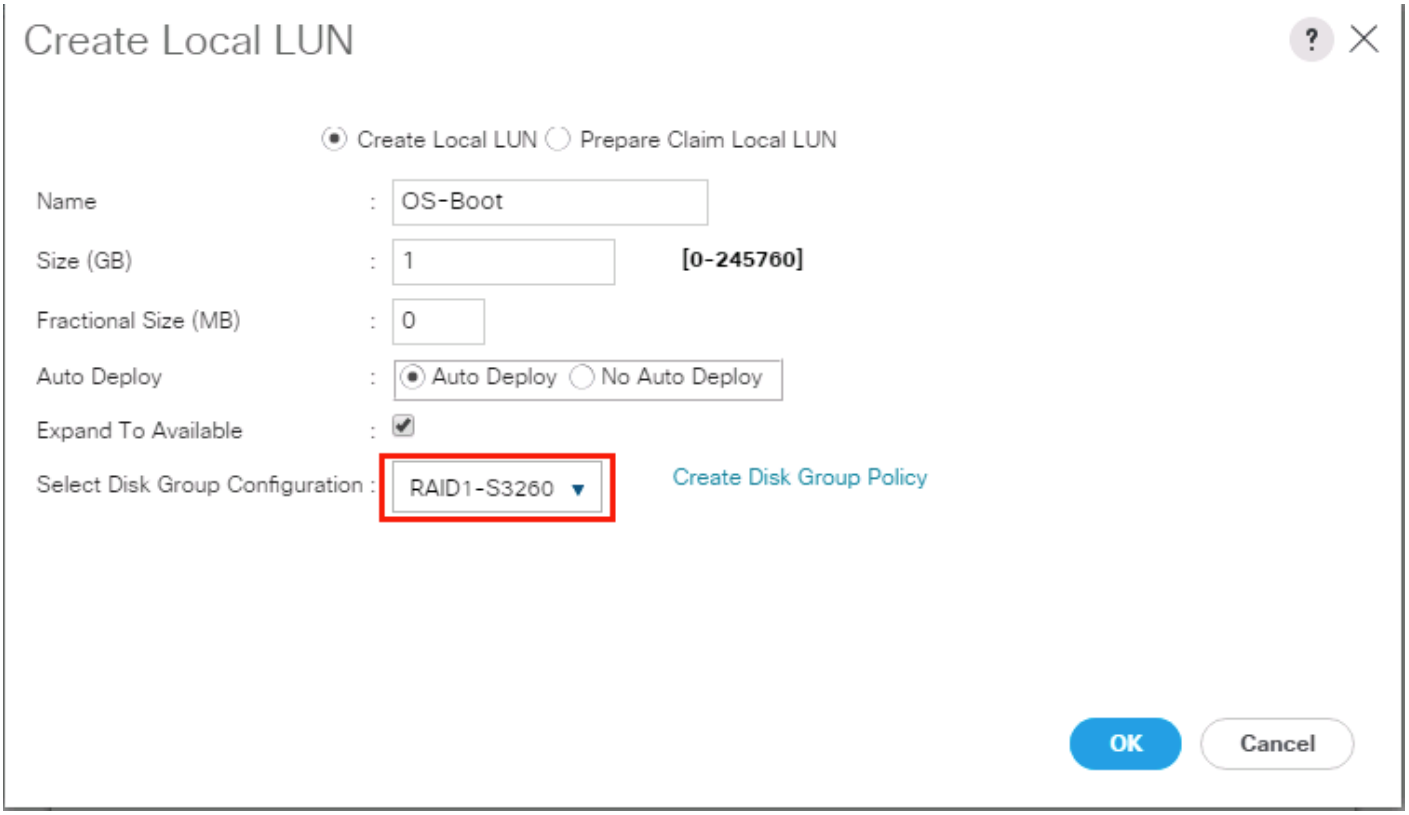
9. Type in **RAID1-S3260** in the Name field.
10. (Optional) Enter a description in the **Description** field.
11. RAID Level = RAID 1 Mirrored.
12. Select Disk Group Configuration (Manual).
13. Click **Add**.
14. Type in **201** for **Slot Number**.
15. Click **OK** and then again **Add**.
16. Type in **202** 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 **OK**

Figure 45 Create Disk Group Policy



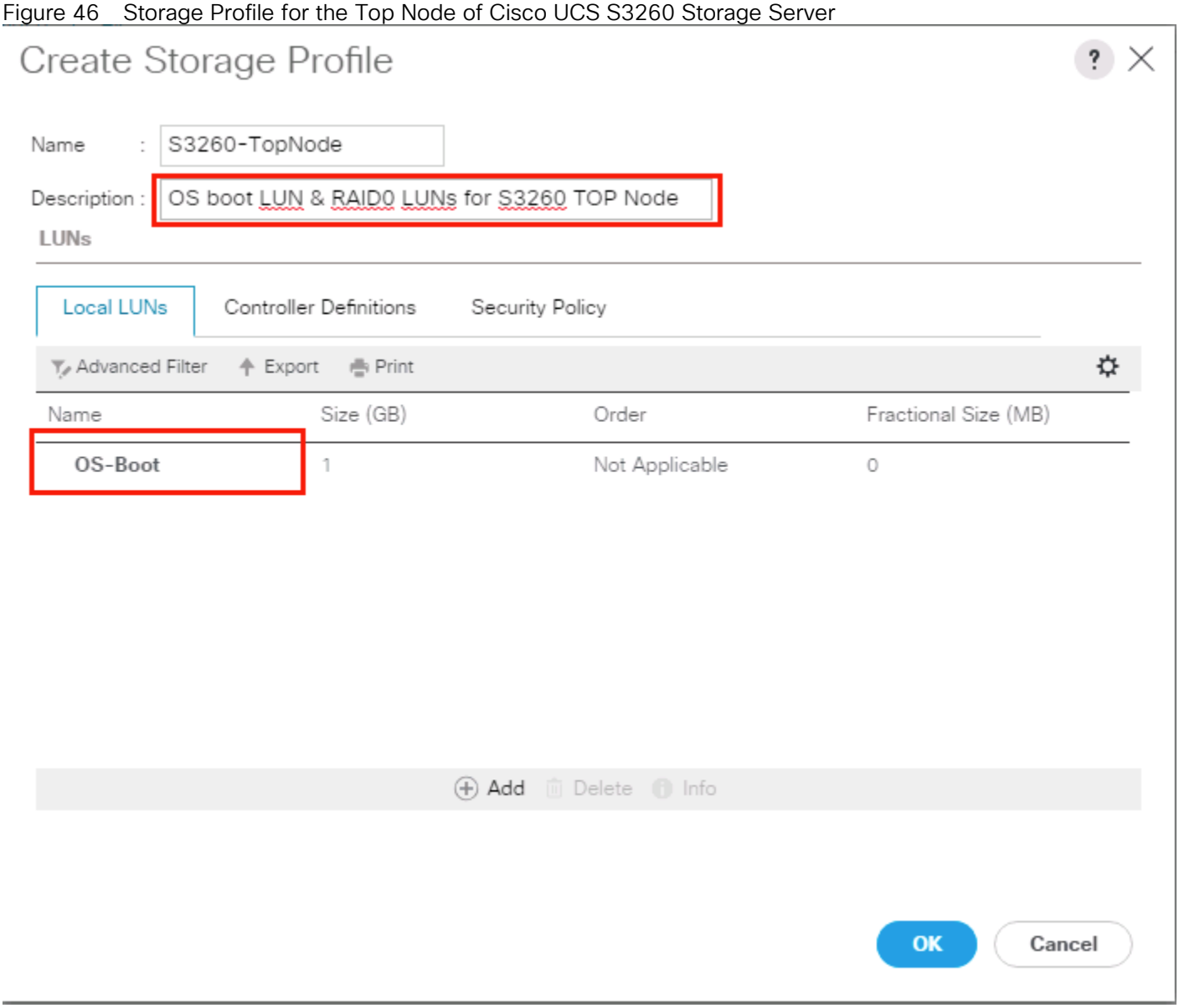
19. Select your previously created Disk Group Policy for the Boot with the radio button under **`Select Disk Group Configuration`**.

20. Select Disk Group Configuration.



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

Figure 46 Storage Profile for the Top Node of Cisco UCS S3260 Storage Server



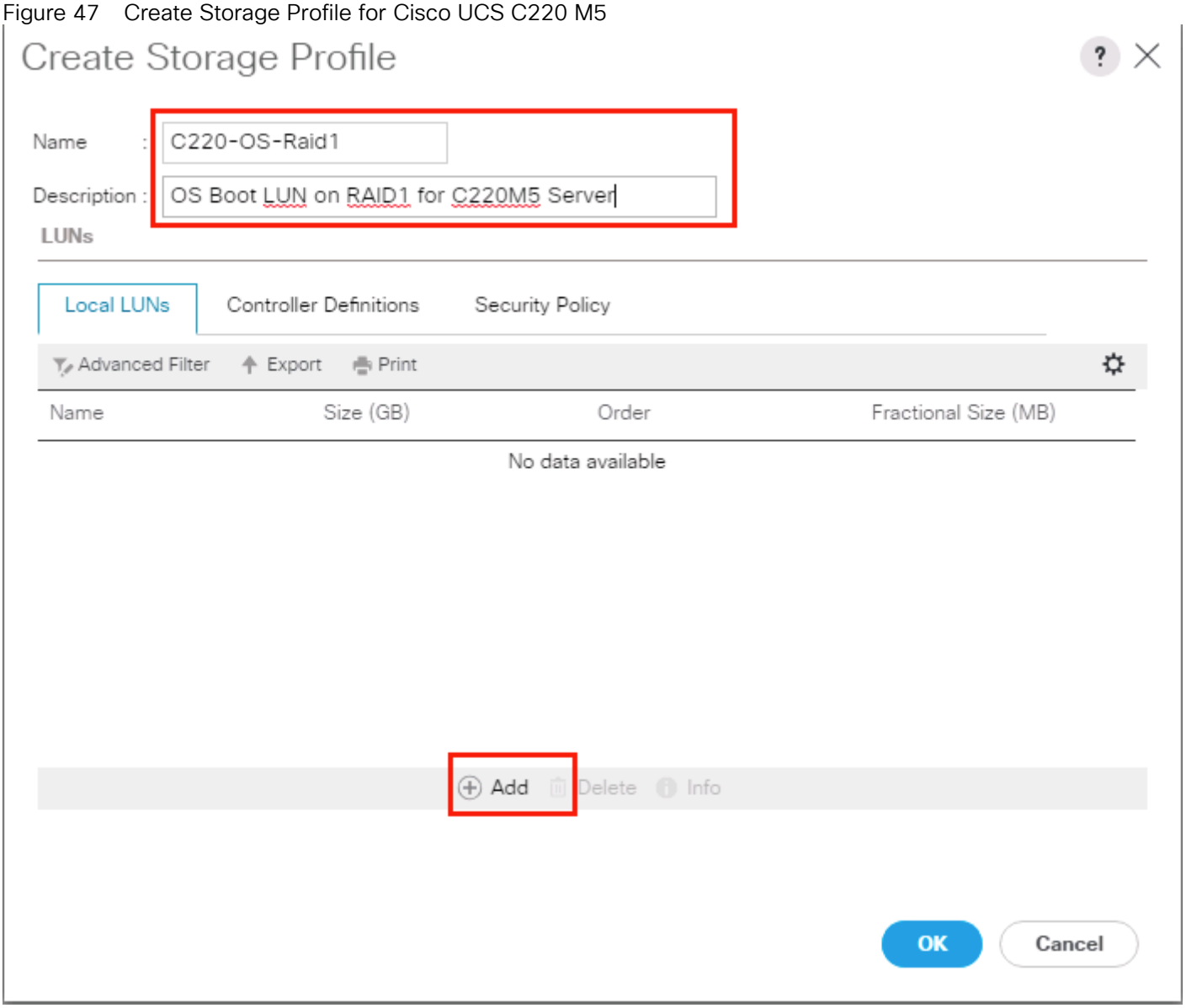
21. Repeat these steps to create the Storage Profile for the bottom Node of the Cisco UCS S3260 Storage Server and name it "S3260-BottomNode."

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

To create a Storage Profile for the Cisco UCS C220 M5, complete the following steps:

1. Select **Storage** in the left pane 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 47 Create Storage Profile for Cisco UCS C220 M5



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

23. Configure as follows:

    Create Local LUN

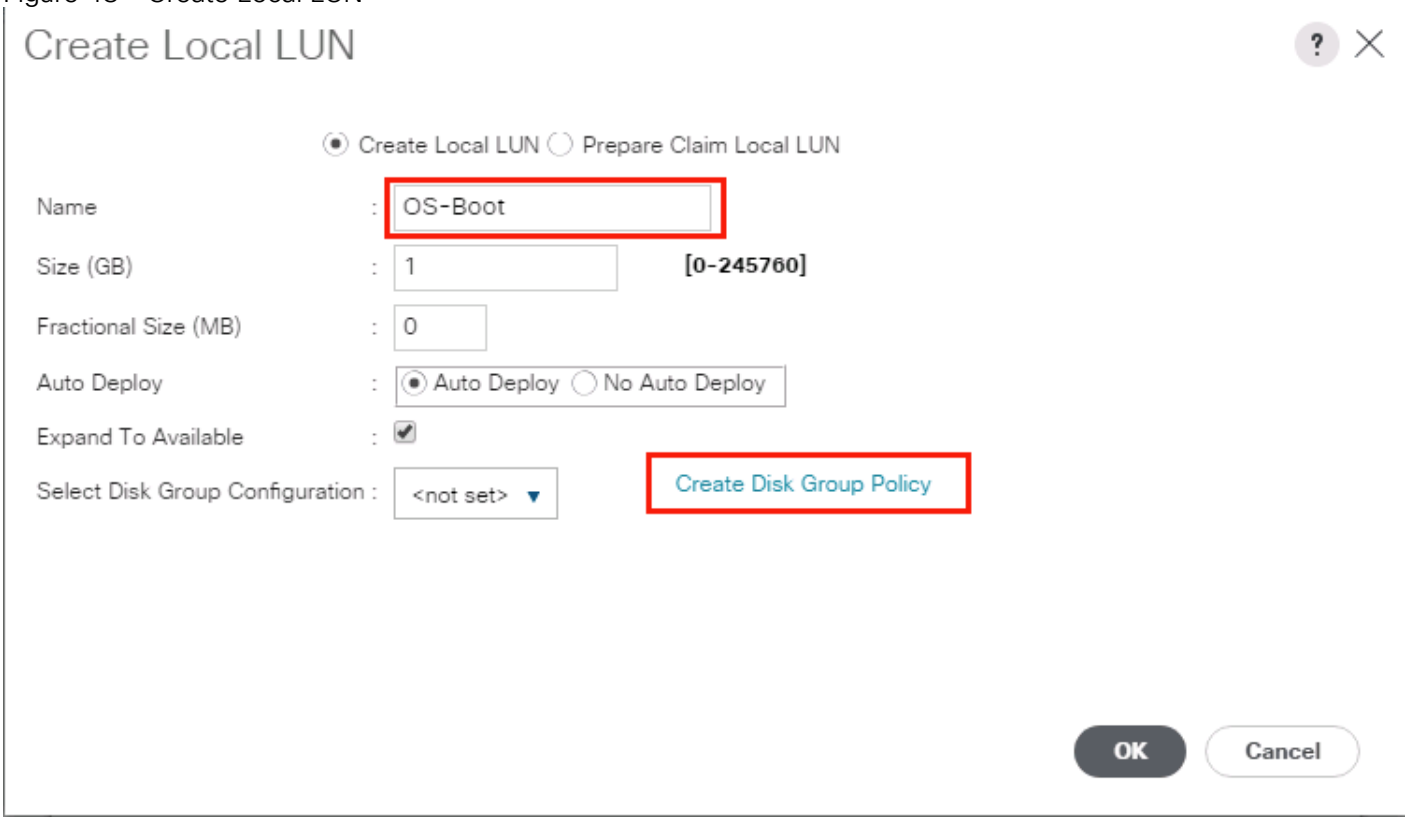
    Size (GB) = 1

    Fractional Size (MB) = 0

    Select Expand To Available

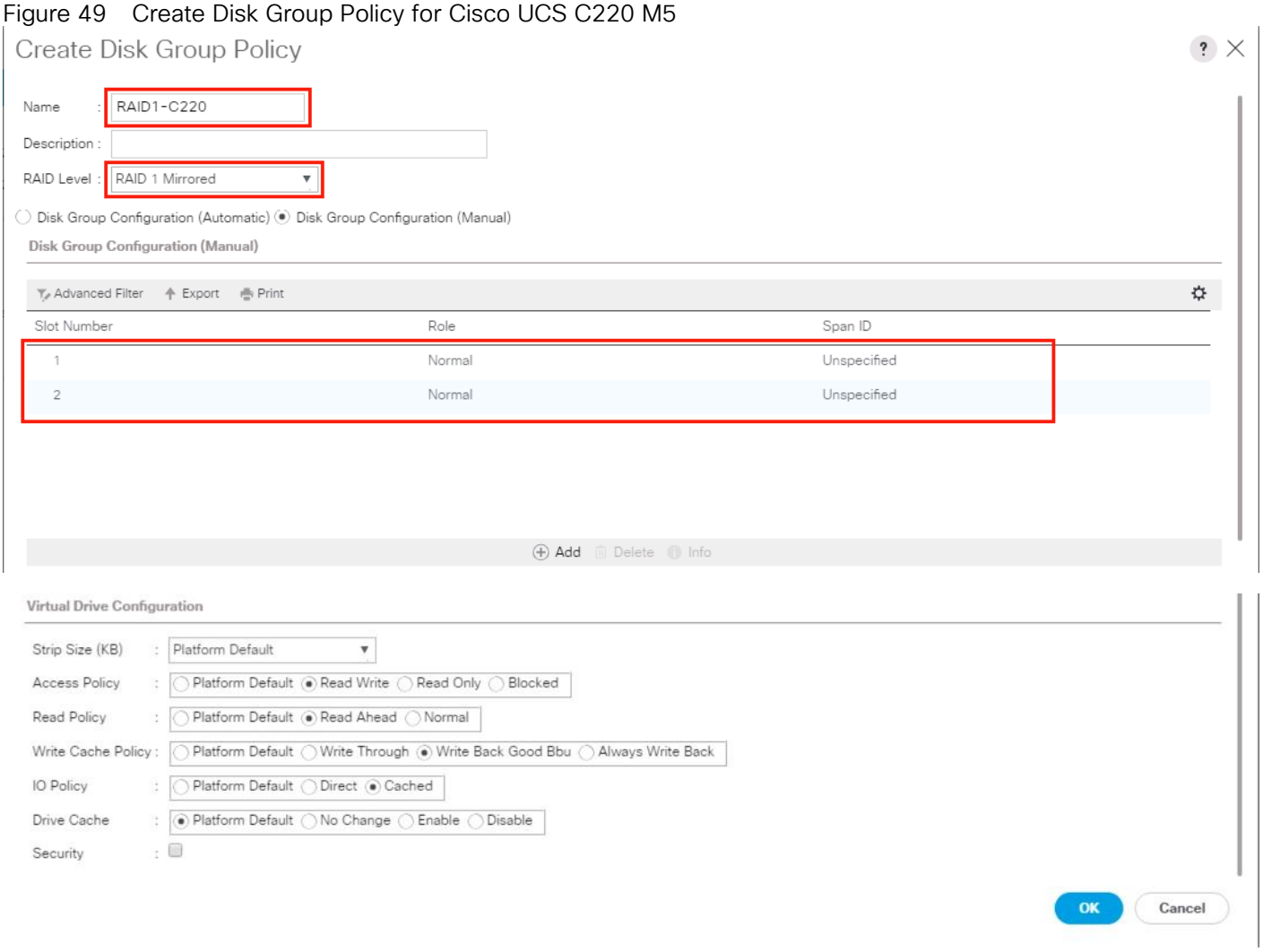
    Auto Deploy

Figure 48 Create Local LUN



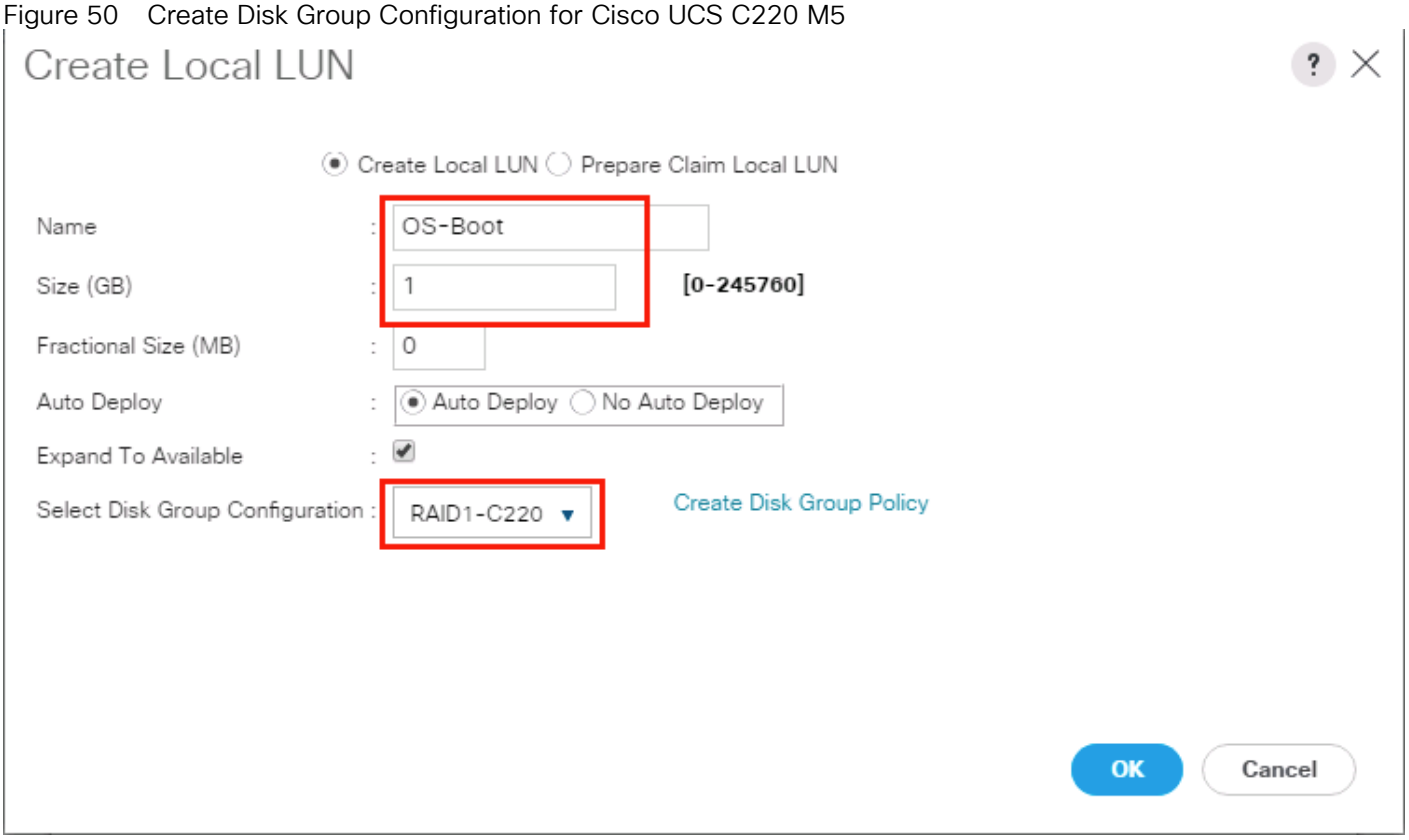
24. Click Create Disk Group Policy to Create RAID1 LUN.
25. Type in **RAID1-C220** in the **Name** field.
26. (Optional) Enter a description in the **Description** field.
27. RAID Level = RAID 1 Mirrored.
28. Select Disk Group Configuration (Manual).
29. Click **Add**.
30. Type in **1** for **Slot Number**.
31. Click **OK** and then again **Add**.
32. Type in **2** for **Slot Number**.
33. 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."
34. Click **OK** and then click **OK** again.

Figure 49 Create Disk Group Policy for Cisco UCS C220 M5



35. Select the previously created Disk Group Policy for the C220 M5 Boot Disks with the radio button under **`Select Disk Group Configuration`**.

Figure 50 Create Disk Group Configuration for Cisco UCS C220 M5



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

Creating a Service Profile Template for S3260 Storage Server

Create Service Profile Template for Cisco UCS S3260 Storage Server Top and Bottom Node

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

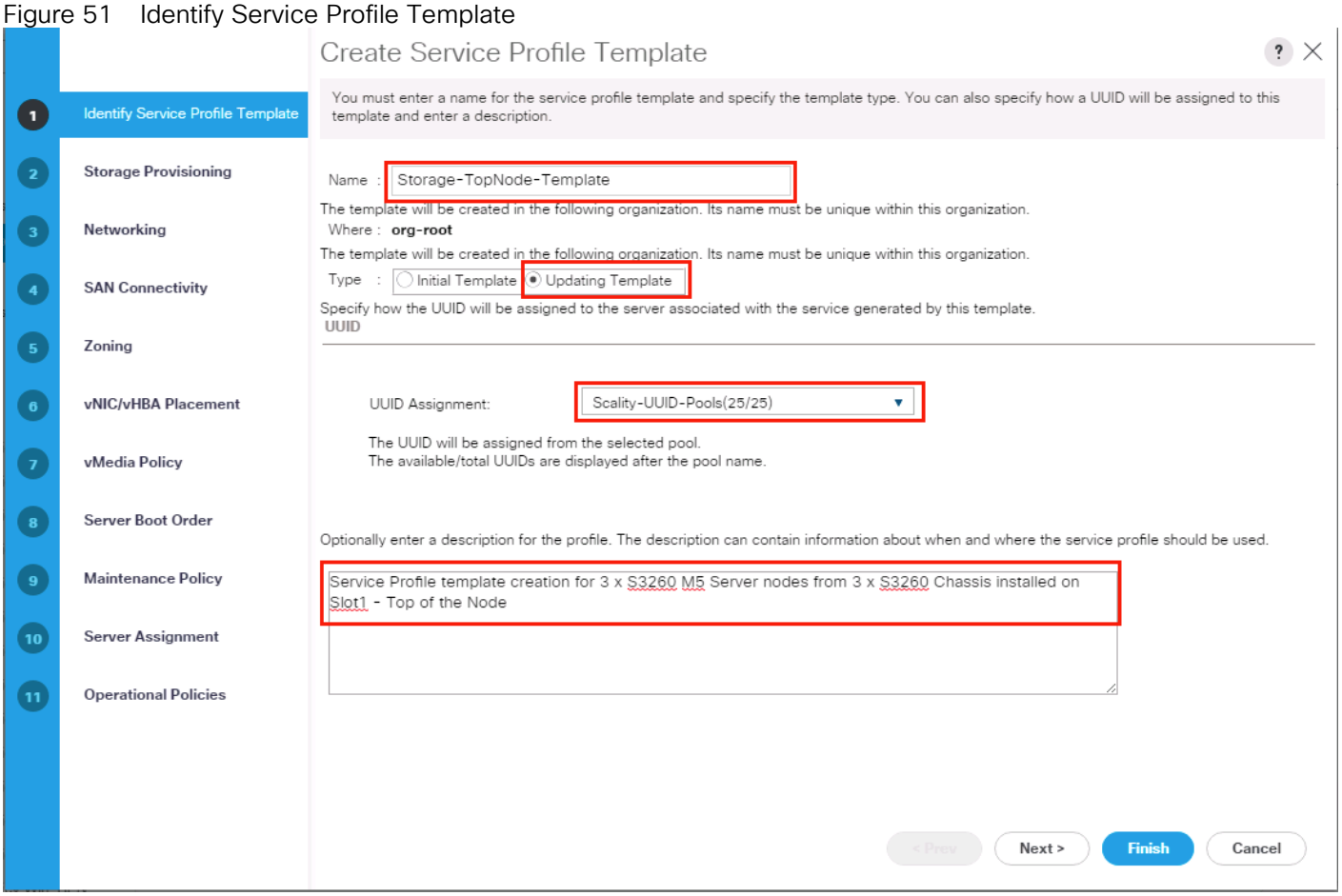
1. Select **Servers** in the left pane of the Cisco UCS Manager GUI.
2. Go to Servers > Service Profile Templates > root and right-click Create Service Profile Template.

Identify Service Profile Template

To identify the Service Profile template, complete the following steps:

1. Type in "Storage-TopNode-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 51 Identify Service Profile Template



5. Click **Next**.

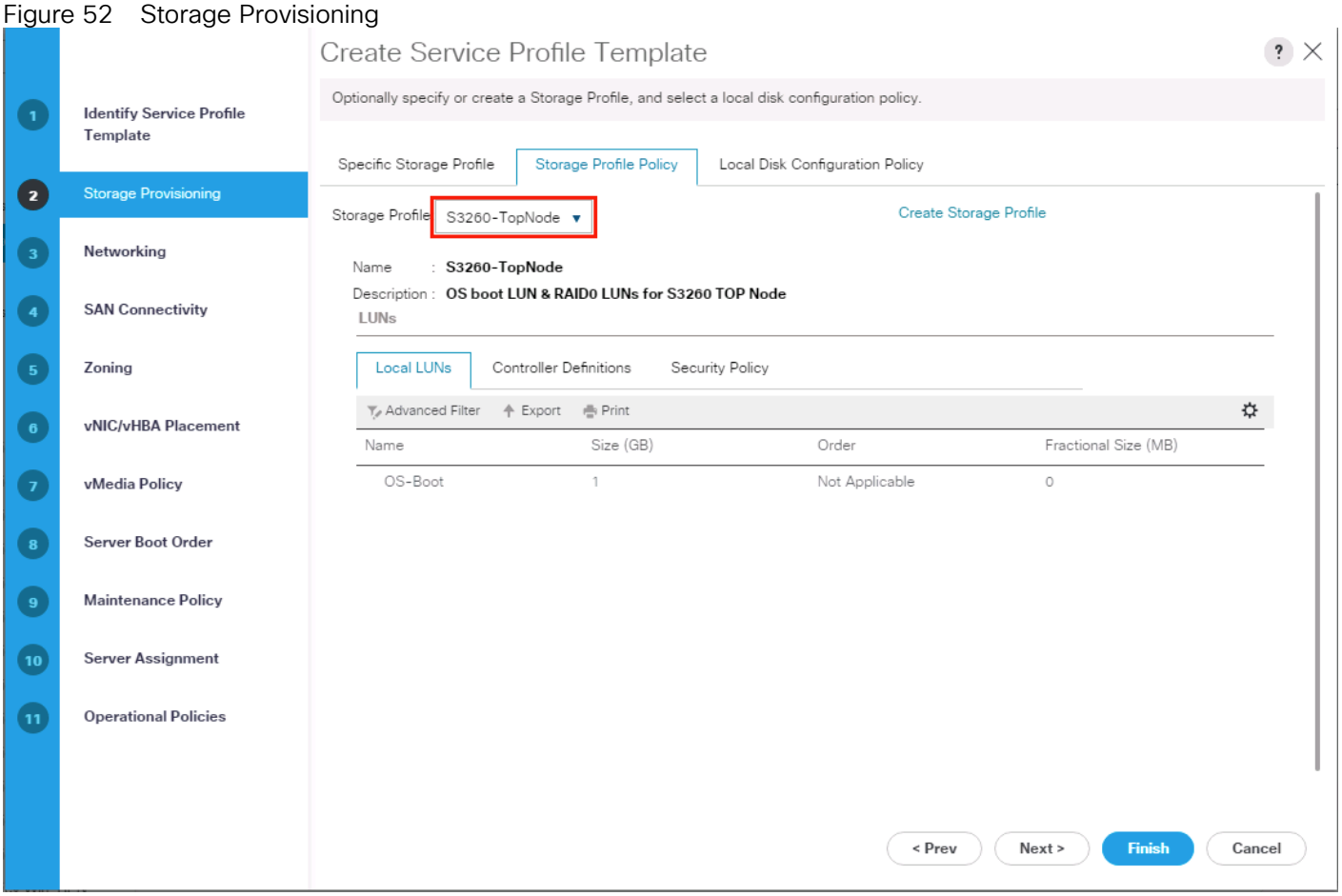
Storage Provisioning

To provision the storage profile, complete the following steps:

1. Go to the **Storage Profile Policy** tab and select the Storage Profile **S3260-TopNode** for the top node of the Cisco UCS S3260 Storage Server you created before.

2. Click **Next**.

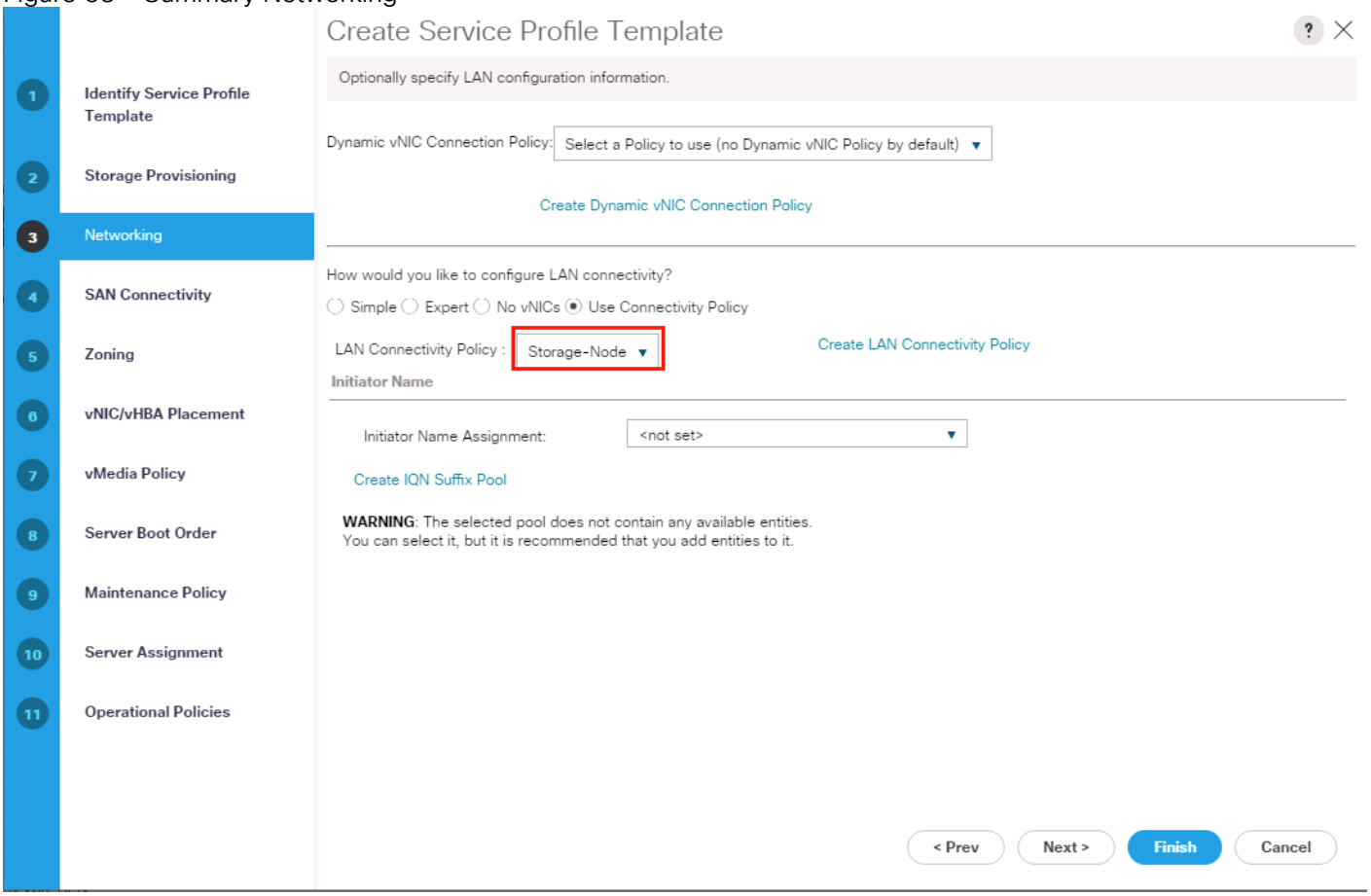
Figure 52 Storage Provisioning



Networking

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.

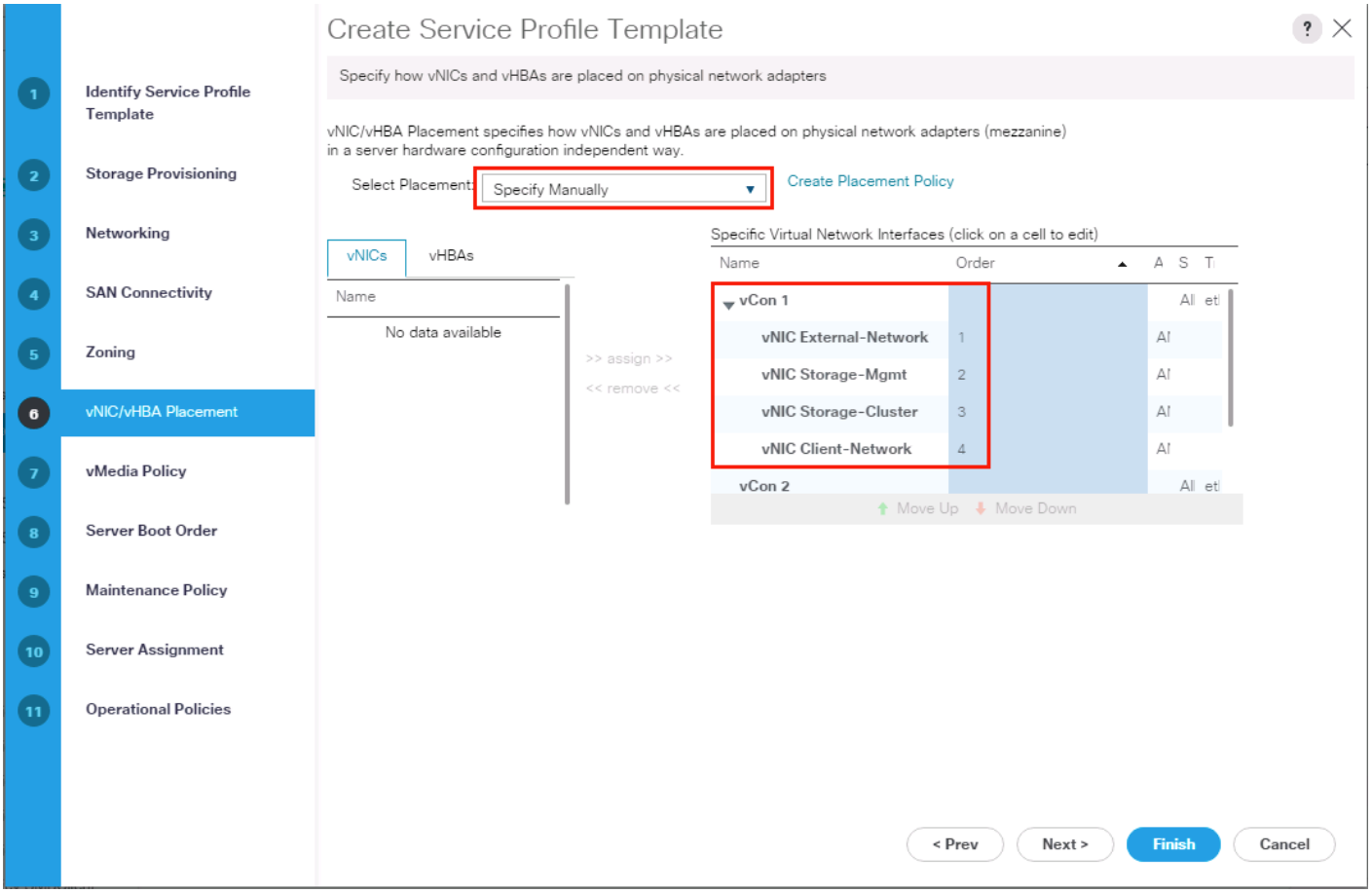
Figure 53 Summary Networking



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

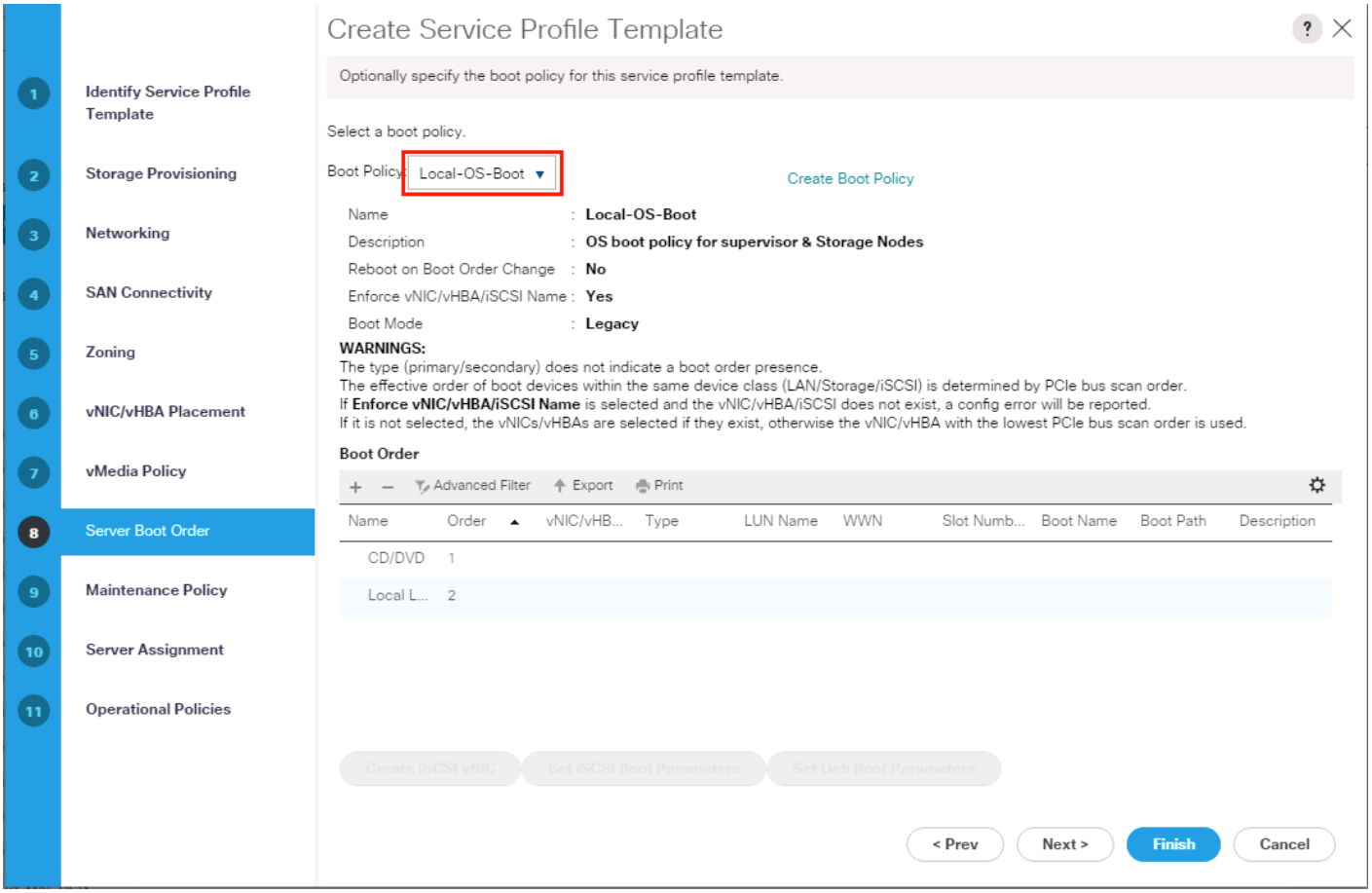
1. Select **Specify Manually** form the drop-down list.
2. Under PCI order section, Sort all the vNICs.
3. Make sure the vNICs order are listed as External-Network > 1, then followed by Storage-Mgmt > 2 and Storage-Cluster > 3 Client-Network > 4.



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

Server Boot Order

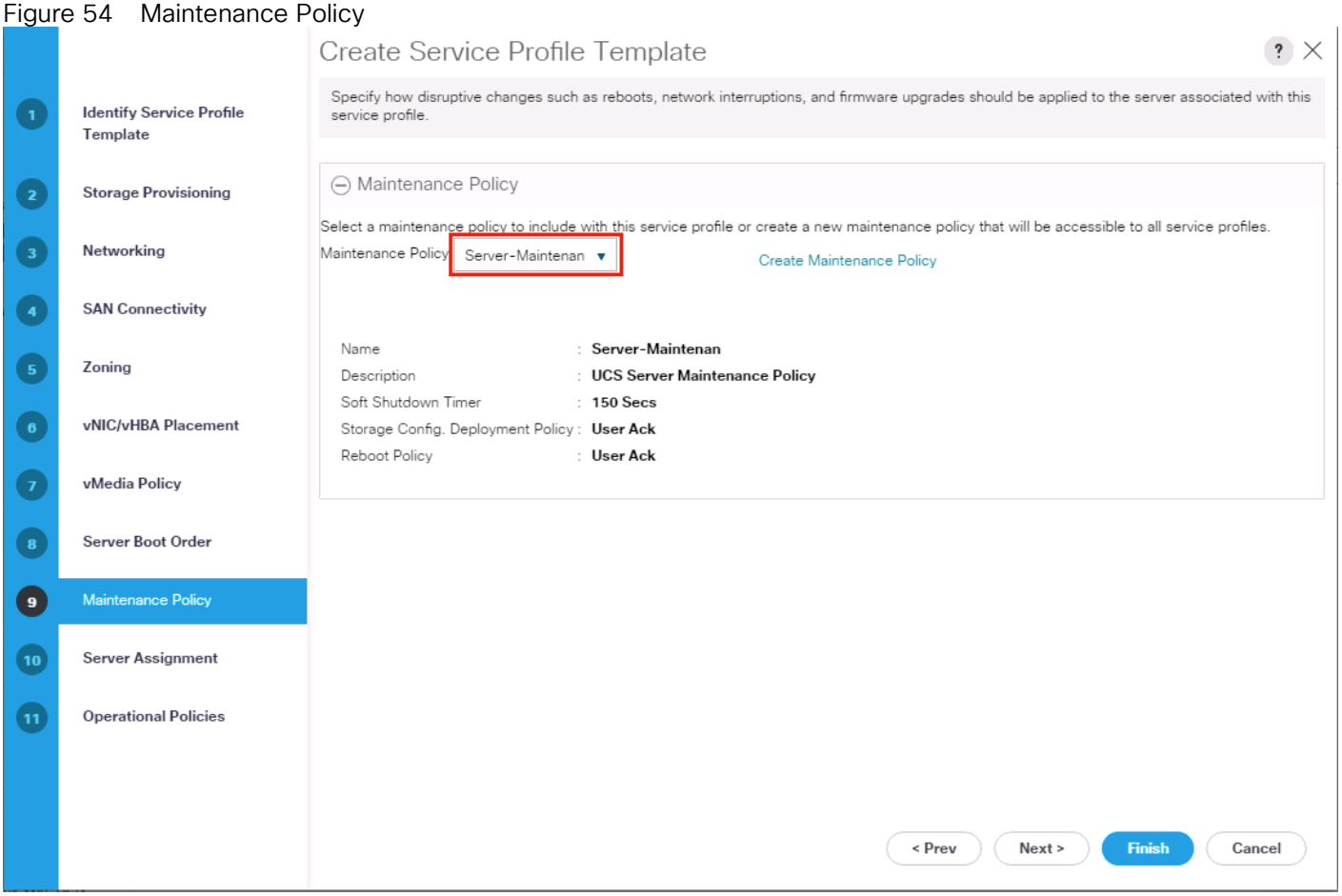
1. Select the Boot Policy “Local-OS-Boot” you created before under Boot Policy.
2. Server Boot Order.
3. Click Next.



Maintenance Policy

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

Figure 54 Maintenance Policy



2. Click **Next**.
3. Under Server Assignment, Leave everything else untouched.
4. Click **Next**.

Operational Policies

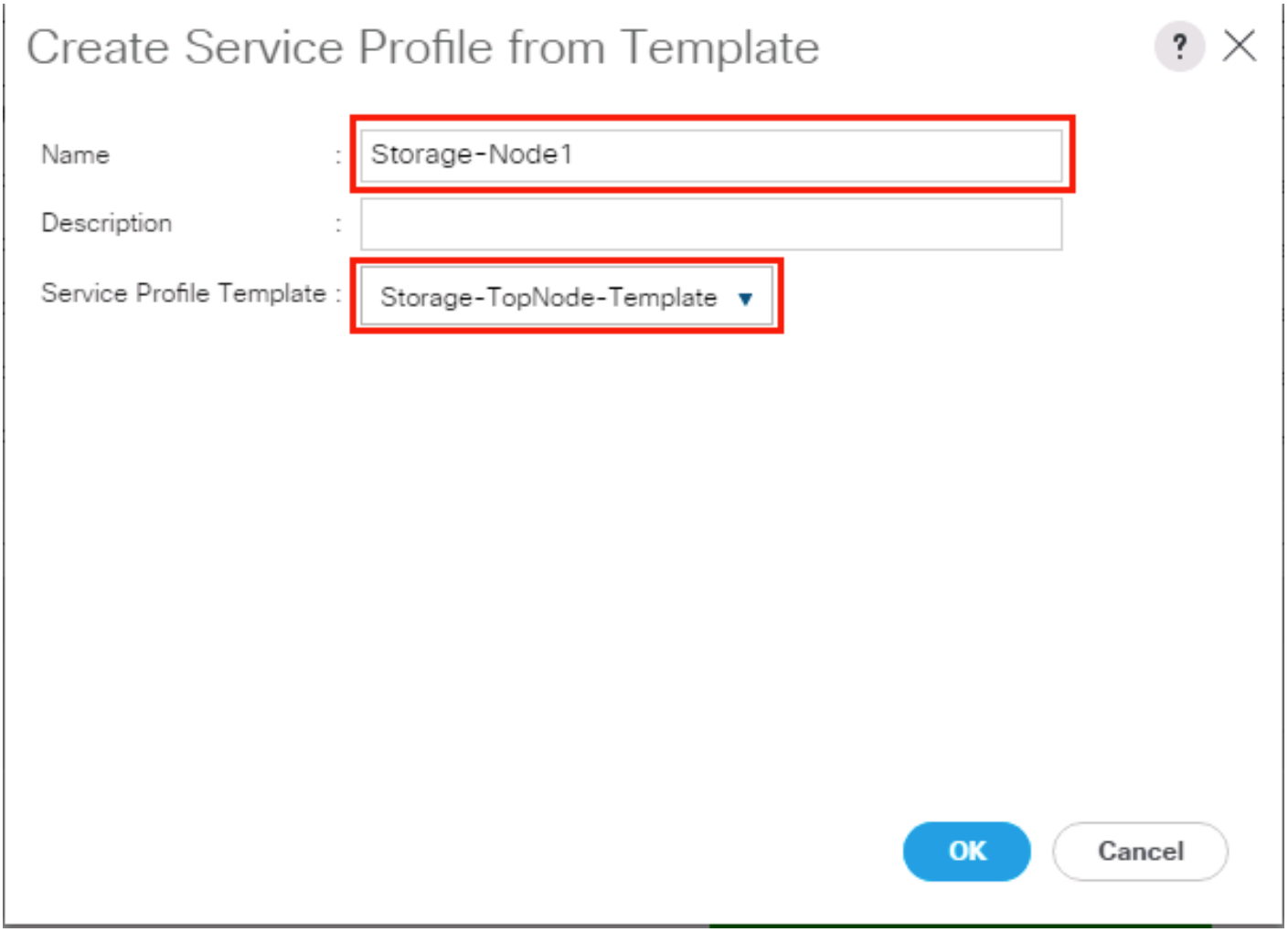
1. Click **Finish** and then click **OK**.
2. Repeat the steps for the bottom node of the Cisco UCS S3260 Storage Server by naming template as "Storage-BottomNode-Template."
3. During Storage Provisioning tab, choose the Storage Profile for the bottom node "S3260-BottomNode" you created previously.

Create Service Profiles from Template

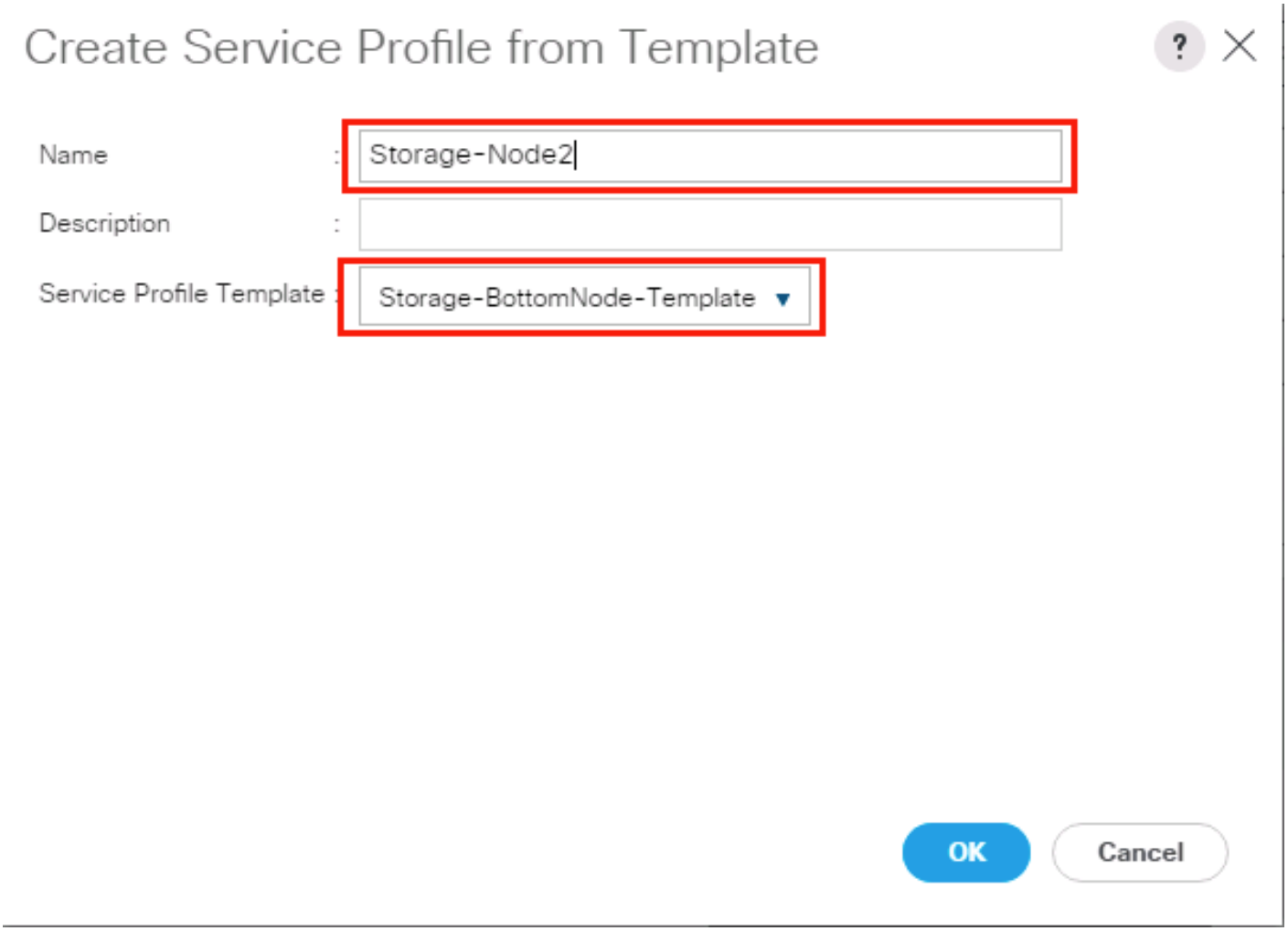
This section details how to create the appropriate Service Profiles from the previous Service Profile Templates. To create the first profile for the top node of the Cisco UCS S3260 Storage Server, complete the following steps:

1. Select **Servers** from the left pane of 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-TopNode-Template"** as the **Service Profile Template** you created before for the top node of the Cisco UCS S3260 Storage Server.
5. Click **OK** and then click **OK** again.



6. Repeat these steps to create Service Profiles for the remaining S3260 M5 server top Nodes from the Template that belongs to top Node "Storage-TopNode-Template". Make sure you name it as "Storage-Node3, Storage-Node5" respectively.
7. For the remaining M5 nodes, again Navigate to Servers > Service Profiles and right-click Create Service Profile from Template.
8. Type in **Storage-Node2** in the Name Prefix field.
9. Choose "**Storage-BottomNode-Template"** as the **Service Profile Template** you created before for the top node of the Cisco UCS S3260 Storage Server.
10. Click **OK** and then click **OK** again.



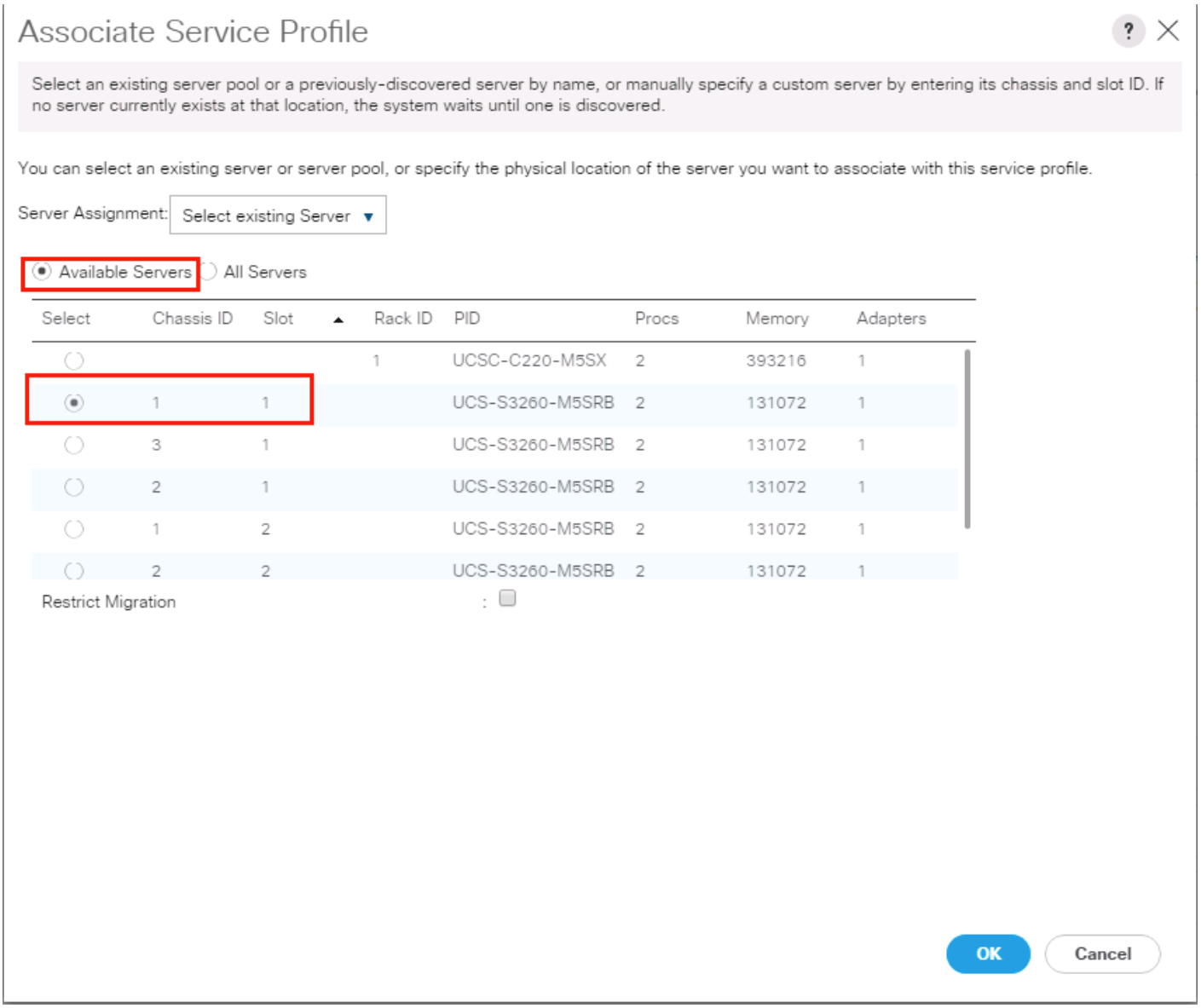
11. Repeat these steps to create Service Profiles for the remaining S3260 M5 server Bottom Nodes from the Template that belongs to bottom Node “Storage-BottomNode-Template”. Make sure you name it as “Storage-Node4, Storage-Node6.”

Associating a Service Profile for Cisco UCS S3260 M5 Server

To associate all the “Storage-NodeX” Service Profiles to the Cisco UCS S3260 M5 Storage Servers, complete the following steps:

1. Select **Servers** from the left pane of 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 Chassis and Slot listed, choose Chassis1/Slot1 for Storage-Node1.

7. Click OK.



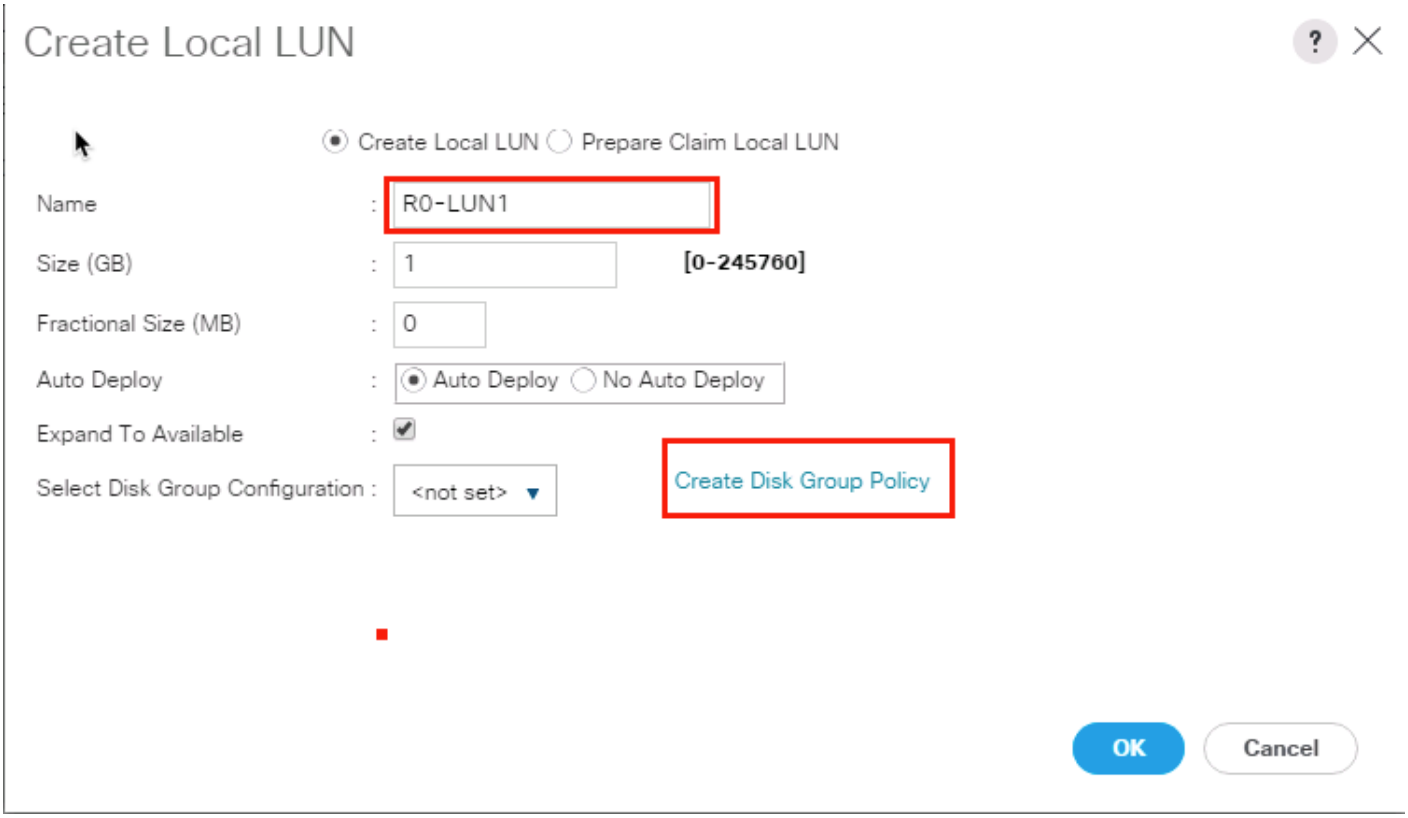
8. Repeat these steps to the Associate Remaining Service profiles “Storage-NodeX” for the Cisco UCS S3260 M5 storage server as listed in the table below.

Service Profile Template	Service Profile	S3260 Chassis	Server Slot ID
Storage-TopNode-Template	Storage-Node1	1	1
Storage-BottomNode-Template	Storage-Node2	1	2
Storage-TopNode-Template	Storage-Node3	2	1
Storage-BottomNode-Template	Storage-Node4	2	2
Storage-TopNode-Template	Storage-Node5	3	1
Storage-BottomNode-Template	Storage-Node6	3	2

Create Individual RAID0 LUNs for Cisco UCS S3260 Top Loading HDDs

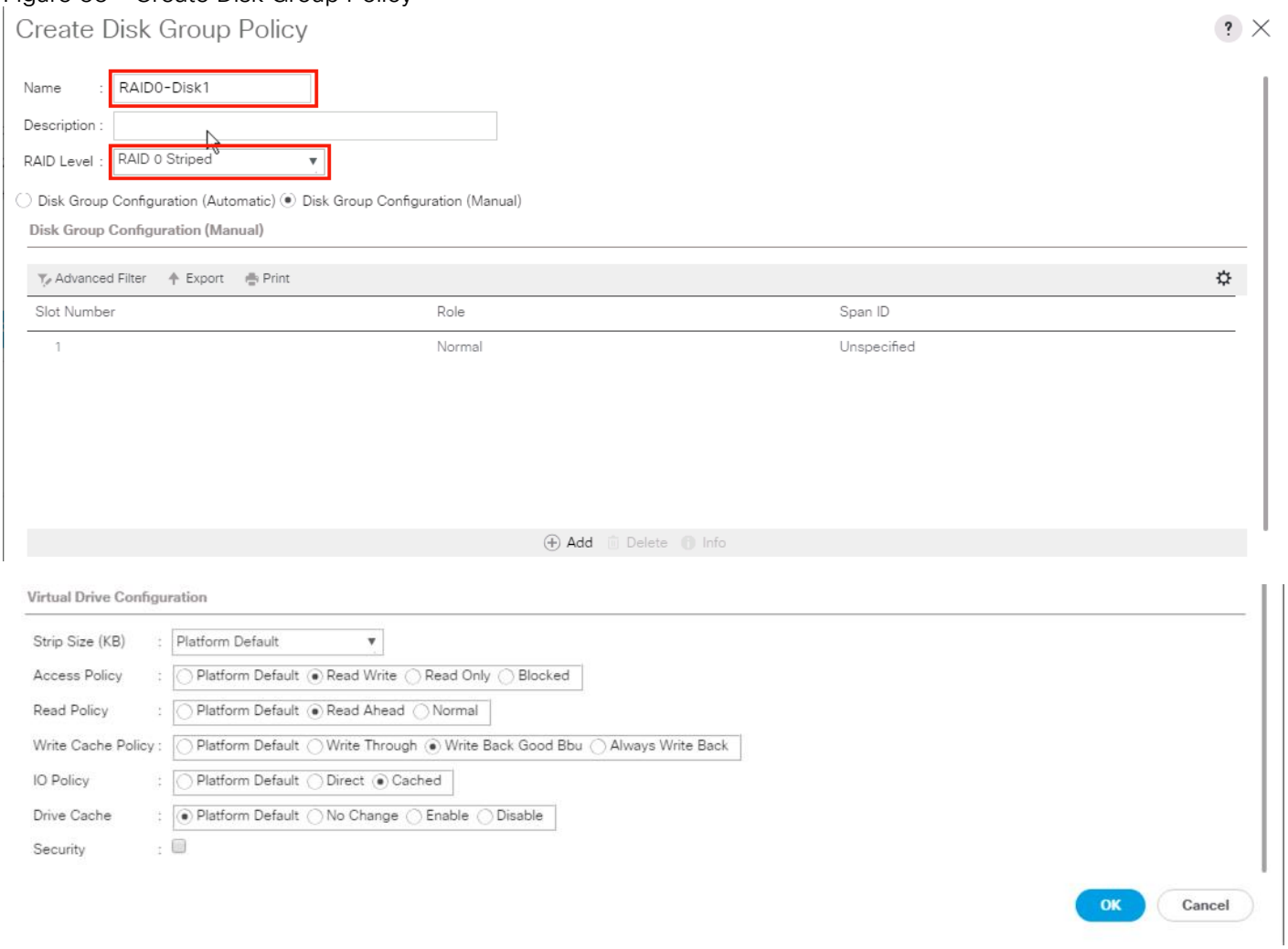
To create individual RAID0 LUNs for the top loading HDDs from Cisco UCS S3260 Storage Server, complete the following steps:

1. Select **Storage** in the left pane of the Cisco UCS Manager GUI.
2. Go to Storage > Storage Profiles -> root and right-click the previously created Storage Profile "S3260-Top-Node"
3. Select "Create Local LUN" radio button.
4. Type in R0-LUN1 in the name field.
5. Size (GB) = 1.
6. Fractional Size (MB) = 0.
7. Auto Deploy.
8. Select Expand To Available.



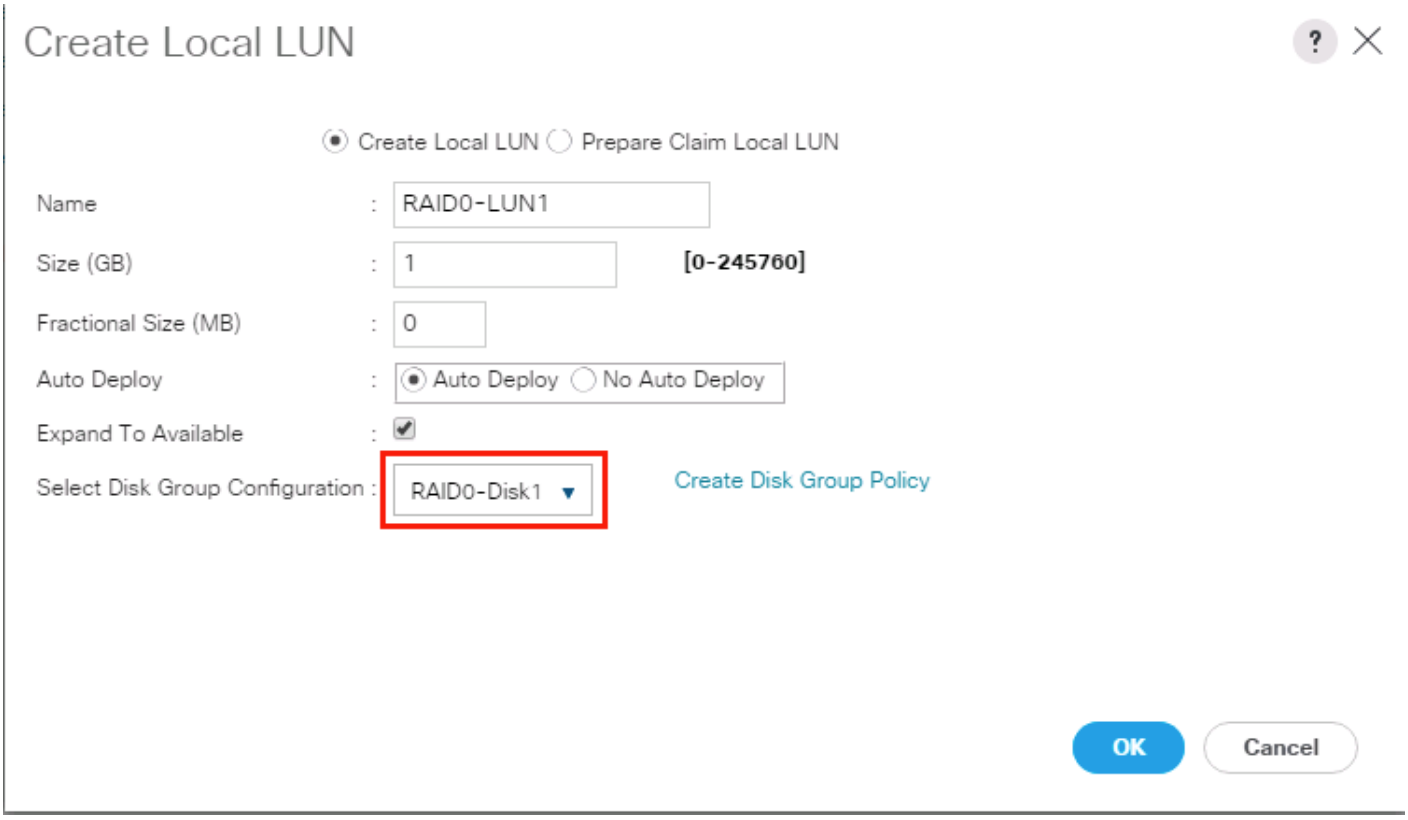
9. Click “Create Disk Group Policy” to Create RAID0 LUN.
10. Type in **`RAID0-Disk1`** in the `Name` field.
11. (Optional) Enter a description in the **`Description`** field.
12. RAID Level = RAID 0 Striped.
13. Select Disk Group Configuration (Manual).
14. Click **`Add`**.
15. Type in 1 Slot Number.
16. Click **`OK`**.
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.”
    c. Click **`OK`** and then **`OK`**.

Figure 55 Create Disk Group Policy



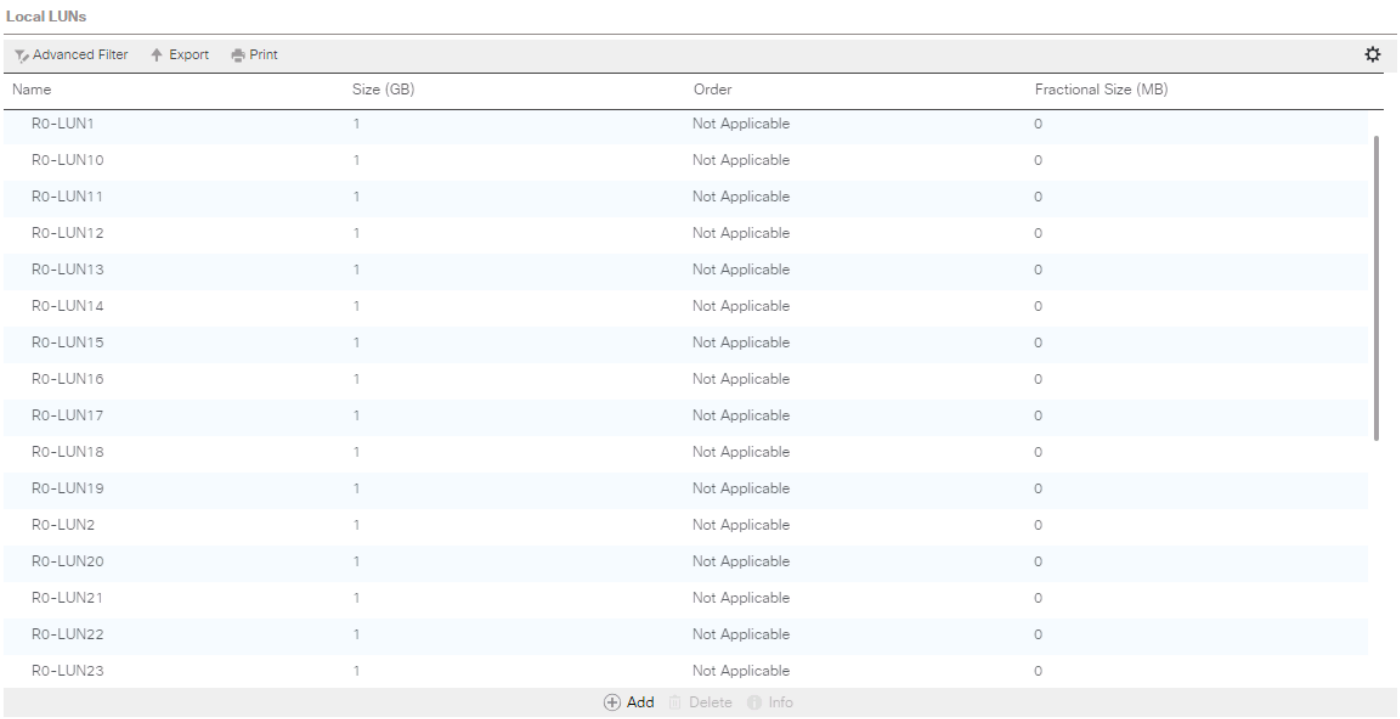
18. Select your previously created Disk Group Policy for the drop-down list under **`Select Disk Group Configuration`**.

19. Select Disk Group Configuration.



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

21. Create the RAID0 LUNs for the remaining top loading HDDs by following all the steps.



Make sure to choose “Storage-TopNode” Storage Profile for creating RAID0 LUNs for HDDs installed from Slot 1-28. Then choose “Storage-BottomNode” Storage Profile for creating RAID0 LUNs for HDDs installed from Slot 29-56.

Create Service Profile for Cisco UCS C220 M5 Server for Scality Supervisor Node

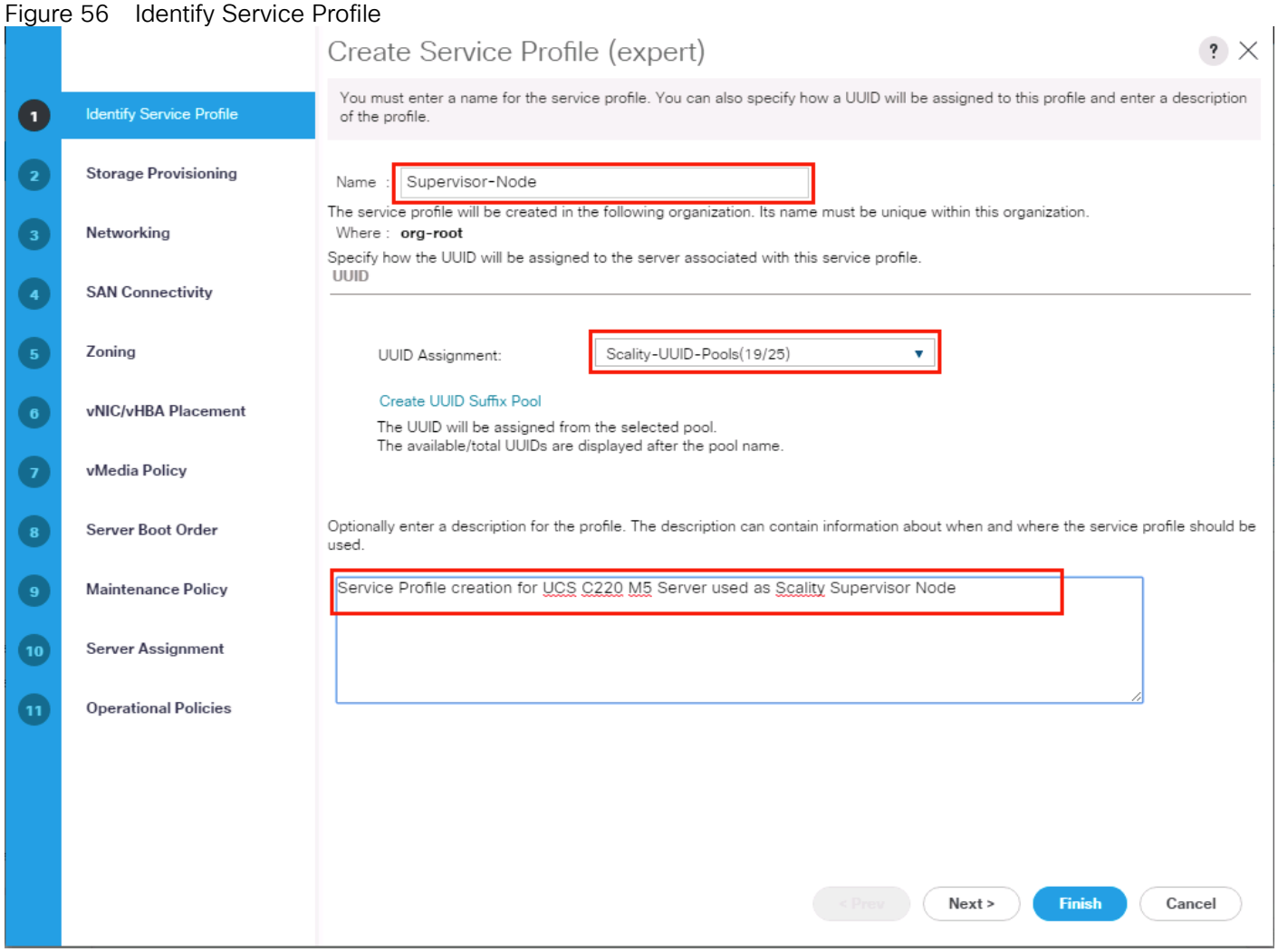
To create a Service Profile, complete the following steps:

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

1. Type in Supervisor-Node 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 56 Identify Service Profile



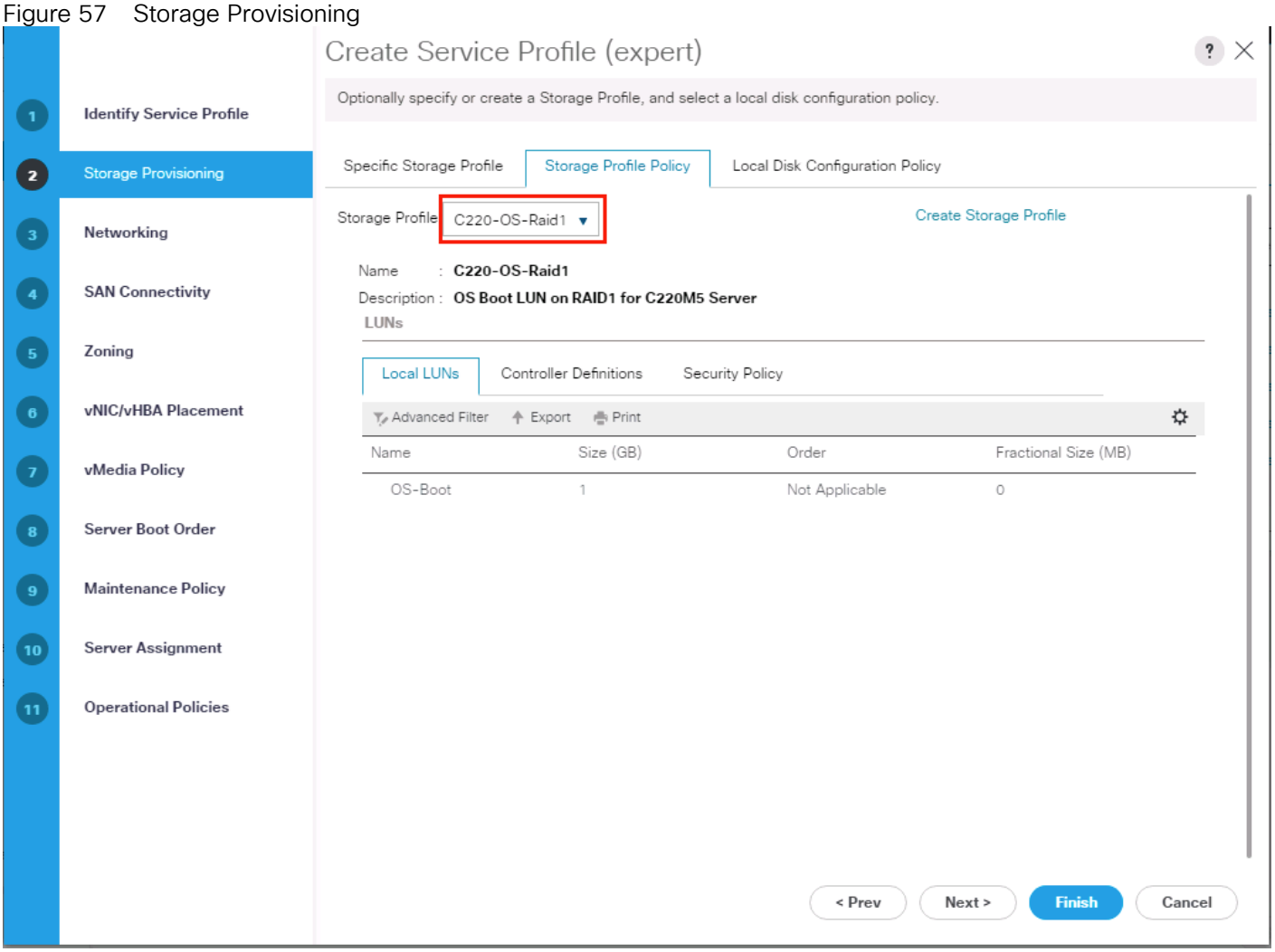
4. Click **Next**.

Storage Provisioning

1. Go to the **Storage Profile Policy** tab and select the Storage Profile **C220-OS-Raid1** for the top node of the Cisco UCS S3260 Storage Server you created before.

2. Click **Next**.

Figure 57 Storage Provisioning



Networking

1. Keep the Dynamic vNIC Connection Policy field at the default.

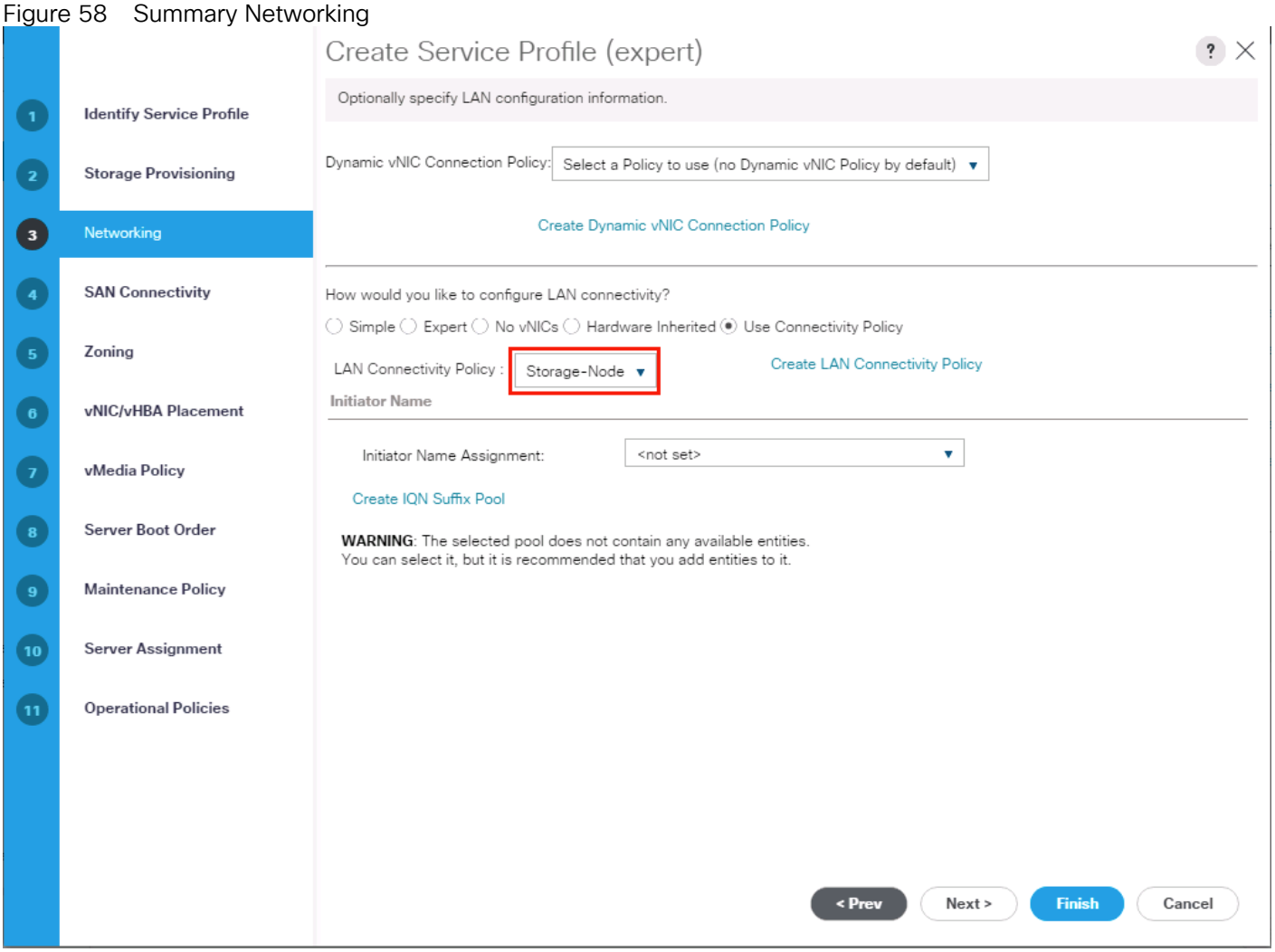
2. Select LAN connectivity to Use Connectivity Policy created previously.

3. From the LAN Connectivity drop-down list, select “Storage-Node” previously created.

Scality Supervisor Node and Storage-Nodes use the same VNIC interfaces.

4. Click Next.

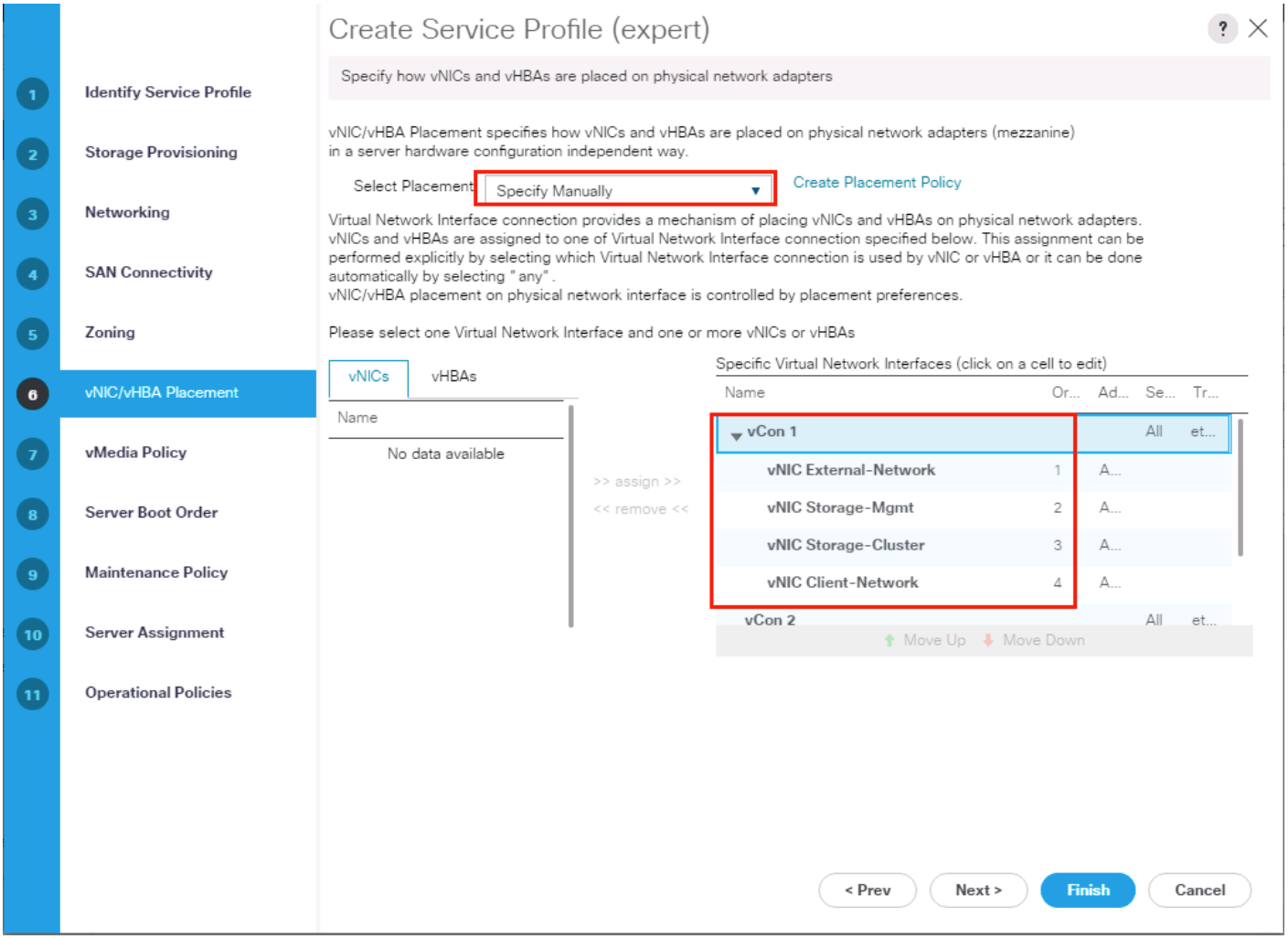
Figure 58 Summary Networking



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

1. Select **Specify Manually** form the drop-down list.
2. Under PCI order section, Sort all the vNICs.
3. Make sure the vNICs order 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.

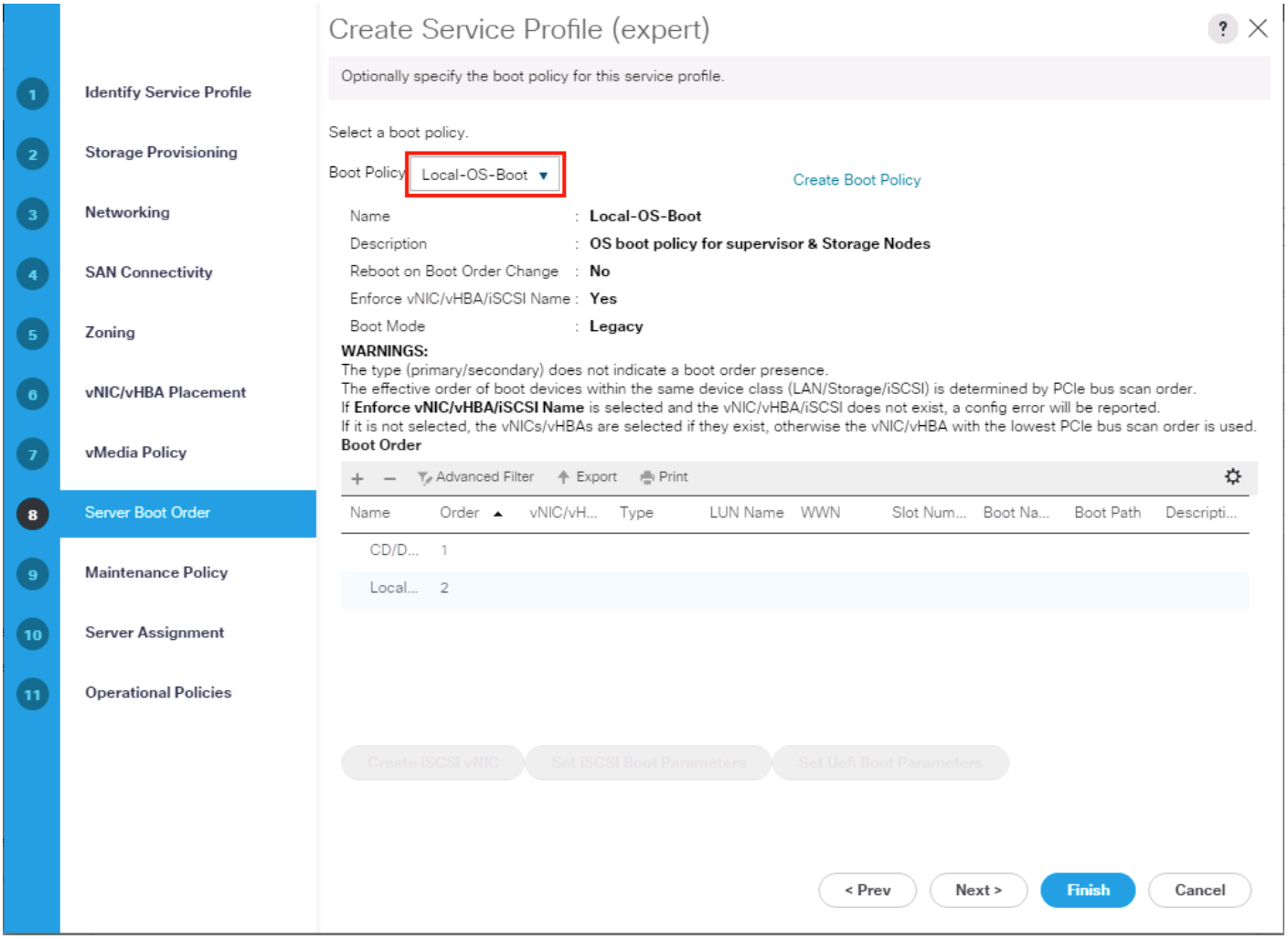
5. Click **Next**.

Server Boot Order

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

2. Server Boot Order.

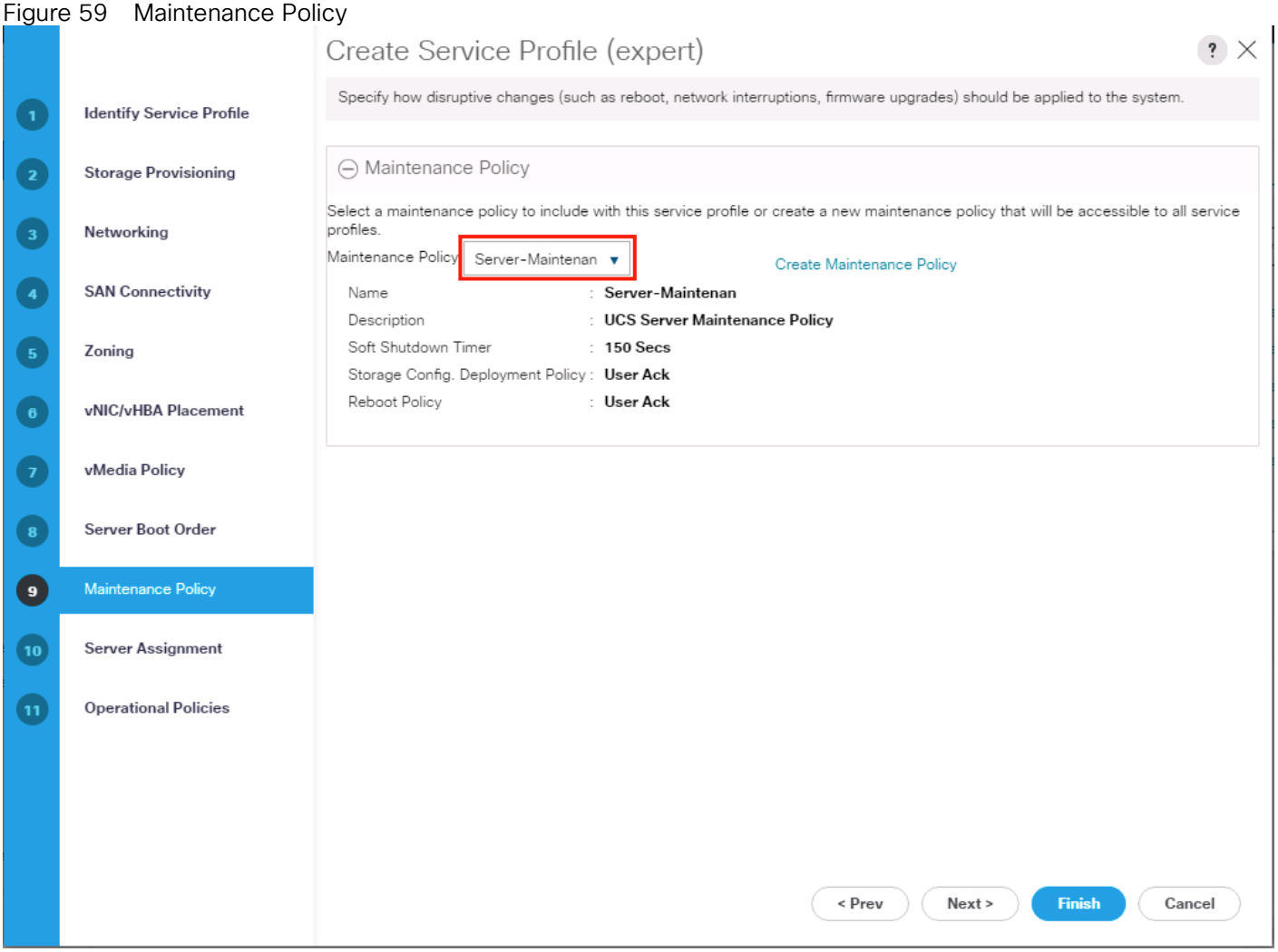
3. Click Next.



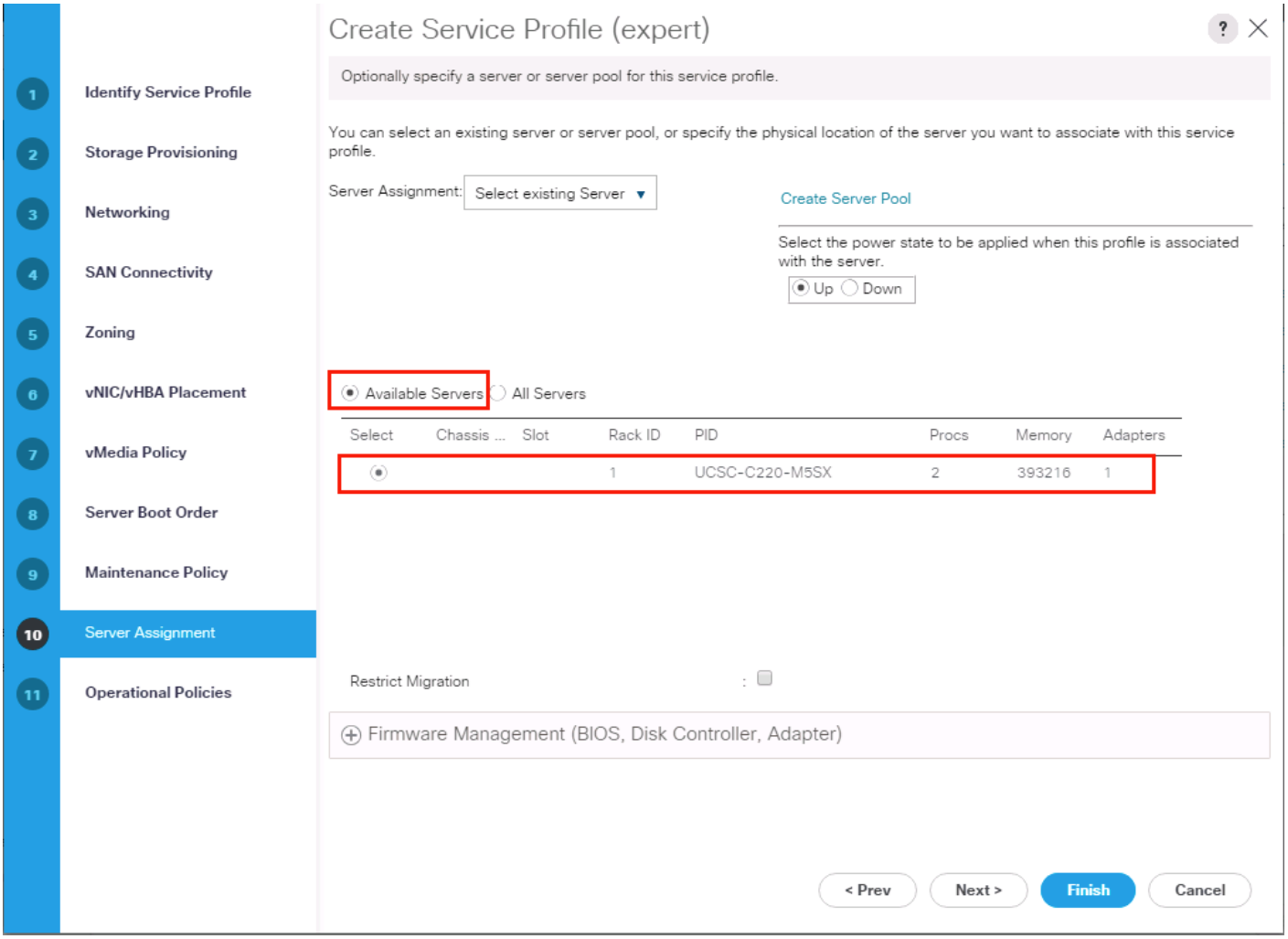
Maintenance Policy

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

Figure 59 Maintenance Policy



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



6. Click **Next**.

Operational Policies

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

Creating Port Channel for Network Uplinks

Create Port Channel for Fabric Interconnect A/B

To create Port Channels to the connected Nexus 9332PQ switches, complete the following steps:

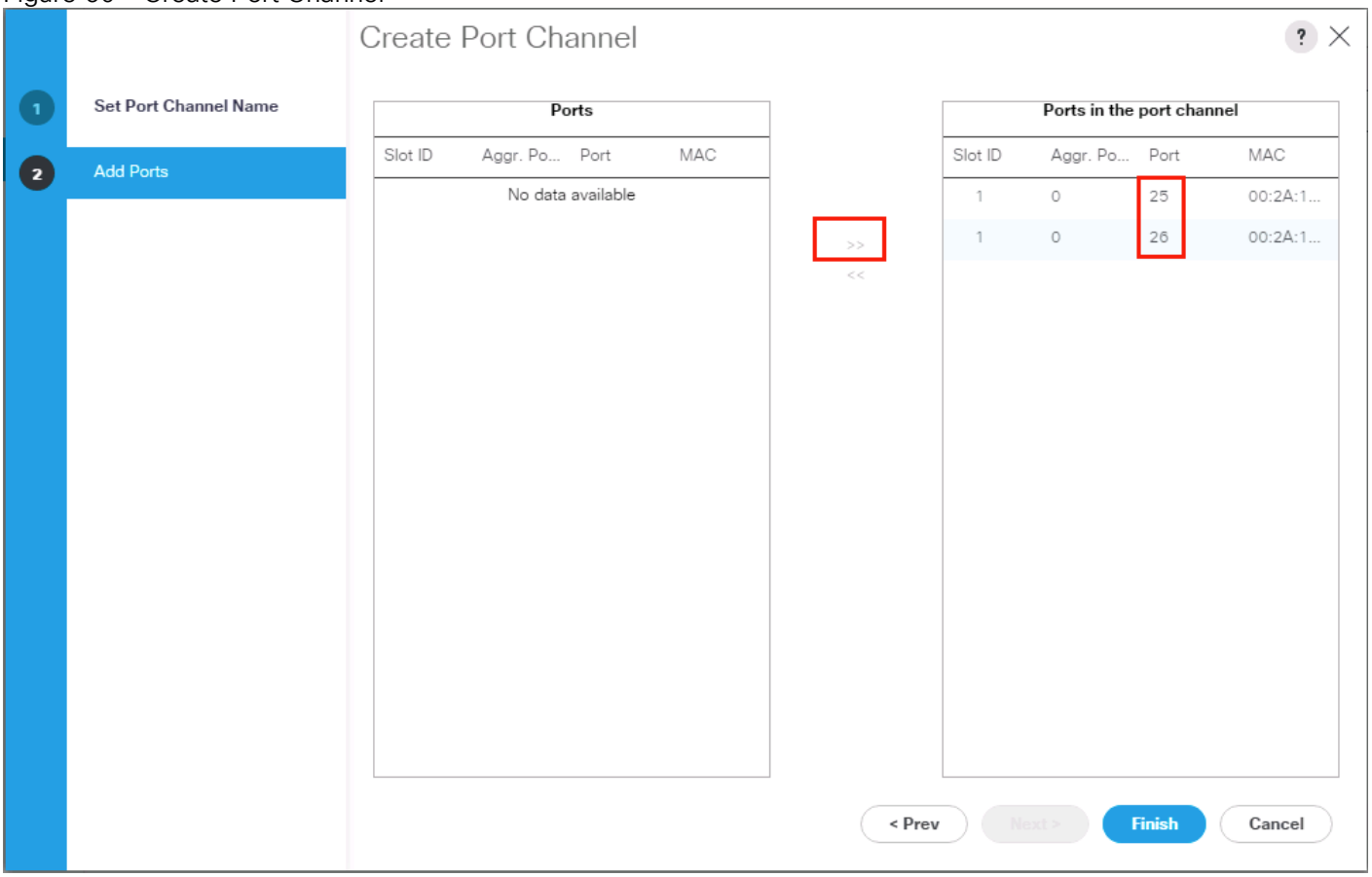
1. Select the **LAN** tab in the left pane of the Cisco UCS Manager GUI.
2. Go to LAN > LAN Cloud > Fabric A > Port Channels and right-click Create Port Channel.
3. Type in **ID 20**.
4. Type in **vPC20** in the `Name` field.

5. Click Next.

6. Select the available ports on the left **25-26** 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 60 Create Port Channel



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

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

10. Type in **ID 30**.

11. Type in **VPC30** name in the Name field.

12. Click **Next**.

13. Select the available ports on the left **25-26** 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 9332PQ switches is finished and next is the installation of the Red Hat Enterprise Linux 7.5 Operating System.

Installing Red Hat Enterprise Linux 7.5 Operating System

This section provides the detailed procedures to install Red Hat Enterprise Linux 7.5 on Cisco UCS C220 M5 and Cisco UCS S3260 Storage Server. The installation uses the KVM console and virtual Media from Cisco UCS Manager.

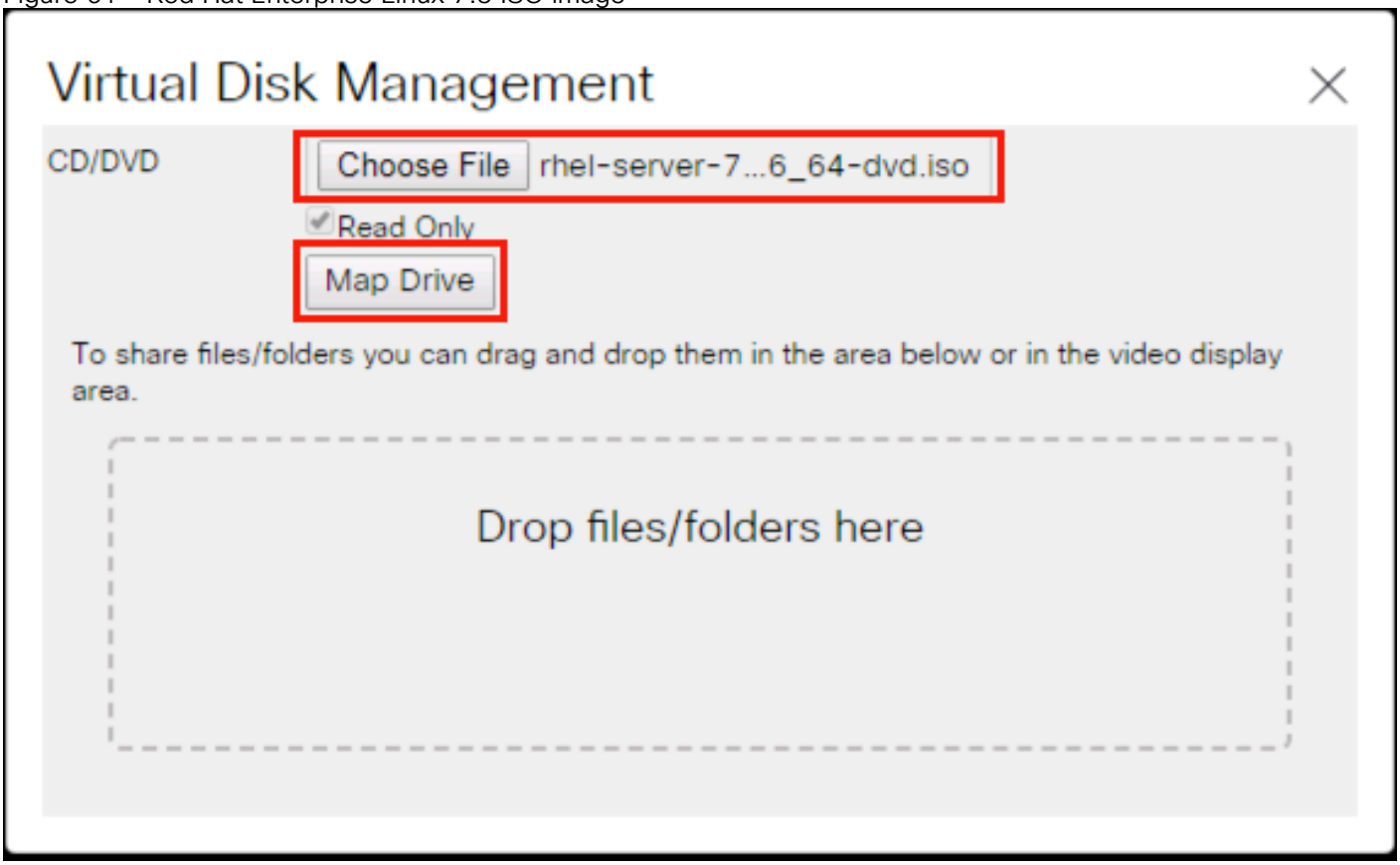
This requires RHEL 7.5 DVD/ISO media for the installation.

Installation of RHEL 7.5 on Cisco UCS C220 M5 and Cisco UCS S3260 M5 Server

To install Red Hat Linux 7.5 operating system on Cisco UCS C220 M5, complete the following steps:

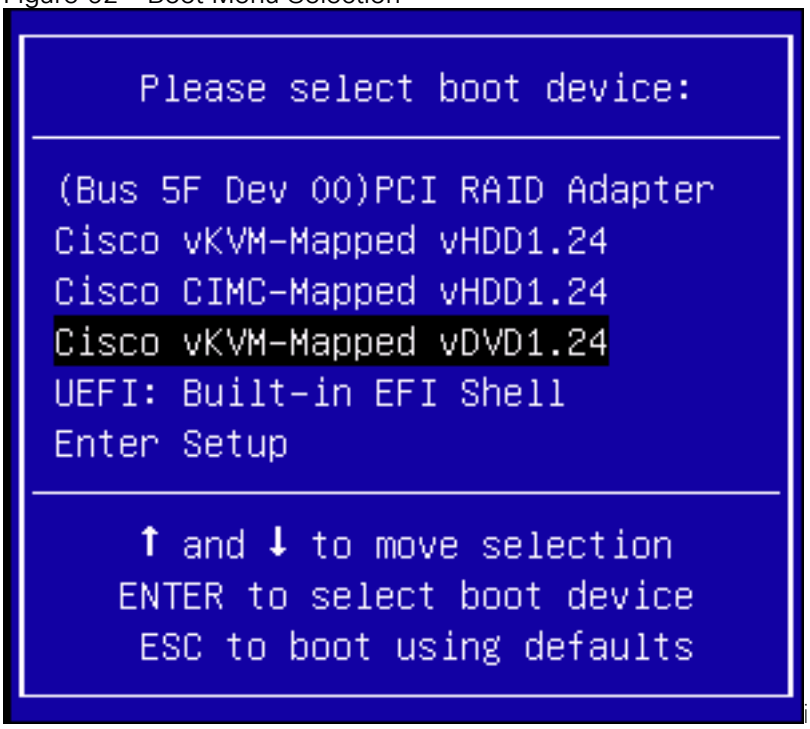
1. Log in to the Cisco UCS Manager and select the **Equipment** tab from the left pane.
2. Go to Equipment > Rack-Mounts > Server > Server 1 (Supervisor) and right-click KVM Console.
3. Launch KVM Console.
4. Click the **Activate Virtual Devices** in the Virtual Media tab.
5. In the UCS KVM window, select the Virtual Media tab and then click **CD/DVD**.
6. Click Choose File and Browse to the Red Hat Enterprise Linux 7.5 installation ISO image and select then click "**Map Drive.**"

Figure 61 Red Hat Enterprise Linux 7.5 ISO image



7. In the KVM window, select the **Macros > Static Macros > Ctrl-Alt-Del** button in the upper left corner.
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 62 Boot Menu Selection



in

11. When the Red Hat Enterprise Linux 7.5 installer appears, press the Tab button for further configuration options.
12. At the prompt type:

```
inst.ks=ftp://192.168.10.2/Supervisor.cfg net.ifnames=0 biosdevname=0
ip=192.168.10.160::192.168.10.1:255.255.255.0:Supervisor: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 Supervisor Node is in Appendix A. This Kickstart file for the Cisco UCS S3260 M5 Server for Storage Nodes is in Appendix B.

13. Repeat these steps to install RHEL7.5 on all the UCS S3260 M5 storage servers.

Preparation of all Nodes for Scality RING Installation

Before installing Scality RING, make sure you prepare all nodes with certain configurations.

A summary of the prerequisites for the entire installation with the appropriate changes to the current environment is listed below.

Step 1 – Configuring Network Time Protocol

In our Kickstart installation file, a time server is included. To enable Network Time Protocol on all servers and configure them to use the same source, complete the following steps:

1. Install NTP on all servers:

```
# yum –y install ntp
# for i in {1..6}; do ssh storage-node{i} 'yum –y install ntp'; done
```

2. Configure /etc/ntp.conf on Supervisor node only with the following contents:

```
# vi /etc/ntp.conf
driftfile /var/lib/ntp/drift
restrict 127.0.0.1
restrict -6 ::1
server 192.168.10.2
fudge 192.168.10.2 stratum 10
includefile /etc/ntp/crypto/pw
keys /etc/ntp/keys
```

3. Start the ntpd daemon on Supervisor Node:

```
# systemctl enable ntpd
# systemctl start ntpd
# systemctl status ntpd
```

4. Copy ntp.conf from Supervisor node to all the Storage nodes:

```
# cd /etc/
# for i in {1..6}; do scp etc/ntp.conf storage-node${i}:/etc/ntp.conf'; done
```

5. Restart the ntpd daemon on all the storage nodes:

```
# for i in {1..6}; do ssh storage-node${i} 'systemctl enable ntpd; done
# for i in {1..6}; do ssh storage-node${i} 'systemctl start ntpd; done
# for i in {1..6}; do ssh storage-node${i} 'systemctl status ntpd; done
```

Scality RING Installation

The following procedures cover the installation and configuration of the Scality RING. The installation process is comprised of five unique stages listed below.

1. Preparing the Environment
2. Running the Pre-Install Suite
3. Installing Scality RING
4. Installing Scality S3 Connector Service
5. Running the Post-Install Suite

Prerequisites

Before beginning download the Scality Offline packages with S3 from packages.scality.com on the supervisor node. The Scality Installer archive comes with three package sets, each of which can be used for RING installation: Offline packages without S3, Offline packages with S3, and Online packages. Scality recommends using Offline packages for installation.

The installer leverages a platform description file to automate the RING installation process. Key to the automated RING installation process, the Platform Description File supplies information to the Scality Installer concerning the infrastructure on which the RING will be installed. It is generated from the Scality Sizing Tool, with system hardware information entered by Sales Engineers and technical details (e.g., minion IDs, IP addresses and RING names) entered by Customer Service Engineers.

The platform description file used for the CVD can be found in Appendix C.

Please contact your Scality sales representative for access to packages.scality.com and help generating the platform description file.

Starting the Installer

After downloading the installer ensure the root execution flag is set on the .run file.

```
$ chmod +x scality-ring-with-s3-offline.run
```

Invoke the installer with the --description-file option and pass the platform description file as the argument.

```
$ ./scality-ring-with-s3-offline.run --description-file /root/scality-sizing-cisco-m5.csv
```

Using the Installer

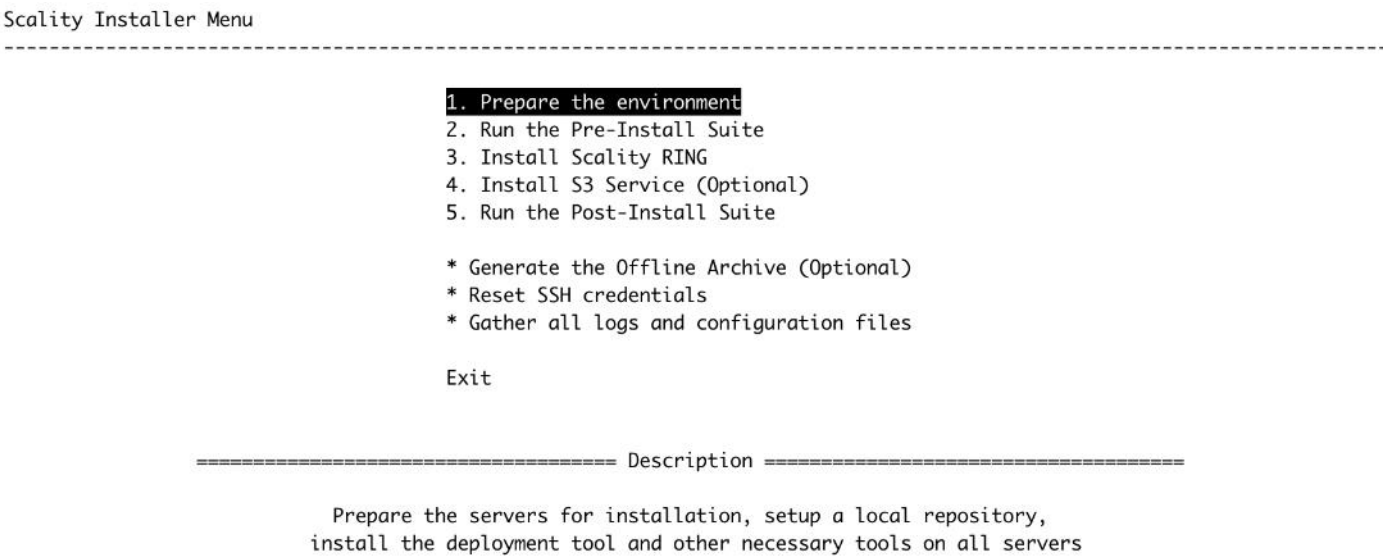
The Scality Installer Menu offers commands that correlate to major RING installation steps. These commands are presented in the sequence in which the installation steps are best followed.

Preparing the Environment

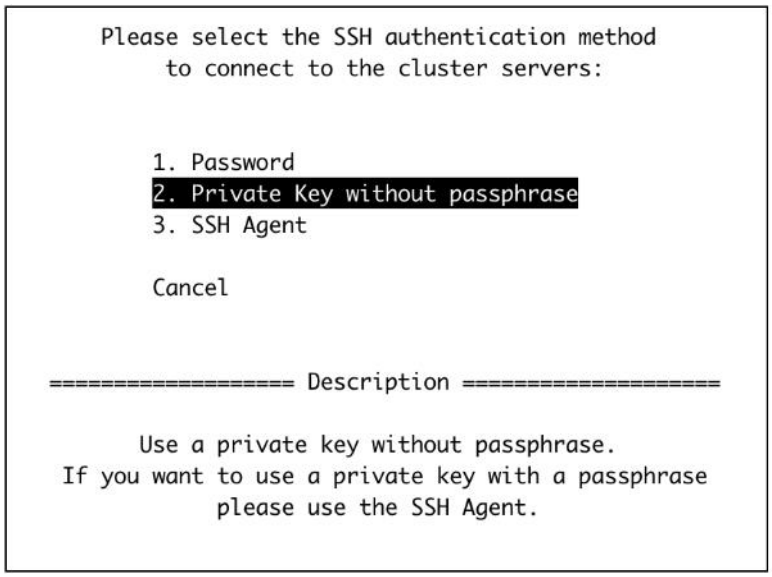
The first step in an Automated Installation is to set up the environment for RING installation, set up the Scality repositories to administer Scality, third-party and offline packages, and to deploy SaltStack on every machine listed in the Platform Description File.

To prepare the environment, complete the following steps:

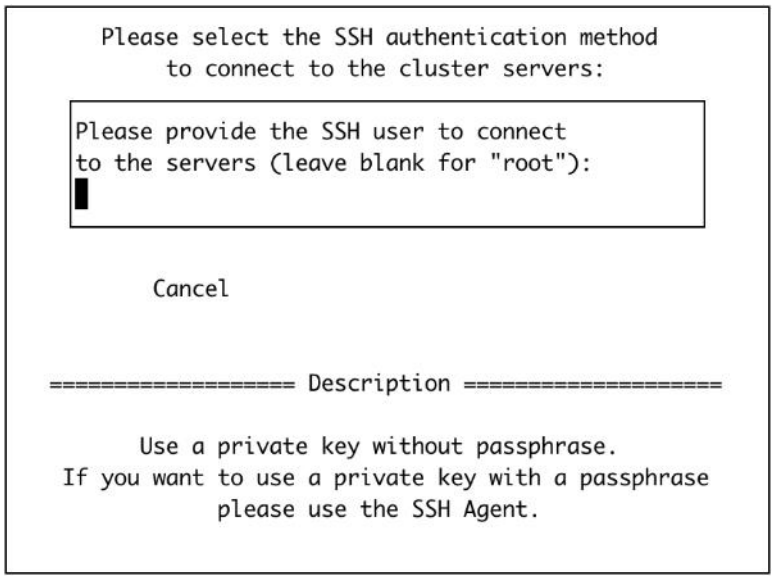
1. From the Scality Installer Menu, invoke the Prepare the Environment command.



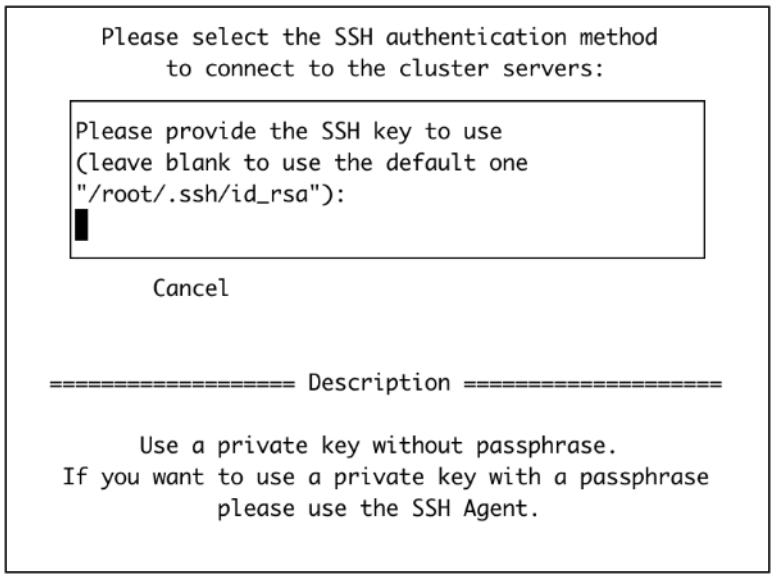
2. The first time an installer command is selected you will be asked to select the SSH authentication method used to connect to each of the servers. Select option 2, Private Key without passphrase.



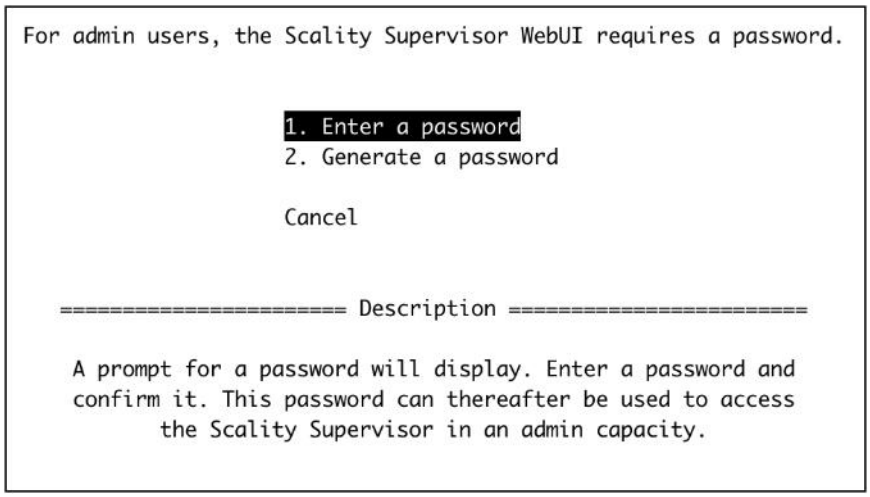
3. Provide the SSH user that will be used to connect to each of the servers.



4. Provide the SSH key that will be used to connect to each of the servers.



5. The Scality Supervisor UI requires a password. Choose option 1, Enter a password, to provide your own password.



The installer will now prepare the environment.

```
[2018-09-25 13:04:06,077] Loading the platform description file '/root/scality_sizing_cisco_m5_cvd.csv'... OK
[2018-09-25 13:04:07,653] Extracting platform description data... OK
[2018-09-25 13:04:07,653] Checking that bootstrap is run from supervisor server... OK
[2018-09-25 13:04:07,877] Generating the salt roster file '/etc/salt/roster'... OK
[2018-09-25 13:04:07,917] Preparing and testing SSH connection on every machine... OK
[2018-09-25 13:04:14,627] Performing server OS version correspondence check... OK
[2018-09-25 13:04:16,764] Generating the pillars for the install... OK
[2018-09-25 13:04:18,374] Installing scality-setup-httpd on '192.168.10.191'... OK
[2018-09-25 13:04:25,053] Setting up the new repository definitions on every machine... OK
[2018-09-25 13:04:33,973] Configuring logging on '192.168.10.191'... OK
[2018-09-25 13:04:43,026] Configuring Scality SSH on every machine... OK
[2018-09-25 13:04:49,743] Installing sreport on every machine... OK
[2018-09-25 13:05:02,178] Installing salt-master on 'supervisor'... OK
[2018-09-25 13:05:21,632] Installing salt-minion on every machine... OK
[2018-09-25 13:05:45,771] Accepting minion key(s) on the master instance... OK
[2018-09-25 13:06:12,746] Syncing configuration on every machine... OK
[2018-09-25 13:06:14,468] Installing and configuring scaldisk on every machine... OK
[2018-09-25 13:06:28,286] Preparing disks for installation... OK
[2018-09-25 13:09:21,054] Restoring repositories on every machine... OK

-- Bootstrap step successful, duration: 0:05:21.599582 --
[2018-09-25 13:09:27,031] The bootstrap step finished successfully...
```

Running the Pre-Install Suite

Execute Run the Pre-Install Suite menu command to check the availability and accessibility of hard- ware servers and components as defined in the Platform Description File. In addition, the Pre-Install Suite also checks and updates OS settings per Scality recommendations.

```
Scality Installer Menu
------------------------------------------------------------------------------------------------------------------------

                                    1. Prepare the environment
                                    2. Run the Pre-Install Suite
                                    3. Install Scality RING
                                    4. Install S3 Service (Optional)
                                    5. Run the Post-Install Suite

                                    * Generate the Offline Archive (Optional)
                                    * Reset SSH credentials
                                    * Gather all logs and configuration files

                                    Exit


=================================== Description ====================================

        Run the Pre-Install Suite to check the availability and accessibility of
      hardware servers and components, as defined in the Platform Description File
                    (the CSV/XLS file provided to the Installer).
       The suite also checks and updates OS settings per Scality recommendations,
              as described in the Scality Setup and Installation Guide.
```

Critical errors detected by the pre-install suite should be addressed before proceeding.

Installing Scality RING

Initiate the Install Scality RING menu command to install the Scality RING and all needed components on every node, as described in the Platform Description File.

```
Scality Installer Menu
------------------------------------------------------------------------------------------------------------------------

                                    1. Prepare the environment
                                    2. Run the Pre-Install Suite
                                    3. Install Scality RING
                                    4. Install S3 Service (Optional)
                                    5. Run the Post-Install Suite

                                    * Generate the Offline Archive (Optional)
                                    * Reset SSH credentials
                                    * Gather all logs and configuration files

                                    Exit


=================================== Description ====================================

        Install Scality RING and all necessary components on every node, as described
        in the Platform Description File (the CSV/XLS file provided to the Installer).
```

```
[2018-09-25 14:02:48,064] INFO     - Launching install, this might take some time
[2018-09-25 14:02:48,078] <salt> Clear the cache and sync modules, grains and pillar ... OK
[2018-09-25 14:02:52,433] <roles> Check storage nodes minions matcher ... OK
[2018-09-25 14:02:52,434] <roles> Ensure grains is deleted everywhere ... OK
[2018-09-25 14:02:54,920] <roles> Setup the group for storage nodes ... OK
[2018-09-25 14:02:55,953] <roles> Setup supervisor role ... OK
[2018-09-25 14:02:56,475] <roles> Setup storage nodes role ... OK
```

```
[2018-09-25 14:02:58,842] <roles> Setup elasticsearch cluster role ... OK
[2018-09-25 14:02:59,359] <roles> Advertise elasticsearch cluster ... OK
[2018-09-25 14:03:04,098] <roles> Setup the group for S3 connectors ... OK
[2018-09-25 14:03:04,618] <roles> Setup S3 role ... OK
[2018-09-25 14:03:09,919] <roles> Setup the group for NFS connectors ... OK
[2018-09-25 14:03:10,437] <roles> Setup NFS role ... OK
[2018-09-25 14:03:11,746] <roles> Setup SPROXYD role ... OK
[2018-09-25 14:03:14,020] <setup> Start scality-setup-httpd ... OK
[2018-09-25 14:03:17,481] <setup> Install python-scality ... OK
[2018-09-25 14:03:24,206] <setup> Install python-scaldisk ... OK
[2018-09-25 14:03:29,951] <setup> Install sreport ... OK
[2018-09-25 14:03:38,212] <setup> Detect the disks ... OK
[2018-09-25 14:03:39,431] <setup> Publish disks infos ... OK
[2018-09-25 14:03:40,565] <sup> Install and configure supervisor ... OK
[2018-09-25 14:04:10,465] <rings> Spread rings membership ... OK
[2018-09-25 14:04:12,381] <rings> Configure the rings on the supervisor ... OK
[2018-09-25 14:04:43,078] <elastic> Install and configure cluster elasticsearch ... OK
[2018-09-25 14:04:52,036] <supapi> Configure the supapi service ... OK
[2018-09-25 14:04:58,829] <supapi> Install the cloud monitoring service ... OK
[2018-09-25 14:05:05,551] <disks> Partition and format disks ... OK
[2018-09-25 14:08:14,052] <disks> Mount all disks ... OK
[2018-09-25 14:35:09,192] <nodes> Install and configure storage nodes ... OK
[2018-09-25 14:37:49,249] <keyspace> Compute the keyspace ... OK
[2018-09-25 14:37:51,877] <keyspace> Spread the keyspace to storage nodes ... OK
[2018-09-25 14:37:55,164] <keyspace> Make storage nodes join rings ... OK
[2018-09-25 14:38:04,546] <conns> Install NFS connectors ... OK
[2018-09-25 14:38:42,437] <conns> Install sproxyd connectors ... OK
[2018-09-25 14:39:40,735] <conns> Install Scality Agent Daemon (sagentd) on S3 connectors ... OK
[2018-09-25 14:40:13,788] <post> Install and configure ringsh ... OK
[2018-09-25 14:40:19,979] <post> Backup the whole platform ... OK
[2018-09-25 14:40:23,783] <post> Install external tools ... OK
[2018-09-25 14:40:39,620] <exit> Removing the credentials ... OK
[2018-09-25 14:40:40,643] <exit> Restoring repositories definitions before exiting ... OK

Press [Enter] to return to the menu or [Ctrl]+c to exit the installer
```

Installing S3 Connector Service

To install the S3 Connector Service, complete the following steps:

1. The Install the S3 Service (Optional) menu command installs the S3 Connector components on the nodes as described in the Platform Description File.

```
Scality Installer Menu
------------------------------------------------------------------------------------------------------------------------

                                        1. Prepare the environment
                                        2. Run the Pre-Install Suite
                                        3. Install Scality RING
                                        4. Install S3 Service (Optional)
                                        5. Run the Post-Install Suite

                                        * Generate the Offline Archive (Optional)
                                        * Reset SSH credentials
                                        * Gather all logs and configuration files

                                        Exit


================================== Description ==================================

 Install the S3 components on the nodes, as described in the Platform Description File
                        (the CSV/XLS file provided to the Installer).
```

```
Using private key '/root/.ssh/id_rsa'.
[2018-09-25 14:50:18,364] Searching S3 offline archive file... OK
[2018-09-25 14:50:18,365] Extracting S3 offline archive... OK
[2018-09-25 14:54:04,909] Generating the S3 inventory from platform description file... OK
[2018-09-25 14:54:10,087] Installing sshpass package... OK
[2018-09-25 14:54:14,476] Generating vault environment configuration... OK
[2018-09-25 14:54:18,475] Running S3 ansible playbook to install the S3 connector... OK
[2018-09-25 15:06:15,570] Setting up the identisee credentials... OK
[2018-09-25 15:06:18,191] The s3 step finished successfully...

Press [Enter] to return to the menu or [Ctrl]+c to exit the installer
```

Running the Post-Install Suite

Issue the Run the Post-Install Suite menu command to validate the installation.

```
Scality Installer Menu
------------------------------------------------------------------------------------------------------------------------

                                        1. Prepare the environment
                                        2. Run the Pre-Install Suite
                                        3. Install Scality RING
                                        4. Install S3 Service (Optional)
                                        5. Run the Post-Install Suite

                                        * Generate the Offline Archive (Optional)
                                        * Reset SSH credentials
                                        * Gather all logs and configuration files

                                        Exit


================================== Description ==================================

                Run the Post-Install Suite on the platform to validate the installation.
```

```
Script installation detail is found in /var/log/postinstall_launcher.log
[2018-09-25 15:12:21,607] Setting up the new repositories definitions on every machine ... OK
[2018-09-25 15:12:26,167] Installing the postinstallchecks ... OK
```

```
Running the postinstallchecks
Running script using salt
Starting checks on supervisor,stroage-node3,stroage-node1,stroage-node5,storage-node4,storage-node6,storage-node2
Checking if server is handled by salt
Checking missing pillars
Gathering info from servers (salt mine.send) for consistency check later
Running tests
The result is found in: /root/post-install-checks-results.tgz
[2018-09-25 15:15:37,055] The postinstall step finished successfully...

Press [Enter] to return to the menu or [Ctrl]+c to exit the installer
```

The results of the Post-Install Suite should be shared with your Scality Service or Support representative for review. The results can be found at /root/post-install-checks-results.tgz

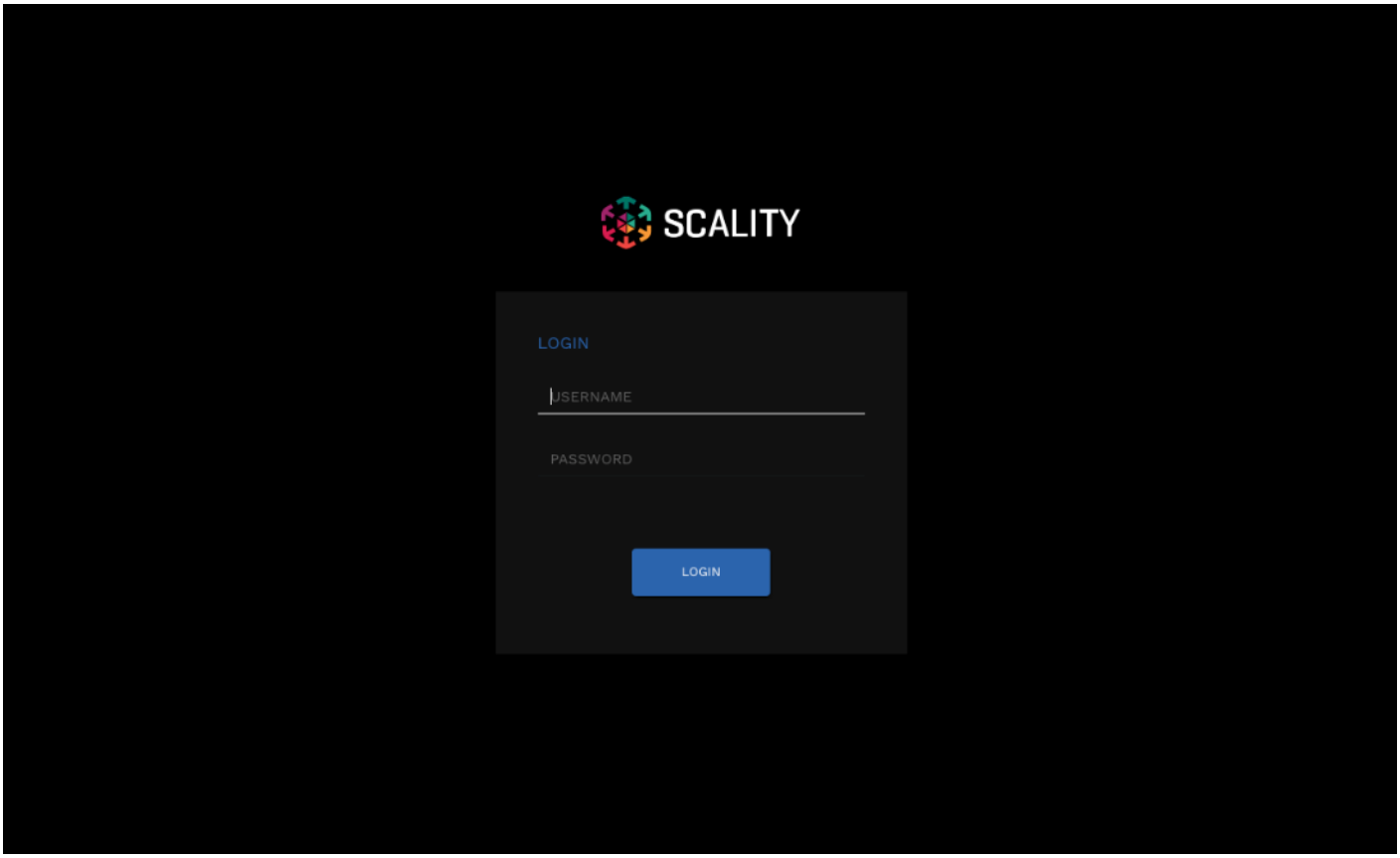
Managing and Monitoring Scality RING

Managing and monitoring the RING is enabled through a cohesive suite of user interfaces, built on top of a family of RESTful interfaces termed the Supervisor API ("SupAPI"). RING 7 includes the new "Scality Supervisor", a browser-based portal for both systems monitoring and management of Scality components. In RING 7, the Supervisor now provides capabilities across object (S3) and file (NFS, SMB, FUSE) Connectors including integrated dashboards including Key Performance Indicators (KPIs) with trending information such as "Global Health", "Performance", "Availability" and "Forecast."

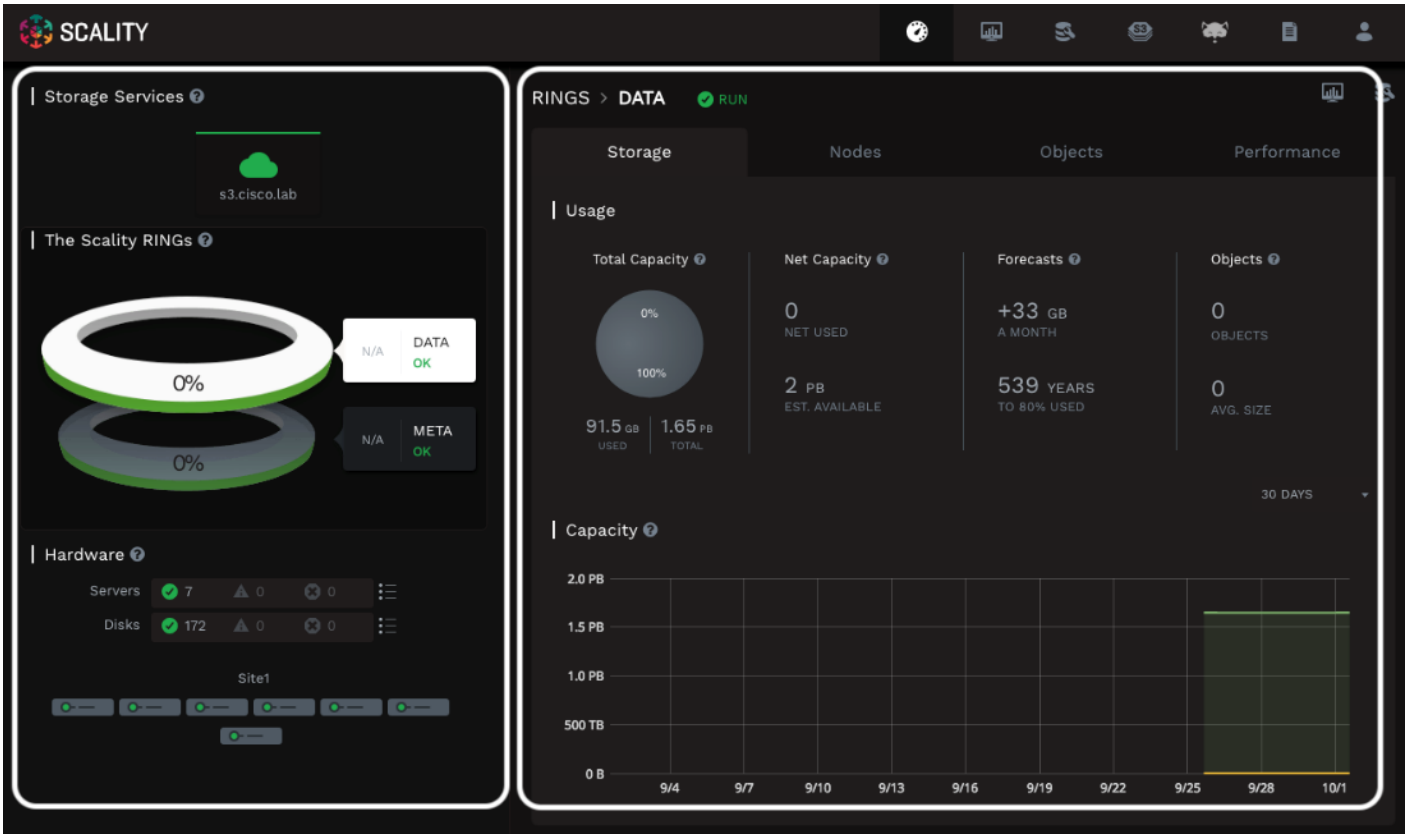
The Scality Supervisor can be accessed in a browser via the IP address of the Supervisor server.

Monitoring Scality RING

1. Launch the Scality Supervisor by navigating to http://<supervisor IP>/gui.



2. Login using the user "admin" and the password provided during the Preparing the Environment step of the installation.



The Component Panel on the left hand side provides an overview of the overall platform health including hardware and services. Services in a failed or critical state will be colored red indicating attention is needed.

The Information Screen on the right hand side provides a capacity overview. The Forecasts section provides the storage administrator with a projected time to 80 percent full.

Managing NFS Connectors

To configure and access NFS exports on the Scality Scale-Out Filesystem (SOFS), complete the following steps:

1. The legacy RING Supervisor is used for creating and managing Volumes and NFS exports. To access the legacy RING Supervisor navigate to http://<supervisor IP>/sup.

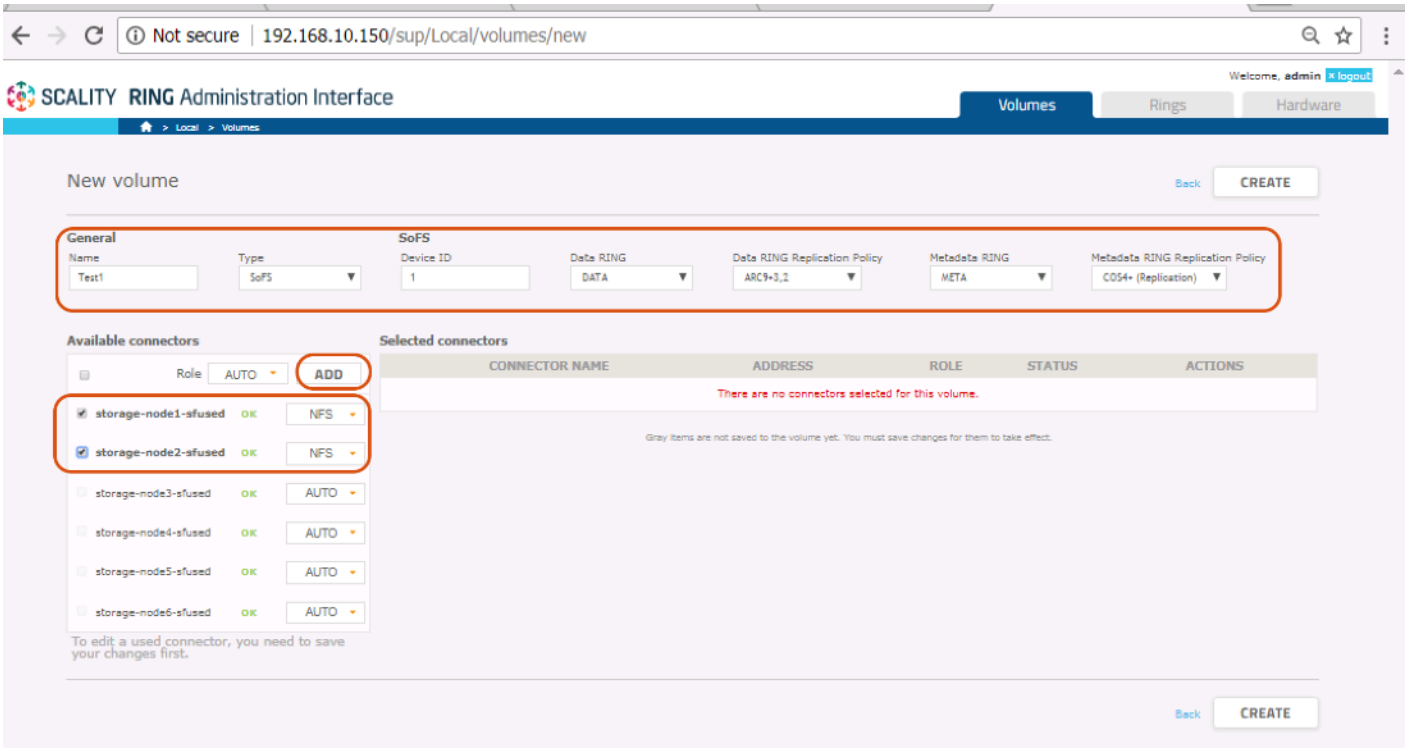
2. Login using the user “admin” and the password provided during the Preparing the Environment step of the installation.

3. Navigate to the Volumes tab and click NEW VOLUME.

4. In the General section provide the following values.

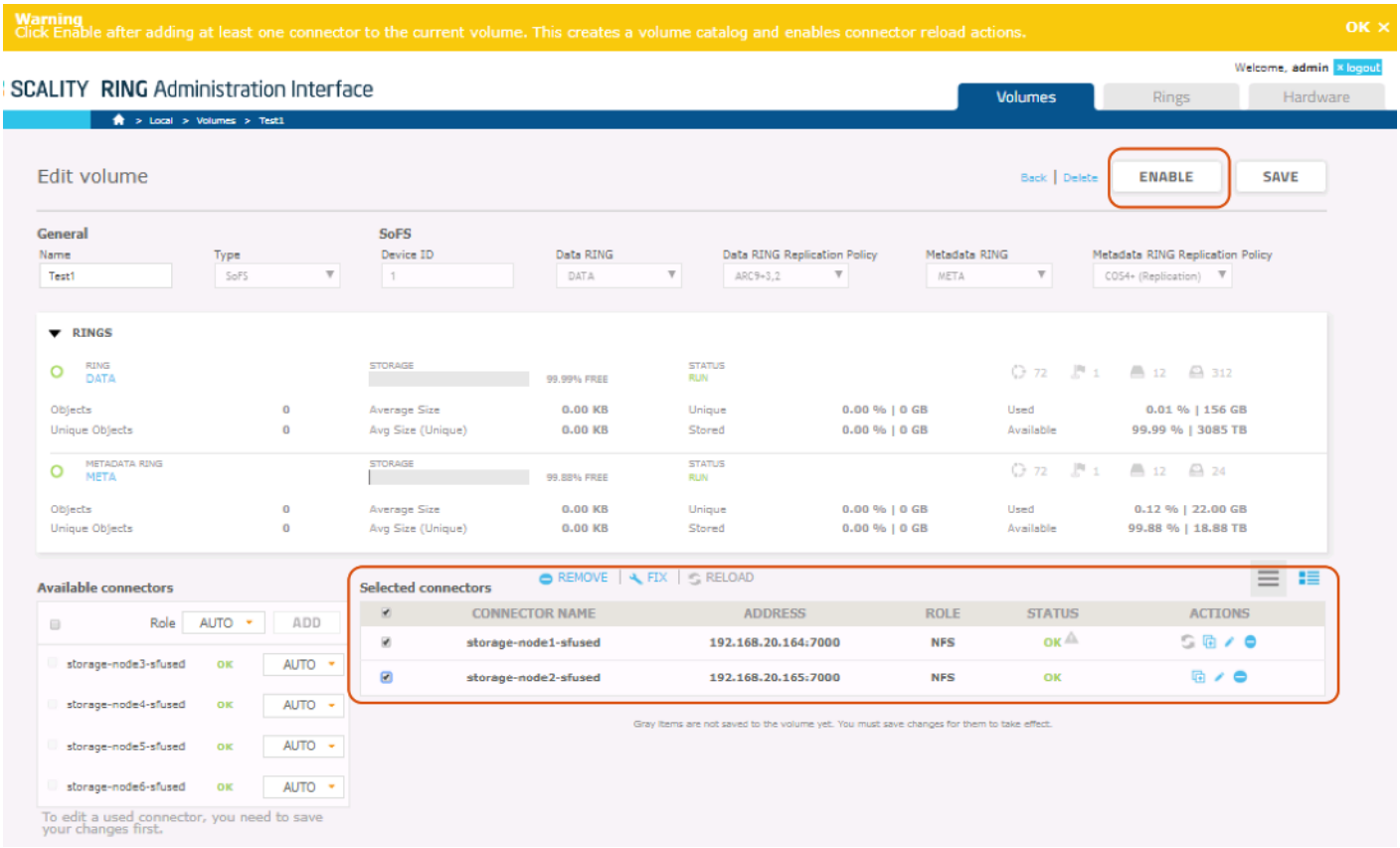
   Name: Cisco
   Type: SoFS
   Device ID: 1
   Data RING: DATA
   Data RING Replication Policy: ARC 9+3
   Metadata RING: META
   Metadata RING Replication Policy: COS 4+ (Replication)

   a. Under Available Connectors select storage-node1-sfused and storage-node2-sfused. Change the Role to NFS and click Add.

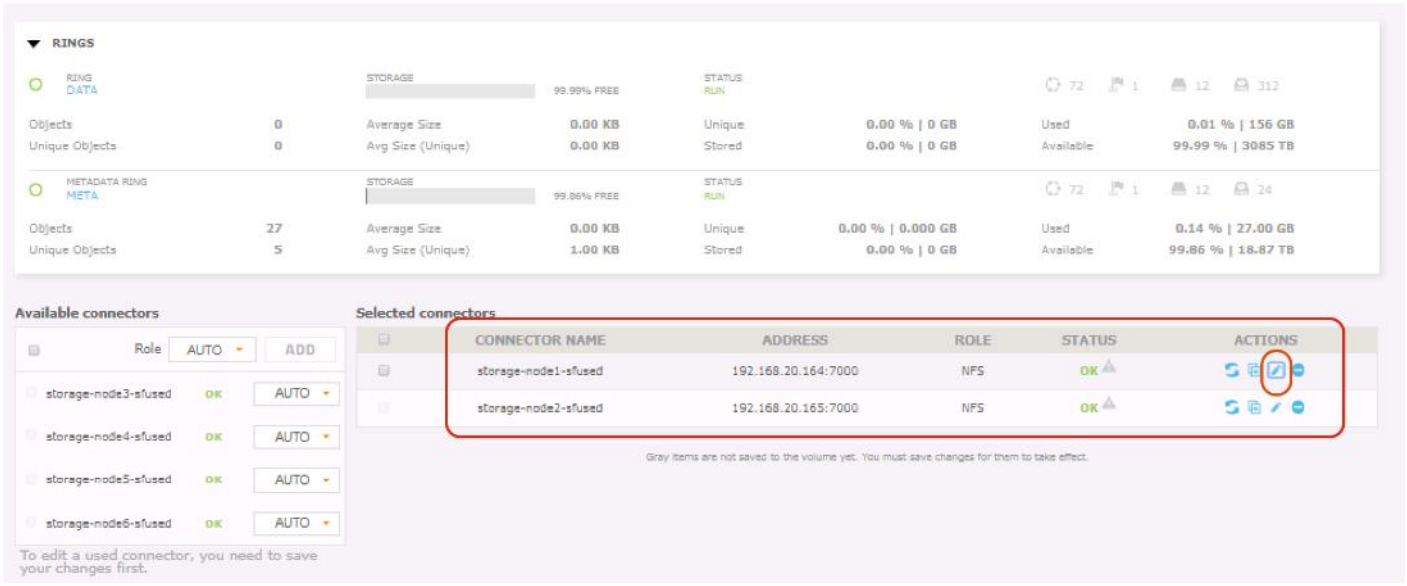


5. Click the Create button.

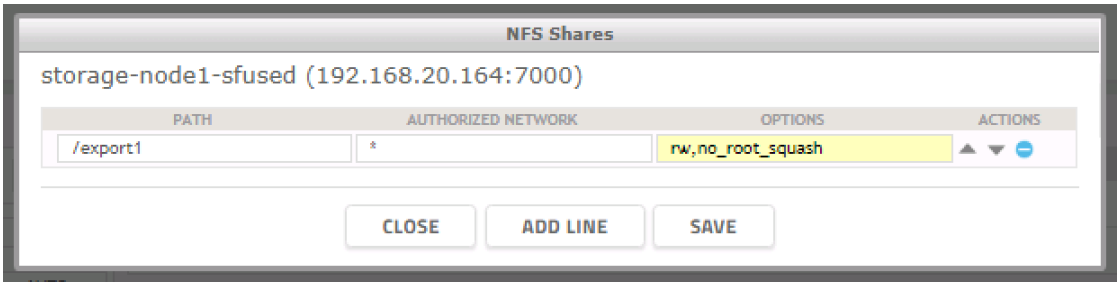
6. Verify that the ROLE of each connector is NFS and click Enable.



7. To create the NFS exports click the Edit NFS action (identified by the pencil symbol) next to the connector storage-node1-sufsed.

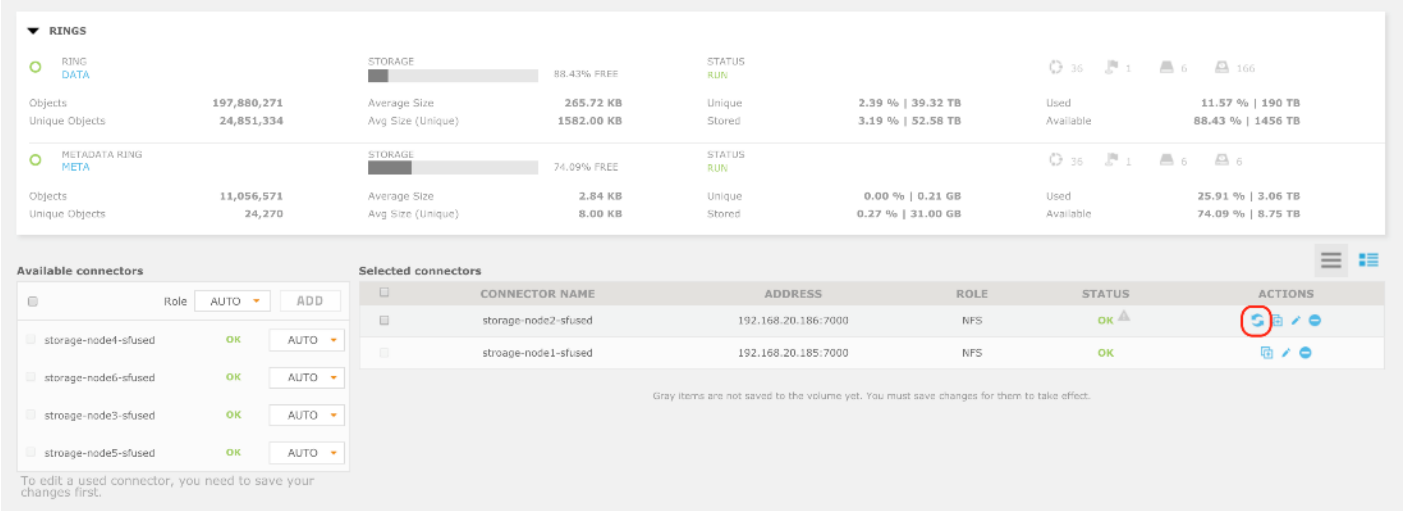


8. Define the path as '/export1' and pass the 'rw,no_root_squash' options.



9. Click the SAVE button.

10. Click the Reload icon next to the connector name to activate the changes.



11. Repeat the steps above for storage-node2-sfused and create a second export, '/export2'.

12. The Supervisor node can be used as an NFS client to test functionality.

```
$ yum -y install nfs-utils
```

13. Mount the exports.

```
$ cd /mnt; mkdir export1 export2

$ mount storage-node1:/export1 /mnt/export1

$ mount storage-node2:/export2 /mnt/export2
```

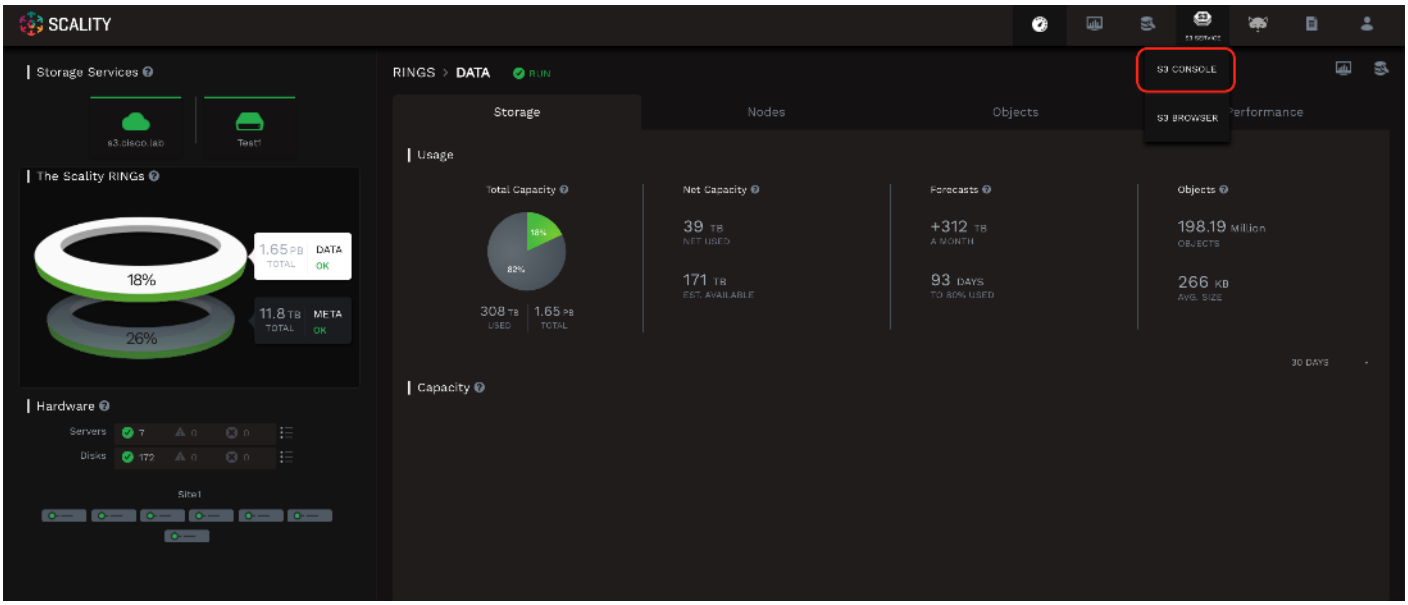
14. A simple functional test may be performed by copying files to and from the NFS-mounted directories.

Managing S3 Connectors

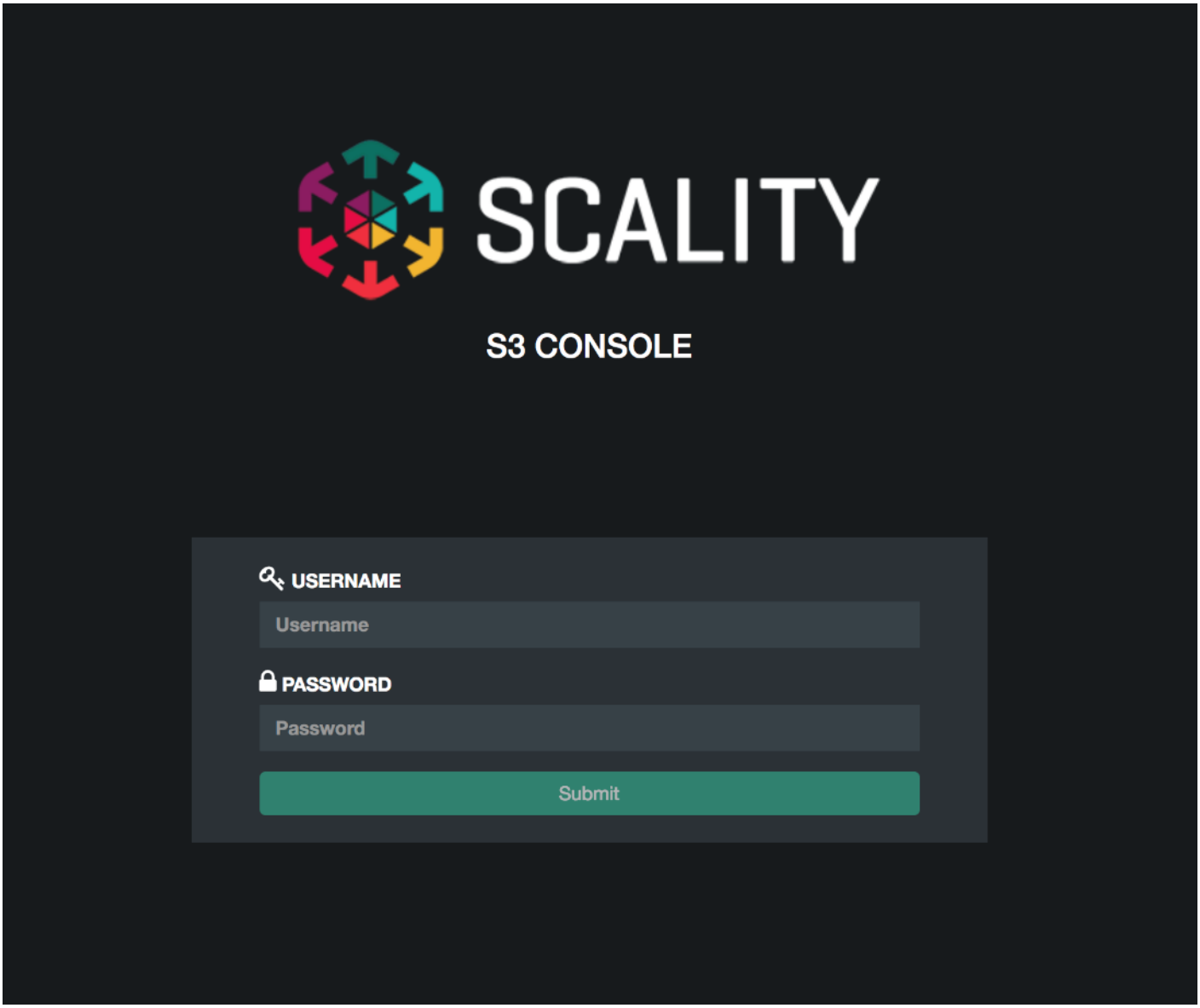
The Scality S3 Connector provides a full implementation of the AWS multi-tenancy and identity management (AWS IAM) model with federated authentication to LDAP and Active Directory to integrate into enterprise deployment environments. In addition to the RING Supervisor management UI, the S3 Service Provider UI is a web-based user interface to manage multi-tenancy accounts, users, group and policies.

The following procedures describe creating a new S3 account through the S3 console.

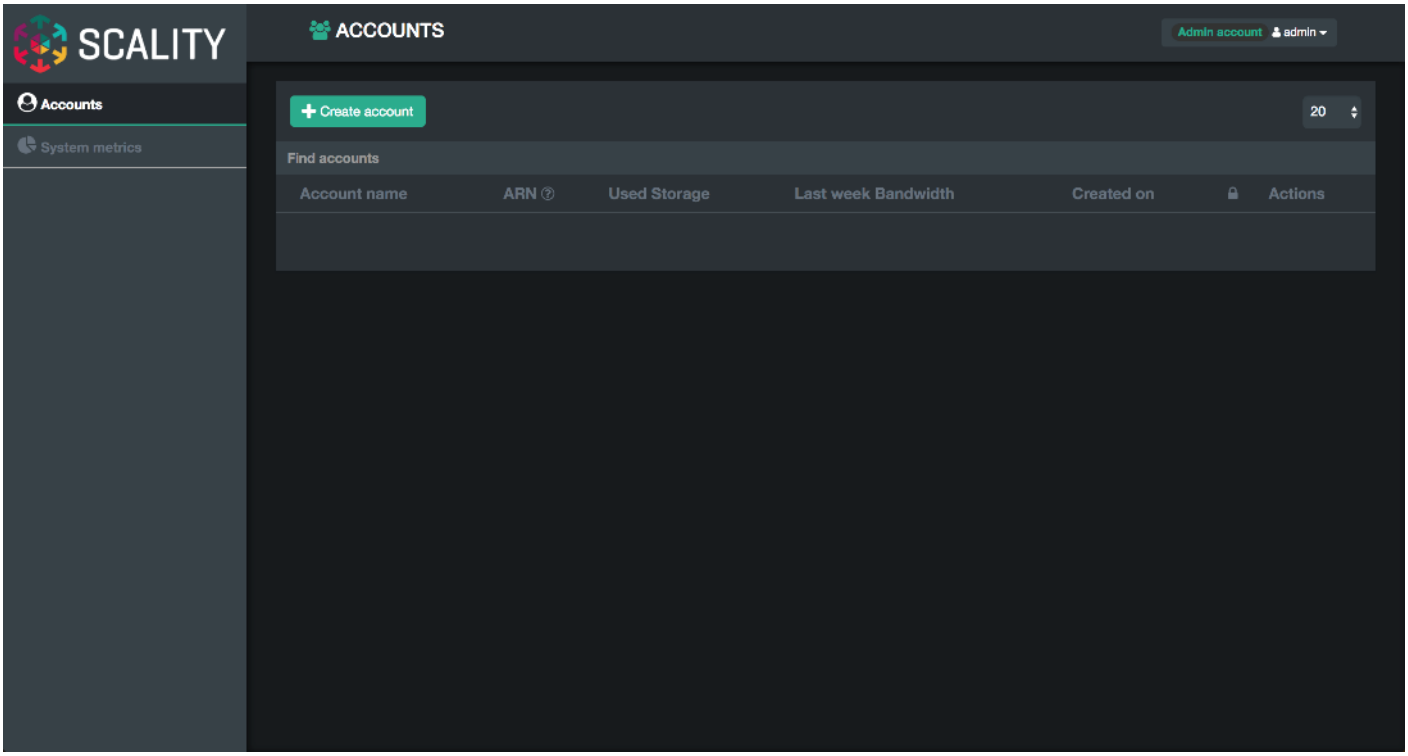
1. To connect to the S3 Console navigate to the S3 Service drop-down list in the Scality Supervisor and select S3 Console.



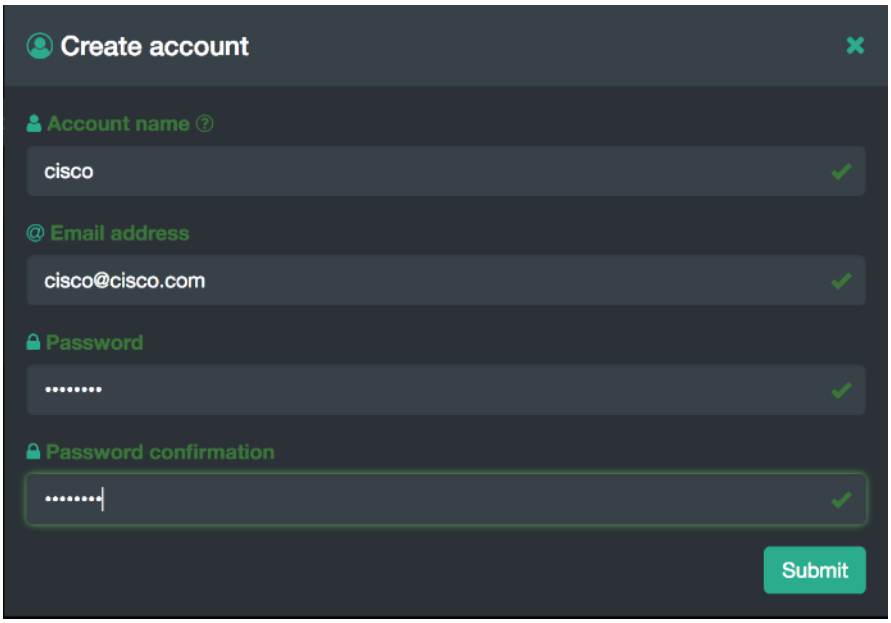
2. Login using the user “admin” and the password provided during the Preparing the Environment step of the installation.



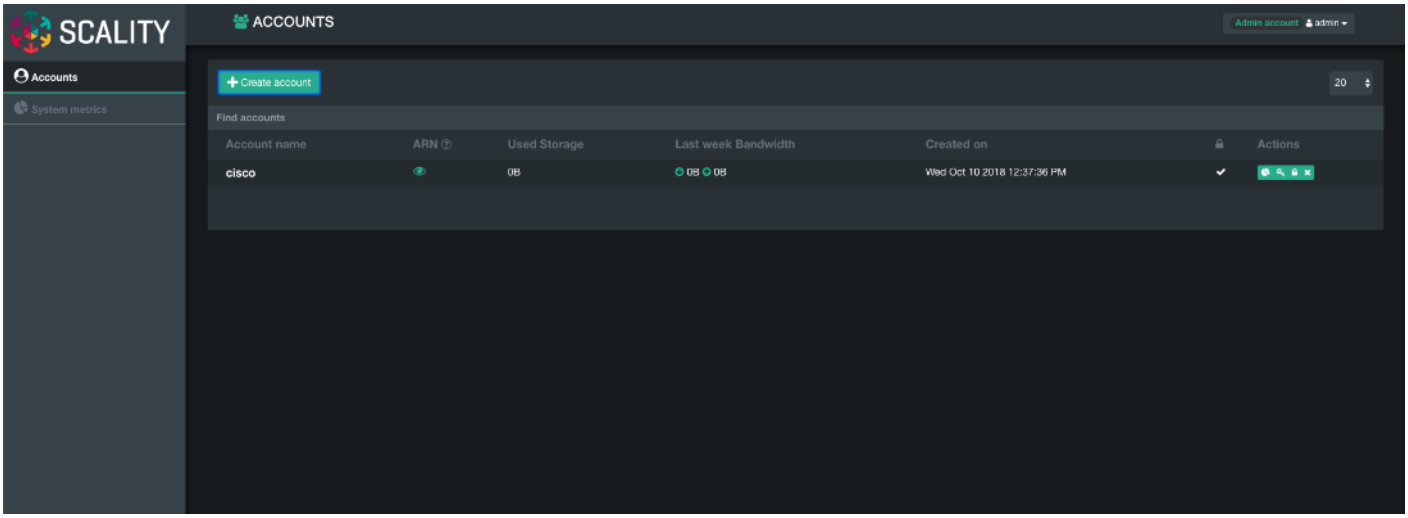
3. In the top left corner click the Create account button to create a new S3 account.



4. Provide an account name, email and password then click submit.

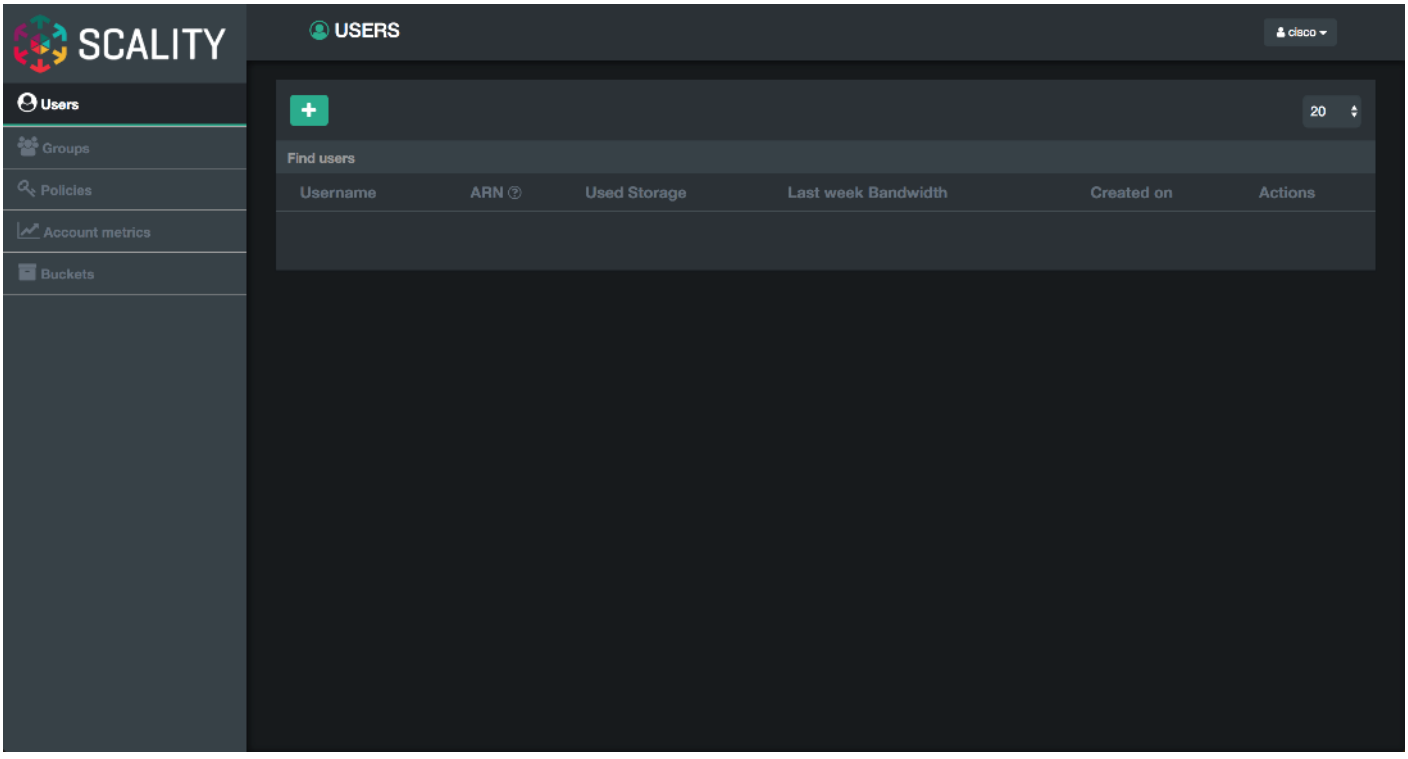


5. The new account will show up in the accounts table.

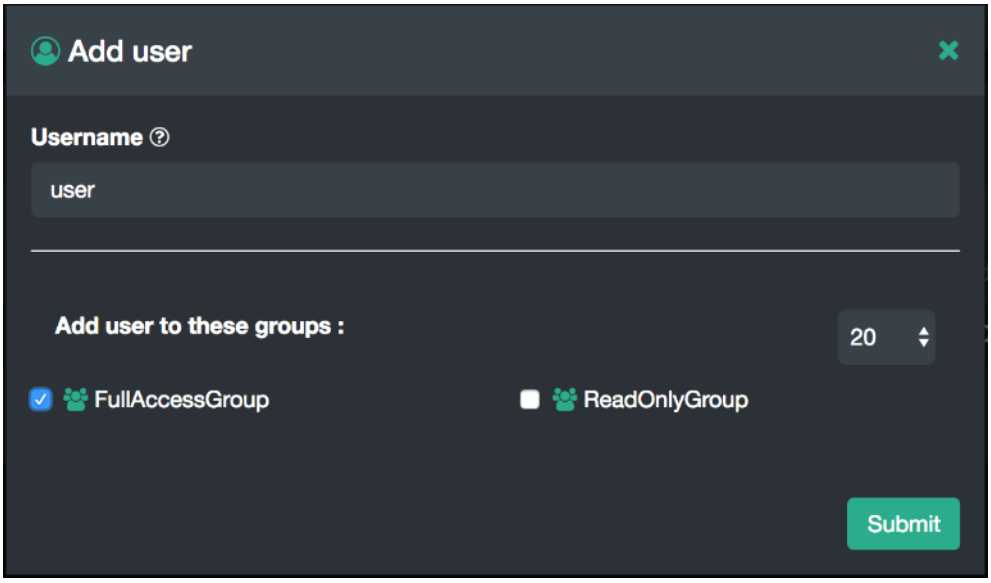


The S3 Console can be used to manage users. To create a new user under the ‘cisco’ account created in the previous section login to the S3 Console using the user ‘cisco’ and the password provided.

6. Click the + button in the top left corner to create a new user.

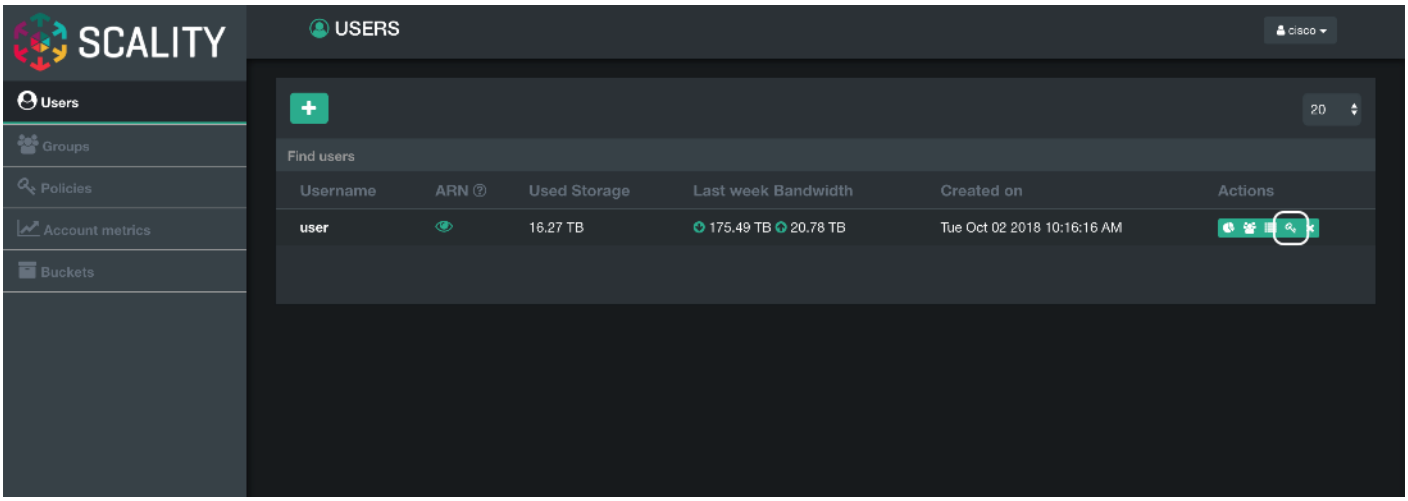


7. Provide a Username, select the FullAccessGroup to grant the user full permissions and click Submit.

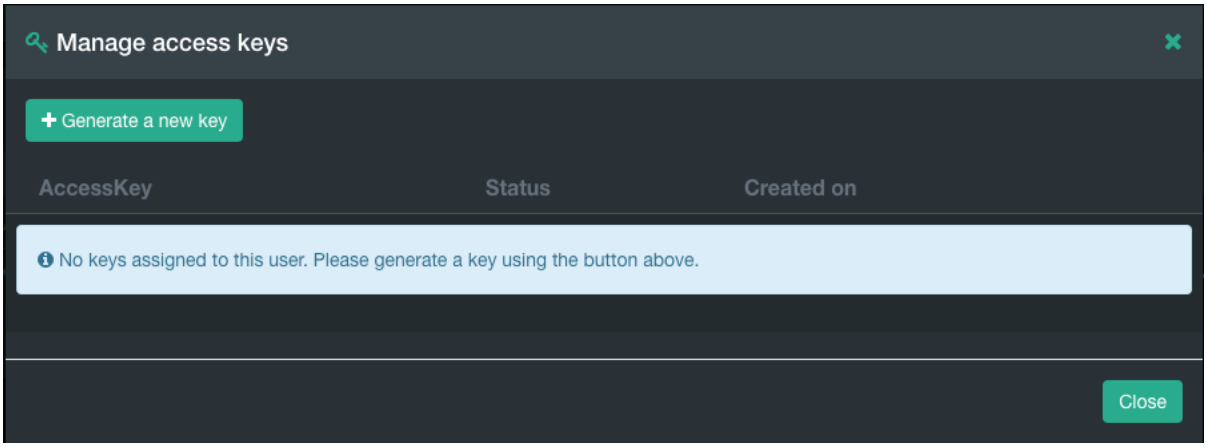


8. After clicking Submit the user will appear in the users table.

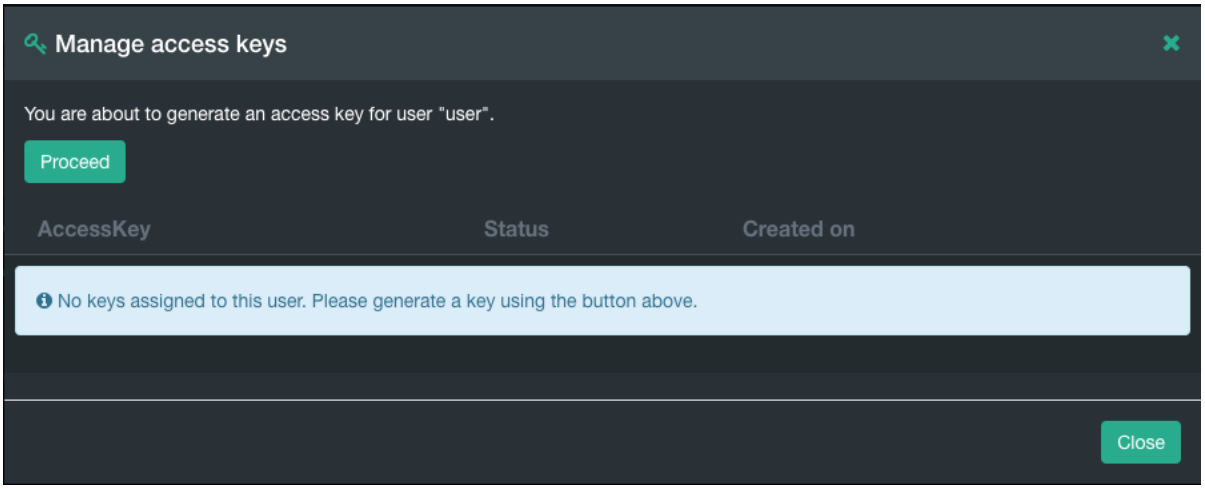
9. To generate the users secret key and access key required to send S3 requests click the Users Key icon in the Actions column.



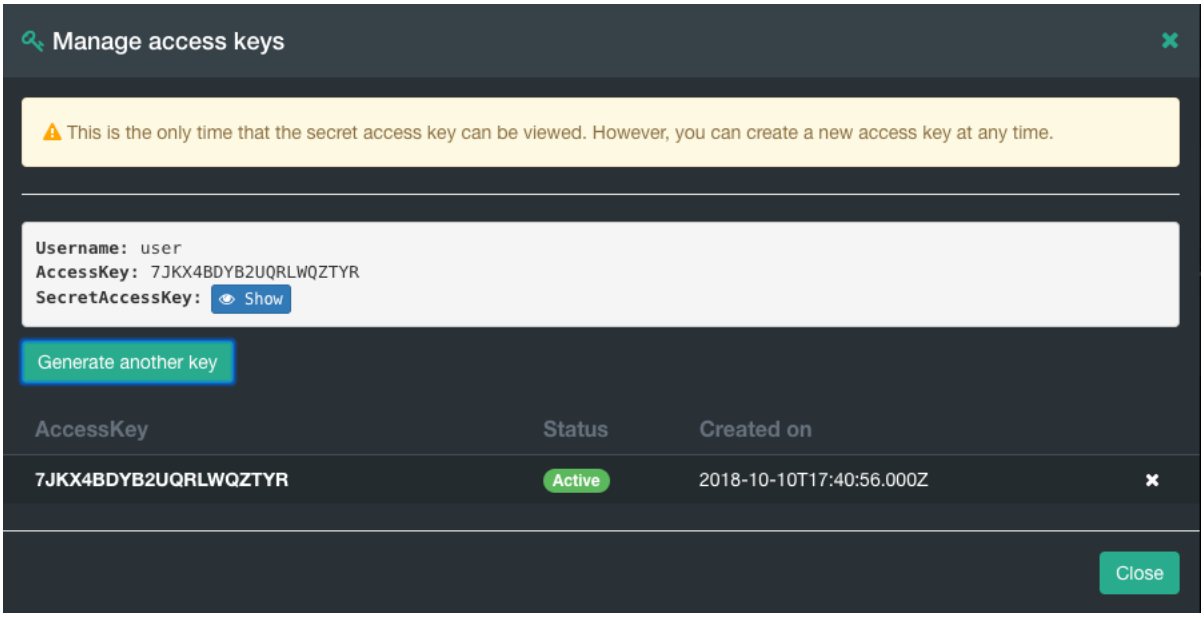
10. Click the Generate a new key button to generate the secrete key and access key.



11. Confirm the key generation by clicking Proceed.

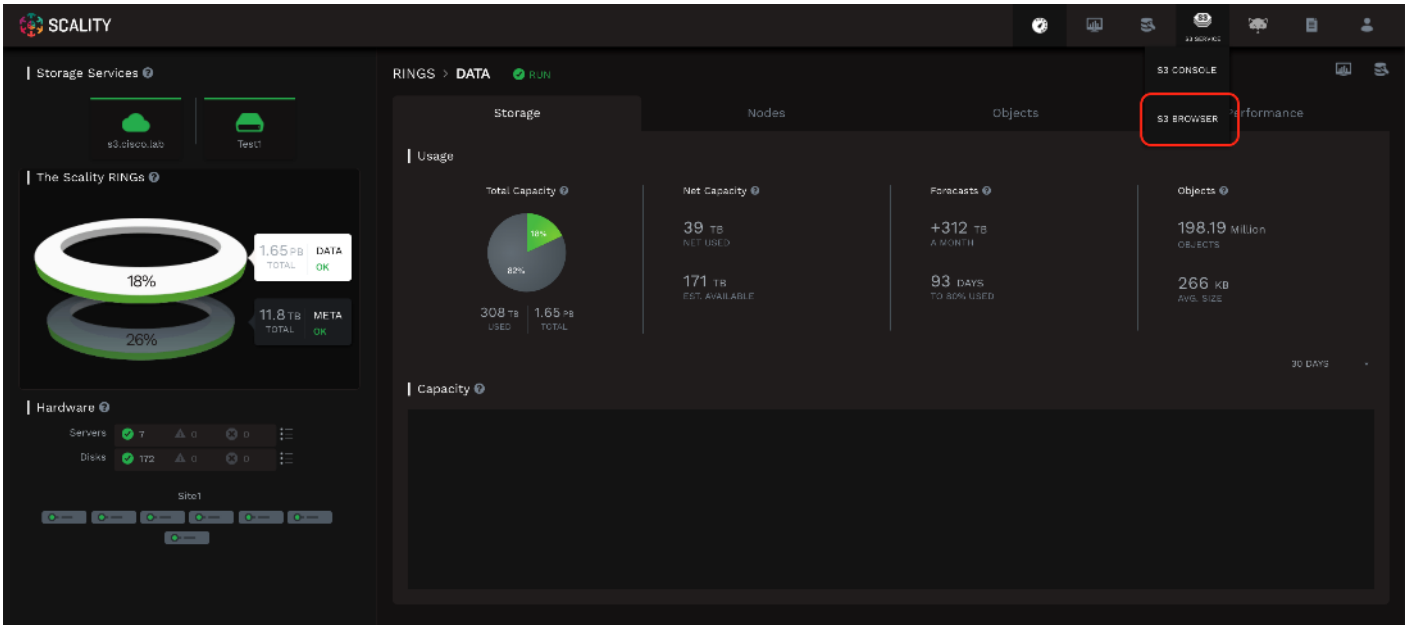


For security reasons, the secret key can only be viewed at creation time. Record the secret key in a safe location. If the secret key is lost generate a new secret key and access key can be generated from the S3 Console.

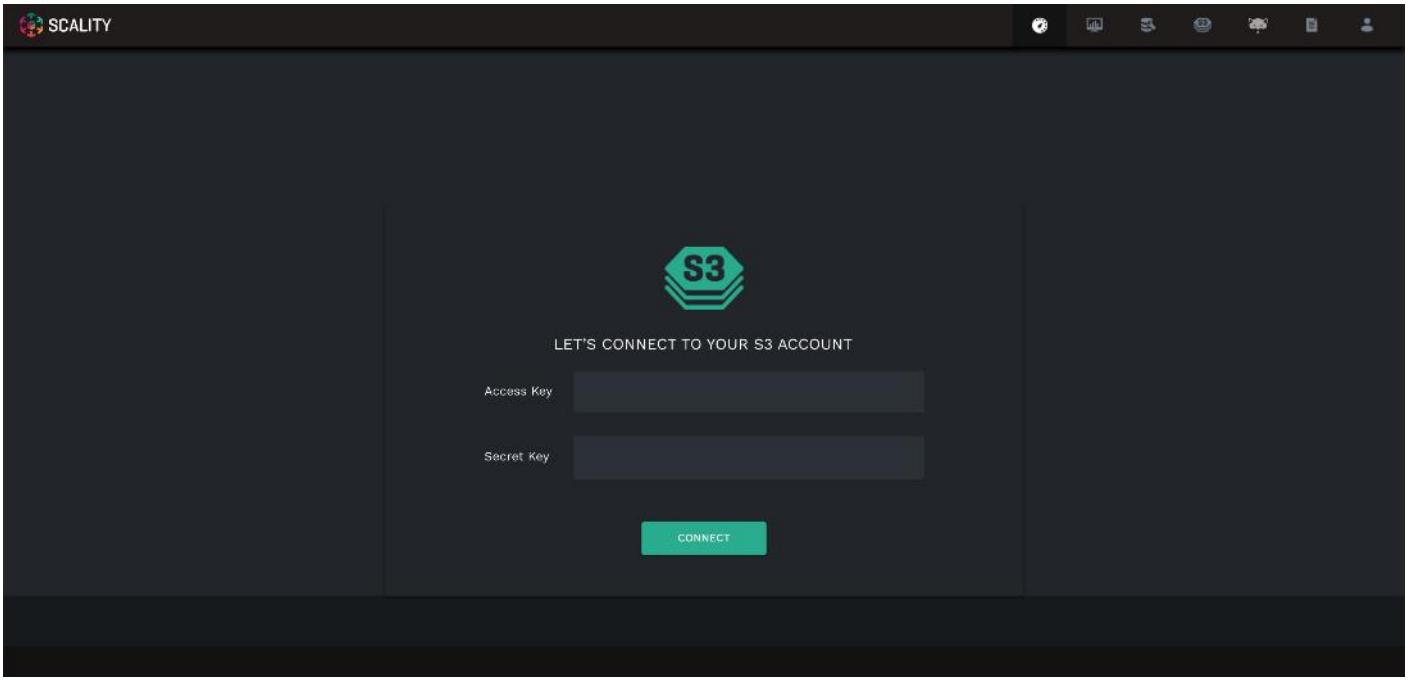


The AccessKey and SecretAccessKey can be used to connect to the S3 endpoint using command line tools like awscli or s3cmd or directly in any application which supports the S3 API.

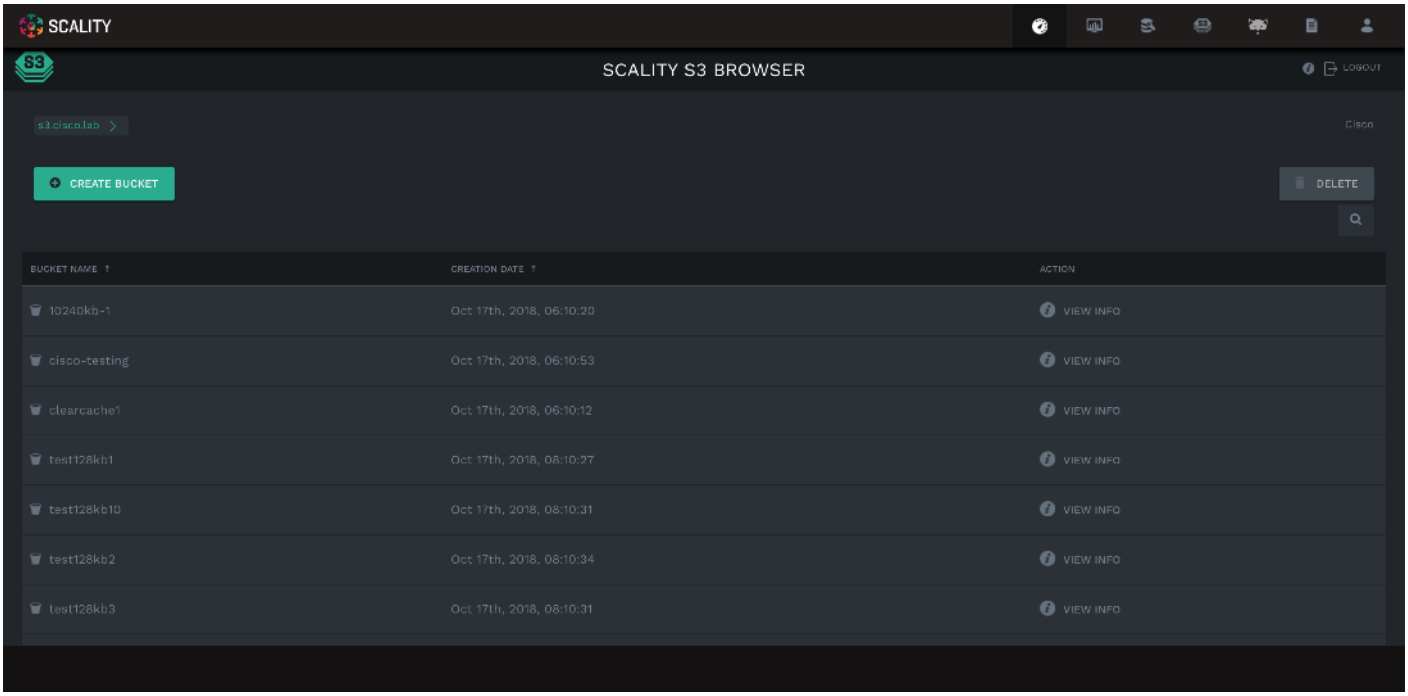
Scality also provides an S3 Browser that can be used browse or create buckets and upload objects. To connect to the S3 Console navigate to the S3 Browser dropdown menu in the Scality Supervisor and select S3 Browser.



The AccessKey and SecretKey can be used to login.



You can browse, upload, download, and delete buckets and objects.

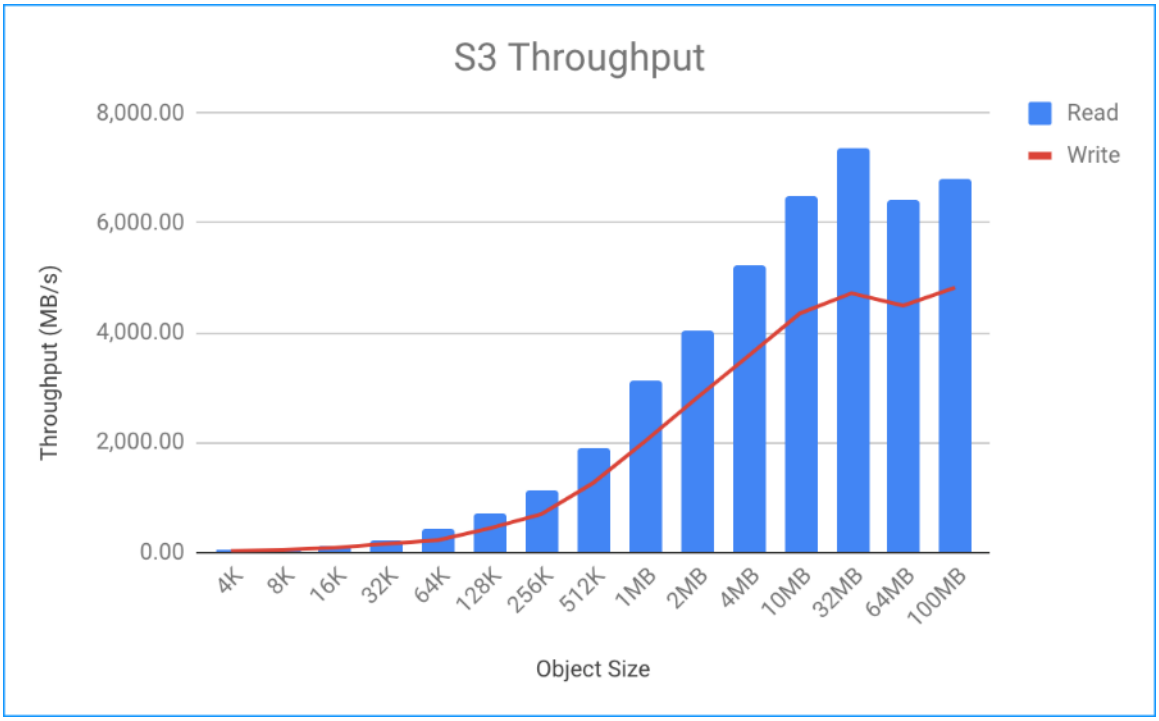


Performance

Performance was evaluated on the Scality RING running on Cisco M5 UCS hardware. The goal of the performance testing was to evaluate peak file and object performance under ideal conditions.

S3 Performance Tests

S3 performance testing was conducted with COSBench the standard cloud object storage benchmark. Five Cisco UCS C220 M5 servers were used as COSBench drivers to generate the object workload.



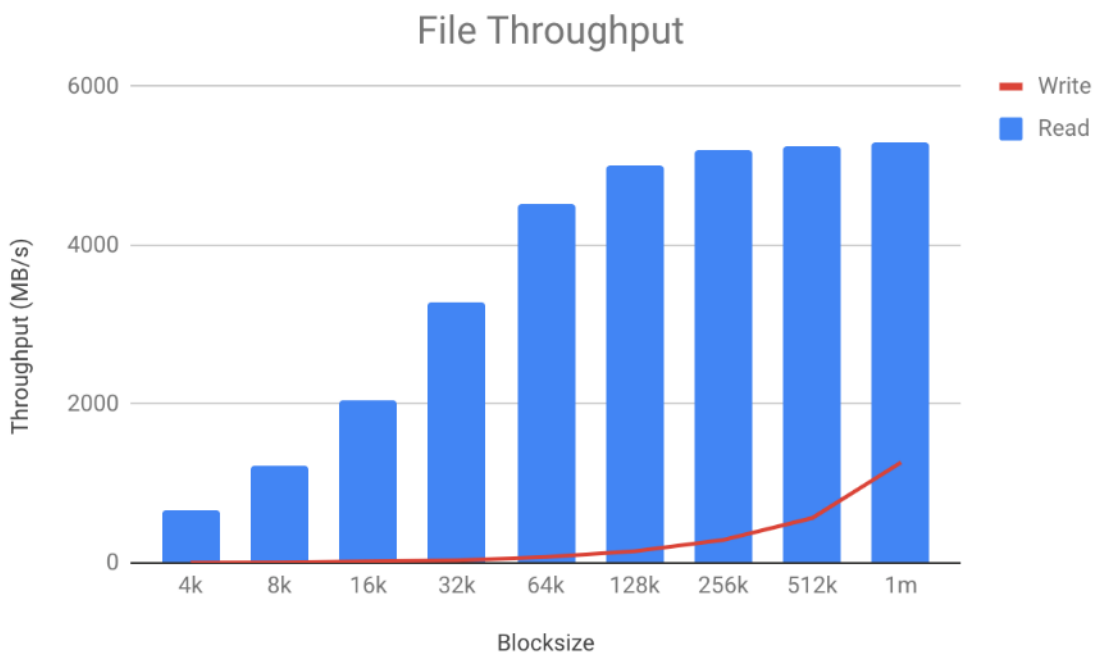
- Read bandwidth peaks at 7.19 GB/s at an object size of 32MB
- Write bandwidth peaks at 4.61 GB/s at an object size of 32MB

The network links on the COSBench drivers are saturated for read results above 4MB.

NFS Performance Tests

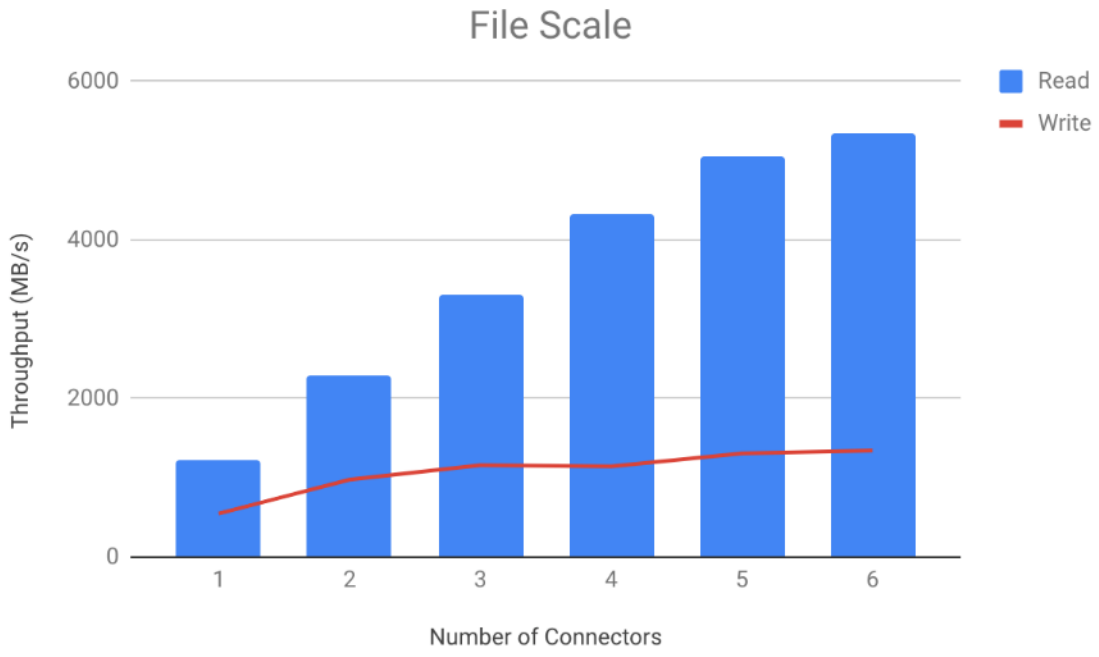
NFS performance testing was conducted with fio a flexible file benchmark tool. Load was generated from six NFS clients running on Cisco C220 M5 servers. Each of the six RING servers were running the NFS file connector.

All write workloads were conducted with direct IO enabled.



- Read bandwidth peaks at 5.18 GB/s at a 1M block size
- Write bandwidth peaks at 1.24 GB/s at a 1M block size

In order to demonstrate the scalability of the Scality RING benchmarks were conducted on 1-6 file connectors.



- Read bandwidth scales nearly linearly from one through 6 connectors
- Write bandwidth scales nearly linearly from one through 3 connectors

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 HA 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 HA tests:

- The Cluster will have reasonable amount of load when the fault is injected. The outputs like bandwidth and IOPS from the cluster will be gathered before and after the fault injection and after the restoration of the failed components.
- 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 HA tests conducted were:

- Fabric Interconnect Failures
- Nexus 9000 Failures
- S3 Connector, NFS Connectors, and 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. Connect Local Management on A

```
scality-pod2-A# connect local-mgmt A
```

……..

scality-pod2-A(local-mgmt)# show cluster extended-state

Cluster Id: 0xba13e47e876d11e7-0x99df002a1029453f

Start time: Sat Nov 11 19:14:33 2017

Last election time: Thu Feb 15 08:51:36 2018

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 7, serial: FOX2036G8U6, state: active

Server 2, serial: FCH2033V31P, state: active

Server 4, serial: FCH2034V0UG, state: active

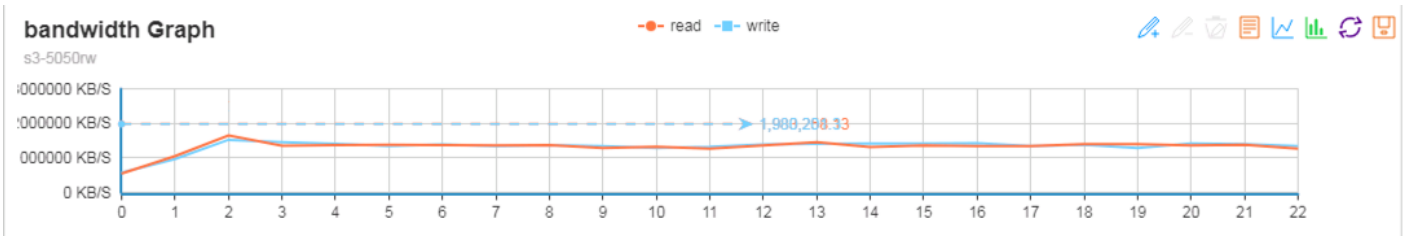
S3 COSBench test started for 10MB block size and with 480 workers.

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

COSBench graphs:

Op-Type	Op-Count	Byte-Count	Avg-ResTime	Avg-ProcTime	Throughput	Bandwidth	Succ-Ratio
op1: read	1.02 kops	10.41 GB	714.75 ms	592.92 ms	202.34 op/s	2.07 GB/S	100%
op2: write	1 kops	10.28 GB	746.46 ms	684.28 ms	200.07 op/s	2.05 GB/S	100%

Bandwidth around 10.41 GB/s:



The cluster was doing about 10.41 GB/s at 10MB objects size before fault injection.

Inject fault into the system:

Rebooted the fabric

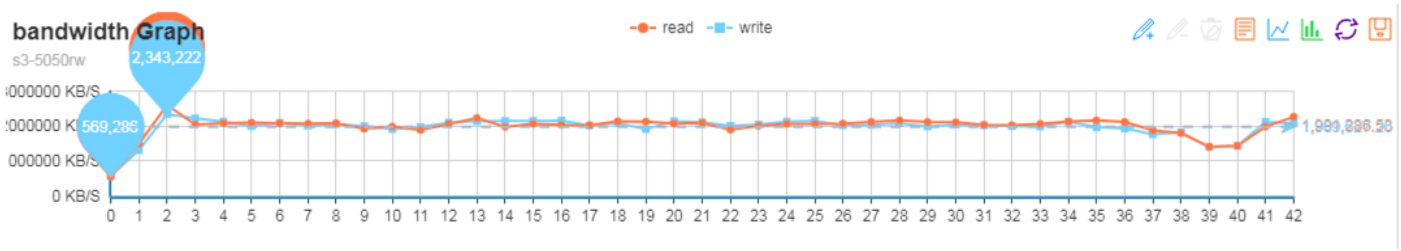
UCS-FI-6332-A# connect local-mgmt a

UCS-FI-6332-A(local-mgmt)# reboot

Before rebooting, please take a configuration backup.

Do you still want to reboot? (yes/no):yes

After the FI reboot:



FI was reboot at 38

When one of the FI's is down, COSBench continues to send the requests. However, the bandwidth comes down to 7.3 GB/sec from 10.41 GB/sec now.

Ring data dashboard reports cluster as healthy:

The ring dash board does not show any faults because of FI failure.


```
UCS-FI-6332-B(local-mgmt)# show cluster extended-state
```

Cluster Id: 0xc6ebf5d6b22f11e8-0x9e07002a1029453f

Start time: Thu Sep 6 16:58:55 2018

Last election time: Tue Oct 16 12:36:49 2018

B: UP, PRIMARY

A: DOWN, INAPPLICABLE

B: memb state UP, lead state PRIMARY, mgmt services state: UP

A: memb state DOWN, lead state INAPPLICABLE, mgmt services state: DOWN

heartbeat state SECONDARY_FAILED

INTERNAL NETWORK INTERFACES:

eth1, UP

eth2, UP

HA NOT READY

Peer Fabric Interconnect is down

Detailed state of the device selected for HA storage:

Chassis 1, serial: FOX2034GCM3, state: active

Chassis 2, serial: FOX2036G8TZ, state: active

Chassis 3, serial: FOX2036G8U3, state: active

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

The FI has come up. However it is not fully ready yet.

UCS-FI-6332-B(local-mgmt)# show cluster extended-state

Cluster Id: 0xc6ebf5d6b22f11e8-0x9e07002a1029453f

Start time: Thu Sep 6 16:58:55 2018

Last election time: Tue Oct 16 12:43:21 2018

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: FOX2034GCM3, state: active

Chassis 2, serial: FOX2036G8TZ, state: active

Chassis 3, serial: FOX2036G8U3, state: active

At this time, FI joined back and HA is in Ready status.

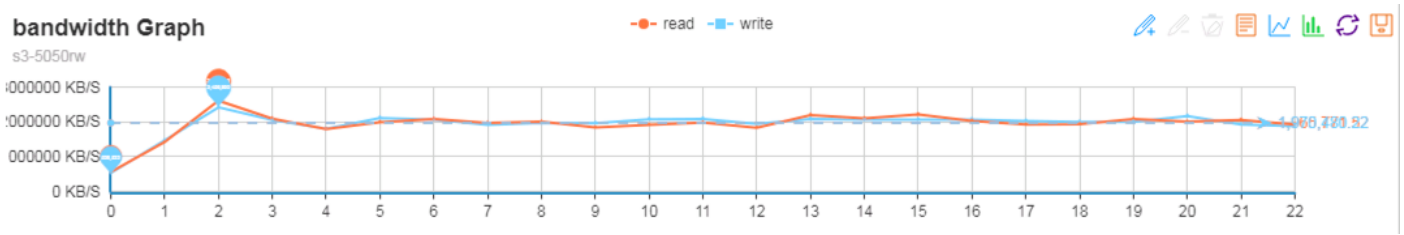
The system recovers after the Fabric joins the cluster and when HA READY. The dip in the graphs show the activity when the FI was rebooted.

Nexus 9000 Switch Failures

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

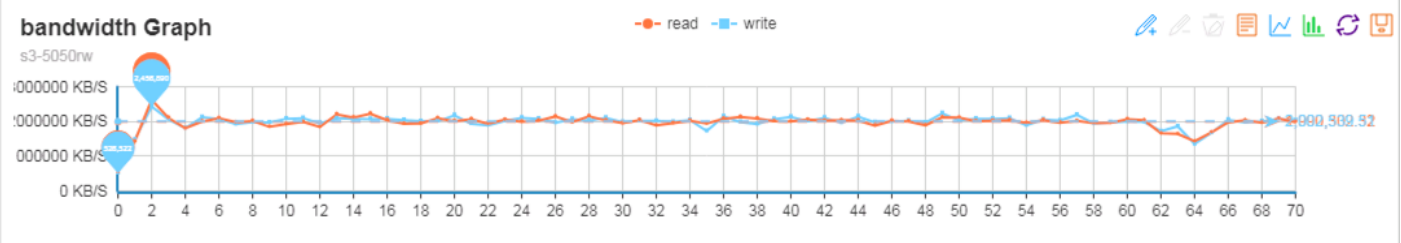
Reloaded the switch to check VPC status and impact on the application.

Similar workload as FI failures above was started on the system:



The N9K switch was reloaded.

```
N9k-Scality-POD2-Fab-A# show version | grep time
Kernel uptime is 63 day(s), 9 hour(s), 46 minute(s), 23 second(s)
N9k-Scality-POD2-Fab-A# configure terminal
N9k-Scality-POD2-Fab-A(config)# reload
This command will reboot the system. (y/n)?  [n] y
N9k-Scality-POD2-Fab-A# show version | grep uptime
Kernel uptime is 0 day(s), 0 hour(s), 3 minute(s), 1 second(s)
```



System was doing writes of around 5.7 GB/s when the Nexus switch was reloaded.

System continues to operate without any interruption.

N9k-Scality-POD2-Fab-A# show vpc brief

Legend:

(*) - local vPC is down, forwarding via vPC peer-link

vPC domain id : 201

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)

vPC Peer-link status

---

id Port Status Active vlans

-- ---- ------ ---

1 Po1 up 1,10,20,30,79

vPC status

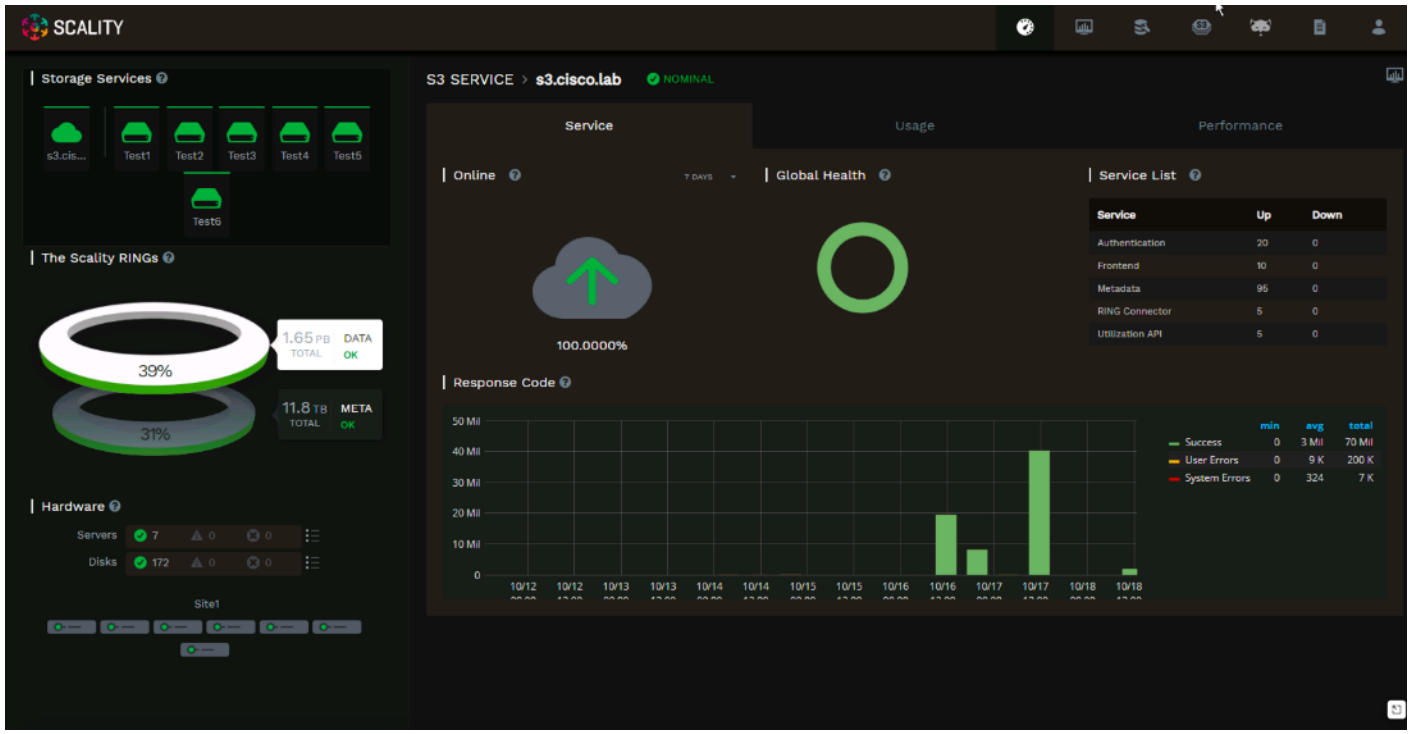
```
--------------------------------------------------------------------------------
id   Port   Status Consistency Reason                Active vlans
--   ----   ------ ----------- ------                ------------
25   Po25   up     success     success               10,20,30,79

26   Po26   up     success     success               10,20,30,79
```

S3 Connector Failures

Client load was generated using COSBench and one of the S3 Connectors (which is also the storage node) was shut down from Cisco UCS.

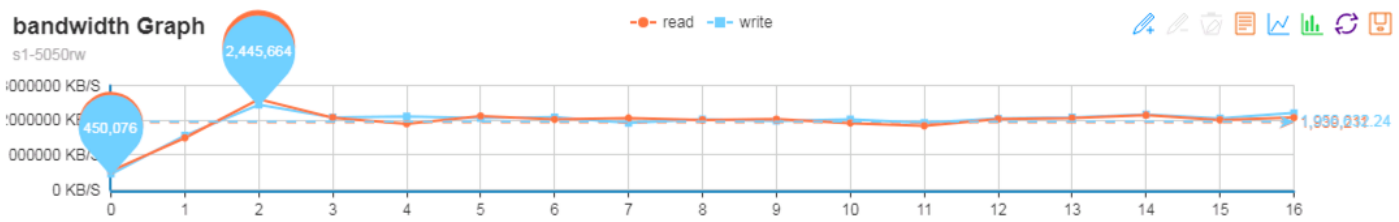
Status of the connectors before fault injection:



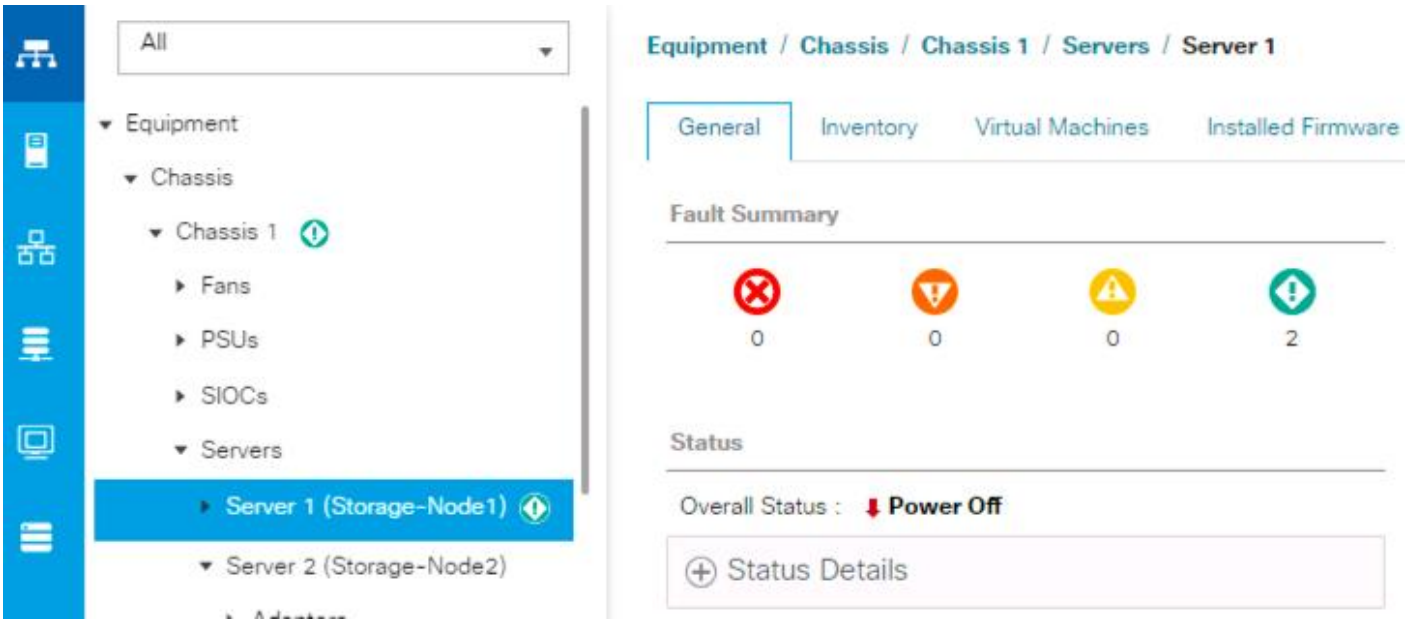
Run COSBench to put some load on the system on all connectors, then power off node2 and check the status.

Writes were approximately 10 GB/s:

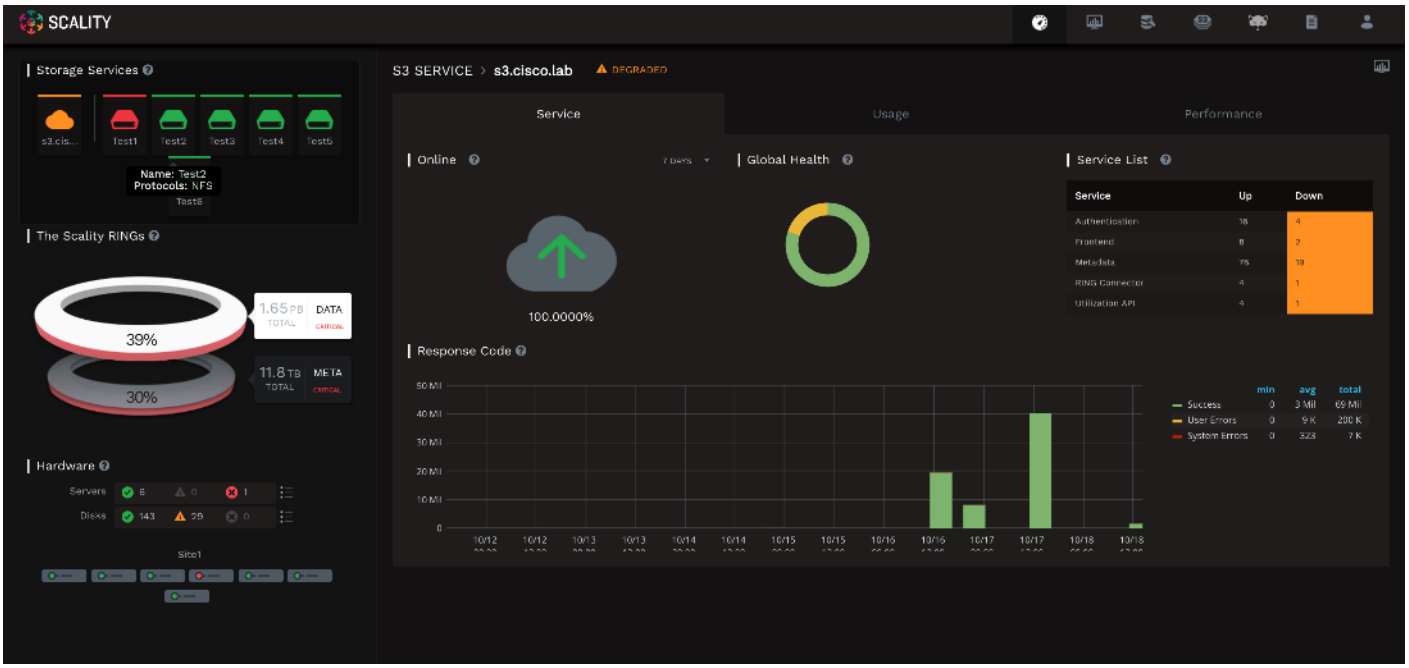
Op-Type	Op-Count	Byte-Count	Avg-ResTime	Avg-ProcTime	Throughput	Bandwidth	Succ-Ratio
op1: read	1.06 kops	10.82 GB	694.13 ms	569.55 ms	205.5 op/s	2.1 GB/S	100%
op2: write	1.05 kops	10.73 GB	764.13 ms	700.92 ms	201.14 op/s	2.06 GB/S	100%



Fault injected by powering off Node 1:

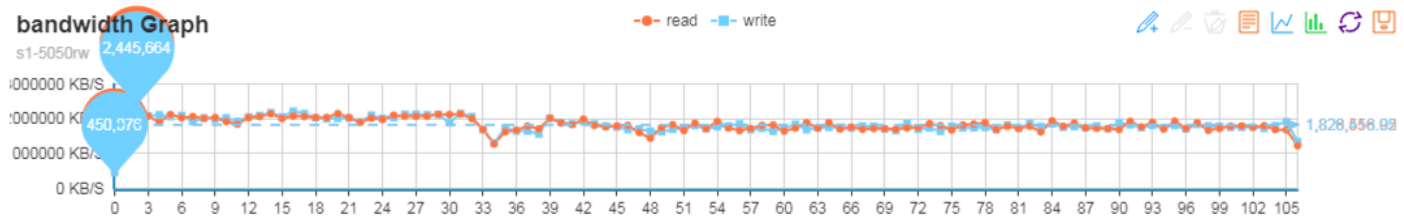


Status as observed from Scality Supervisor console:



The COSBench bandwidth drops to 9.4 GB/s of writes from 10.48 GB/s as shown below. It should be noted that we brought down 1 of the 6 x S3 connectors/storage nodes here.

Op-Type	Op-Count	Byte-Count	Avg-ResTime	Avg-ProcTime	Throughput	Bandwidth	Succ-Ratio
op1: read	822 ops	8.42 GB	1004.83 ms	931.94 ms	165.29 op/s	1.69 GB/S	100%
op2: write	813 ops	8.33 GB	753.42 ms	689.87 ms	160.47 op/s	1.64 GB/S	100%



After running for few minutes the server was brought up again.

Server comes up at:

[root@storage-node1 ~]# uptime

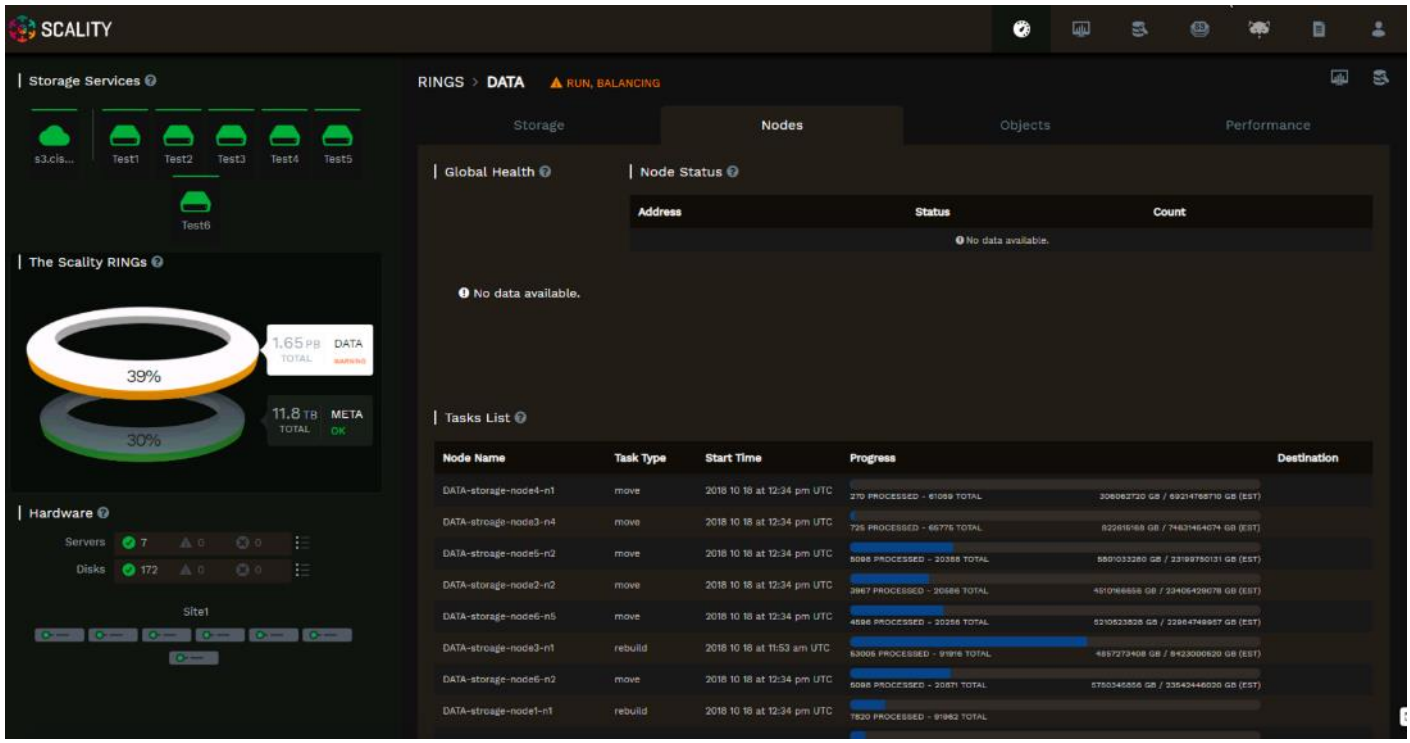
12:25:32 up 1 min, 1 user, load average: 175.66, 47.91, 16.31

Supervisor shows node the failed node as up now:

The storage-node1 joins the cluster and starts sharing the load.

Balancing Kicks Off on Nodes after storage-node1 joins the cluster:

When balancing is on, the following background tasks were observed:



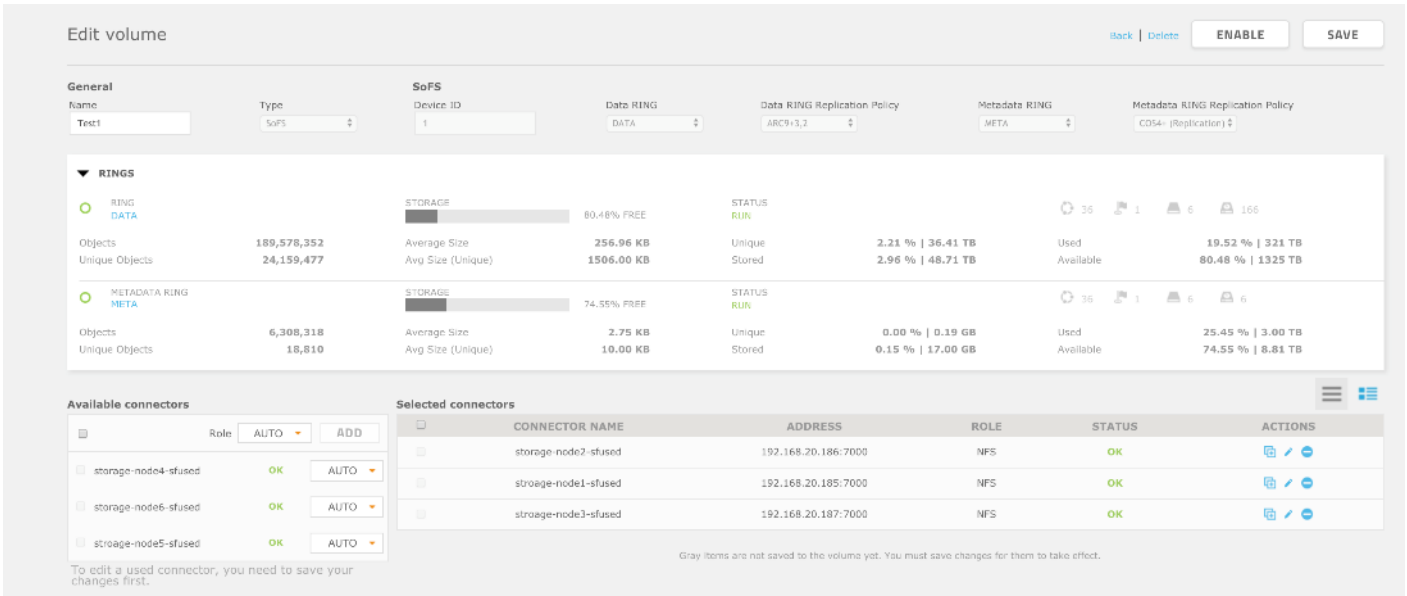
The output drops to 5.5 GB/sec while rebuild is on:

Everything back to normal:

NFS Connector Failures

Scality provides highly available file services with the Scality Virtual Server Daemon (SVSD) a distributed system to manage pools of customer-defined Virtual IP (VIP) addresses. SVSD monitors hardware and file services for failures and will automate VIP failover.

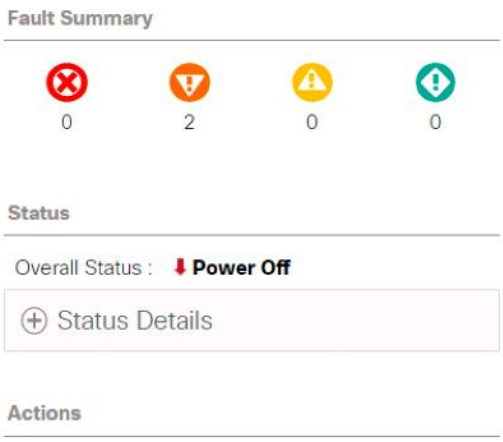
SVSD was configured on three file connectors and a unique NFS export was associated with each of the three VIPs. Three NFS clients were used to generate file IO using the benchmark utility fio during the failover testing.



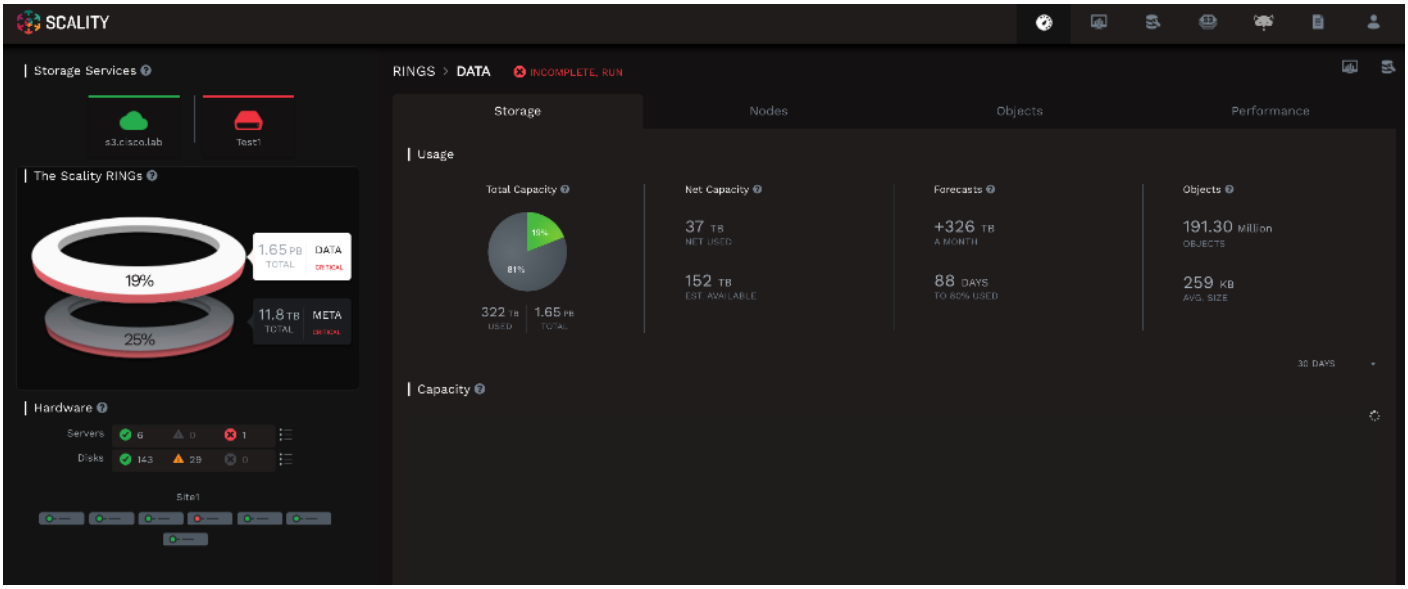
The following output shows the three VIPs 192.168.10.147, 192.168.10.148, 192.168.10.149 located on the network interfaces associated with the real IPs (RIP) 192.168.10.185 (storage-node1), 192.168.10.186 (storage-node2), and 192.168.10.187 (storage-node3) respectively.

```
$ svsdcli -c /etc/svsd.conf -z nfs show
INFO:kazoo.client:Connecting to 192.168.20.188:2181
INFO:kazoo.client:Zookeeper connection established, state: CONNECTED
/scality/svsd/nfs/configs/global:
{
    "vips": {
        "192.168.10.147": {
            "connected": true,
            "mask": 24,
            "preferred": [
                "*"
            ],
            "rip": "192.168.10.185"
        },
        "192.168.10.148": {
            "connected": true,
            "mask": 24,
            "preferred": [
                "*"
            ],
            "rip": "192.168.10.186"
        },
        "192.168.10.149": {
            "connected": true,
            "mask": 24,
            "preferred": [
                "*"
            ],
            "rip": "192.168.10.187"
        }
    }
}
```

Using the Cisco UCS Manager web interface storage-node1 was powered off.



The outage is immediately reported by the Scality Supervisor console.



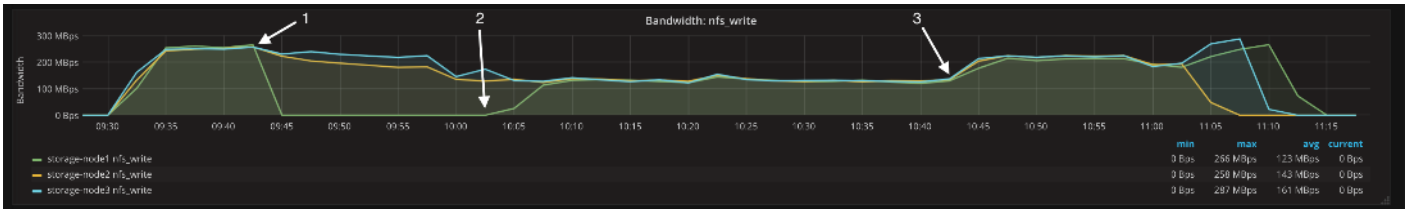
Using svsdcli we can see that the first VIP, 192.168.10.147, has been failed over to the RIP 192.168.10.187 which is associated with storage-node3.

```
$ svsdcli -c /etc/svsd.conf -z nfs show
INFO:kazoo.client:Connecting to 192.168.20.186:2181
INFO:kazoo.client:Zookeeper connection established, state: CONNECTED
/scality/svsd/nfs/configs/global:
{
    "vips": {
        "192.168.10.147": {
            "connected": true,
            "mask": 24,
            "preferred": [
                "*"
            ],
            "rip": "192.168.10.187"
        },
        "192.168.10.148": {
            "connected": true,
            "mask": 24,
            "preferred": [
                "*"
            ],
            "rip": "192.168.10.186"
        },
        "192.168.10.149": {
            "connected": true,
            "mask": 24,
            "preferred": [
                "*"
            ],
            "rip": "192.168.10.187"
        }
    }
}
```

When the server is powered back on the administrator must manually fail the VIP back to storage-node01.

```
$ svsdcli -c /etc/svsd.conf -z nfs move -t 192.168.20.185 192.168.20.92 --force
```

The following bandwidth graph was captured during the outage and illustrates each phase of the outage.

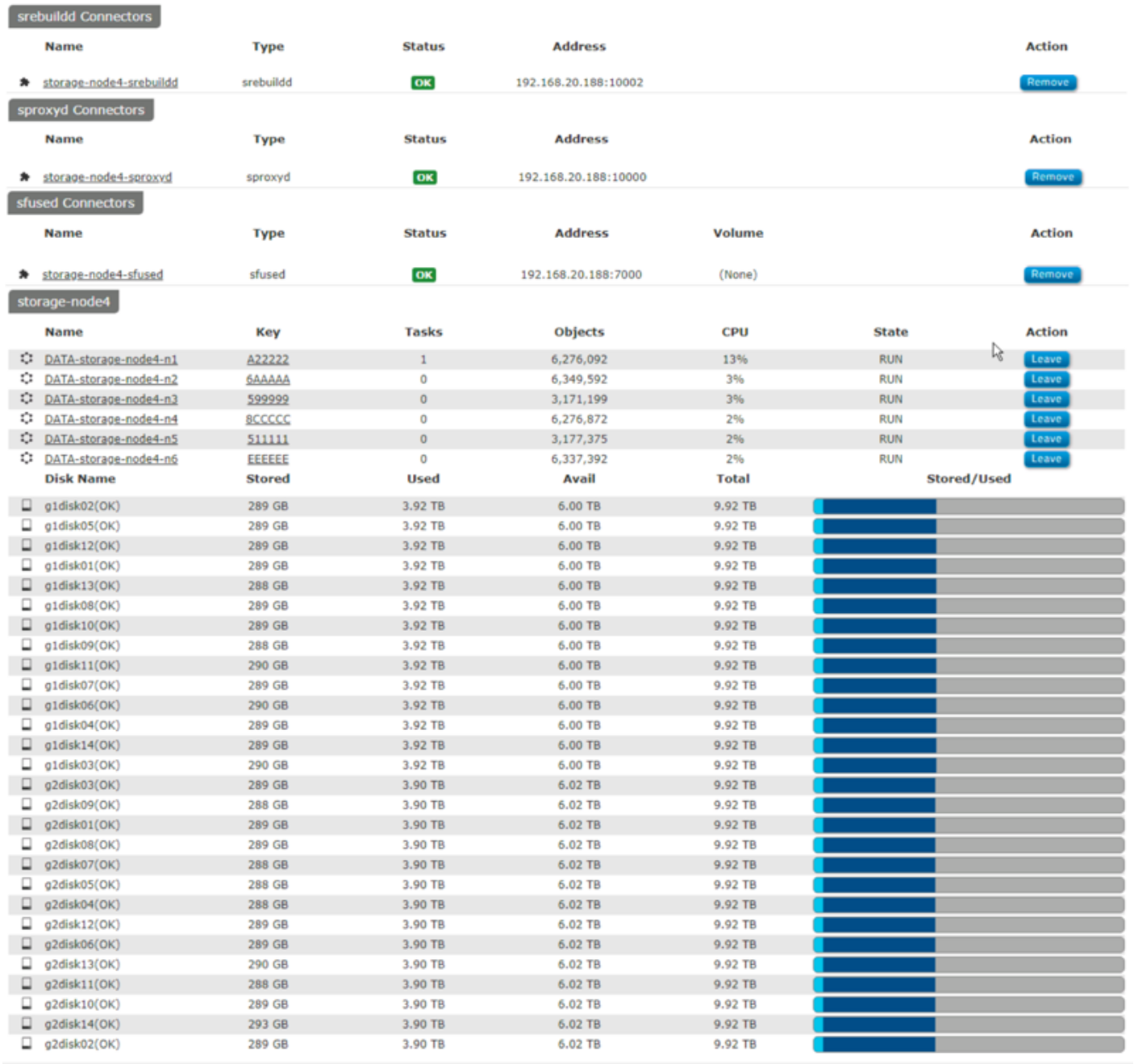


1. The server, storage-node1, was powered off and the VIP failed over to storage-node3.
2. The server storage-node1 was powered on and the VIP was manually failed back.
3. Balances complete and the DATA and META RINGs are in the RUN state.

Disk Failure Tests

Disk Failure was simulated to understand the procedure needed from Cisco UCS and the Scality side to replace a failed disk.

The figure below shows the healthy disk on node 4 as reported by Supervisor and Cisco UCS.



srebuildd Connectors

Name	Type	Status	Address	Action
storage-node4-srebuildd	srebuildd	OK	192.168.20.188:10002	Remove

sproxyd Connectors

Name	Type	Status	Address	Action
storage-node4-sproxyd	sproxyd	OK	192.168.20.188:10000	Remove

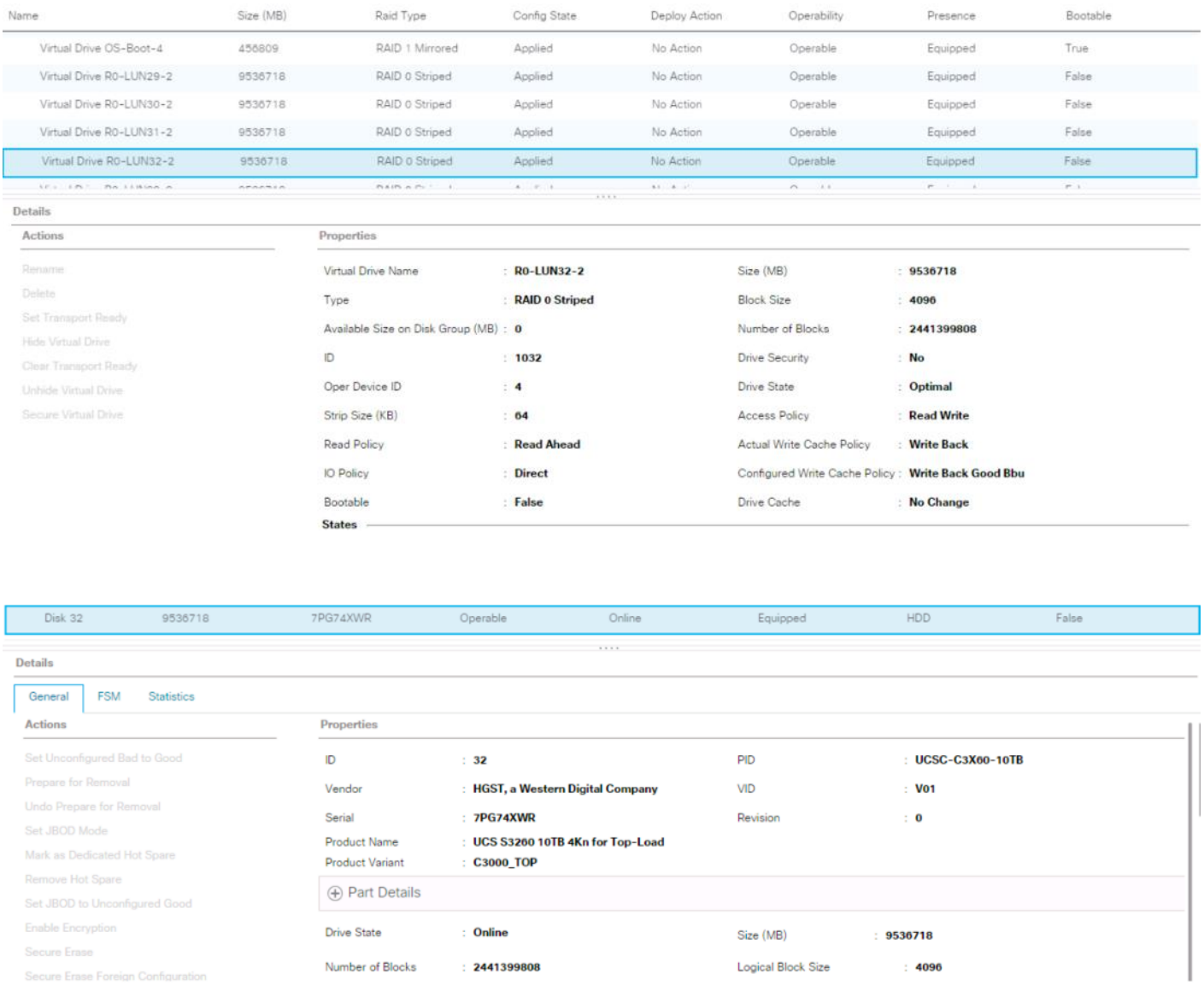
sfused Connectors

Name	Type	Status	Address	Volume	Action
storage-node4-sfused	sfused	OK	192.168.20.188:7000	(None)	Remove

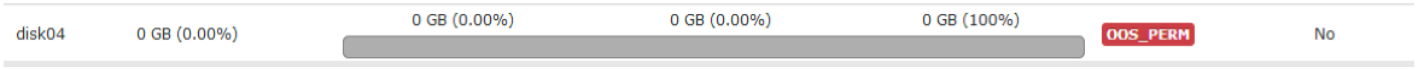
storage-node4

Name	Key	Tasks	Objects	CPU	State	Action
DATA-storage-node4-n1	A22222	1	6,276,092	13%	RUN	Leave
DATA-storage-node4-n2	6AAAAA	0	6,349,592	3%	RUN	Leave
DATA-storage-node4-n3	599999	0	3,171,199	3%	RUN	Leave
DATA-storage-node4-n4	8CCCCC	0	6,276,872	2%	RUN	Leave
DATA-storage-node4-n5	511111	0	3,177,375	2%	RUN	Leave
DATA-storage-node4-n6	EEEEEE	0	6,337,392	2%	RUN	Leave

Disk Name	Stored	Used	Avail	Total	Stored/Used
g1disk02(OK)	289 GB	3.92 TB	6.00 TB	9.92 TB	
g1disk05(OK)	289 GB	3.92 TB	6.00 TB	9.92 TB	
g1disk12(OK)	289 GB	3.92 TB	6.00 TB	9.92 TB	
g1disk01(OK)	289 GB	3.92 TB	6.00 TB	9.92 TB	
g1disk13(OK)	288 GB	3.92 TB	6.00 TB	9.92 TB	
g1disk08(OK)	289 GB	3.92 TB	6.00 TB	9.92 TB	
g1disk10(OK)	289 GB	3.92 TB	6.00 TB	9.92 TB	
g1disk09(OK)	288 GB	3.92 TB	6.00 TB	9.92 TB	
g1disk11(OK)	290 GB	3.92 TB	6.00 TB	9.92 TB	
g1disk07(OK)	289 GB	3.92 TB	6.00 TB	9.92 TB	
g1disk06(OK)	290 GB	3.92 TB	6.00 TB	9.92 TB	
g1disk04(OK)	289 GB	3.92 TB	6.00 TB	9.92 TB	
g1disk14(OK)	289 GB	3.92 TB	6.00 TB	9.92 TB	
g1disk03(OK)	290 GB	3.92 TB	6.00 TB	9.92 TB	
g2disk03(OK)	289 GB	3.90 TB	6.02 TB	9.92 TB	
g2disk09(OK)	288 GB	3.90 TB	6.02 TB	9.92 TB	
g2disk01(OK)	289 GB	3.90 TB	6.02 TB	9.92 TB	
g2disk08(OK)	289 GB	3.90 TB	6.02 TB	9.92 TB	
g2disk07(OK)	288 GB	3.90 TB	6.02 TB	9.92 TB	
g2disk05(OK)	288 GB	3.90 TB	6.02 TB	9.92 TB	
g2disk04(OK)	288 GB	3.90 TB	6.02 TB	9.92 TB	
g2disk12(OK)	289 GB	3.90 TB	6.02 TB	9.92 TB	
g2disk06(OK)	289 GB	3.90 TB	6.02 TB	9.92 TB	
g2disk13(OK)	290 GB	3.90 TB	6.02 TB	9.92 TB	
g2disk11(OK)	288 GB	3.90 TB	6.02 TB	9.92 TB	
g2disk10(OK)	289 GB	3.90 TB	6.02 TB	9.92 TB	
g2disk14(OK)	293 GB	3.90 TB	6.02 TB	9.92 TB	
g2disk02(OK)	289 GB	3.90 TB	6.02 TB	9.92 TB	

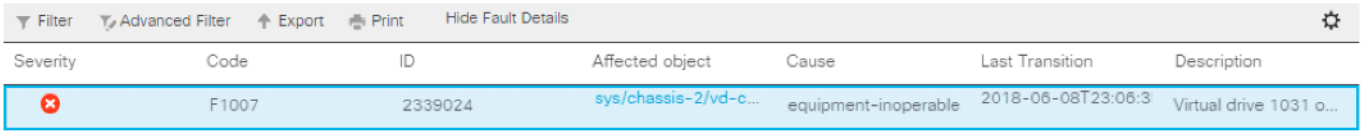


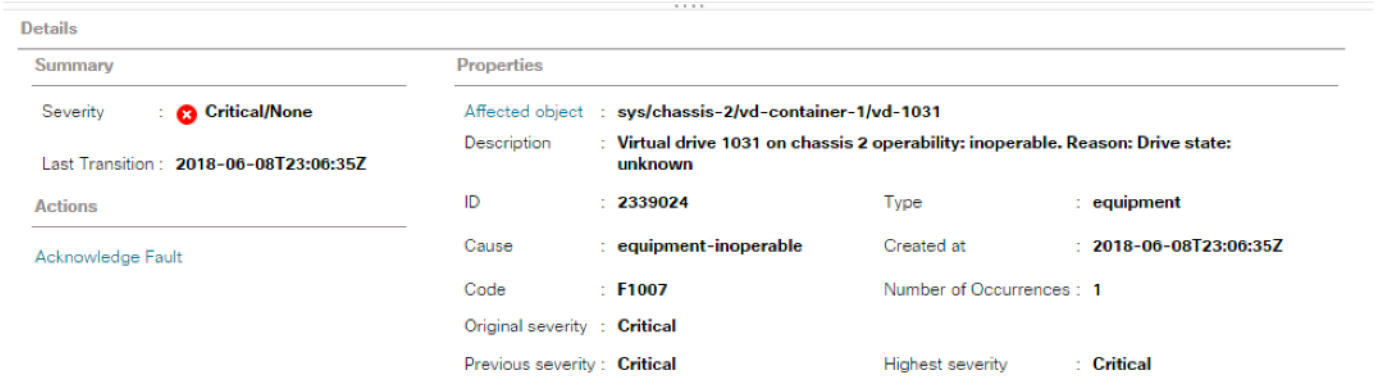
After failing the disk, it displays as a failure in Supervisor.



This disk has to be replaced with a new disk.

Re-Acknowledged the server:

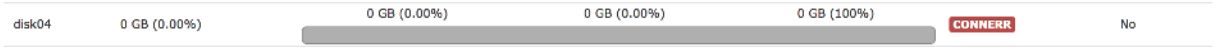




The disk was removed and replaced with a new disk.

If this is a used disk, you may clear foreign configuration in Cisco UCS before assigning it as Unconfigured Good.

The disk is replaced now in Cisco UCS, but supervisor still reports as CONNERR as shown below:

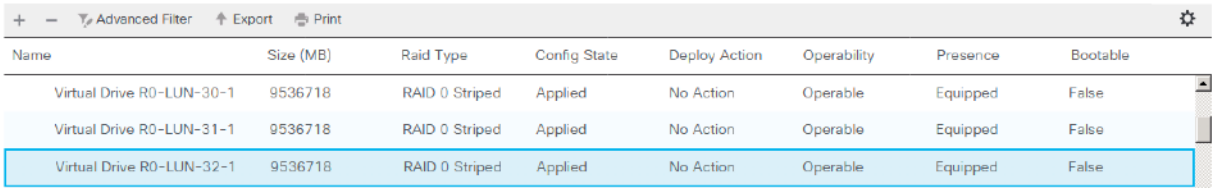


Disk is visible now to the OS, but there is no partition created.

[root@storage-node4 ~]# cat /proc/partitions | grep sdg

8 96 9765599232 sdg

Cisco UCS shows that lun32 is a RAID-0 LUN



Run Scality's scaldisk command to make the disk join the cluster.

```
[root@storage-node4 biziod]# scaldisk replace -d g1disk04 -c /dev/sde
```

Switch on the LED of the disk: g1disk04

Disk g1disk04 with basepath /scality/g1disk04

g1disk04 is going to be replaced with /dev/sde

Please confirm [y/N] y

create a new GPT disk label on /dev/sde (Module function partition.mklabel executed)

make a primary partition on /dev/sde (Module function partition.mkpart executed)

name the partition on /dev/sde (Module function partition.name executed)

reread the partition table on /dev/sde before formatting (Command "sleep 3 && blockdev --rereadpt /dev/sde || true" run)

format /dev/sde1 with an ext4 filesystem (Module function extfs.mkfs executed)

tune the ext4 filesystem on /dev/sde1 (Module function extfs.tune executed)

leave udev time to update /dev/disk/by-* for /dev/sde (Command "udevadm settle" run)

mount /scality/g1disk04 and persist in fstab (check_cmd determined the state succeeded)

update systemd auto generated mount unit (Module function service.systemctl_reload executed)

mount /scality/g1disk04 if systemd unmounted it (Command "systemctl start --now /scality/g1disk04" run)

wipe bizobj.bin for g1disk04 at /scality/ssd1/bizobj-g1disk04/DATA/0/ (Removed file /scality/ssd1/bizobj-g1disk04/DATA/0/bizobj.bin)

clear OOS PERM flag for g1disk04, ring DATA (OOS PERM flag is cleared for ring DATA on disk g1disk04)

/scality/g1disk04/DATA/0 (Directory /scality/g1disk04/DATA/0 updated)

ensure store for DATA on disk g1disk04 exists (Created store for ring DATA on disk g1disk04)

make node DATA-storage-node4-n1 from ring DATA reload keys on g1disk04 (Node DATA-storage-node4-n1 reloaded keys for ring DATA from disk g1disk04)

make node DATA-storage-node4-n2 from ring DATA reload keys on g1disk04 (Node DATA-storage-node4-n2 reloaded keys for ring DATA from disk g1disk04)

make node DATA-storage-node4-n3 from ring DATA reload keys on g1disk04 (Node DATA-storage-node4-n3 reloaded keys for ring DATA from disk g1disk04)

Switch off the LED of the disk: g1disk04

Disk g1disk04 with basepath /scality/g1disk04

DISK g1disk04 SUCCESSFULLY REPLACED

- device name: /dev/sde
- mountpoint: /scality/g1disk04
- used by nodes:
    - DATA-storage-node4-n3 (is running: True)
    - DATA-storage-node4-n2 (is running: True)
    - DATA-storage-node4-n1 (is running: True)

Disks

IOD Name	Stored	Capacity Disk used	Avail	Total	State	Full ?
disk01	0 GB (0.00%)	1.28 GB (0.01%)	9.92 TB (99.99%)	9.92 TB (100%)	OK	No
disk02	0 GB (0.00%)	1.28 GB (0.01%)	9.92 TB (99.99%)	9.92 TB (100%)	OK	No
disk03	0 GB (0.00%)	1.28 GB (0.01%)	9.92 TB (99.99%)	9.92 TB (100%)	OK	No
disk04	0 GB (0.00%)	0.35 GB (0.00%)	9.92 TB (100.00%)	9.92 TB (100%)	OK	No

For more information about scaldisk replace, please refer to the Scality documentation.

Bill of Materials

This section provides the BOM for the entire Scality Storage and Cisco UCS S3260 solution.

Table 7 Bill of Materials for Cisco UCS, Nexus.

Component	Model	Quantity	Comments
Scality Storage Node (also the Connector Node)	Cisco UCS S3260 M5 Chassis	3	• 2 x UCS S3X60 M5 Server Nodes per Chassis (Total = 6nodes) • Per Server Node – 2 x Intel Xeon Silver 4114 (2.2GHz/10cores), 192 GB RAM – Cisco 12G SAS RAID Controller – 2 x 1.6 TB SSD for OS – 2 x 800G SSD for Metadata or 1 x 2 TB of NVMe. – 26 x 10TB HDDs for Data, (28 x 10TB HDD's if using NVMe's for Metadata) – Dual-port 40 Gbps VIC
Scality Supervisor Node	Cisco UCS C220 M5 Rack server	1	• 2 x Intel Xeon Silver 4110 (2.1GHz/8 Cores), 96GB RAM • Cisco 12G SAS RAID Controller • 2 x 600GB SAS for OS • Dual-port 40 Gbps VIC
UCS Fabric Interconnects	Cisco UCS 6332 Fabric Interconnects	2	
Switches	Cisco Nexus 9332PQ Switches	2	

Appendix

Appendix A – Kickstart File of Supervisor Node for Cisco UCS C220 M5

Kickstart File for Supervisor Node

```
#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=128.107.79.211 --netmask=255.255.255.0 --onboot=on --
gateway=128.107.79.1 --nameserver=171.70.168.183 --ipv6=auto --activate

network  --bootproto=static --device=eth1 --ip=192.168.10.191 --netmask=255.255.255.0 --onboot=on --ipv6=auto --
activate

network  --bootproto=static --device=eth2 --ip=192.168.20.191 --netmask=255.255.255.0 --onboot=on --ipv6=auto -
activate

network  --bootproto=static --device=eth3 --ip=192.168.30.191 --netmask=255.255.255.0 --onboot=on --ipv6=auto --
activate


network  --hostname=supervisor
```

```
# Root password

rootpw --iscrypted
$6$yfE2jHtdy.OSmO8g$InneiVXQI9Kc9m4w2cEiS8/og6BKUlu5HSR0eCYgh5dVaeCV54Q6piS7k10lalXignLCBvAZPqmw4dvY
gy66V1

# 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
192.168.10.191        supervisor salt
192.168.10.185        storage-node1
192.168.10.186        storage-node2
192.168.10.187        storage-node3
192.168.10.188        storage-node4
192.168.10.189        storage-node5
192.168.10.190        storage-node6
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
5yI8hxn0in+RmferTq8WwyZihMV0EyN4q5HfT+gdbSY6xPMM9UHF89+IYNNxdZ4/
VuBcbBskey3UbQ332KqA7wS+Sra2DXmnfysWbwIDAQABAoIBAQCbeRFUXiyR5IP9
5Iyw9k9HYRX/OfGLLumSMnJyb1wzzP9cHcPeh/V8QihLadxHVZTHXZRXcHG19pFE
7rx2y7RVU2gUIDCkchd4nEG9EYKvF1u66GLE3I7zH5Nwj/sQkfAKMZ26rTC8sUsG
mBUUWKzE+K7Fklj6ud7WidZHxKH32ok1IEcFOsH/nK1BXR29XmQ/O/Kg2h0V/KiM
1Y9CJngpghnybcDzlvpV6LS8bEiRieHJGT5RTyDk+ad0uSv+f2YtlpvSUly7NAft
e1feAq3RWT82ZGyKTHWGTFNbfltcUjzPI/dcyS8AurYf+oQjJVAKhAI+yln7IUrL
V6xKsdYBAoGBANwNb96gJHZUeSoOP/JCnTps+MeOhT1vyrhRRZf1IaFnEmX7hXmE
RKXaQUvGcOSPumZMkKYyqRN22B2PLM7n1D0ypKshRmk1eq6tZ/W9gkYfldn0+QAx
AAVfUA8vJm9XLgkCAE4o2BHvtQ1w63CfygoF4V3OAsQv677F6ItROeiBAoGBAM6A
9quEOrPiRDiF25HnXXFUeRUXM4H77QB6WRV3AKggJjVlBXkhNt34g8Jr6/MfW4WO
SebQEwwBYH6NN7IG1Q0PeDRzrcv2voqzM7bV7I1rpc2E2BQhplcSyGr/aA6lW0OA
LI/HZIdqb6OXXR8lmcP0rfxuqUJ8e6SHskG6qAbvAoGAlrw4QXMT7I3NNndDXtFn
EjbrWkzD+XuxC0FA9Aisw1aKz/BRFGptj6SRFA4B+gI6ETXay3FJwRnMaXYVQ5/S
n8pjteOtwqO/dt1GgMLmUn1NkaMavw39C9wMvijaL08apC9drvjBiqtE8Bc4Avlm
KUjeVzlStHdABkAlQgCTXlECgYEAur6BU4YWmAnsa7kRYRZ7uDsN7Ha4y7mJED+U
RAcD/wZjxzF+C5ZvybgtXyq9i3U2DMcqKaLNNrQgERGf5kyrak4tBDlAX0zZ7xAz
mgplrw7kN8EErt/nTyLbP3eNIlGE0LwgM9lbHeKw5p3BRok+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/DtYlaOaG67+u6stWg
D3R+NkNBjpoQB0dlf6jbuYfqF+QxWrK6fBDq7cy1SqqTBErY20QmokGldnD35uaCh/IViXnAFJY8YiKDSRvyb5wGbS5GgTIlUW4A
Zzu6weR8gWU5BB32/P2Ho5fxtrdzrJQBkPNZKe3a53ls5OpXhl+lBjg7Y29iCbVWluUe9S+Y/ti7nKXyHGfSKf5GZ96tOrxbDJmKE
xXQTl3irkd9P6B1tJjrE8wz1QcXz36Vg03F1nj9W4FxsGyR7LdRtDffYqoDvBL5KtrYNead/KxZv root@storage-node7

EOF6

cat > /root/.ssh/authorized_keys <<EOF7

ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAABAQCxgarFbFB0ZSyJTNp9gu5foxUPeaOrevopovRBw1K5nwXdd/DtYlaOaG67+u6stWg
D3R+NkNBjpoQB0dlf6jbuYfqF+QxWrK6fBDq7cy1SqqTBErY20QmokGldnD35uaCh/IViXnAFJY8YiKDSRvyb5wGbS5GgTIlUW4A
Zzu6weR8gWU5BB32/P2Ho5fxtrdzrJQBkPNZKe3a53ls5OpXhl+lBjg7Y29iCbVWluUe9S+Y/ti7nKXyHGfSKf5GZ96tOrxbDJmKE
xXQTl3irkd9P6B1tJjrE8wz1QcXz36Vg03F1nj9W4FxsGyR7LdRtDffYqoDvBL5KtrYNead/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 of Storage Nodes for Cisco UCS S3260 M5 Server

Kickstart File for Storage-node1

```
#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=128.107.79.205 --netmask=255.255.255.0 --onboot=on --
gateway=128.107.79.1 --nameserver=171.70.168.183 --ipv6=auto --activate

network  --bootproto=static --device=eth1 --ip=192.168.10.185 --netmask=255.255.255.0 --onboot=on --ipv6=auto --
activate

network  --bootproto=static --device=eth2 --ip=192.168.20.185 --netmask=255.255.255.0 --onboot=on --ipv6=auto –
activate

network  --bootproto=static --device=eth2 --ip=192.168.30.185 --netmask=255.255.255.0 --onboot=on --ipv6=auto --
activate

network  --hostname=storage-node1


# Root password

rootpw --iscrypted
$6$yfE2jHtdy.OSmO8g$InneiVXQI9Kc9m4w2cEiS8/og6BKUlu5HSR0eCYgh5dVaeCV54Q6piS7k10lalXignLCBvAZPqmw4dvY
gy66V1

# 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 nr_cpus=24";
```

```
tuned-adm profile latency-performance;
systemctl enable ntpd;
################
#Preconfigure /etc/hosts
################
cat >> /etc/hosts <<EOF4
192.168.10.191        supervisor salt
192.168.10.185        storage-node1
192.168.10.186        storage-node2
192.168.10.187        storage-node3
192.168.10.188        storage-node4
192.168.10.189        storage-node5
192.168.10.190        storage-node6
EOF4
################
#Setup ssh keys
################
mkdir /root/.ssh;
cat > /root/.ssh/id_rsa <<EOF5
-----BEGIN RSA PRIVATE KEY-----
MIIEpAIBAAKCAQEAsYGqxWxQdGUsiUzafYLuX6MVD3mjq3r6KaL0QcNSuZ8F3Xfw
7WJWjmhuu/rurLVoA90fjZDQY6aEAdHSH+o27mH6hfkMVqyunwQ6u3MtUqqkwRK2
NtEJqJBiHZw9+bmgofyFYl5wBSWPGlig0kb8m+cBm0uRoE5SFFuAGc7usHkfIFIO
QQd9vz9h6OX8ba3c6yUAZDzWSnt2udyLOTqV4SPpQY4O2NvYgm1VpbIHvUvmP7Yu
5yI8hxn0in+RmferTq8WwyZihMV0EyN4q5HfT+gdbSY6xPMM9UHF89+IYNNxdZ4/
VuBcbBskey3UbQ332KqA7wS+Sra2DXmnfysWbwIDAQABAoIBAQCbeRFUXiyR5IP9
5Iyw9k9HYRX/OfGLLumSMnJyb1wzzP9cHcPeh/V8QihLadxHVZTHXZRXcHG19pFE
7rx2y7RVU2gUIDCkchd4nEG9EYKvF1u66GLE3I7zH5Nwj/sQkfAKMZ26rTC8sUsG
mBUUWKzE+K7Fklj6ud7WidZHxKH32ok1IEcFOsH/nK1BXR29XmQ/O/Kg2h0V/KiM
1Y9CJngpghnybcDzlvpV6LS8bEiRieHJGT5RTyDk+ad0uSv+f2YtlpvSUly7NAft
e1feAq3RWT82ZGyKTHWGTFNbfltcUjzPI/dcyS8AurYf+oQjJVAKhAI+yln7IUrL
V6xKsdYBAoGBANwNb96gJHZUeSoOP/JCnTps+MeOhT1vyrhRRZf1laFnEmX7hXmE
```

RKXaQUvGcOSPumZMkKYyqRN22B2PLM7n1D0ypKshRmk1eq6tZ/W9gkYfldn0+QAx

AAVfUA8vJm9XLgkCAE4o2BHvtQ1w63CfygoF4V3OAsQv677F6ItROeiBAoGBAM6A

9quEOrPiRDiF25HnXXFUeRUXM4H77QB6WRV3AKggJjVlBXkhNt34g8Jr6/MfW4WO

SebQEwwBYH6NN7lG1Q0PeDRzrcv2voqzM7bV7l1rpc2E2BQhplcSyGr/aA6lW0OA

LI/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/DtYlaOaG67+u6stWg
D3R+NkNBjpoQB0dlf6jbuYfqF+QxWrK6fBDq7cy1SqqTBErY20QmokGldnD35uaCh/lViXnAFJY8YiKDSRvyb5wGbS5GgTllUW4A
Zzu6weR8gWU5BB32/P2Ho5fxtrdzrJQBkPNZKe3a53ls5OpXhl+lBjg7Y29iCbVWluUe9S+Y/ti7nKXyHGfSKf5GZ96tOrxbDJmKE
xXQTl3irkd9P6B1tJjrE8wz1QcXz36Vg03F1nj9W4FxsGyR7LdRtDffYqoDvBL5KtrYNead/KxZv root@storage-node7

EOF6

cat > /root/.ssh/authorized_keys <<EOF7

ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAABAQCxgarFbFB0ZSyJTNp9gu5foxUPeaOrevopovRBw1K5nwXdd/DtYlaOaG67+u6stWg
D3R+NkNBjpoQB0dlf6jbuYfqF+QxWrK6fBDq7cy1SqqTBErY20QmokGldnD35uaCh/lViXnAFJY8YiKDSRvyb5wGbS5GgTllUW4A
Zzu6weR8gWU5BB32/P2Ho5fxtrdzrJQBkPNZKe3a53ls5OpXhl+lBjg7Y29iCbVWluUe9S+Y/ti7nKXyHGfSKf5GZ96tOrxbDJmKE
xXQTl3irkd9P6B1tJjrE8wz1QcXz36Vg03F1nj9W4FxsGyR7LdRtDffYqoDvBL5KtrYNead/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

Appendix C – Platform Description File

ring,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,
sizing_version,customer_name,#ring,data_ring_name,meta_ring_name,HALO API key,S3 endpoint,cos,arc-data,arc-coding,,,,,,,,,,,,,,,,,,
18.9,Cisco CVD,2,DATA,META,0,s3.cisco.lab,2,9,3,,,,,,,,,,,,,,,,,,
,,,,,,,,,,,,,,,,,,,,,,,,,,
servers,,,,,,,,,,,,,,,,,,,,,,,,,,,,
data_ip,data_iface,mgmt_ip,mgmt_iface,s3_ip,s3_iface,svsd_ip,svsd_iface,ring_membership,role,minion_id,enclosure,site,#cpu,cpu,ram,#nic,nic_size,#os_disk,os_disk_size,#data_disk,data_disk_size,#raid_card,raid_cache,raid_card_type,#ssd,ssd_size,#ssd_for_s3,ssd_for_s3_size
192.168.20.185,eth2,192.168.10.185,eth1,,,,,"DATA,META","storage,native rest,s3,s3_md,nfs,elastic",storage-node1,Cisco UCS S3260 M5 (Dual node),site1,2,Intel Xeon Gold 5118 (2.30GHz/12 cores),192,2,40,2,500,28,10000,1,2,Cisco 12Gbps Modular RAID PCIe Gen 3.0,1,2000,0,0
192.168.20.186,eth2,192.168.10.186,eth1,,,,,"DATA,META","storage,native rest,s3,s3_md,nfs,elastic",storage-node2,Cisco UCS S3260 M5 (Dual node),site1,2,Intel Xeon Gold 5118 (2.30GHz/12 cores),192,2,40,2,500,28,10000,1,2,Cisco 12Gbps Modular RAID PCIe Gen 3.0,1,2000,0,0
192.168.20.187,eth2,192.168.10.187,eth1,,,,,"DATA,META","storage,native rest,s3,s3_md,nfs,elastic",storage-node3,Cisco UCS S3260 M5 (Dual node),site1,2,Intel Xeon Gold 5118 (2.30GHz/12 cores),192,2,40,2,500,28,10000,1,2,Cisco 12Gbps Modular RAID PCIe Gen 3.0,1,2000,0,0
192.168.20.188,eth2,192.168.10.188,eth1,,,,,"DATA,META","storage,native rest,s3,s3_md,nfs,elastic",storage-node4,Cisco UCS S3260 M5 (Dual node),site1,2,Intel Xeon Gold 5118 (2.30GHz/12 cores),192,2,40,2,500,28,10000,1,2,Cisco 12Gbps Modular RAID PCIe Gen 3.0,1,2000,0,0
192.168.20.189,eth2,192.168.10.189,eth1,,,,,"DATA,META","storage,native rest,s3,s3_md,nfs,elastic",storage-node5,Cisco UCS S3260 M5 (Dual node),site1,2,Intel Xeon Gold 5118 (2.30GHz/12 cores),192,2,40,2,500,28,10000,1,2,Cisco 12Gbps Modular RAID PCIe Gen 3.0,1,2000,0,0
192.168.20.190,eth2,192.168.10.190,eth1,,,,,"DATA,META","storage,native rest,nfs,elastic",storage-node6,Cisco UCS S3260 M5 (Dual node),site1,2,Intel Xeon Gold 5118 (2.30GHz/12 cores),192,2,40,2,500,28,10000,1,2,Cisco 12Gbps Modular RAID PCIe Gen 3.0,1,2000,0,0
192.168.20.191,eth2,192.168.10.191,eth1,,,,,,supervisor,supervisor,Cisco UCS C220 M5,site1,2,Intel Xeon Platinum 8180 (2.50GHz/28 cores),96,2,40,2,500,0,0,0,0,,0,0,0,0

About the Authors

Muhammad Ashfaq, Cisco Systems, Inc.

Muhammad Ashfaq is a Technical Marketing Engineer in Cisco UCS and Data Center Solutions Group. He has over 10 years of experience in IT Infrastructure, Server Virtualization and Cloud Computing. His current role includes building Cloud Computing, Software defined Storage, Automation & Management, Converged and Hyper-Converged Solutions on Cisco UCS platforms. Muhammad also holds Cisco Internetwork Expert Data Center Certification (CCIE-DC).

William Kettler, Scality

William Kettler is a Customer Solution Engineer Partner within Scality's Technical Services group. His current role includes helping customers deploy their petabyte-scale storage solutions, certifying strategic ISVs, and being a technical resource for Scality partners like Cisco.

Acknowledgements

For their support and contribution to the design, validation, and creation of this Cisco Validated Design, we would like to acknowledge the following people for their significant contribution and expertise that resulted in developing this document:

- RamaKrishna Nishtala, Cisco Systems, Inc.
- Jawwad Memon, Cisco Systems, Inc.
- Maziar Tamadon, Scality