



NetApp Verified Architecture

Cisco Secure AI Factory with NVIDIA & FlexPod

With NetApp AFX & AIDE

NVA Design & Deployment

Bobby Oommen, Abhinav Singh, Roney Daniel - NetApp

May 2026 | NVA-1187-DEPLOY-0426

In partnership with



Abstract

This NetApp Verified Architecture describes a secure, scalable enterprise AI Factory built on Cisco® AI POD powered by FlexPod® Converged Infrastructure. It combines Cisco compute with NVIDIA AI-accelerated architectures with high-performance Cisco networking and FlexPod's enterprise data foundation using NetApp® AFX disaggregated solution. The solution enables customers to operationalize AI from training and RAG pipelines to inference on a validated platform that reduces deployment risk, accelerates time to value, and scales securely from pilot to production.

TABLE OF CONTENTS

Executive summary	5
Target audience	5
Business Value	5
Why AI on FlexPod	5
Advantages of Running AI Workloads on NetApp	6
Running AI Workloads on Cisco AI Accelerated Compute	6
Cisco Secure AI Factory with NVIDIA and FlexPod	6
Solution overview	9
Technology requirements	9
Hardware requirements	9
Software requirements	9
Concepts and Components	10
Compute Layer (Cisco UCS C240 M8 with NVIDIA RTX PRO 6000)	11
Networking and Specifications	12
Cisco Intersight for Manageability	12
NetApp AFX 1K	12
AFX 1K Storage Nodes	13
NX224 NVMe enclosure	13
NetApp ONTAP 9	13
NetApp ONTAP S3	13
NetApp Trident & Trident Protect	14
Red Hat OpenShift	14
Red Hat OpenShift AI	15
NVIDIA AI Factory Eco-system	15
Infrastructure and Application Monitoring with Splunk Observability	16
Solution Design	16
Data Services – with NetApp AIDE	20
Solution Deployment	20
Network configuration	21
Cisco UCS C-Series server configuration	26

NetApp AFX installation and configuration	27
OpenShift installation	32
Post installation setup	39
NVIDIA Network and GPU Operator	49
NetApp Trident	57
Splunk Observability	59
Solution Validation	61
Elbencho Benchmarking	62
RDMA Validation	64
Validation of GDS and NFSoRDMA	70
NVIDIA Dynamo	74
NVIDIA AIPerf-LLM Metrics	76
Solution Scaling Guidance	80
Cisco UCS C240 M8 configuration - AI Acceleration	80
Licensing	85
Conclusion	85
Acknowledgments	86
Bill of Materials	86
Additional Information	88
NetApp Documentation	88
Cisco UCS M8 Rack Servers & Management Software	88
Cisco Nexus Switches	88
NVIDIA Documentation	88
Red Hat OpenShift	88
Transceiver	89
Version history	89

LIST OF TABLES

Table 1) Hardware requirements	9
Table 2) Software requirements	9
Table 3) MGX vs. HGX - At a glance	11

Table 4) Solution Networking.....	12
Table 5) DNS and IP Address.....	32
Table 6) Host information.....	33
Table 7) Scaling on C240 MGX configuration.....	81
Table 8) MGX with Cisco UCS C845A M8 in 2-8-5-400 listing E/W Topology	82
Table 9) MGX with UCS C845A M8 in 2-8-5-400 listing N/S Topology	83
Table 10) HGX with Cisco UCS C885A M8 in 2-8-9-400 listing E/W Topology.....	85
Table 11) HGX with Cisco UCS C885A M8 in 2-8-9-400 listing N/S Topology.....	85

LIST OF FIGURES

Figure 1) Cisco Secure AI Factory with NVIDIA & FlexPod.....	7
Figure 2) Cisco Nexus 9332D-GX2B.....	12
Figure 3) NetApp Trident deployed on a High Availability Kubernetes Cluster	14
Figure 4) Solution Topology of AFX 1K	17
Figure 5) Cisco UCS Compute Topology.....	18
Figure 6) NFS - nconnect vs. session trunking	19
Figure 7) Topology with AIDE Data Services.....	20
Figure 8) Cisco UCS C240 M8 configuration.....	80
Figure 9) Scaling with C845A M8 in MGX up to 32 Nodes	82
Figure 10) Scaling with C885A M8 in HGX up to 32 Nodes	84

Executive summary

This NetApp FlexPod Verified Architecture delivers a secure, scalable enterprise AI Factory, combining Cisco AI POD innovation with FlexPod as the foundational platform for enterprise-ready AI. The solution brings together Cisco servers with AI-accelerated computing based on NVIDIA MGX and HGX architectures, high-performance Cisco networking, and cloud-native lifecycle management to enable customers to deploy AI infrastructure that scales predictably from pilot to production.

FlexPod, powered by NetApp AFX storage, ONTAP® data services, and NetApp AIDE (AI Data Engine), provides the unified data and storage layer required to sustain data-intensive AI workloads, including model training, RAG pipelines, and high-throughput inference. Validated end-to-end with real-world performance, networking, and MLOps benchmarks on Red Hat OpenShift AI, this architecture reduces deployment risk, simplifies operations, and accelerates time to value helping enterprises turn data into secure, governed AI outcomes at scale.

This document is structured to support both architectural planning and deployment execution. The initial sections up to NetApp ONTAP S3 focus on the overall solution architecture and design principles. The subsequent sections provide validated, step-by-step guidance for deploying and operationalizing the solution in production environments.

Target audience

This document is intended for Data Scientists, AI Architects/Engineers, and Field Consultants seeking to utilize infrastructure optimized for AI workloads. Familiarity with AI, cloud-native environments, networking, storage, and related components will facilitate a more effective implementation process.

Business Value

Cisco® compute & networking combined with NetApp® storage provide key benefits to customers such as:

- Flexible reference architecture for any AI workloads
- Optimized infrastructure for high-performance and scalability of compute with any NVIDIA GPUs
- Disaggregated data management solution from NetApp scaling performance and capacity independently
- Solution guidance in terms of Compute, Network, Storage and validation using benchmarking tools & AI models
- Reliability and security - Data protection, built-in ransomware protection, etc

Why AI on FlexPod

AI on FlexPod enables enterprises to move from AI experimentation to production with confidence, using a proven, modular architecture. It simplifies operations, reduces risk, and scales AI workloads predictably across training, RAG, and inference.

Advantages of Running AI Workloads on NetApp

The objective of this solution is to help enterprises to be successful in their AI journey, with an architecture that provides a powerful data management plane which provides the following benefits:

- Velocity – Handling large datasets at high velocity for versioning
- Disaggregation – Ability to scale performance and capacity independently
- Reliability – Data protection, built-in ransomware protection, and dynamic provisioning of storage
- Cloud Adjacency – Connect with resources across multiple data center and clouds to load GenAI knowledgebase repositories
- Efficiency – Enterprise storage features such as compression, deduplication thin provisioning on data sets
- Secure Multi-Tenancy – Adaptive QoS to isolate multiple AI workloads along with encryption for data at-rest and in-flight and using multiple SVMs [Storage Virtual Machine]
- Automated Ransomware Protection

Running AI Workloads on Cisco AI Accelerated Compute

Using Cisco AI-accelerated servers and infrastructure such as UCS X-Series, UCS C845A M8, UCS C885A M8 and UCS C240 M8 offers key technical and strategic benefits over general-purpose or legacy hardware. Advantages include better performance, flexibility, scalability, cost savings, and simpler operations, especially for enterprise and distributed AI applications.

Cisco servers offer high-density NVIDIA GPUs (including models such as H100, H200, and RTX PRO to provide the large-scale parallel processing required by AI workloads accelerating both model training and inference. Tools such as Run:ai on Cisco UCS offer smart GPU scheduling, quota management, fractional sharing, and multi-instance support to maximize compute resource efficiency.

By efficiently utilizing GPUs and CPUs and incorporating energy-saving designs, total cost of ownership (TCO) can be lowered compared to relying on an excess of generic servers, while also increasing throughput. Centralized tools like Cisco Intersight and partner integrations automate, monitor, and control policies for AI infrastructure, minimizing operational effort.

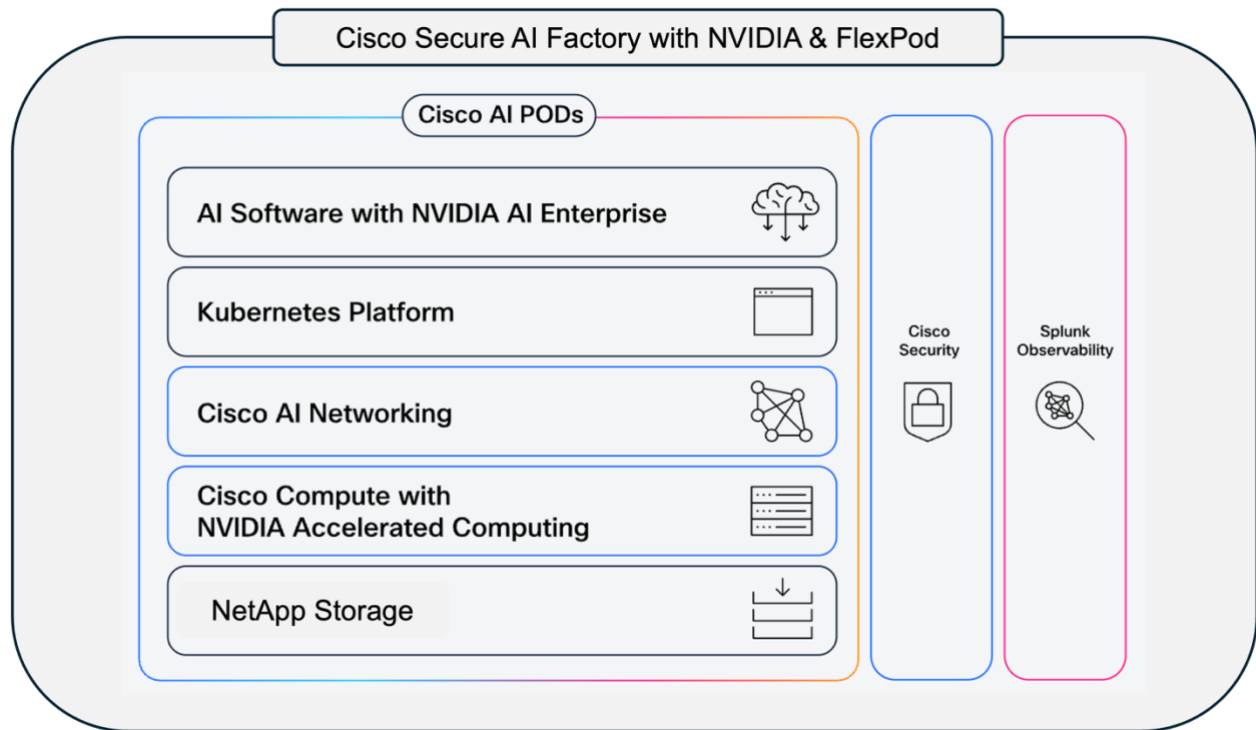
Cisco networking and zero-trust framework can integrate with AI nodes to protect data essential for regulated industries. Cisco switches and fabrics deliver fast, reliable connections between GPUs, storage, and apps which are key for distributed training.

Cisco Secure AI Factory with NVIDIA and FlexPod

Cisco® Secure AI Factory with NVIDIA and NetApp ([Figure 1](#)) is the “blueprint/framework for enterprise AI infrastructure.” The concept of an AI Factory, popularized by Cisco Systems in partnership with NVIDIA and NetApp which is a NVIDIA-certified storage partner, represents a fundamental shift in how enterprises think about AI infrastructure. Instead of traditional IT environments that run applications, AI Factories are designed to manufacture intelligence at scale continuous training, refining, and serving models across the enterprise.

At the core of this vision, FlexPod is tightly integrated into Cisco AI POD to form a modular, validated, and scalable AI infrastructure stack that delivers a complete AI infrastructure platform. The main benefits give customers reduced deployment risk and accelerated time to value, while providing scalable AI clusters with optimal tokens per watt.

Figure 1) Cisco Secure AI Factory with NVIDIA & FlexPod



What is Cisco Secure AI Factory with NVIDIA?

Cisco Secure AI Factory is not a single product, rather it is a reference architecture and operational model that enables:

- Continuous AI lifecycle
- Secure, governed AI pipelines
- Scalable GPU infrastructure
- High-performance data movement with NetApp ONTAP when FlexPod is used

The framework provides an industrialized AI platform, using data as the raw material to deliver outcomes in the form of models and decision-making. The framework is positioned as a solution that emphasizes security, observability, and governance across the AI lifecycle.

Cisco AI POD: The Core Building Block

Cisco AI PODs are modular, full-stack infrastructure solutions designed to support AI workloads such as training, fine-tuning, inferencing, and optimization. Each POD integrates accelerated compute, high-bandwidth/low-latency networking, a Kubernetes (or HPC) software stack, and unified management, so it can be deployed as a repeatable unit and scaled out over time.

Cisco AI POD components

Compute: Cisco UCS® GPU-accelerated servers (for example, UCS C-Series platforms aligned to NVIDIA HGX/MGX designs) delivering dense, scalable GPU capacity.

Cisco AI Networking: Cisco Nexus data center switching forming high-bandwidth fabrics for both East–West GPU traffic and North–South connectivity to storage, services, and users.

Workload platform: Enterprise Kubernetes (commonly Red Hat OpenShift) or HPC-oriented stacks, enabling scheduling, multi-tenancy, and lifecycle management of AI workloads.

Management: Cisco Intersight for compute lifecycle management and Cisco Nexus Dashboard for fabric operations, provisioning, and telemetry.

Splunk Observability: Cisco Splunk Observability Cloud for full-stack visibility across infrastructure, platform, and AI workloads.

Cisco Security in the Cisco Secure AI Factory is built in by design, delivering end-to-end protection for AI data, models, workloads, and infrastructure across the full AI lifecycle.

- **Cisco AI Defense** provides purpose-built controls to detect and mitigate AI-specific risks such as model misuse, prompt injection, and unsafe agent behaviour, while policy-driven network segmentation enforces zero-trust access across the factory.
- **Splunk** delivers continuous enterprise-wide visibility, correlating signals across network, infrastructure, and AI workloads to enable real-time threat detection, compliance monitoring, and rapid response.

Together, these integrated capabilities create a security-first AI platform that enables customers to confidently scale AI from core to edge while maintaining governance, compliance, and operational resilience.

Cisco & NVIDIA-certified Storage Partner: NetApp who is one of the NVIDIA-certified storage partner provides the storage layer delivering:

- High throughput data pipelines, Parallel reads for training & Persistent storage for checkpoints and models.
- NetApp All-flash systems and scale-out options consistently delivers performance to keep GPUs efficient to meet any AI inference demands.
- Snapshot-based workflows accelerate dataset versioning, experiment repeatability, and fast rollback.
- CSI integration via NetApp Trident enables persistent volumes for AI workloads running on Kubernetes platforms like OpenShift.
- NetApp adds a strong data security foundation, with native capabilities such as immutable Snapshots, ransomware detection, encryption, secure multi-tenancy, and data governance across platforms.

NVIDIA AI Enterprise is a full-stack, cloud-native AI software platform that includes pre-optimized frameworks, tools, and runtime environments for building, deploying, and managing AI and generative AI applications at scale. It provides enterprise-grade support, security, and integration with Kubernetes and platforms like OpenShift, enabling consistent AI operations across on-premises and hybrid cloud environments. Within solutions like Cisco AI PODs or FlexPod AI, it acts as the standardized software layer that accelerates model development, simplifies deployment, and ensures production-ready AI workflows.

The integrated architecture can be viewed secure building block connected to shared enterprise services, with NetApp providing the storage substrate with each AI POD containing the GPU compute and high-

performance fabrics required for AI workloads. The storage ecosystem connects via the AI POD frontend network and provides persistent, governed storage for datasets, feature stores, vector databases, model artifacts, checkpoints, logs, and backups. Security and observability span the entire stack to reduce blind spots and help enforce policy.

Solution overview

This engineering solution offers the following benefits:

- Run AI training workloads on a verified platform with NetApp AI Accelerated data management systems, Cisco AI Accelerated compute and Cisco high speed networking
- Consistent facilitation and efficient implementation and oversight of comprehensive full-stack AI solutions through thorough testing and documentation of end-to-end infrastructure
- Scalable, reliable platform for creating reusable components
- A unified storage platform designed to support containerized applications as well as serve as an object store for extensive AI pipelines

Technology requirements

This section covers the hardware and software which are needed for this solution.

Hardware requirements

Table 1 lists all the hardware used for this solution and a detailed bill of materials is available in the last section of this document.

Table 1) Hardware requirements.

Hardware	Quantity	Comment
Cisco UCS C240 M8	7	Compute Node with GPU/K8s worker nodes, AIDE
Cisco UCS C240 M7	3	K8s Control Node
NetApp AFX 1K	4	Minimum 4 nodes
NetApp NX224 Disk shelf	2	Storage disk shelves (48 * 7.6T drives)
Cisco Nexus 9332D-GX2B (400G)	6	Networking

Software requirements

Table 2 lists all the software stack which was used for this solution.

Table 2) Software requirements

Software	Version	Comment
Red Hat OpenShift	4.20	OpenShift cluster
Kubernetes	v1.33.6	K8s version
NFD Operator	4.20.0	For GPU node discovery
NVIDIA GPU Operator	v26.3.1	Manages GPU resources and drivers in a K8s cluster

NVIDIA Driver Version	580.126.20	NVIDIA GPU driver
NVIDIA CUDA Version	13.0	Software for GPU-accelerated computing applications
NVIDIA Network Operator	v26.1.0	For DOCA driver
NVIDIA-Mellanox ConnectX-7	28.46.1006	Firmware for ConnectX adapter
Cisco Nexus 9332D GXB2	10.4(2)	All 4 switches
Cisco UCS C240 M8	6.0.2a.0.012 6261758	BIOS version
Cisco UCS C240 M8	6.0(2.260044)	CIMC version
NetApp Trident™ software	26.2.0	Fully supported open-source storage orchestrator for containers and K8s distributions, including Red Hat OpenShift

Concepts and Components

This section introduces the foundational building blocks of the FlexPod AI architecture and explains how each layer comes together to enable a scalable, enterprise-ready AI Factory. It covers Cisco servers with AI-accelerated computing platforms based on NVIDIA MGX and HGX designs, high-bandwidth networking for GPU and storage traffic, and NetApp AFX storage with ONTAP-based data services. The section also describes how NetApp AIDE, Red Hat OpenShift, and OpenShift AI integrate into the stack to support end-to-end AI and MLOps workflows. Together, these components form a modular, validated architecture that can scale from right-sized enterprise AI deployments to large, multi-node AI training and inference environments while maintaining consistent operations, performance, and governance.

AI Accelerated Compute

Cisco delivers AI accelerated computing through Cisco UCS rack servers built on NVIDIA MGX and NVIDIA HGX architectures, enabling everything from flexible enterprise inference to extreme-scale AI training. Cisco IMC (CIMC) is primarily used for initial bring-up, node-level configuration, and out-of-band management of C845A and C885A servers, complementing Cisco Intersight's centralized visibility feature and limited server actions at the time of this solution creation.

Cisco UCS with NVIDIA MGX Architecture

Cisco UCS C845A M8 servers are based on NVIDIA MGX leverage a modular reference architecture that allows customers to right-size AI infrastructure and scale over time without redesigning platforms with key characteristics –

- Modular, building-block design for CPUs, GPUs, DPUs, networking, power, and cooling
- Supports PCIe GPUs (e.g., NVIDIA H100 PCIe, H200 NVL, future generations)
- Choice of x86 CPUs (Intel™ Xeon or AMD EPYC)
- PCIe Gen5 expansion for high-speed networking and DPUs
- Managed through Cisco IMC for lifecycle management

Value proposition MGX based Cisco UCS servers provide maximum flexibility and investment protection, enabling customers to start small and scale GPU count and capability as AI workloads mature.

Cisco UCS with NVIDIA HGX Architecture

Cisco UCS servers [C880A M8, C885A M8] built on NVIDIA HGX are designed for the most demanding AI and HPC workloads, delivering maximum GPU density and high-bandwidth interconnects with key characteristics -

- NVIDIA HGX baseboard with 8 SXM GPUs (H100, H200, or newer generations)
- GPUs interconnected via NVLink and NVSwitch for all-to-all, high-bandwidth GPU communication
- Designed for large-scale training, fine-tuning, and high-throughput inference
- High-speed networking options using
 - NVIDIA Spectrum-X Ethernet
 - NVIDIA BlueField DPUs / SuperNICs
- Optimized for scale-out clusters and AI factories
- Integrated with Cisco AI POD / Cisco Secure AI Factory architectures

Value proposition HGX-based Cisco UCS servers deliver maximum performance per node, enabling customers to train and run frontier-scale models efficiently while integrating seamlessly with Cisco networking and AI reference architectures. Table 3 shows the comparison between Cisco UCS MGX and HGX systems.

Table 3) MGX vs. HGX - At a glance

Dimension	Cisco UCS MGX	Cisco UCS HGX
GPU count	Flexible (2–8 typical)	Fixed, dense 8-GPU
Interconnect	PCIe	NVLink + NVSwitch
Primary focus	Flexibility, inference, hybrid AI	Extreme training & Scale
Scalability model	Incremental, modular growth	High-density scale-out
Typical buyer	Enterprise IT, line-of-business AI	AI factories, Research

C240 M8 with 2 GPUs (NVIDIA Blackwell RTX™ PRO 6000 Server Edition, H200 NVL) is the smallest configuration which comes with PCIe optimized NVIDIA certified compute node with 2 * NVIDIA ConnectX-7 NIC, and two CPUs and was chosen for this solution.

Compute Layer (Cisco UCS C240 M8 with NVIDIA RTX PRO 6000)

The 2RU, two-socket Cisco UCS C240 M8 Server with the NVIDIA RTX PRO 6000 GPU delivers flexible I/O and has a higher storage capacity. Built on the latest Intel processors, it is a versatile, general-purpose application and infrastructure server that provides performance and efficiency for a wide range of workloads, including AI, big-data analytics, databases, collaboration, virtualization, and high-performance computing.

You can deploy Cisco UCS C-Series Rack Servers as standalone systems or manage them with Cisco Intersight® to simplify server administration. This helps free your IT staff to focus on mission-critical, value-added projects.

Networking and Specifications

Figure 2) Cisco Nexus 9332D-GX2B



Cisco Nexus 9332D-GX2B (Figure 2) is a compact 1RU fixed-port data center switch designed for high-density 400G leaf/spine fabrics. It delivers 25.6 Tbps of throughput with 32x 400G QSFP-DD ports plus 2x 1/10G SFP+ ports, with flexible breakout options to support 200G/100G/50G/25G/10G speeds. The platform supports Cisco NX-OS or Cisco ACI operation and includes wire-rate MACsec capability on select ports for secure, high-performance connectivity. A total of 6 * Nexus 9332D-GX2B were used for the solution, two were used for storage cluster backend and rest were for E/W and N/S traffic for the C240 M8 servers.

Table 4) Solution Networking

Front End/Back End/Storage Cluster	Quantity
Nexus 9332D-GX2B	6

Cisco Intersight for Manageability

Cisco Intersight serves as the cloud-native operational control plane for FlexPod® AI, delivering consistent, policy-driven lifecycle management across Cisco UCS C240 M8 rack servers. Integrated with the FlexPod architecture, Intersight enables Day-0 to Day-N automation, firmware compliance, and deep GPU, power, and performance telemetry while complementing NetApp ONTAP based data services for high-performance AI pipelines. By unifying compute operations at scale and reducing operational friction across heterogeneous AI infrastructure, Cisco Intersight helps FlexPod customers accelerate AI deployment, improve infrastructure reliability, and maximize the business value of their AI investments. Server configuration and management can be handled using CIMC especially for Cisco MGX and HGX configurations for deployments. The end-user is solely responsible for deploying and operating these tools.

NetApp AFX 1K

AI is no longer an emerging experiment, it has become a core enterprise workload on par with virtualization, databases, and other mission-critical systems. As with these established workloads, AI demands enterprise-grade capabilities including robust data protection, predictable quality of service, operational resilience, and high availability. However, AI also introduces distinct requirements: rapid elasticity, exceptionally high throughput, and sustained performance for dynamic, data-intensive pipelines that support real-time inference and large-scale model training. Agentic AI raises expectations further, requiring secure, seamless integration with enterprise applications and data sources to automate essential processes and deliver measurable business value.

NetApp AFX: disaggregated AI storage

NetApp AFX is disaggregated AI storage built on NetApp ONTAP and will give:

- Decouple performance and capacity: Scale storage resources independently from AFX compute, aligning cost to demand while maintaining predictable latency
- Extend and mobilize data across hybrid cloud with native ONTAP integrations, SnapMirror® replication, and FlexCache® intelligent caching.
- Eliminates the throughput/latency at scale which has been a bottleneck for AI infrastructure as the AI pipelines become dynamic.
- Standardized data services: Use one storage operating environment across tiers to simplify provisioning, protection, and compliance
- Support hybrid strategies: Extend ONTAP data services to cloud targets for backup, disaster recovery, or burst workloads

AFX 1K Storage Nodes

AFX 1K storage controller Built on NetApp engineered hardware, the AFX 1K storage controller delivers extreme performance in a resilient and easily serviced form factor.

NX224 NVMe enclosure

The NX224 NVMe enclosure provides ultrafast, low-latency storage for AI datasets. Customers can add enclosures to scale capacity seamlessly as data volumes grow.

NetApp AIDE for Data preparation

Operationalize AI with NetApp® AI Data Engine. Accelerate AI pipelines by unifying fragmented tools into an ONTAP-integrated solution with real-time metadata, inline vectorization, and semantic search. Get a global, current view of your data estate for faster curation, deep platform integration, and built-in governance for secure, compliant AI operations.

NetApp ONTAP 9

NetApp ONTAP® 9 is the latest generation of storage management software from NetApp that enables businesses to modernize infrastructure and to transition to a cloud-ready data center. With industry-leading data management capabilities, ONTAP enables you to manage and protect your data with a single set of tools regardless of where that data resides. You can also move data freely to wherever you need it: the edge, the core, or the cloud. ONTAP 9 includes numerous features that simplify data management, accelerate and protect your critical data, and future-proof your infrastructure across hybrid cloud architectures. For more information, please refer to the documentation.

NetApp ONTAP S3

NetApp® ONTAP is a unified data platform that supports file (NFS/SMB), block (iSCSI/FC), and S3-compatible object access. In an AI/MLOps FlexPod reference architecture, S3-compatible object storage is commonly used as the system of record for training datasets, feature data, pipeline artifacts, experiment outputs, and model artifacts that need to be accessed by multiple tools and stages of the ML lifecycle. ONTAP S3 provides this object capability on the same ONTAP platform that also delivers persistent storage for Kubernetes workloads (for example, via Astra Trident-backed NFS/iSCSI volumes), enabling a consistent data management and protection approach across notebooks, pipelines, and model-serving workflows.

For FlexPod AI/MLOps deployments, ONTAP S3 is typically enabled by creating an S3 server on an ONTAP storage virtual machine (SVM) and provisioning buckets for specific purposes (for example,

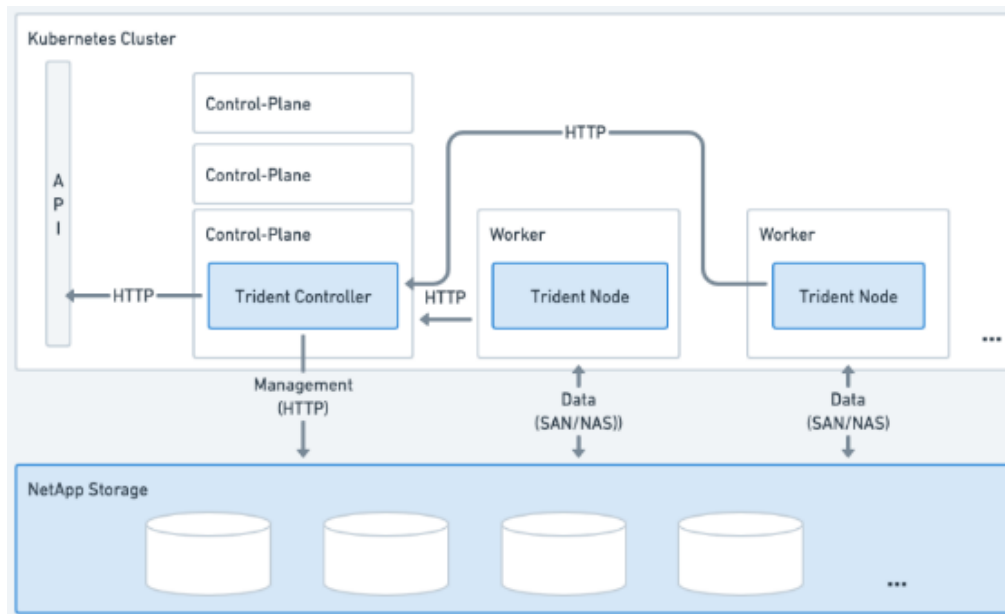
pipeline artifacts, a model repository, or curated datasets). S3 clients—such as Red Hat OpenShift AI data science pipelines and model-serving components—connect through dedicated S3 data network interfaces (data LIFs), often on an isolated storage VLAN with an MTU and bandwidth appropriate for large-object transfers. Use TLS (commonly port 443) with CA-signed certificates and least-privilege S3 users/credentials for secure access; for data protection and recovery requirements, consider S3 SnapMirror for bucket replication as part of the overall backup and disaster recovery strategy.

NetApp Trident & Trident Protect

NetApp Trident™ (Figure 3) is a fully supported, open-source Container Storage Interface (CSI) storage provisioner and orchestrator maintained by NetApp. Trident runs as pods in the Kubernetes cluster and enables developers and platform teams to consume NetApp storage through familiar Kubernetes objects such as Storage Classes, Persistent Volume Claims (PVCs), and Volume Snapshot resources. In a FlexPod reference architecture, Trident is commonly used with NetApp ONTAP to provide dynamic provisioning of persistent volumes (for example, over NFS or iSCSI), as well as capabilities such as volume expansion, snapshots, and cloning to support stateful container and AI/ML workloads.

NetApp Trident Protect adds Kubernetes-native application data management on top of Trident-provisioned storage. Trident Protect uses custom resources (CRs) and the `tridentctl-protect` CLI to define applications and perform orchestrated, application-consistent snapshots and backups that capture both Kubernetes resources (metadata) and persistent volume data. Backup and snapshot metadata and backup data are stored in a vault backed by object storage (for example, ONTAP S3, StorageGRID, or other S3-compatible targets), and data protection workflows can be extended to include replication (SnapMirror) for disaster recovery and mobility across clusters when required.

Figure 3) NetApp Trident deployed on a High Availability Kubernetes Cluster



Red Hat OpenShift

Red Hat® OpenShift Container Platform (OCP) is Red Hat's enterprise Kubernetes application platform for building, deploying, and managing containerized applications across on-premises, hybrid cloud, and multi-cloud environments. OpenShift provides an integrated set of developer and operations services with automated installation, upgrades, and lifecycle management.

Key Features:

- **OpenShift Serverless:** Provides Kubernetes-native serverless capabilities (based on Knative) for event-driven applications, helping teams deploy services that can scale based on demand without managing the underlying infrastructure for each service
- **OpenShift Pipelines:** A Kubernetes-native CI/CD capability based on Tekton that enables teams to define and run build, test, and deployment workflows using custom resources and containerized tasks
- **OpenShift GitOps:** Delivers declarative, Git-driven continuous delivery using Argo CD. It monitors desired state stored in Git repositories and reconciles application and configuration changes to keep clusters consistent
- **OpenShift Virtualization:** Enables running and managing virtual machines alongside containers in the same OpenShift cluster, supporting VM lifecycle operations and use cases such as migration and modernization

OpenShift uses Red Hat Enterprise Linux CoreOS (RHCOS), a container-optimized operating system managed through Operators and designed for automated installation and in-place upgrades. In a FlexPod reference architecture, NetApp ONTAP storage is commonly integrated with OpenShift using the NetApp Astra Trident CSI driver to provide dynamic provisioning of persistent volumes (for example, over NFS or iSCSI). For application-aware data protection and mobility, NetApp Trident protect can be added to help manage and protect Kubernetes applications running on OpenShift.

Red Hat OpenShift AI

Red Hat® OpenShift AI is Red Hat's supported MLOps and AI application platform that runs on Red Hat OpenShift. It provides a consistent, Kubernetes-native environment for data scientists, ML engineers, and platform teams to collaborate on the end-to-end AI/ML lifecycle from experimentation and data preparation through training, evaluation, and production deployment. OpenShift AI organizes work into projects and provides self-service workbenches (for example, Jupyter-based environments) with curated images and accelerator profiles to help standardize development across teams.

OpenShift AI includes data science pipelines (based on KubeFlow Pipelines) to automate repeatable workflows and promote reproducibility, and it provides model serving capabilities built for deploying both predictive models and larger generative AI models. OpenShift AI can leverage GPU-accelerated infrastructure (for example, NVIDIA GPUs) and supports common frameworks and runtimes used across the industry. It also integrates with complementary services to help operationalize inference at scale (for example, NVIDIA NIM microservices where applicable) while relying on OpenShift for enterprise security, multi-tenancy, and lifecycle management.

In FlexPod AI/MLOps designs, OpenShift AI commonly uses S3-compatible object storage for pipeline artifact storage and, in many cases, for curated datasets and model artifacts that must be shared across workbenches, pipeline runs, and model-serving components. NetApp ONTAP S3 can provide this S3 endpoint on the same ONTAP platform used for enterprise file and block storage, enabling consistent data services and protection. In disaggregated architectures such as NetApp AFX where compute and storage scale independently to provide a scalable, high-throughput object store to support MLOps workflows (for example, buckets dedicated to pipeline artifacts and a model repository) while maintaining a unified operational model across the AI data pipeline.

NVIDIA AI Factory Eco-system

NVIDIA's AI Factory concept describes an end-to-end, production approach to AI combining accelerated GPU compute, a high-performance network fabric, an enterprise data layer, and operational software to turn data into deployed models and AI services. It emphasizes a repeatable pipeline for ingest/curate, train or fine-tune, and delivers governed inference through standard APIs and platform operations.

FlexPod AI aligns well to this blueprint by combining Cisco networking with NetApp storage and NVIDIA-accelerated servers either in the form factor of Cisco HGX C885A or Cisco MGX C845A. HGX nodes typically anchor large-model training and high-throughput fine-tuning, while MGX nodes provide right-sized capacity for scalable inference, RAG / embedding pipelines, and supporting pre/post-processing. The NetApp data management layer provides persistent storage for datasets, checkpoints, and enterprise content, and FlexPod validated designs help standardize deployment and expansion.

Cisco Silicon One based platforms help deliver the scale-out, low-latency Ethernet fabric needed for GPU-cluster E/W traffic, while keeping operations consistent across the FlexPod environment. On top, NVIDIA AI Enterprise (NVAIE) provides the software layer through NVIDIA NIM and delivers GPU-optimized inference as microservices that can scale on the MGX tier, and models such as NVIDIA Nemotron (where available/licensed) can be served via NIM and adapted with RAG and/or HGX-based fine-tuning.

Infrastructure and Application Monitoring with Splunk Observability

Splunk Observability is Splunk's cloud-based platform for full-stack observability, helping organizations monitor, troubleshoot, and optimize the performance of applications, infrastructure, and user experiences in real time. It combines metrics, traces, logs, and events into a single platform so teams can quickly detect issues and improve reliability.

Explore the benefits of Splunk Infrastructure Monitoring, Splunk APM, Splunk RUM, Splunk Synthetic Monitoring and Splunk Log Observer Connect, all in one interface (one user seat):

- Easily detect potential issues and identify root causes using high-resolution, streaming analytics
- Instantly see how every user experiences your app and how each code change impacts site performance
- Keep total control over your data with no vendor lock-in and Open Telemetry — instrument once, observe everywhere

Solution Design

A common AI factory pattern uses two complementary network fabrics: a backend (east–west) fabric optimized for GPU-to-GPU communication in distributed workloads, and a frontend (north–south) fabric for connectivity to storage, management & shared services. This separation supports predictable performance for AI training while keeping integration with the broader data center clean and controllable.

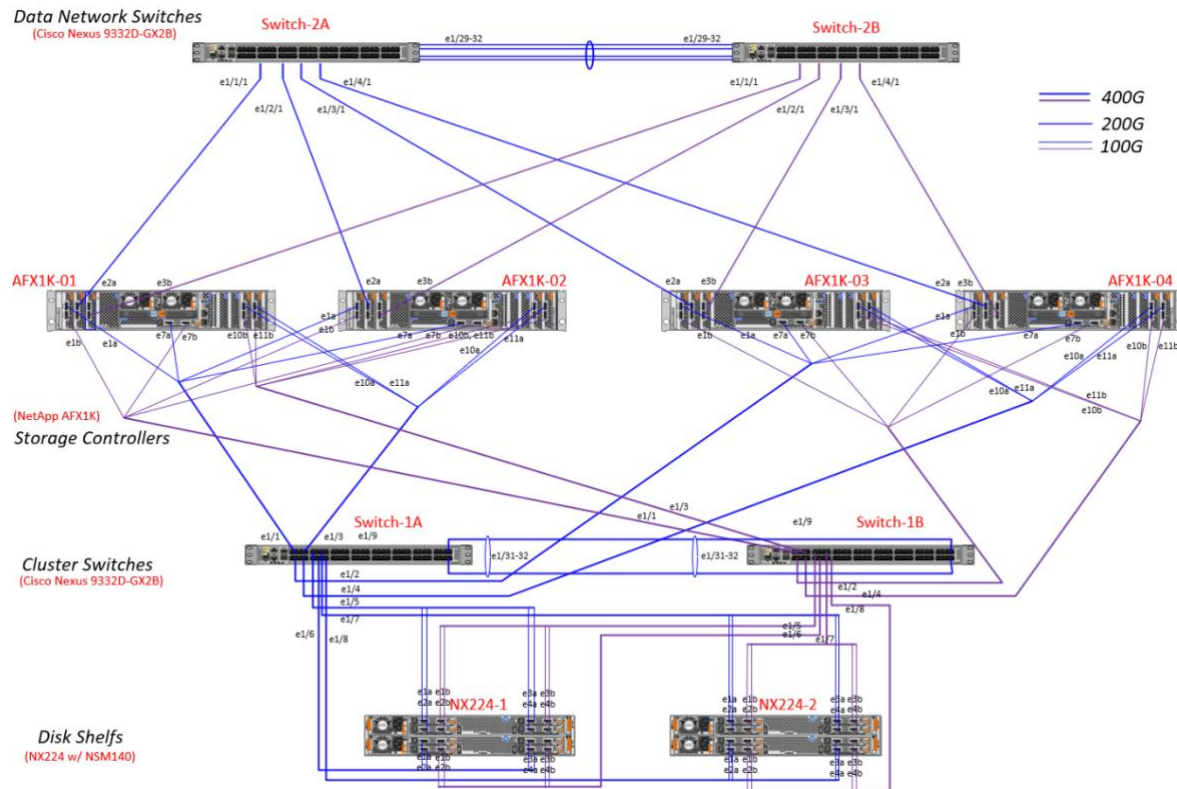
The solution topology (Figure 4) consists of four NetApp AFX 1K controllers, two fully populated NX224 disk shelves with NSM140 and 7.6TB NVMe/SSD drives, two Cisco Nexus 9332D-GX2B switches for cluster and North/South storage traffic and, two additional Cisco Nexus 9332D-GX2B switches for the host side East/West GPU traffic. Each controller has 2-port 40G/100G (X50130A) cards installed in slots 1, 7, 10 & 11. Cards 1 & 7 are used for HA and Cluster traffic respectively while cards 10 & 11 are used for storage traffic. Port-a on each card connects to the cluster switch A (Switch-1A) and port-b connects to the other cluster switch (Switch1-B) using 400G to 100G breakout cables.

The cluster switches are pre-configured using NetApp provided RCF configuration file. Each of the first 30 ports are configured as 400G to 100G breakout ports with identical switch port configuration, so the controllers and disk shelves can be connected to any of the breakout ports. Port 31 & 32 are configured as local ISL port-channel and each port connects to the identical port on the partner switch.

Each controller has 2-port 100G/200G/400G (X50131B) cards installed in slots 2 and 3. For link level and card level redundancy, port 2a connects to data network switch (Switch-2A) while Port 3b connects to the

other data network switch (Switch-2B) using 400G to 200G breakout cables. These ports carry the NFSoRDMA traffic.

Figure 4) Solution Topology of AFX 1K



The NX224 disk shelf comes with two NSM140 controllers, each has four 2-port 100G (X60135B) adapters. Port-a on each adapter connects to cluster switch A (Switch-1A) and port-b connects to cluster switch B (Switch-1B).

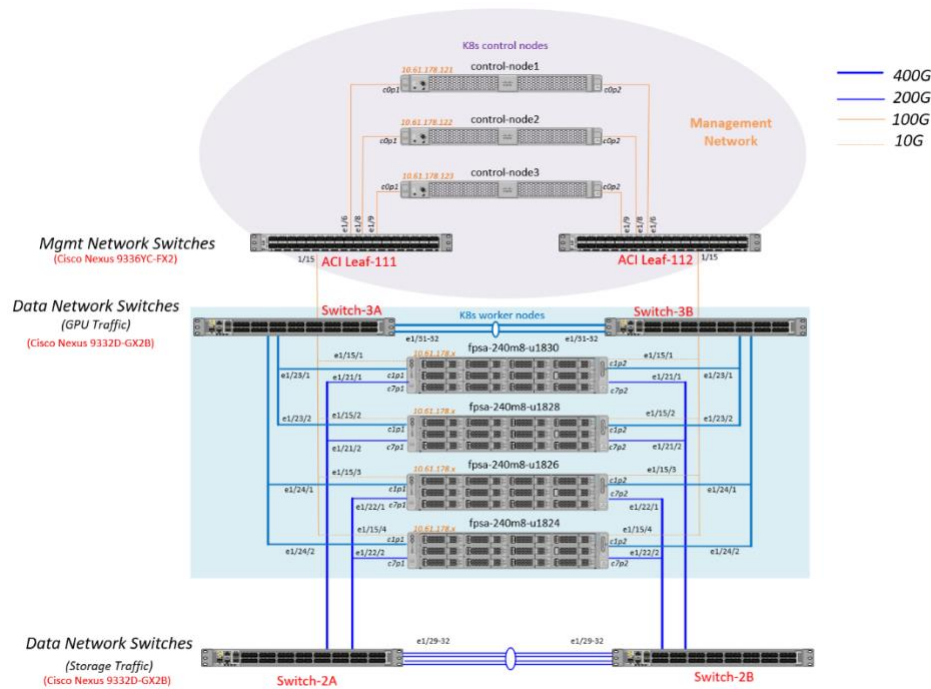
Each of the devices is connected to a management network infrastructure as shown in (Figure 4)

Cisco UCS C240M8 with NVIDIA RTX PRO 6000 in NVIDIA MGX

This section describes how Cisco's NVIDIA accelerated server platforms can be used within NetApp FlexPod architecture to demonstrate end-to-end scalability across Compute, Network, and ONTAP-based storage. The goal is to show a repeatable scale-out building-block approach that keeps performance predictable as GPU and CPU demand grows.

FlexPod provides a validated, repeatable architecture that combines Cisco compute, Cisco networking, and NetApp storage running ONTAP. By introducing MGX nodes as the compute layer, FlexPod can showcase modern accelerated workloads while preserving the operational model that customers expect from a converged, reference-architected solution and a MGX server, C240 M8 was chosen for this solution and FlexPod compute can scale by adding additional MGX nodes as demand grows.

Figure 5) Cisco UCS Compute Topology



Each Cisco UCS C240M8 server is equipped with two dual port ConnectX-7 NIC (4x 100/200GbE ports per server), installed in slot1 & slot7. The ports on card 1 are connected individually to the backend switches (Switch-3A & Switch-3B) and carry the GPU traffic. The ports on card 7 are connected to the front-end switches (Switch-2A & Switch-2B) via vPC and carry the storage traffic. Each server is equipped with NVIDIA RTX PRO 6000 in slot2 & slot5. Each server also has a 4-port 10/25Gbps card (Cisco UCS VIC 15427) installed in the mLOM slot for in-band management connectivity.

Networking

Cisco switching (typically leaf/spine designs for scale) provides predictable east-west bandwidth for distributed training/inference and north-south connectivity to users, services as well as connecting to storage backend. As GPU density increases, the network must scale in two ways: (1) more ports as nodes are added, and (2) more aggregate throughput to avoid bottlenecks for distributed jobs and storage access. FlexPod can demonstrate this by scaling the leaf layer with additional switch capacity and expanding the spine as needed, preserving consistent latency and predictable oversubscription ratios as the environment grows. A starter configuration consists of 3 pairs of Nexus 9332D-GX2B switches.

- 2 * Nexus 9332D-GX2B 400GbE switches for the GPU traffic E/W
- 2 * Nexus 9332D-GX2B 400GbE switches for the storage traffic N/S
- 2 * Nexus 9332D-GX2B 400GbE switches for the AFX cluster traffic

Additionally, you would need switching infrastructure for out of band (OOB) management connectivity as well.

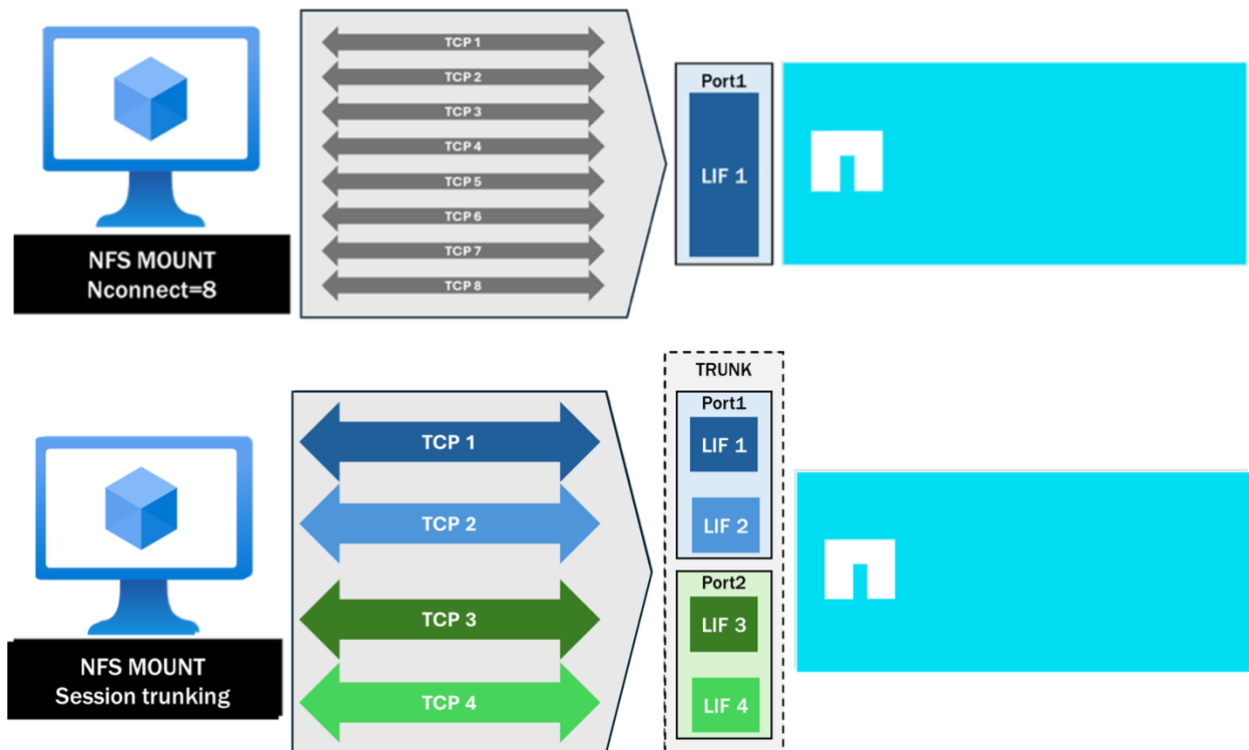
Storage

ONTAP® AFX solution scales to meet accelerated workload needs by expanding capacity and performance independently while maintaining consistent data management. Within a FlexPod design, storage can be sized for high-throughput dataset reads, sustained checkpoint writes, and multi-tenant access patterns. ONTAP also provides enterprise capabilities such as space-efficient cloning for rapid dataset and environment replication, and snapshot-based protection, so the platform scales not only in raw performance, but also in operational efficiency.

With this approach, FlexPod can position MGX and HGX as the scalable compute engine, Cisco networking as the elastic, high-bandwidth fabric, and ONTAP as the shared storage foundation that grows with the data. The result is a solution story that scales from pilot to production for AI/ML pipelines, large language model tuning/inference, analytics, and other accelerated workloads while retaining consistent operations and enterprise-grade data management.

Session Trunking vs. nconnect

Figure 6) NFS - nconnect vs. session trunking



With nconnect, you can increase throughput by using multiple parallel TCP connections for a single NFS mount; the client opens “n=8” (Figure 6) connections to the same ONTAP LIF (same IP/port). With NFSv4.1 session trunking, a single NFSv4.1 client session can use multiple network paths/LIFs (more IPs and ports per mount), while still operating under a shared state/locking context. In general, nconnect can be a better fit for metadata-heavy workloads because it can help mitigate any RPC slot limitation, while session trunking is often a better fit for maximizing aggregate throughput and resiliency by spreading traffic across multiple LIFs on the storage node.

Data Services – with NetApp AIDE

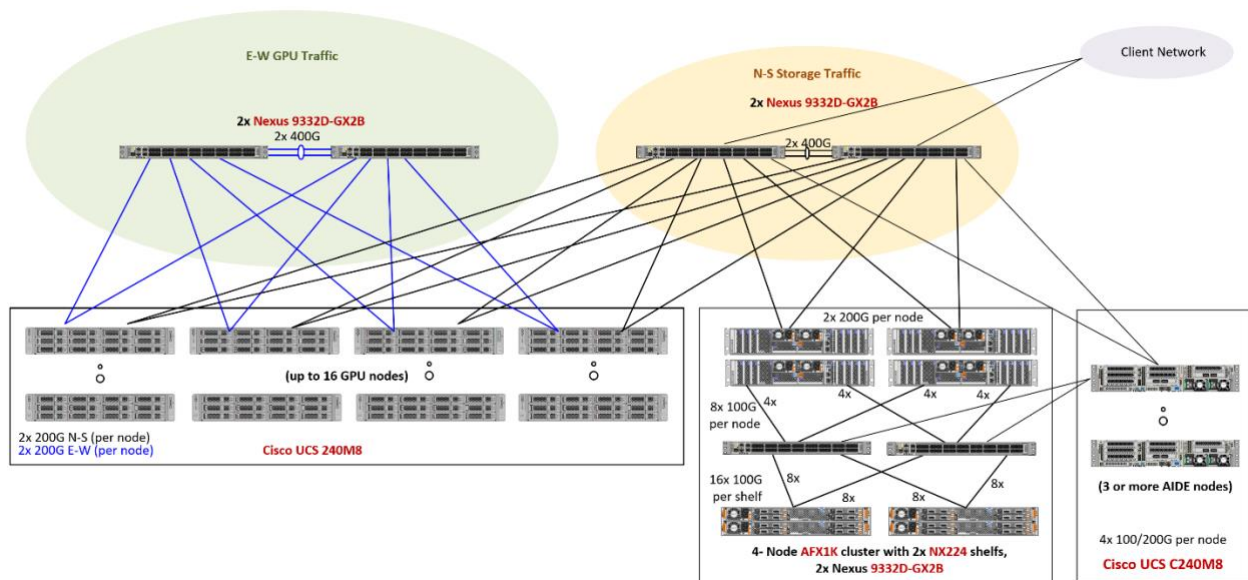
NetApp AI Data Engine (AIDE)

AIDE is a storage-integrated AI data service designed to make enterprise data “AI-ready” for GenAI use cases like RAG, agentic AI, and AI factories by reducing data sprawl, keeping data fresh, and enforcing governance/security close to where the data live. AIDE is part of NetApp’s AI portfolio and is built on ONTAP to bring AI data services “to the data”, rather than copying data into many separate tools and in practical terms:

- Builds a global, structured view of your data estate (especially unstructured file data) via a large-scale metadata catalog/engine, so teams can discover and understand what data exists
- Curates data into trusted collections so you can pick the “right” documents/objects for an AI use case rather than dumping everything into a pipeline
- Applies policy-driven “guardrails” classification, monitoring, and controls to help protect sensitive data and support compliance
- Enables real-time, continuous vectorization and semantic retrieval so curated data can be published as retrieval endpoints for RAG systems, while staying current through automated change detection and sync

The AIDE cluster consists of 3x Cisco UCS C240M8 servers as shown in Figure 7

Figure 7) Topology with AIDE Data Services



Solution Deployment

This section describes the end-to-end deployment of the Cisco AI POD built on FlexPod® Converged Infrastructure, covering infrastructure configuration, platform installation, and integration of AI-optimized components. It provides validated, step-by-step guidance to help customers deploy a secure, scalable AI environment with consistent Day-0 to Day-N operations.

Network configuration

This section goes over the network switch configuration. A pair of Nexus 9332D-GX2B switches are configured as the backend cluster switches connecting the storage controllers and disk shelves. Two additional pairs of Nexus 9332D-GX2B switches are configured as front-end and backend switches, carrying the storage traffic to the servers (N-S) and GPU traffic (E-W) respectively.

Nexus 9332D-GX2B backend cluster switches

The backend cluster switches are pre-configured as part of the AFX bundle and the configuration is available as an RCF file from NetApp Support site. Ports 1 to Port 30 are configured as a 4-port 100G breakout and ports 31 & 32 are configured as Intra-Cluster ISL port-channel at line rate. The traffic is classified into HA, Storage, RDMA and default classes with appropriate Cos (Class of Service) or DSCP (Differentiated Services Code Point) values and the policy is applied at the ingress port. Network QoS and queuing policies are also created and applied at the system level. Priority Flow Control (PFC) is enabled on all interfaces as well.

The RCF configuration is shown below.

```
# Installation Notes:
#
# IMPORTANT NOTES
# In certain conditions, NX9332D-GX2B may not be able to auto-negotiate port
# speed correctly, and port speed has to be manually set in config mode, e.g.
# int e1/1
# speed 100000
# int e1/3
# speed 400000
#

banner motd #
*****
* NetApp Reference Configuration File (RCF)
*
* Switch   : NX9332D-GX2B
* Filename : NX9332D-GX2B-RCF-v10.0-Shared.txt
* Date     : 07-22-2025
* Version  : v10.0
* Port Usage:
* Ports   1-30: 100GbE Intra-Cluster/HA/Storage Ports, int e1/{1-30}/1-4
* Ports 31-32: Intra-Cluster ISL Ports, int e1/31-32
*
* IMPORTANT NOTES
* Interface port-channel999 is reserved to identify the version of this file.
*****
#

interface breakout module 1 port 1-30 map 100g-4x

feature lACP
feature lldp
feature ssh
feature sftp-server
feature scp-server

vlan 17,18,30,40
exit

cdp enable
cdp advertise v1
cdp timer 5
system default switchport
no system default switchport shutdown
```

```

snmp-server community cshml! group network-operator
errdisable recovery interval 30
port-channel load-balance src-dst ip-l4port-vlan
ip domain-lookup
logging console 1

hardware access-list tcam region ing-racl 1024
hardware access-list tcam region egr-racl 1024
hardware access-list tcam region ing-l2-qos 1536

#***** Keyless SSH for SHM *****
ssh key ecdsa 521
#***** PFC/QoS/ECN *****

class-map type qos match-all HA
match cos 5
exit

class-map type qos match-all STORAGE
match cos 3
exit

class-map type qos match-any RDMA
match dscp 16
match cos 2

policy-map type qos HA_STORAGE_POLICY
class type qos HA
    set qos-group 5
class type qos STORAGE
    set qos-group 3
class type qos RDMA
    set qos-group 2
class type qos class-default
    set qos-group 0

policy-map type queuing EGRESS_POLICY
class type queuing c-out-8q-q7
bandwidth remaining percent 0
class type queuing c-out-8q-q6
bandwidth remaining percent 0
class type queuing c-out-8q-q5
bandwidth remaining percent 0
random-detect threshold burst-optimized ecn
class type queuing c-out-8q-q4
bandwidth remaining percent 0
class type queuing c-out-8q-q3
random-detect minimum-threshold 150 kbytes maximum-threshold 1500 kbytes drop-probability 100
weight 0 ecn
bandwidth remaining percent 0
class type queuing c-out-8q-q2
bandwidth remaining percent 0
random-detect threshold burst-optimized ecn
class type queuing c-out-8q-q1
bandwidth remaining percent 0
class type queuing c-out-8q-q-default
bandwidth remaining percent 0
random-detect threshold burst-optimized ecn

policy-map type network-qos NETQOS_POLICY
class type network-qos c-8q-nq5
pause pfc-cos 5
mtu 9216
class type network-qos c-8q-nq3
pause pfc-cos 3
mtu 9216
class type network-qos c-8q-nq2

```

```

mtu 9216
class type network-qos c-8q-nq-default
mtu 9216
exit
exit

system qos
service-policy type network-qos NETQOS_POLICY
service-policy type queuing output EGRESS_POLICY

copp profile strict

#***** Port Profiles *****

port-profile type ethernet CLUSTER_STORAGE_INTERFACE
description 4x100GbE Breakout Cluster/HA/Controller/Shelf Storage Nodes
spanning-tree port type edge trunk
spanning-tree bpduguard enable
switchport mode trunk
switchport trunk allowed vlan 1,17,18,30,40
no shutdown
mtu 9216
state enabled
exit

#***** Interfaces *****

interface e1/1/1-4,e1/2/1-4,e1/3/1-4,e1/4/1-4,e1/5/1-4,e1/6/1-4,e1/7/1-4,e1/8/1-4,e1/9/1-
4,e1/10/1-4,e1/11/1-4,e1/12/1-4,e1/13/1-4,e1/14/1-4,e1/15/1-4,e1/16/1-4,e1/17/1-4,e1/18/1-
4,e1/19/1-4,e1/20/1-4,e1/21/1-4,e1/22/1-4,e1/23/1-4,e1/24/1-4,e1/25/1-4,e1/26/1-4,e1/27/1-
4,e1/28/1-4,e1/29/1-4,e1/30/1-4
description CL-1,ST-1 Cluster/HA, Storage 100G Port
inherit port-profile CLUSTER_STORAGE_INTERFACE
priority-flow-control mode on
priority-flow-control watch-dog-interval on
service-policy type qos input HA_STORAGE_POLICY no-stats

#***** Intra-Cluster ISL ports *****

interface port-channel1
description LOCAL_ISL Switch
switchport mode trunk
priority-flow-control mode on
priority-flow-control watch-dog-interval on
switchport trunk allowed vlan 1,40
mtu 9216
service-policy type qos input HA_STORAGE_POLICY
no shutdown

interface Ethernet1/31
description LOCAL_ISL Switch 400G Port
priority-flow-control mode on
priority-flow-control watch-dog-interval on
switchport mode trunk
switchport trunk allowed vlan 1,40
mtu 9216
channel-group 1 mode active
no shutdown

interface Ethernet1/32
description LOCAL_ISL Switch 400G Port
priority-flow-control mode on
priority-flow-control watch-dog-interval on
switchport mode trunk
switchport trunk allowed vlan 1,40
mtu 9216
channel-group 1 mode active
no shutdown

```

```

interface port-channel999
 shutdown
 description RCF NX9332D-GX2B v10.0 1-CLUSTER 1-STORAGE AFX
end

```

Nexus 9332D-GX2B data network switches

Two pairs of Nexus 9332D-GX2B switches are used for data network, one pair for the front-end North-South storage traffic and another pair for the backend East-West GPU traffic.

The front-end switch pair is configured as virtual Port Channel (vPC) pair with 2 or more vPC member links. VLAN is defined for storage traffic, and the peer-link is configured to allow this VLAN. Each C240M8 server connects to this switch pair over vPC. Appropriate QoS classification, marking and queuing policies are defined and applied at the system level and interface level as shown in the sample configuration below. The backend switches could use similar configuration with the exception that the servers are connected to leaf switches via non-vPC links.

```

#***** PFC/QoS/ECN *****

# Storage Traffic (NFSorDMA) - CoS 3 (Matches NetApp PFC Queue 3)
class-map type qos match-all STORAGE
 match cos 3
exit

# RDMA GPU-to-GPU Traffic - DSCP 26 or CoS 2
class-map type qos match-any RDMA
 match dscp 26
 match cos 2
exit

# Ingress QoS Policy - Map traffic to QoS Groups
policy-map type qos RDMA_STORAGE_POLICY

 # NFSorDMA Storage -> QoS Group 3 (PFC Lossless)
 class type qos STORAGE
   set qos-group 3

 # GPU-to-GPU RDMA -> QoS Group 2 (Lossless)
 class type qos RDMA
   set qos-group 2

 # Everything else -> QoS Group 0 (Best Effort)
 class type qos class-default
   set qos-group 0

exit

# Egress Queuing Policy with ECN
policy-map type queuing EGRESS_POLICY

 class type queuing c-out-8q-q7
   bandwidth remaining percent 0

 class type queuing c-out-8q-q6
   bandwidth remaining percent 0

 class type queuing c-out-8q-q4
   bandwidth remaining percent 0

 # Queue 3 - NFSorDMA Storage (Lossless + ECN)
 # Matches NetApp PFC Queue 3
 class type queuing c-out-8q-q3
   bandwidth remaining percent 0
   random-detect minimum-threshold 150 kbytes maximum-threshold 1500 kbytes drop-probability 100
 weight 0 ecn

```

```

# Queue 2 - GPU-to-GPU RDMA (Lossless + ECN)
class type queuing c-out-8q-q2
    bandwidth remaining percent 0
    random-detect threshold burst-optimized ecn

class type queuing c-out-8q-q1
    bandwidth remaining percent 0

# Queue 0 - Best Effort (Default)
class type queuing c-out-8q-q-default
    bandwidth remaining percent 0
    random-detect threshold burst-optimized ecn

exit

# Network QoS - PFC and MTU settings
policy-map type network-qos NETQOS_POLICY

    # Queue 3 - NFSoRDMA (PFC Enabled - Lossless Q3)
    class type network-qos c-8q-nq3
        pause pfc-cos 3
        mtu 9216

    # Queue 2 - GPU-to-GPU RDMA
    class type network-qos c-8q-nq2
        mtu 9216

    # Default Queue
    class type network-qos c-8q-nq-default
        mtu 9216

    exit
exit

# Apply QoS Policies System Wide
system qos
    service-policy type network-qos NETQOS_POLICY
    service-policy type queuing output EGRESS_POLICY

copp profile strict

# Interface -NFSoRDMA

interface Ethernet1/X
    description NetApp-NFSoRDMA-Storage
    switchport mode trunk
    switchport trunk allowed vlan <NFSO_RDMA_VLAN>
    priority-flow-control mode on
    service-policy type qos input RDMA_STORAGE_POLICY
    mtu 9216
    spanning-tree port type edge trunk
    no shutdown
exit

# GPU-GPU RDMA

interface Ethernet1/X
    description GPU-Server-CX7-RDMA
    switchport access vlan <GPU_RDMA_VLAN>
    priority-flow-control mode on
    service-policy type qos input RDMA_STORAGE_POLICY
    mtu 9216
    spanning-tree port type edge
    no shutdown
exit

```

```

# vPC peer link
# Physical interfaces for vPC Peer Link

interface Ethernet1/X
  description vPC-Peer-Link-Member
  switchport mode trunk
  switchport trunk allowed vlan all
  priority-flow-control mode on
  mtu 9216
  channel-group 1 mode active
  no shutdown
exit

# Port Channel for vPC Peer Link

interface port-channel1
  description vPC-Peer-Link
  switchport mode trunk
  switchport trunk allowed vlan all
  spanning-tree port type network
  priority-flow-control mode on
  mtu 9216
  vpc peer-link
  no shutdown
exit

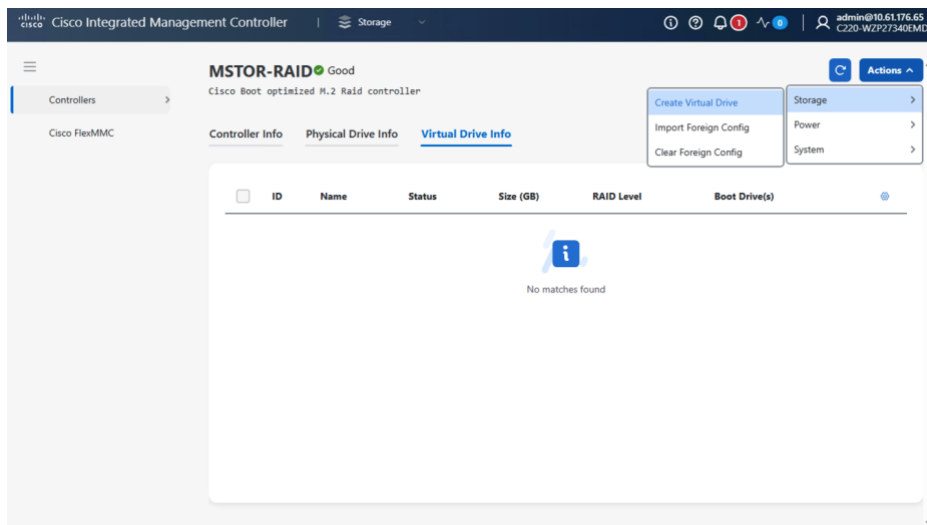
```

Cisco UCS C-Series server configuration

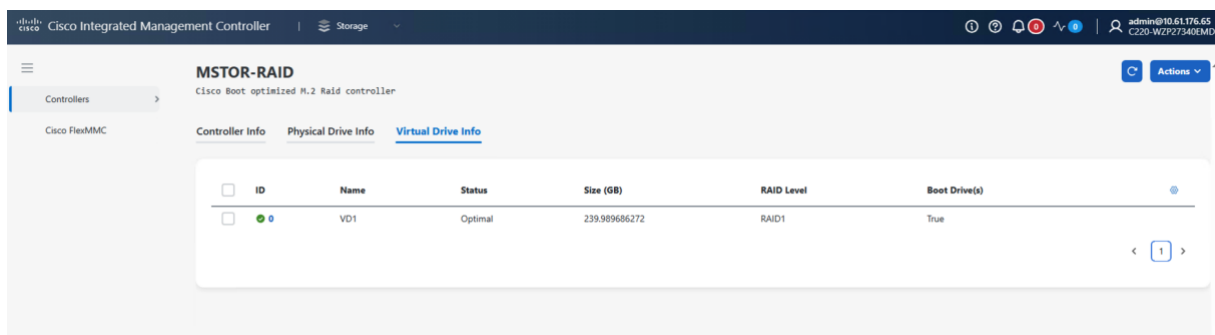
This section discusses about the Cisco C series server's initial setup, before an operating system can be installed. In this section, we will talk about the CIMC and Local Boot configuration. The major steps are listed below.

- Configure local boot from M.2 boot drives
- Configure Virtual Interface Cards
- Configure server boot order
- BIOS settings for UCS M8 servers

1. Go to Storage and Under Actions > Storage, click Create Virtual Drive.



1. Select Controller to create a RAID volume- MSTOR-RAID
2. Create Virtual Drive from Unused Physical Drives
3. Select RAID types as RAID1
4. Enter Virtual Drive name and click Next
5. Verify the Summary tab and click Create.
6. Virtual Drive created successfully.



2. To configure the Virtual Interface Cards networking refer to [Configuring Network-Related Settings](#)
3. For configuring server boot order refer to [Managing the Server](#).
4. To configure BIOS settings for UCS M8 servers, refer to [BIOS recommendation for enterprise workloads](#).

NetApp AFX installation and configuration

Installation

1. The NetApp AFX storage system is a disaggregated hardware and software solution focused on the unique needs of high-performance NAS and S3 customers. To understand the installation and setup workflow for AFX 1K storage systems, refer to the [AFX Documentation](#).
2. To install and configure the software for a Cisco 9332D-GX2B switch and to install or upgrade the Reference Configuration File (RCF), follow these [steps](#).
3. After your AFX hardware has been installed, you can complete the ONTAP cluster setups by following [steps](#).

Cluster Setup

After your AFX hardware has been installed, you can complete the ONTAP cluster setup. This process involves related configuration phases that you should perform sequentially. As part of this, you can optionally add the data compute nodes (DCNs) running the AI Data Engine (AIDE), enabling the AFX storage cluster and DCN cluster to operate as a single system that's managed through the same ONTAP control plane. To set up the AFX system refer [perform initial cluster setup](#).

AFX Configuration

Once the AFX storage systems has been installed and cluster set up is completed, follow the below section to configure the storage.

1. Open an SSH connection to either the cluster IP or the host name.
2. Log into the admin user with the password you provided earlier.
3. Delete any Default-N automatically created broadcast domains.

```
network port broadcast-domain delete -broadcast-domain <Default-N> -ip-space Default
network port broadcast-domain show
```

Note: Delete the Default broadcast domains with Network ports (Default-1, Default-2, and so on). This does not include Cluster ports and management ports.

4. Disable Flow Control on the Data Ports which will be part of interface groups.

```
network port modify -node fpsa-afx1k-0* -port e2b,e3a -flowcontrol-admin none

8 entries were modified.
```

5. To enable the Cisco Discovery Protocol (CDP) on the NetApp storage controllers, run the following command.

```
node run -node * options cdpd.enable on
```

6. Enable LLDP on all ports of all nodes in the cluster.

```
node run -node * options lldp.enable on
```

7. To configure time synchronization on the cluster, run the following command.

```
timezone -timezone <timezone>
```

8. To configure physical ports for RoCE, run the following commands.

```
network port modify -node * -port e2a -flowcontrol-admin pfc -pfc-queues-admin 3
network port modify -node * -port e3b -flowcontrol-admin pfc -pfc-queues-admin 3
```

9. Create broadcast domain for normal NFS, NFSoRDMA and IB-Mgmt with the required MTU.

```
network port broadcast-domain create -broadcast-domain IB-Mgmt -mtu 1500 -ip-space Default
network port broadcast-domain create -broadcast-domain OCP-NFSoRDMA -mtu 9000 -ip-space Default
network port broadcast-domain create -broadcast-domain Infra-NFS -mtu 9000 -ip-space Default
```

10. To create the LACP interface groups for the data interfaces, run the following commands.

```
network port ifgrp create -node fpsa-afx1k-01 -ifgrp a0a -distr-func port -mode multimode_lacp

network port ifgrp add-port -node fpsa-afx1k-01 -ifgrp a0a -port e2b
network port ifgrp add-port -node fpsa-afx1k-01 -ifgrp a0a -port e3a

network port ifgrp create -node fpsa-afx1k-02 -ifgrp a0a -distr-func port -mode multimode_lacp
```

```

network port ifgrp add-port -node fpsa-afx1k-02 -ifgrp a0a -port e2b
network port ifgrp add-port -node fpsa-afx1k-02 -ifgrp a0a -port e3a

network port ifgrp create -node fpsa-afx1k-03 -ifgrp a0a -distr-func port -mode multimode_lacp

network port ifgrp add-port -node fpsa-afx1k-03 -ifgrp a0a -port e2b
network port ifgrp add-port -node fpsa-afx1k-03 -ifgrp a0a -port e3a

network port ifgrp create -node fpsa-afx1k-04 -ifgrp a0a -distr-func port -mode multimode_lacp

network port ifgrp add-port -node fpsa-afx1k-04 -ifgrp a0a -port e2b
network port ifgrp add-port -node fpsa-afx1k-04 -ifgrp a0a -port e3a

```

11. Create NFS VLAN and Add RoCE ports to Broadcast domain.

```

network port vlan create -node fpsa-afx1k-01 -vlan-name e2a-2287
network port vlan create -node fpsa-afx1k-02 -vlan-name e2a-2287
network port vlan create -node fpsa-afx1k-03 -vlan-name e2a-2287
network port vlan create -node fpsa-afx1k-04 -vlan-name e2a-2287

network port vlan create -node fpsa-afx1k-01 -vlan-name e3b-2287
network port vlan create -node fpsa-afx1k-02 -vlan-name e3b-2287
network port vlan create -node fpsa-afx1k-03 -vlan-name e3b-2287
network port vlan create -node fpsa-afx1k-04 -vlan-name e3b-2287

broadcast-domain add-ports -broadcast-domain OCP-NFSorDMA -ports fpsa-afx1k-01:e2a-2287,fpsa-afx1k-01:e3b-2287,fpsa-afx1k-02:e2a-2287,fpsa-afx1k-02:e3b-2287,fpsa-afx1k-03:e2a-2287,fpsa-afx1k-03:e3b-2287,fpsa-afx1k-04:e2a-2287,fpsa-afx1k-04:e3b-2287

```

12. Create NFS VLAN and Add interface groups ports to Broadcast domain.

```

network port vlan create -node fpsa-afx1k-01 -vlan-name a0a-2287
network port vlan create -node fpsa-afx1k-02 -vlan-name a0a-2287
network port vlan create -node fpsa-afx1k-03 -vlan-name a0a-2287
network port vlan create -node fpsa-afx1k-04 -vlan-name a0a-2287

broadcast-domain add-ports -broadcast-domain Infra-NFS -ports fpsa-afx1k-01:a0a-2287,fpsa-afx1k-02:a0a-2287,fpsa-afx1k-03:a0a-2287,fpsa-afx1k-04:a0a-2287

```

13. Create IB-Mgmt VLAN ports

```

network port vlan create -node fpsa-afx1k-01 -vlan-name a0a-2285
network port vlan create -node fpsa-afx1k-02 -vlan-name a0a-2285
network port vlan create -node fpsa-afx1k-03 -vlan-name a0a-2285
network port vlan create -node fpsa-afx1k-04 -vlan-name a0a-2285

network port broadcast-domain add-ports -ipSpace Default -broadcast-domain IB-MGMT -ports fpsa-afx1k-01:a0a-2285,fpsa-afx1k-02:a0a-2285,fpsa-afx1k-03:a0a-2285,fpsa-afx1k-04:a0a-2285

```

14. Create the SVM (Storage Virtual Machine). Run the vserver create command.

```

vserver create -vserver ocp-svm
vserver create -vserver svm1

```

15. Add the required data protocols to the SVM and remove the unused data protocols from the SVM.

```

vserver add-protocols -vserver ocp-svm -protocols nfs
vserver add-protocols -vserver svm1 -protocols nfs,s3

vserver remove-protocols -vserver ocp-svm -protocols cifs,s3
vserver remove-protocols -vserver svm1 -protocols cifs

```

16. Add the storage availability zone to the ocp-svm and enable and run the NFS protocol in the SVM:

```

vserver modify -vserver ocp-svm -aggr-list storage_availability_zone_0

set -privilege advanced

```

```
vserver nfs create -vserver ocp-svm -udp disabled -v3 enabled -v4.1-pnfs enabled -v4.1 enabled -  
v4.1-trunking enabled -tcp-max-xfer-size 1048576 -rdma enabled  
  
set -privilege admin
```

17. Add the storage availability zone to the svm1 and enable and run the NFS protocol in the SVM.

```
vserver modify -vserver svm1 -aggr-list storage_availability_zone_0  
  
set -privilege advanced  
  
vserver nfs create -vserver svm1 -udp disabled -v3 enabled -v4.1-pnfs enabled -v4.1 enabled -  
v4.1-trunking enabled -tcp-max-xfer-size 262144  
  
set -privilege admin
```

18. Create new rule for the SVMs NFS subnet in the default export policy and assign the policy to the SVM's root volume.

```
vserver export-policy rule create -vserver ocp-svm -policyname default -ruleindex 1 -protocol nfs  
-clientmatch 172.22.87.0/24 -rorule sys -rwrule sys -superuser sys -allow-suid true  
  
volume modify -vserver ocp-svm -volume ocp_svm_root -policy default  
  
vserver export-policy rule create -vserver svm1 -policyname default -ruleindex 1 -protocol nfs -  
clientmatch 172.22.87.0/24 -rorule sys -rwrule sys -superuser sys -allow-suid true  
  
volume modify -vserver svm1 -volume svm1_root -policy default
```

19. Create and enable audit log for both SVMs.

```
volume create -vserver svm1 -volume svm1_audit_log -state online -size 100GB -policy default -  
junction-path /audit_log -space-guarantee none -percent-snapshot-space 0  
  
vserver audit create -vserver svm1 -destination /audit_log  
vserver audit enable -vserver svm1  
  
volume create -vserver ocp-svm -volume ocp_svm_audit_log -state online -size 100GB -policy  
default -junction-path /audit_log -space-guarantee none -percent-snapshot-space 0  
  
vserver audit create -vserver ocp-svm -destination /audit_log  
vserver audit enable -vserver ocp-svm
```

20. Run the following commands to create RDMA NFS Logical Interfaces (LIFs).

```
network interface create -vserver ocp-svm -lif afx01-nfs-lif01 -service-policy default-data-files  
-home-node fpsa-afx1k-01 -home-port e2a-2287 -address 172.22.87.51 -netmask 255.255.255.0 -  
status-admin up -failover-policy broadcast-domain-wide -auto-revert true -rdma-protocols roce  
  
network interface create -vserver ocp-svm -lif afx01-nfs-lif02 -service-policy default-data-files  
-home-node fpsa-afx1k-01 -home-port e3b-2287 -address 172.22.87.52 -netmask 255.255.255.0 -  
status-admin up -failover-policy broadcast-domain-wide -auto-revert true -rdma-protocols roce  
  
network interface create -vserver ocp-svm -lif afx02-nfs-lif03 -service-policy default-data-files  
-home-node fpsa-afx1k-02 -home-port e2a-2287 -address 172.22.87.53 -netmask 255.255.255.0 -  
status-admin up -failover-policy broadcast-domain-wide -auto-revert true -rdma-protocols roce  
  
network interface create -vserver ocp-svm -lif afx02-nfs-lif04 -service-policy default-data-files  
-home-node fpsa-afx1k-02 -home-port e3b-2287 -address 172.22.87.54 -netmask 255.255.255.0 -  
status-admin up -failover-policy broadcast-domain-wide -auto-revert true -rdma-protocols roce  
  
network interface create -vserver ocp-svm -lif afx03-nfs-lif05 -service-policy default-data-files  
-home-node fpsa-afx1k-03 -home-port e2a-2287 -address 172.22.87.55 -netmask 255.255.255.0 -  
status-admin up -failover-policy broadcast-domain-wide -auto-revert true -rdma-protocols roce
```

```
network interface create -vserver ocp-svm -lif afx03-nfs-lif06 -service-policy default-data-files
-home-node fpsa-afx1k-03 -home-port e3b-2287 -address 172.22.87.56 -netmask 255.255.255.0 -
status-admin up -failover-policy broadcast-domain-wide -auto-revert true -rdma-protocols roce
```

```
network interface create -vserver ocp-svm -lif afx04-nfs-lif07 -service-policy default-data-files
-home-node fpsa-afx1k-04 -home-port e2a-2287 -address 172.22.87.57 -netmask 255.255.255.0 -
status-admin up -failover-policy broadcast-domain-wide -auto-revert true -rdma-protocols roce
```

```
network interface create -vserver ocp-svm -lif afx04-nfs-lif08 -service-policy default-data-files
-home-node fpsa-afx1k-04 -home-port e3b-2287 -address 172.22.87.58 -netmask 255.255.255.0 -
status-admin up -failover-policy broadcast-domain-wide -auto-revert true -rdma-protocols roce
```

21. Run the following commands to create NFS Logical Interfaces (LIFs).

```
network interface create -vserver svml -lif afx01-nfs-lif01-a0a -service-policy default-data-
files -home-node fpsa-afx1k-01 -home-port a0a-2287 -address 172.22.87.11 -netmask 255.255.255.0 -
status-admin up -failover-policy broadcast-domain-wide -auto-revert true
```

```
network interface create -vserver svml -lif afx02-nfs-lif02-a0a -service-policy default-data-
files -home-node fpsa-afx1k-02 -home-port a0a-2287 -address 172.22.87.12 -netmask 255.255.255.0 -
status-admin up -failover-policy broadcast-domain-wide -auto-revert true
```

```
network interface create -vserver svml -lif afx03-nfs-lif03-a0a -service-policy default-data-
files -home-node fpsa-afx1k-03 -home-port a0a-2287 -address 172.22.87.13 -netmask 255.255.255.0 -
status-admin up -failover-policy broadcast-domain-wide -auto-revert true
```

```
network interface create -vserver svml -lif afx04-nfs-lif04-a0a -service-policy default-data-
files -home-node fpsa-afx1k-04 -home-port a0a-2287 -address 172.22.87.14 -netmask 255.255.255.0 -
status-admin up -failover-policy broadcast-domain-wide -auto-revert true
```

22. Create SVM management LIF.

```
network interface create -vserver svml -lif svml-mgmt-lif -service-policy default-management -
home-node fpsa-afx1k-01 -home-port a0a-2285 -address 172.22.85.11 -netmask 255.255.255.0 -status-
admin up -failover-policy broadcast-domain-wide -auto-revert true
```

```
network interface create -vserver ocp-svm -lif ocp-svm-mgmt-lif -service-policy default-
management -home-node fpsa-afx1k-01 -home-port a0a-2285 -address 172.22.85.51 -netmask
255.255.255.0 -status-admin up -failover-policy broadcast-domain-wide -auto-revert true
```

23. Create a default route that enables the SVM management interface to reach the outside world.

```
network route create -vserver svml -destination 0.0.0.0/0 -gateway 172.22.85.1
```

```
network route create -vserver ocp-svm -destination 0.0.0.0/0 -gateway 172.22.85.1
```

24. Set a password for both SVMs vsadmin user and unlock the user.

```
fpsa-afx1k::> security login password -username vsadmin -vserver svml
```

Enter a new password:

Enter it again:

```
fpsa-afx1k::> security login unlock -username vsadmin -vserver svml
```

```
fpsa-afx1k::> security login password -username vsadmin -vserver ocp-svm
```

Enter a new password:

Enter it again:

```
fpsa-afx1k::> security login unlock -username vsadmin -vserver ocp-svm
```

25. Add DNS servers to the SVMs.

```
dns create -vserver svml -domains fpmc.sa -name-servers 10.61.176.251,10.61.176.252
```

```
dns create -vserver ocp-svm -domains fpmc.sa -name-servers 10.61.176.251,10.61.176.252
```

OpenShift installation

OpenShift 4.20 is deployed on the Cisco UCS infrastructure as M.2 booted bare metal servers. The Cisco UCS C220 M7 and Cisco C240 M8 servers are equipped with M.2 drives. Three control-plane (UCS C220 M7) nodes and four (UCS C240 M8) worker nodes are deployed in the validation environment and additional worker nodes can easily be added to increase the scalability of the solution. This document will guide you through the process of using the Assisted Installer to deploy OpenShift 4.20.

Prerequisites

The FlexPod for OpenShift solution utilizes the Assisted Installer for deploying OpenShift. As a result, when provisioning and managing the FlexPod infrastructure, you must supply all supporting cluster infrastructure components and resources. These include the installer VM or host, networking, storage, and the individual cluster machines required for deployment.

The following supporting cluster resources are required for installation:

- Control plane and compute machines that form the OpenShift cluster
- Cluster networking, including required VLANs, IP addressing, and connectivity
- Storage for both cluster infrastructure components and application workloads
- Installer VM or host

Network Requirements

The following infrastructure services need to be deployed to support the OpenShift cluster, during the validation of this solution we have provided VMs on your hypervisor of choice to run the required services. You can use existing DNS and DHCP services available in the data center.

There are various infrastructure services prerequisites for deploying OpenShift 4.20. These prerequisites are as follows:

- DNS and DHCP services – these services were configured on Microsoft Windows Server VMs
- NTP Distribution was done with the Cisco Nexus switches
- Specific DNS entries for deploying OpenShift – added to the DNS server
 - Base Domain: fpmc.sa
 - OpenShift Cluster Name: flexpod-afx
- A Linux VM/Physical server for generating installer ISO and cluster management – a Rocky Linux 9 / RHEL 9 VM with appropriate packages

The DNS domain name for the OpenShift cluster should be the cluster name followed by the base domain, for example, flexpod-afx.fpmc.sa. Table 5 lists the information for fully qualified domain names used during validation.

Table 5) DNS and IP Address

Usage	Hostname	IP Address
API	api.flexpod-afx.fpmc.sa	10.61.178.130
Ingress LB (apps)	*.apps.flexpod-afx.fpmc.sa	10.61.178.131
control-01	control-01.flexpod-afx.fpmc.sa	10.61.178.121
control-02	control-02.flexpod-afx.fpmc.sa	10.61.178.122
control-03	control-03.flexpod-afx.fpmc.sa	10.61.178.123

compute-01	compute-01.flexpod-afx.fpmc.sa	10.61.178.124
compute-02	compute-02.flexpod-afx.fpmc.sa	10.61.178.125
compute-03	compute-03.flexpod-afx.fpmc.sa	10.61.178.126
compute-04	compute-04.flexpod-afx.fpmc.sa	10.61.178.127

DHCP

For the bare metal network, a network administrator must reserve several IP addresses, including:

- One IP address for the API endpoint
- One IP address for the wildcard Ingress endpoint
- One IP address for each control-plane node (DHCP server assigns to the node)
- One IP address for each worker node (DHCP server assigns to the node)

Note: Obtain the MAC addresses of the bare metal Interfaces (Table 6) from the UCS Server CIMC for each node to be used in the DHCP configuration to assign reserved IP addresses (reservations) to the nodes.

Table 6) Host information

Hostname	IP Address	BareMetal MAC Address (eno5)
control-01	10.61.178.121	98:A2:C0:66:87:DC
control-02	10.61.178.122	3C:26:E4:F2:23:CC
control-03	10.61.178.123	98:A2:C0:66:77:B0
compute-01	10.61.178.124	44:1A:5C:7C:36:80
compute-02	10.61.178.125	FC:72:88:B2:8A:7C
compute-03	10.61.178.126	98:A2:C0:66:87:DC
compute-04	10.61.178.127	b4:ca:dd:25:af:94

Deployment Steps

1. SSH to Linux workstation to manage the OpenShift cluster using CLI.
2. Create a directory with the name AFX and change directory to AFX.
3. To enable SSH access to OpenShift cluster, go to the newly created directory and run the following commands to generate a SSH key pair to enable SSH access to the OpenShift cluster nodes.

```
cd <new directory for cluster>

ssh-keygen -t rsa -N '' -f <path>/<file_name>

eval "$(ssh-agent -s)"

ssh-add <path>/<file_name>
```

4. To Install OpenShift cluster using Assisted Installer from Red Hat Hybrid Cloud Console, open web browser and go to console.redhat.com and login with your Red Hat account.
5. Go to Red Hat OpenShift > OpenShift tile and click OpenShift.
6. From the left navigation pane, select Cluster List and click Create cluster.
7. Go to the Datacenter tab, then click Create cluster under Assisted Installer option.

- In Install OpenShift with the Assisted Installer workflow, for Cluster Details, specify the Cluster name and Base domain. Select the OpenShift version from the drop-down list. Select Static IP, bridges, and bonds under Host's network configuration and click Next.

Cluster details

☒ Technology Preview ☐ I'm installing on a disconnected/air-gapped/secured environment

Cluster name *
flexpod-afx

Base domain *
fpmc.sa
Enter the name of your domain [domainname] or [domainname.com]. This cannot be changed after cluster installed. All DNS records must include the cluster name and be subdomains of the base you enter. The full cluster address will be: flexpod-afx.fpmc.sa

OpenShift version *
OpenShift 4.20.15
[Learn more about OpenShift releases](#)

CPU architecture
x86_64

☐ Edit pull secret ⓘ

Integrate with external partner platforms
No platform integration

Number of control plane nodes ⓘ
3 (highly available cluster)

Hosts' network configuration
☐ DHCP only ☒ Static IP, bridges, and bonds

- Next to Configure via: select YAML view. In the YAML text box, click Start from scratch. Build and copy a YAML file specifying the LACP bond interface into the text box. Also add MAC address of both interfaces. Repeat the setup for remaining control and compute hosts.

Host 1

YAML

```

1 Interfaces:
2   - name: bond0
3     description: Bond with ports eno5 and eno6
4     type: bond
5     state: up
6     ipv4:
7       dhcp: true
8       enabled: true
9     ipv6:
10      enabled: false
11    link-aggregation:
12      mode: 802.3ad
13    options:
14      ethtool: '100'
15    port:
16      - eno5
17      - eno6

```

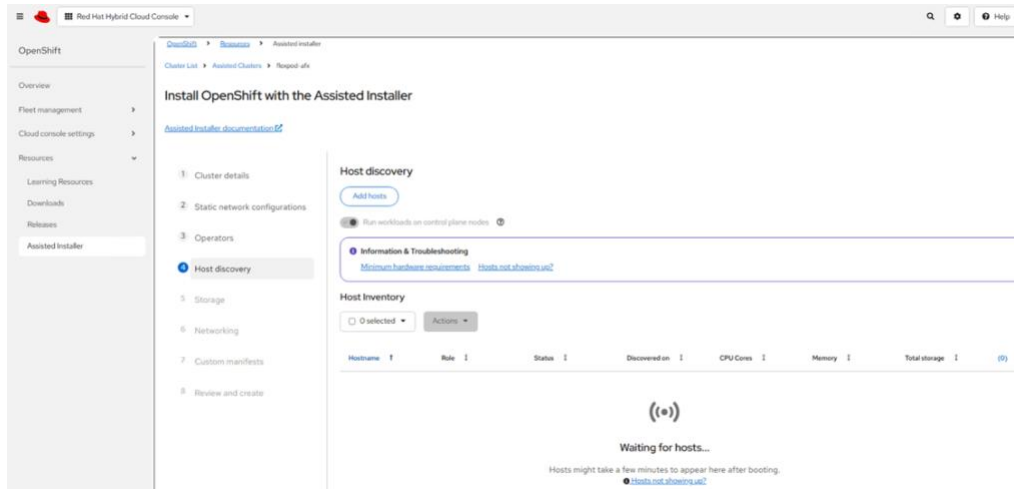
MAC to interface name mapping ⓘ

MAC address *	Interface name *
98:A2:C0:66:87:DC	eno5
98:A2:C0:66:87:DD	eno6

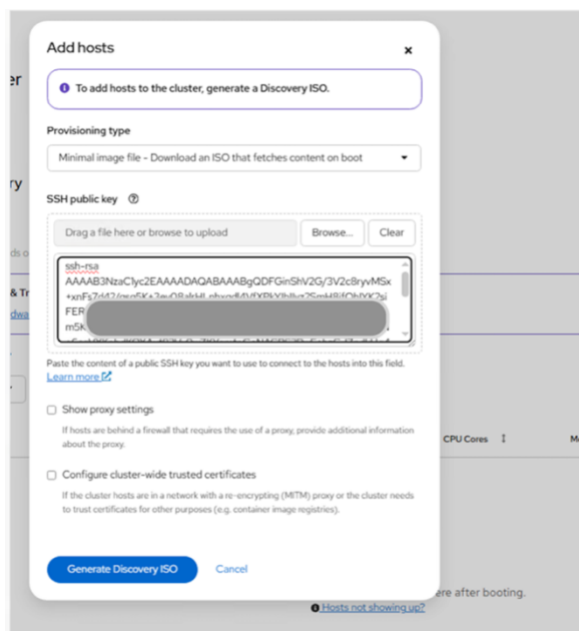
[Add another MAC to interface name mapping](#)

[Add another host configuration](#) ☒ Copy the YAML content

- Under Host discovery, click Add hosts



- For Provisioning type, select Minimal image file. Browse to and select the SSH public key file used in the original cluster installation. Click Generate Discovery ISO.



- Click Download Discovery ISO to download the file on your machine.
- Log into the CIMC of all Cisco UCS servers and launch KVM. In the KVM, from the Virtual Media drop-down list to select Map image. Click Drop file here or click to upload. Navigate to and select the Discovery ISO in your Downloads folder and click Open. Click Upload.
- Boot the server and press F6 and select KVM-mapped-DVD to boot from the Discovery ISO.
- Once the server has booted from the Discovery ISO, return to the Red Hat Hybrid Cloud Console. The nodes should appear in a few minutes. Wait for the Status to become Ready then click Next.

1 Cluster details

2 Static network configurations

3 Operators

4 Host discovery

5 Storage

6 Networking

7 Custom manifests

8 Review and create

Host discovery

Add hosts

Run workloads on control plane nodes

Information & Troubleshooting

[Minimum hardware requirements](#)
[Hosts not showing up?](#)

Host Inventory

0 selected

Actions

	Hostname	Role	Status	Discovered on	CPU Cores	Memory	Total stora...	
	control-01.flexpod-afx.flexc.sa	Worker	Ready	4/14/2026, 4:16:13 AM	128	768.00 GiB	962.48 GiB	
	control-02.flexpod-afx.flexc.sa	Worker	Ready	4/14/2026, 4:10:46 AM	128	768.00 GiB	962.48 GiB	
	control-03.flexpod-afx.flexc.sa	Worker	Ready	4/14/2026, 4:05:30 AM	128	768.00 GiB	962.48 GiB	
	control-04.flexpod-afx.flexc.sa	Worker	Ready	4/14/2026, 4:20:32 AM	128	768.00 GiB	962.48 GiB	
	control-01.flexpod-afx.flexc.sa	Control plane node	Ready	4/14/2026, 3:35:42 AM	128	128.00 GiB	242.27 GiB	
	control-02.flexpod-afx.flexc.sa	Control plane node	Ready	4/14/2026, 3:43:18 AM	128	128.00 GiB	239.99 GiB	
	control-03.flexpod-afx.flexc.sa	Control plane node	Ready	4/14/2026, 4:03:14 AM	128	128.00 GiB	242.27 GiB	

Note: Click the arrow to the left of the server(s) Hostname. Make sure the correct M.2 boot disk is selected.

16. Enter Cluster Network address, VIP and Ingress IP details and click Next.

Cluster List > Assisted Clusters > flexpod-afx

1 Cluster details

2 Static network configurations

3 Operators

4 Host discovery

5 Storage

6 Networking

7 Custom manifests

8 Review and create

Install OpenShift with the Assisted Installer

[Assisted Installer documentation](#)

Networking

Network Management

Cluster-Managed Networking

User-Managed Networking

Networking stack type

IPv4

Dual-stack

Network type

Open Virtual Networking (OVN)

Machine network *

10.61.178.0/24 (10.61.178.0 - 10.61.178.255)

API IP *

10.61.178.130

Ingress IP *

10.61.178.131

Use advanced networking

Configure advanced networking properties (e.g. CIDR ranges).

17. Review and click Create Cluster. The installation will start after this.

OpenShift

Overview

Fleet management

Cloud console settings

Resources

Learning Resources

Downloads

Releases

Assisted installer

Installation progress

Started on 4/14/2026, 7:57:44 AM

Installing 12%

Control Planes Installing 3 control plane nodes

Workers Installing 4 workers

Installation Pending

Abort installation

Download kubeconfig

View cluster events

Download installation logs

Download and save your kubeconfig file in a safe place. This file will be automatically deleted from Assisted Installer's service in 20 days.

Host inventory (7)

Hostname	Role	Status	Discovered on	CPU Cores	Memory	Total Storage
compute-01.flexpod-afx.fpmc.sa	Worker	Starting installation	4/14/2026, 4:16:13 AM	128	768.00 GiB	962.48 GiB
compute-02.flexpod-afx.fpmc.sa	Worker	Starting installation	4/14/2026, 4:10:46 AM	128	768.00 GiB	962.48 GiB
compute-03.flexpod-afx.fpmc.sa	Worker	Starting installation	4/14/2026, 4:05:30 AM	128	768.00 GiB	962.48 GiB
compute-04.flexpod-afx.fpmc.sa	Worker	Starting installation	4/14/2026, 4:20:32 AM	128	768.00 GiB	962.48 GiB
control-01.flexpod-afx.fpmc.sa	Control plane node	Starting installation	4/14/2026, 9:01:44 AM	128	128.00 GiB	242.27 GiB
control-02.flexpod-afx.fpmc.sa	Control plane node	Starting installation	4/15/2026, 12:48:57 AM	128	128.00 GiB	239.99 GiB
control-03.flexpod-afx.fpmc.sa	Control plane node (bootstrap)	Starting installation	4/14/2026, 4:03:14 AM	128	128.00 GiB	242.27 GiB

18. Click the arrow to the left of any node and verify the bond status which we created earlier.

control-01.flexpod-afx.fpmc.sa

Control plane node (bootstrap)

Installing 3/11

4/14/2026, 9:01:44 AM

128

128.00 GiB

239.99 GiB

Host details

UID c4e03bca-0efa-f27-6a38-b182f9f52e19

Manufacturer Cisco Systems Inc

Product UCSC-C220-M75

Serial number WZP2734OEM8

Memory capacity 128.00 GiB

CPU model name Intel(R) Xeon(R) Platinum 8462Y+

CPU cores and clock speed 128 cores (hyper-threaded) at 4,100 MHz

CPU architecture x86_64

Hardware type Bare metal

BMC address 10.61.176.118

Boot mode uefi

Disk

Name	Role	Limitations	Drive type	Size	Serial	Model	WWN
sdd (bootable)	Installation disk		HDD	239.99 GiB	415a1a722e470010	CISCO_VD	

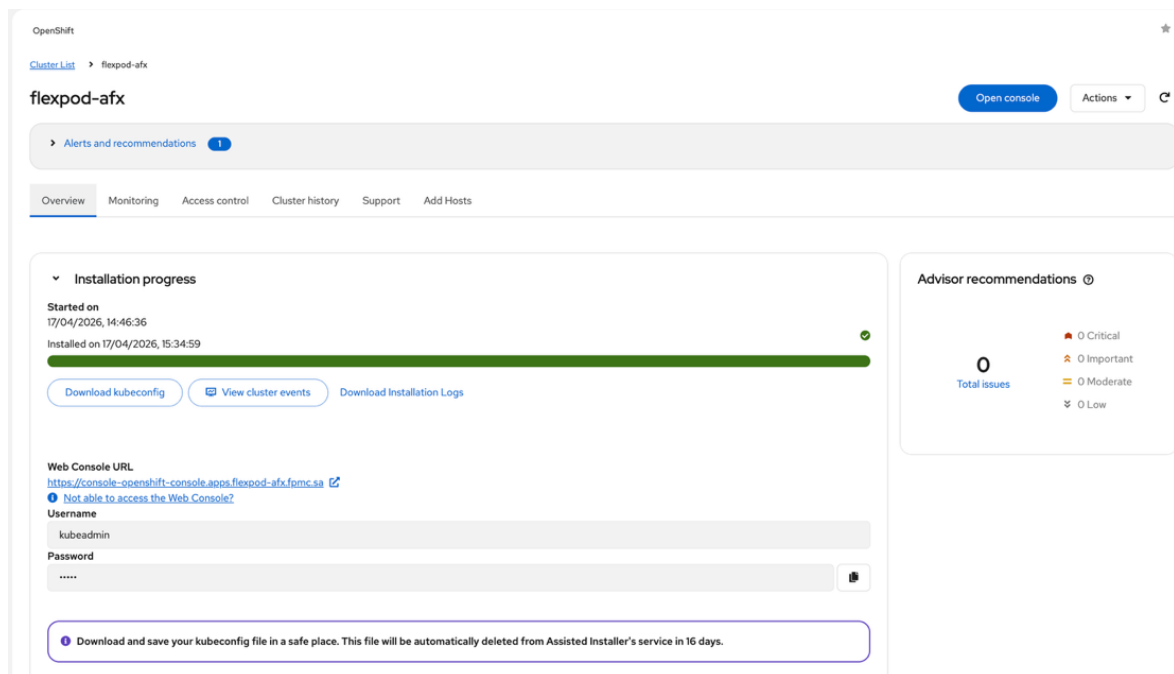
NICs

Name	MAC address	IPv4 address	IPv6 address	Speed
bond0	98:a2:c0:66:87:dc	10.61.178.12/24		200000 Mbps
eno5	98:a2:c0:66:87:dc			100000 Mbps
eno6	98:a2:c0:66:87:dd			100000 Mbps

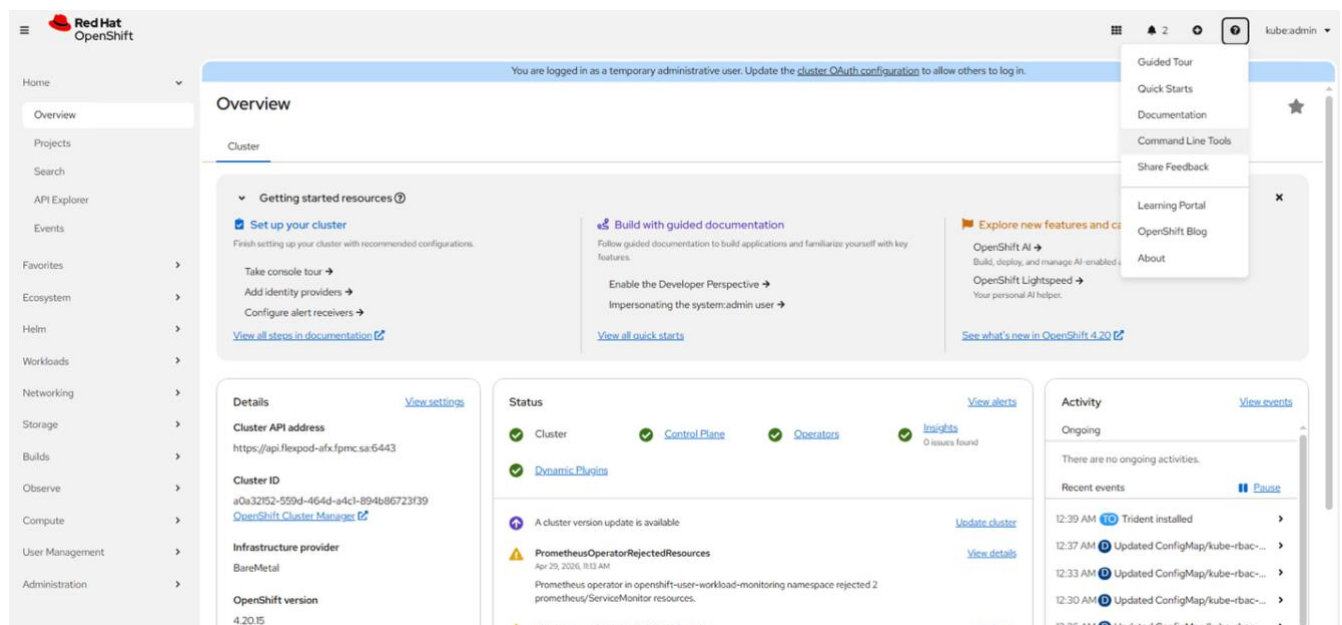
19. Wait for approx. 45 minutes and the OpenShift cluster installation will complete.

Hostname	Role	Status	Discovered on	CPU Cores	Memory
compute-01.flexpod-afx.fpmc.sa	Worker	Installed	4/14/2026, 9:08:32 AM	128	768.00 GiB
compute-02.flexpod-afx.fpmc.sa	Worker	Installed	4/14/2026, 9:33:09 AM	128	768.00 GiB
compute-03.flexpod-afx.fpmc.sa	Worker	Installed	4/14/2026, 9:35:24 AM	128	768.00 GiB
compute-04.flexpod-afx.fpmc.sa	Worker	Installed	4/14/2026, 9:37:28 AM	128	768.00 GiB
control-01.flexpod-afx.fpmc.sa	Control plane node (bootstrap)	Installed	4/14/2026, 9:01:44 AM	128	128.00 GiB
control-02.flexpod-afx.fpmc.sa	Control plane node	Installed	4/15/2026, 12:48:57 AM	128	128.00 GiB
control-03.flexpod-afx.fpmc.sa	Control plane node	Installed	4/14/2026, 9:23:55 AM	128	128.00 GiB

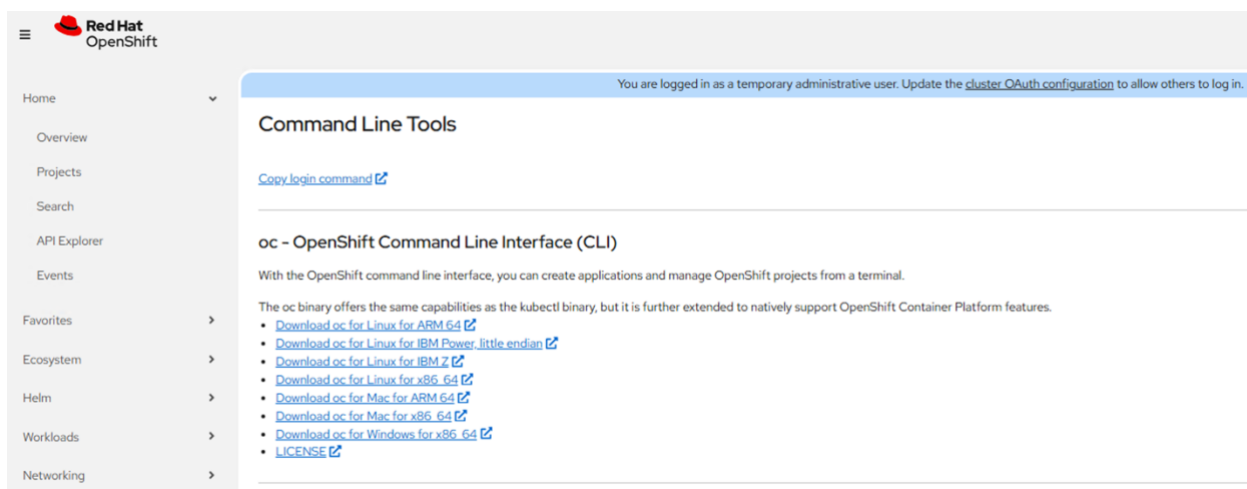
20. Download and save the kubeconfig file. Use the web console URL to open OpenShift web console and enter the credential to login.



21. From the top menu bar, click the ? icon and select Command Line Tools from the drop-down list.



22. Click Download oc for Linux for x86_64.



23. Copy the file to the Linux workstation, move and save into the sub directory.

```
mkdir oc-client
cd oc-client
mv ../oc.tar .
tar xvf oc.tar
oc
ls
oc oc.tar
sudo mv oc /usr/local/bin[sudo]
password for admin:
```

Post installation setup

1. Enable dynamic resource allocation for kubelet. Create below .yaml files

```
cat worker-kubeletconfig.yaml

apiVersion: machineconfiguration.openshift.io/v1
kind: KubeletConfig
metadata:
  name: dynamic-node
spec:
  autoSizingReserved: true
  machineConfigPoolSelector:
    matchLabels:
      pools.operator.machineconfiguration.openshift.io/worker: ""

cat control-plane-kubeletconfig.yaml

apiVersion: machineconfiguration.openshift.io/v1
kind: KubeletConfig
metadata:
  name: dynamic-node-control-plane
spec:
  autoSizingReserved: true
  machineConfigPoolSelector:
    matchLabels:
```

```
pools.operator.machineconfiguration.openshift.io/master: ""
```

2. Apply to reserve cpu and memory resources for system components (kubelet, kube-proxy, etc.) on the nodes.

```
#Apply
oc apply -f worker-kubeletconfig.yaml
oc apply -f control-plane-kubeletconfig.yaml
```

3. To setup NTP on both control and compute nodes, follow the following steps.

```
cd afx-ocp
mkdir machine-configs
cd machine-configs
curl https://mirror.openshift.com/pub/openshift-v4/clients/butane/latest/butane --output butane
chmod +x butane
```

4. Create the following files in the machine config directory.

```
cat 99-control-plane-chrony-conf-override.bu
variant: openshift
version: 4.20.0
metadata:
  name: 99-control-plane-chrony-conf-override
  labels:
    machineconfiguration.openshift.io/role: master
storage:
  files:
    - path: /etc/chrony.conf
      mode: 0644
      overwrite: true
      contents:
        inline: |
          driftfile /var/lib/chrony/drift
          makestep 1.0 3
          rtcsync
          logdir /var/log/chrony
          server 10.61.176.251 iburst
          server 10.61.176.252 iburst

cat 99-worker-chrony-conf-override.bu
variant: openshift
version: 4.17.0
metadata:
  name: 99-worker-chrony-conf-override
  labels:
    machineconfiguration.openshift.io/role: worker
storage:
  files:
```

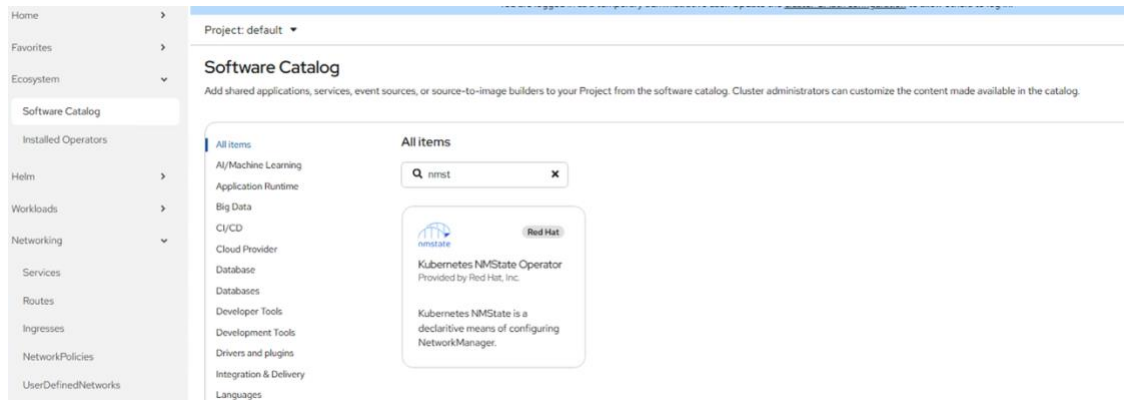
```
- path: /etc/chrony.conf
  mode: 0644
  overwrite: true
  contents:
    inline: |
      driftfile /var/lib/chrony/drift
      makestep 1.0 3
      rtcsync
      logdir /var/log/chrony
      server 10.61.176.251 iburst
      server 10.61.176.252 iburst
```

5. Create .yaml file from the above butane file and apply both files.

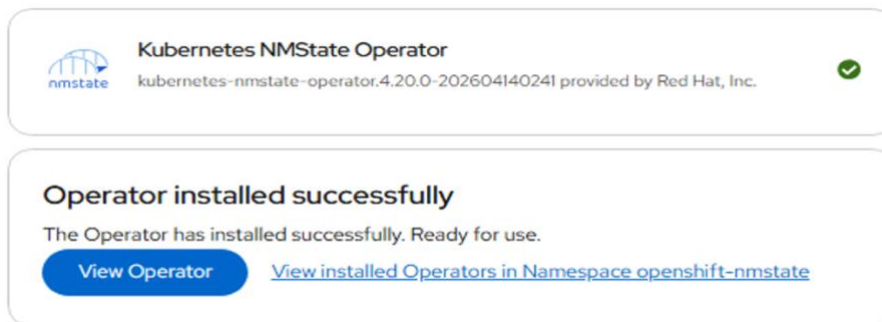
```
./butane 99-control-plane-chrony-conf-override.bu -o ./99-control-plane-chrony-conf-override.yaml
./butane 99-worker-chrony-conf-override.bu -o ./99-worker-chrony-conf-override.yaml

oc create -f 99-control-plane-chrony-conf-override.yaml
oc create -f 99-worker-chrony-conf-override.yaml
```

6. To configure the storage and GPU networking interfaces on the workers nodes, we will use Kubernetes NMState Operator. In the OpenShift console, select Ecosystem > Software Catalog. In the search box, type nmsat. Select Kubernetes NMState Operator.



7. Click Install. Click Install again to deploy the operator. When the operator is installed, click View Operator



8. Select the NMState tab. On the right, click Create NMState. Leave all defaults in place and click Create. The nmstate will be created. You will also need to refresh the console because additional items will be added under Networking.
9. Create the following file to configure bond interface and a vlan interface on top of that bond interface.

```
cat nfs-bond.yaml
apiVersion: nmstate.io/v1
kind: NodeNetworkConfigurationPolicy
metadata:
  name: nfs-bond99-vlan2287-workers
spec:
  nodeSelector:
    node-role.kubernetes.io/worker: ""
  desiredState:
    interfaces:
      # Bond interface
      - name: bond99
        type: bond
        state: up
        mtu: 9000
        link-aggregation:
          mode: 802.3ad
          options:
            miimon: "100"
            xmit_hash_policy: layer3+4
        ports:
          - ens117f0np0
          - ens117f1np1
      # Bond slaves
      - name: ens117f0np0
        type: ethernet
        state: up
        mtu: 9000
      - name: ens117f1np1
        type: ethernet
        state: up
        mtu: 9000
      # VLAN on top of bond
      - name: bond99.2287
        type: vlan
        state: up
        mtu: 9000
        vlan:
          base-iface: bond99
          id: 2287
        ipv4:
```

```

enabled: true
dhcp: true
ipv6:
  enabled: false

```

10. Apply the file to create the bond interface.

NodeNetworkConfigurationPolicy

★ Create

Filter Name Search by name

Name

Matched nodes

Enactment status

info-bond09-yn287-workers

4 nodes

4 Available

11. Create a node label for Cisco UCS C240 M8 GPU worker nodes.

Note: This is required so that machine configs (requires reboot) and other policies can be applied without impacting all nodes in the cluster. GPU-dense nodes may require different policies so creating roles allow you to apply, especially machine configurations that require a reboot.

```

oc get nodes
NAME                                STATUS    ROLES                                AGE    VERSION
compute-01.flexpod-afx.fpmc.sa     Ready    worker                               30m    v1.33.6
compute-02.flexpod-afx.fpmc.sa     Ready    worker                               30m    v1.33.6
compute-03.flexpod-afx.fpmc.sa     Ready    worker                               30m    v1.33.6
compute-04.flexpod-afx.fpmc.sa     Ready    worker                               29m    v1.33.6
control-01.flexpod-afx.fpmc.sa     Ready    control-plane,master                36m    v1.33.6
control-02.flexpod-afx.fpmc.sa     Ready    control-plane,master                59m    v1.33.6
control-03.flexpod-afx.fpmc.sa     Ready    control-plane,master                59m    v1.33.6

```

12. Apply label to worker nodes.

```
oc label node compute-01.flexpod-afx.fpmc.sa node-role.kubernetes.io/worker-ucs-c240m8=
```

13. Similarly label the remaining worker nodes and verify

```

oc get nodes
NAME                                STATUS    ROLES                                AGE    VERSION
compute-01.flexpod-afx.fpmc.sa     Ready    worker,worker-ucs-c240m8            35m    v1.33.6
compute-02.flexpod-afx.fpmc.sa     Ready    worker,worker-ucs-c240m8            35m    v1.33.6
compute-03.flexpod-afx.fpmc.sa     Ready    worker,worker-ucs-c240m8            35m    v1.33.6
compute-04.flexpod-afx.fpmc.sa     Ready    worker,worker-ucs-c240m8            35m    v1.33.6
control-01.flexpod-afx.fpmc.sa     Ready    control-plane,master                41m    v1.33.6
control-02.flexpod-afx.fpmc.sa     Ready    control-plane,master                64m    v1.33.6
control-03.flexpod-afx.fpmc.sa     Ready    control-plane,master                64m    v1.33.6

```

14. Create Machine Config Pool for UCS C240 M8 servers including the new node label.

```

cat worker-ucs-c240m8-mcp.yaml
apiVersion: machineconfiguration.openshift.io/v1
kind: MachineConfigPool
metadata:
  name: worker-ucs-c240m8
spec:
  machineConfigSelector:

```

```

    matchExpressions:
      - {key: machineconfiguration.openshift.io/role, operator: In, values: [worker,worker-ucs-
c240m8]}
    nodeSelector:
      matchLabels:
        node-role.kubernetes.io/worker-ucs-c240m8: ""

```

15. Apply the file configure Machine Config Pool.

```
oc create -f worker-ucs-c240m8-mcp.yaml
```

MachineConfigPools ★ [Create MachineConfigPool](#)

Name Search by name...

Name	Configuration	Degraded	Update status
MCP master	MC rendered-master-d4986e2a5b667463c98d975b4b938e40b	False	Up to date
MCP worker	MC rendered-worker-62d6f2ea9fa3fed349ecee753b17dc6e	False	Up to date
MCP worker-ucs-c240m8	MC rendered-worker-ucs-c240m8-62d6f2ea9fa3fed349ecee753b17dc6e	False	Up to date

16. Create Machine Config by setting kernel arguments to disable irdma.

```

cat 99-worker-ucs-c240m8-kernel-args.yaml

apiVersion: machineconfiguration.openshift.io/v1
kind: MachineConfig
metadata:
  labels:
    machineconfiguration.openshift.io/role: worker-ucs-c240m8
  name: 99-worker-ucs-c240m8-kernel-args
spec:
  kernelArguments:
    - "module_blacklist=irdma"
    - "modprobe.blacklist=usnic_verbs"
    - intel_iommu=off
    - amd_iommu=off
    - iommu=pt

```

Note: Cisco UCS systems with VIC (mLOM) adapters may load the usnic_verbs kernel module by default, enabling Cisco usNIC RDMA functionality. In this deployment, RDMA workloads rely on Mellanox/NVIDIA adapters (for example, ConnectX-7 for RoCE), the usnic_verbs module is not required and can interfere with the Infiniband (ib_core) stack.

16. Apply the file to add Machine Config.

```
oc apply -f 99-worker-ucs-c240m8-kernel-args.yaml
```

17. Verify MachineConfigPools. Wait until the Machine Config Pool has moved from “Updating” to “Up to date”.

MachineConfigPools ★ [Create MachineConfigPool](#)

Name Search by name...

Name	Configuration	Degraded	Update status
MCP master	MC rendered-master-d4986e7a5b567463c98d95b4b938e40b	False	Up to date
MCP worker	MC rendered-worker-6246f2ea9fa3fed349cece753b7dc6e	False	Up to date
MCP worker-ucs-c240m8	-	False	Updating

Note: Each worker will reboot during the config apply process

18. Login to each worker node and verify the parameters.

```
cat /proc/cmdline

BOOT_IMAGE=(hd0,gpt3)/boot/ostree/rhcos-
b69fabcb811ae38fcb9fbf9e0e3d9dc993cbcd4a858c156bd69d74670ef55d372/vmlinuz-5.14.0-
570.104.1.el9_6.x86_64 ignition.platform.id=metal
ostree=/ostree/boot.0/rhcos/b69fabcb811ae38fcb9fbf9e0e3d9dc993cbcd4a858c156bd69d74670ef55d372/0
root=UUID=6a205dd5-7b45-4c41-aa26-ed1ee25167b6 rw rootflags=prjquota boot=UUID=61ab5060-2a6d-
45f5-b938-71b2960beb01 systemd.unified_cgroup_hierarchy=1 cgroup_no_v1=all psi=0
module_blacklist=irdma intel_iommu=off amd_iommu=off iommu=pt

lsmod | grep irdma
<no output>
```

Disable PCIe Access Control Services

Access Control Services (ACS) is a PCIe I/O virtualization technology that can impact GPUDirect, including GPUDirect RDMA and GPUDirect Storage, by preventing direct communication between devices (GPU<->NIC) on the same PCIe bus. While ACS provides security in virtualized environments, it can degrade performance by redirecting this traffic through the PCIe Root Complex (CPU).

19. To disable use the below script.

```
#!/bin/bash
# must be root to access extended PCI config space
if [ "$EUID" -ne 0 ]; then
    echo "ERROR: $0 must be run as root"
    exit 1
fi

for BDF in `lspci -d "*:*:*" | awk '{print $1}'`; do

    # skip if it doesn't support ACS
    setpci -v -s ${BDF} ECAP_ACS+0x6.w > /dev/null 2>&1
    if [ $? -ne 0 ]; then
        #echo "${BDF} does not support ACS, skipping"
        continue
    fi

    logger "Disabling ACS on `lspci -s ${BDF}`"
    setpci -v -s ${BDF} ECAP_ACS+0x6.w=0000
    if [ $? -ne 0 ]; then
        logger "Error disabling ACS on ${BDF}"
        continue
    fi

    NEW_VAL=`setpci -v -s ${BDF} ECAP_ACS+0x6.w | awk '{print $NF}'`
```

```

if [ "${NEW_VAL}" != "0000" ]; then
    logger "Failed to disable ACS on ${BDF}"
    continue
fi
done
exit 0

```

20. Create base64 encoded.

```
cat disable-accs.sh | base64 -w 0
```

```

IyEvYmluL2Jhc2gKIyBtdXN0IGJlIHJvb3QgdG8gYWNjZXNzIGV4dGVuZGVkIFBDSSBjb25maWcg3BhY2UKaWYgWyAiJEVVS
UQiIC1uZSAwIF07IHRoZW4KICBlY2hvICJFUlJPUjogJDAGbXVzdCBiZSB5dW4gYXMGcm9vdCIKICBlG10IDEKZmkKCmZvci
BCREYgaW4gYGxzGnPiC1kICiQOio6KiIgfCBhd2sgJ3twcmludCAkMX0nYDsgZG8KCIAgICAjIHNraXAgawYgaXQgZG9lc24
ndCBzdXBwb3J0IEFDUwogICAgc2V0cGnPiC12IC1zICR7QkRGfSBFQ0FQX0FDUysweDYudyA+IC9kZXYvbnVsbCAyPiYxCiAg
ICBpZiBbICQ/IC1uZSAwIF07IHRoZW4KICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgIC
2tpcHBpbmciCiAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgIC
kgLXMgJHtCREZ9YCIKICAgIHNldHBjaSAtdiAtcyAke0JERn0gRUNBUf9BQ1MrMHg2Lnc9MDAwMCAgICAgIAogICAgawYgWyA
kPyAtbmUgMCAwOyB0aGVuCiAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgIC
ICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAg
ICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgIC
Cd7cHJpbmQgJE5GfSdgCiAgICBpZiBbICike05FVl9WQUx9IiAhPSAiMDAwMCIgXTsgdGhlgogICAgICAgICAgICAgICAg
FpbGVkIHRvIGRpc2FibGUGQUNTIG9uICR7QkRGfSIKICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAg
=

```

21. Create machineconfig file with the following information.

```

apiVersion: machineconfiguration.openshift.io/v1

kind: MachineConfig

metadata:

  name: 99-machine-config-disable-accs

  labels:

    machineconfiguration.openshift.io/role: worker

spec:

  config:

    ignition:

      version: 3.4.0

    systemd:

      units:

        - name: 99-machine-config-disable-accs

          enabled: true

          contents: |

storage:

  files:

    - filesystem: root

      path: /usr/local/bin/disable-accs.sh

      contents:

```

[illegible]

oc get mcp worker-ucs-c240m8					UPDATED
NAME	CONFIG	MACHINECOUNT	READYMACHINECOUNT	UPDATEDMACHINECOUNT	
DEGRADED	AGE				
worker-ucs-c240m8	rendered-worker-ucs-c240m8-9fe275c48b3089b1520b0d20ac697c3a				False
False	4	0	0	0	True 14h

[illegible]

```

DirectTrans-      ACSctl: SrcValid- TransBlk- ReqRedir- CmpltdRedir- UpstreamFwd- EgressCtrl-
DirectTrans-      ACSctl: SrcValid- TransBlk- ReqRedir- CmpltdRedir- UpstreamFwd- EgressCtrl-
DirectTrans-      ACSctl: SrcValid- TransBlk- ReqRedir- CmpltdRedir- UpstreamFwd- EgressCtrl-
DirectTrans-      ACSctl: SrcValid- TransBlk- ReqRedir- CmpltdRedir- UpstreamFwd- EgressCtrl-
DirectTrans-      ACSctl: SrcValid- TransBlk- ReqRedir- CmpltdRedir- UpstreamFwd- EgressCtrl-
DirectTrans-      ACSctl: SrcValid- TransBlk- ReqRedir- CmpltdRedir- UpstreamFwd- EgressCtrl-
DirectTrans-      ACSctl: SrcValid- TransBlk- ReqRedir- CmpltdRedir- UpstreamFwd- EgressCtrl-
DirectTrans-      ACSctl: SrcValid- TransBlk- ReqRedir- CmpltdRedir- UpstreamFwd- EgressCtrl-
- :::::

```

24. Setup password for core user.

```
mkpasswd -m SHA-512 <password>
```

25. create machineconfig file.

```

apiVersion: machineconfiguration.openshift.io/v1
kind: MachineConfig
metadata:
  labels:
    machineconfiguration.openshift.io/role: worker
  name: set-core-user-password
spec:
  config:
    ignition:
      version: 3.5.0
    passwd:
      users:
        - name: core
          passwordHash: <password>

```

26. Apply the file.

```

oc apply -f core-user-password.yaml
machineconfig.machineconfiguration.openshift.io/set-core-user-password created

```

27. Verify mcp status.

```

oc get mcp
NAME                                CONFIG                                UPDATED
UPDATING  DEGRADED  MACHINECOUNT  READYMACHINECOUNT  UPDATEDMACHINECOUNT  AGE
DEGRADEDMACHINECOUNT
master    False      3              3                    0                      2d19h
worker    False      0              0                    0                      2d19h
worker-ucs-c240m8  False      4              2                    0                      12h

```

28. Verify the password is applied.

```

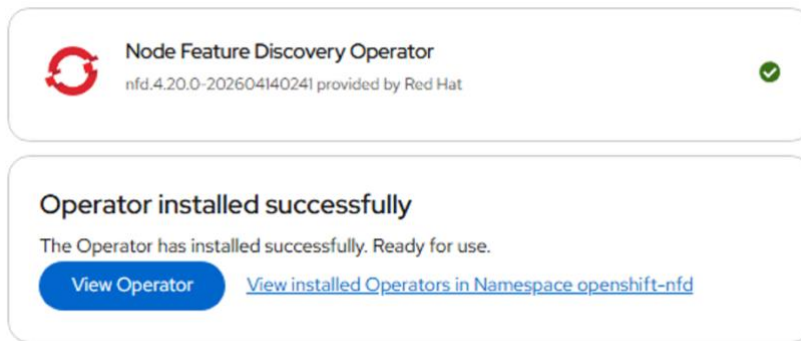
oc debug node/compute-01.flexpod-afx.fpmc.sa
Starting pod/compute-01flexpod-afxfpmsa-debug-jd7rt ...
To use host binaries, run `chroot /host`
Pod IP: 10.61.178.124
If you don't see a command prompt, try pressing enter.
sh-5.1# chroot /host

```

```
sh-5.1# cat /etc/shadow |grep core:
core:$6$m8Sjc9udlcbkTcEr$XU.8p/wtN/oNnHbbZgR9LDC5DA3oWrKDtOSGKjQDrx96ckgkm/qmLwtY34VGjfOJDCYJ0uJx
YZ5uNuthk8bRf/:20541:0:99999:7::::
```

NVIDIA Network and GPU Operator

1. Install Node Feature Discovery Operator. In the OpenShift console, select Ecosystem > Software Catalog. In the search box, type Node Feature. Select Node Feature Discovery Operator. Click Install. Click Install again to deploy the operator. When the operator is installed, click View Operator.



2. Select the NodeFeatureDiscovery tab and click Create NodeFeatureDiscovery. Use the below YAML file to create the instance.

```
apiVersion: nfd.openshift.io/v1
kind: NodeFeatureDiscovery
metadata:
  name: nfd-instance
  namespace: openshift-nfd
spec:
  instance: ''
  operand:
    servicePort: 12000
  prunerOnDelete: false
  topologyUpdater: false
  workerConfig:
    configData: |
      core:
        sleepInterval: 60s
      sources:
        pci:
          deviceClassWhitelist:
            - "02"
```

```

- "03"

- "0200"

- "0207"

- "12"

deviceLabelFields:

- "vendor"

```

3. Wait for nfd-instance to reach Status as **Available, Upgradable**.

Project: openshift-nfd

Installed Operators > Operator details

Node Feature Discovery Operator
4.20.0-20260410241 provided by Red Hat

Details | YAML | Subscription | Events | All instances | **NodeFeatureDiscovery** | NodeFeatureGroup | NodeFeatureRule | NodeFeatureRule | NodeFeature | NodeFeature

NodeFeatureDiscoveries Create NodeFeatureDiscovery

Name Search by name...

Name	Kind	Status	Labels	Last updated
NFD nfd-instance	NodeFeatureDiscovery	Conditions: Available, Upgradable	No labels	1 minute ago

4. To install NVIDIA Network Operator, select Ecosystem > Software Catalog. In the search box, type NVIDIA Network. Select NVIDIA Network Operator. Click Install. Click Install again to deploy the operator.

NVIDIA Network Operator
nvidia-network-operator.v26.1.0 provided by NVIDIA

Operator installed successfully
The Operator has installed successfully. Ready for use.

[View Operator](#)

[View installed Operators in Namespace nvidia-network-operator](#)

5. Verify the pod status.

```

oc get pods -n nvidia-network-operator
NAME                                READY   STATUS    RESTARTS   AGE
nvidia-network-operator-controller-manager-779b99f76f-r2lpf  1/1     Running   0           3m39s

```

6. Go to NicClusterPolicy, create NIC cluster policy in NVIDIA Network Operator.

```

apiVersion: mellanox.com/v1alpha1
kind: NicClusterPolicy
metadata:
  name: nic-cluster-policy
spec:
  ofedDriver:
    image: doca-driver
    repository: nvcr.io/nvidia/mellanox
    version: doca3.3.0-26.01-1.0.0.0-0
    env:
      - name: ENABLE_NFSRDMA

```

```

    value: "true"

- name: UNLOAD_STORAGE_MODULES

  value: "true"

- name: RESTORE_DRIVER_ON_POD_TERMINATION

  value: "true"

- name: CREATE_IFNAMES_UDEV

  value: "true"

- name: ENTRYPOINT_DEBUG

  value: "true"

startupProbe:
  initialDelaySeconds: 10
  periodSeconds: 20
livenessProbe:
  initialDelaySeconds: 30
  periodSeconds: 30
readinessProbe:
  initialDelaySeconds: 10
  periodSeconds: 30
rdmaSharedDevicePlugin:

config: |

  {

    "configList": [

      {

        "resourceName": "rdma_shared_device_1",

        "rdmaHcaMax": 63,

        "selectors": {

          "ifNames": [

            "ens1f0np0",

            "ens1f1np1"

          ]

        }

      },

      {

        "resourceName": "rdma_shared_device_2",

        "rdmaHcaMax": 63,

        "selectors": {

          "ifNames": ["bond99"]

        }

      }

    ]

  }

```

```

    ]
  }

  image: k8s-rdma-shared-dev-plugin
  repository: nvcr.io/nvidia/mellanox
  version: 'sha256:cd489dfcddeb8ef67284d177cabc347f01d6b4fba0a44335a2026f6b6e11c847'
nvIpam:
  image: nvidia-k8s-ipam
  repository: nvcr.io/nvidia/mellanox
  version: network-operator-v26.1.0
  imagePullSecrets: []
  enableWebhook: false

```

7. Verify pod in nvidia-network-operator namespace. Mofed pod should be listed for each work node.

```
oc project nvidia-network-operator
```

```
oc get pods
```

NAME	READY	STATUS	RESTARTS	AGE
mofed-rhel9.6-9dbd55678-ds-5dm4w	2/2	Running	0	57m
mofed-rhel9.6-9dbd55678-ds-5wmb5	2/2	Running	0	57m
mofed-rhel9.6-9dbd55678-ds-frfpr	2/2	Running	0	57m
mofed-rhel9.6-9dbd55678-ds-lsxwf	2/2	Running	0	57m
nv-ipam-controller-7d4f48f577-nm6js	1/1	Running	0	57m
nv-ipam-controller-7d4f48f577-t8ccd	1/1	Running	0	57m
nv-ipam-node-2pr68	1/1	Running	0	57m
nv-ipam-node-cwn6g	1/1	Running	0	57m
nv-ipam-node-kjwqf	1/1	Running	0	57m
nv-ipam-node-wq6mr	1/1	Running	0	57m
nvidia-network-operator-controller-manager-5df9db5f4c-2lvqk	1/1	Running	0	61m
rdma-shared-dp-ds-2xfdl	1/1	Running	0	31s
rdma-shared-dp-ds-9frgt	1/1	Running	0	5m
rdma-shared-dp-ds-jd8zj	1/1	Running	0	18m
rdma-shared-dp-ds-m5lnl	1/1	Running	0	11ms

8. In the OpenShift console we can verify the drivers by running the following command on one of the mofed pod.

```
#oc exec -it mofed-rhel9.6-9dbd55678-ds-5dm4w -- bash
```

```
Defaulted container "mofed-container" out of: mofed-container, openshift-driver-toolkit-ctr,
network-operator-init-container (init)
```

```
#ofed_info -s
```

```
OFED-internal-26.01-1.0.0:
```

```
#ibdev2netdev -v
```

```

0000:0b:00.0 mlx5_0 (MT4129 - MCX755106AS-HEAT) NVIDIA ConnectX-7 Adapter Card, 200GbE / NDR200,
Dual-port QSFP112, PCIe 5.0x16 with x16 PCIe extension option, Crypto Disabled, Secure Boot
Enabled fw 28.46.1006 port 1 (ACTIVE) ==> ens1f0np0 (Up)
0000:0b:00.1 mlx5_1 (MT4129 - MCX755106AS-HEAT) NVIDIA ConnectX-7 Adapter Card, 200GbE / NDR200,
Dual-port QSFP112, PCIe 5.0x16 with x16 PCIe extension option, Crypto Disabled, Secure Boot
Enabled fw 28.46.1006 port 1 (ACTIVE) ==> ens1f1np1 (Up)
0000:c6:00.0 mlx5_bond_0 (MT4129 - MCX755106AS-HEAT) NVIDIA ConnectX-7 Adapter Card, 200GbE /
NDR200, Dual-port QSFP112, PCIe 5.0x16 with x16 PCIe extension option, Crypto Disabled, Secure
Boot Enabled fw 28.46.1006 port 1 (ACTIVE) ==> bond99 (Up)

```

9. The **NICClusterPolicies** status should be ready.

Project: nvidia-network-operator ▾

[Installed Operators](#) ▸ Operator details

**NVIDIA Network Operator**
26.1.0 provided by NVIDIA

★ [Actions ▾](#)

[Details](#) [YAML](#) [Subscription](#) [Events](#) [All instances](#) [HostDeviceNetwork](#) [IPoBNetwork](#) [MacvlanNetwork](#) **[NicClusterPolicy](#)**

NicClusterPolicies

Create NicClusterPolicy

Name ▾

Search by name...



Name ▴ ▾	Kind ▴ ▾	Status ▴ ▾	Labels ▴ ▾	Last updated ▴ ▾	
 nic-cluster-policy	NicClusterPolicy	State: ready	No labels	 Apr 22, 2026, 6:48 PM	

10. Create MacvlanNetwork by using the below .yaml file.

```
mac-vlan-network-0.yaml

apiVersion: mellanox.com/v1alpha1
kind: MacvlanNetwork
metadata:
  name: be-mac-vlan-network-0
spec:
  networkNamespace: default
  master: ens1f0np0
  mode: bridge
  mtu: 9000
  ipam: '{"type": "whereabouts", "range": "192.168.2.0/24" }'

mac-vlan-network-1.yaml

apiVersion: mellanox.com/v1alpha1
kind: MacvlanNetwork
metadata:
  name: be-mac-vlan-network-1
spec:
  networkNamespace: default
  master: ens1f1np1
  mode: bridge
  mtu: 9000
  ipam: '{"type": "whereabouts", "range": "192.168.2.0/24" }'
```

11. Verify the status.

Project: nvidia-network-operator ▾

[Installed Operators](#) ▸ Operator details

 **NVIDIA Network Operator**
26.1.0 provided by NVIDIA

★ Actions ▾

Details YAML Subscription Events All instances HostDeviceNetwork IPolBNetwork **MacvlanNetwork** NicClusterPolicy

MacvlanNetworks

Create MacvlanNetwork

Name ▾ Search by name... 🔍

Name	Kind	Status	Labels	Last updated
 be-mac-vlan-network-0	MacvlanNetwork	State: ready	No labels	🕒 Just now
 be-mac-vlan-network-1	MacvlanNetwork	State: ready	No labels	🕒 Just now

12. To deploy NVIDIA GPU Operator, select Ecosystem > Software Catalog. In the search box, type NVIDIA GPU Select NVIDIA GPU Operator. Click Install. Click Install again to deploy the operator.

 **NVIDIA GPU Operator**
gpu-operator-certified.v26.3.1 provided by NVIDIA Corporation

Operator installed successfully
The Operator has installed successfully. Ready for use.
[View Operator](#) [View installed Operators in Namespace nvidia-gpu-operator](#)

13. We will use the latest nvidia-fs driver available in NGC.

Container

 **NVIDIA GDS Driver**
NVIDIA GDS Driver
NVIDIA AI Enterprise Supported

Get Container ▾

Release Types ▾ 2.27.3-rhel9.6 Signed

14. Apply below .yaml file to create GPU cluster policy

```
apiVersion: nvidia.com/v1
kind: ClusterPolicy
metadata:
  name: gpu-cluster-policy
spec:
  vgpuDeviceManager:
    config:
      default: default
    enabled: true
  migManager:
    config:
```

```

    default: all-disabled
    name: default-mig-parted-config
    enabled: true
operator:
  defaultRuntime: crio
  initContainer: {}
  runtimeClass: nvidia
  use_ocp_driver_toolkit: true
dcgm:
  enabled: true
gfd:
  enabled: true
dcgmExporter:
  config:
    name: ''
  serviceMonitor:
    enabled: true
    enabled: true
cdi:
  default: false
  enabled: true
driver:
  licensingConfig:
    nlsEnabled: true
    secretName: ''
  kernelModuleType: auto
  certConfig:
    name: ''
  rdma:
    enabled: true
    useHostMofed: false
  kernelModuleConfig:
    name: ''
upgradePolicy:
  autoUpgrade: true
  drain:
    deleteEmptyDir: false
    enable: false
    force: false
    timeoutSeconds: 300
  maxParallelUpgrades: 1
  maxUnavailable: 25%
  podDeletion:
    deleteEmptyDir: false
    force: false
    timeoutSeconds: 300
  waitForCompletion:
    timeoutSeconds: 0
repoConfig:
  configMapName: ''
virtualTopology:
  config: ''
  enabled: true
  useNvidiaDriverCRD: false
devicePlugin:
  config:
    name: ''
    default: ''
mps:
  root: /run/nvidia/mps
  enabled: true
gdrcopy:
  enabled: false
kataManager:
  config:
    artifactsDir: /opt/nvidia-gpu-operator/artifacts/runtimeclasses
mig:
  strategy: single

```

```

sandboxDevicePlugin:
  enabled: true
validator:
  plugin:
    env: []
nodeStatusExporter:
  enabled: true
daemonsets:
  rollingUpdate:
    maxUnavailable: '1'
  updateStrategy: RollingUpdate
sandboxWorkloads:
  defaultWorkload: container
  enabled: false
gds:
  enabled: true
  image: nvidia-fs
  repository: nvcr.io/nvidia/cloud-native
version: 2.27.3
vgpuManager:
  enabled: false
vfioManager:
  enabled: true
toolkit:
  installDir: /usr/local/nvidia
  enabled: true

```

Note: Make sure to enable RDMA and disable useHostMofed in the driver section in the spec of the ClusterPolicy CR.

15. Verify the pods in nvidia-gpu-operator project.

```
#oc project nvidia-gpu-operator
```

```
#oc get pod
```

NAME	READY	STATUS	RESTARTS	AGE
gpu-feature-discovery-d989j	1/1	Running	0	12m
gpu-feature-discovery-mpzrz	1/1	Running	0	12m
gpu-feature-discovery-swvbg	1/1	Running	0	12m
gpu-feature-discovery-xx8z5	1/1	Running	0	12m
gpu-operator-5586fd8bbf-kdpdr	1/1	Running	0	24m
nvidia-container-toolkit-daemonset-dzx5j	1/1	Running	0	12m
nvidia-container-toolkit-daemonset-jvxfx	1/1	Running	0	12m
nvidia-container-toolkit-daemonset-tlkpr	1/1	Running	0	12m
nvidia-container-toolkit-daemonset-vnz76	1/1	Running	0	12m
nvidia-cuda-validator-6rgr2	0/1	Completed	0	8m21s
nvidia-cuda-validator-8jpc5	0/1	Completed	0	8m17s
nvidia-cuda-validator-k58wv	0/1	Completed	0	8m32s
nvidia-cuda-validator-wxs5j	0/1	Completed	0	8m21s
nvidia-dcgm-46w2h	1/1	Running	0	12m
nvidia-dcgm-exporter-g2x6x	1/1	Running	6 (4m47s ago)	12m
nvidia-dcgm-exporter-h2t6b	1/1	Running	6 (4m40s ago)	12m
nvidia-dcgm-exporter-jjndl	1/1	Running	6 (4m39s ago)	12m
nvidia-dcgm-exporter-rfpqk	1/1	Running	6 (4m39s ago)	12m
nvidia-dcgm-j8q8b	1/1	Running	0	12m
nvidia-dcgm-msq7j	1/1	Running	0	12m
nvidia-dcgm-rbshh	1/1	Running	0	12m
nvidia-device-plugin-daemonset-2wc42	1/1	Running	0	12m
nvidia-device-plugin-daemonset-j5wqw	1/1	Running	0	12m
nvidia-device-plugin-daemonset-pcmtz	1/1	Running	0	12m
nvidia-device-plugin-daemonset-qhg59	1/1	Running	0	12m
nvidia-driver-daemonset-9.6.20260217-1-67q8v	3/3	Running	0	12m
nvidia-driver-daemonset-9.6.20260217-1-6lrr6	3/3	Running	0	12m
nvidia-driver-daemonset-9.6.20260217-1-757bt	3/3	Running	0	12m
nvidia-driver-daemonset-9.6.20260217-1-9rzgj	3/3	Running	0	12m
nvidia-mig-manager-4rbdc	1/1	Running	0	7m54s

nvidia-mig-manager-bc8tc	1/1	Running	0	7m54s
nvidia-mig-manager-gdprv	1/1	Running	0	7m54s
nvidia-mig-manager-lsxdh	1/1	Running	0	7m33s
nvidia-node-status-exporter-7trj7	1/1	Running	0	12m
nvidia-node-status-exporter-bcqq6	1/1	Running	0	12m
nvidia-node-status-exporter-fqw2x	1/1	Running	0	12m
nvidia-node-status-exporter-mk2hs	1/1	Running	0	12m
nvidia-operator-validator-9vm79	1/1	Running	0	12m
nvidia-operator-validator-hzkj4	1/1	Running	0	12m
nvidia-operator-validator-l2p5f	1/1	Running	0	12m
nvidia-operator-validator-swqdk	1/1	Running	0	12m

16. The state of gpu cluster policy needs to be ready.

Project: nvidia-gpu-operator ▾

Installed Operators ▸ Operator details



NVIDIA GPU Operator
26.31 provided by NVIDIA Corporation

★ Actions ▾

Details YAML Subscription Events All instances ClusterPolicy NVIDIADriver

ClusterPolicies Create ClusterPolicy

Name ▾ Search by name... 

Name ▴	Kind ▴	Status ▴	Labels ▴	Last updated ▴
 gpu-cluster-policy	ClusterPolicy	State: ready	No labels	 Apr 22, 2026, 8:07 PM

17. Also, the status Verify nvidia_fs module is loaded.

```
#oc rsh nvidia-driver-daemonset-9.6.20260217-1-67q8v
sh-5.1#
sh-5.1#
sh-5.1# lsmod | grep nvidia
nvidia_fs             323584  0
nvidia_peermem        20480  0
nvidia_modeset        1826816  0
nvidia_uvm            4079616  12
nvidia                14438400  49 nvidia_uvm,nvidia_peermem,nvidia_fs,nvidia_modeset
video                 77824  1 nvidia_modeset
ib_uverbs             233472  3 nvidia_peermem,rdma_ucm,mlx5_ib
drm                   815104  5 drm_kms_helper,ast,drm_shmem_helper,nvidia
```

NetApp Trident

1. In the OpenShift console, select Ecosystem > Software Catalog. In the search box, type Trident. Select NetApp Trident Certified edition. Click Install. Click Install again to deploy the operator.



NetApp Trident
trident-operator.v26.2.0 provided by NetApp, Inc.

✓

Operator installed successfully

The Operator has installed successfully. Ready for use.

View Operator
[View installed Operators in Namespace openshift-operators](#)

2. Click on View Operator and go to Trident Orchestartor, click Create TridentOrchestrators
3. Select Configure via Form View and leave everything default and click Create
4. Verify Trident pods in trident namespace

```
oc get pod -n trident
```

NAME	READY	STATUS	RESTARTS	AGE
trident-controller-6669c4cc5b-m6jbj	6/6	Running	0	113s
trident-node-linux-7mg8p	2/2	Running	0	113s
trident-node-linux-8m88n	2/2	Running	0	113s
trident-node-linux-ftw22	2/2	Running	0	113s
trident-node-linux-g55jm	2/2	Running	0	113s
trident-node-linux-pmrhw	2/2	Running	0	113s
trident-node-linux-qwddp	2/2	Running	0	113s
trident-node-linux-rsbvq	2/2	Running	0	113s

5. Create backend for NFSoRDMA using below yaml file.

```
cat backend_AFX_NFS_RDMA.yaml
---
version: 1
storageDriverName: ocp-afx-rdma-nfs
backendName: ontap-nas
managementLIF: 172.22.84.10
dataLIF: 172.22.87.51
svm: ocp-svm
username: vsadmin
password: <password>
useREST: true
defaults:
  spaceReserve: none
  exportPolicy: default
  snapshotPolicy: default
  snapshotReserve: '0'
  nameTemplate:
"{{.config.StoragePrefix}}_{{.config.BackendName}}_{{.volume.Namespace}}_{{.volume.RequestName}}"
labels:
  service: "rdma"
  perf: "extreme"
```

6. Apply the file to create the backend and verify.

```
tridentctl -n trident create backend -f backend_AFX_NFS_flexgroup.yaml
```

7. Create storage class for RDMA.

```
cat storage-class-afx-nfs-rdma.yaml
---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: ontap-nfs-rdma
  annotations:
    storageclass.kubernetes.io/is-default-class: "false"
provisioner: csi.trident.netapp.io
parameters:
  backendType: "ontap-nas"
  selector: "service=rdma"
  provisioningType: "thin"
  snapshots: "true"
mountOptions:
["vers=4.1", "proto=rdma", "max_connect=16", "rsize=262144", "wsize=262144", "write=eager", "hard", "noa
time", "nodiratime"]
allowVolumeExpansion: true
```

Note: Only “ontap-nas” is a supported backend for AFX system.

8. Create backend for NFS

```
version: 1
storageDriverName: ontap-nas-flexgroup
backendName: ocp-afx-nfs
managementLIF: 172.22.85.10
dataLIF: 172.22.87.11
svm: svml
username: vsadmin
password: <password>
useREST: true
defaults:
  spaceReserve: none
  exportPolicy: default
  snapshotPolicy: default
  snapshotReserve: '0'
  nameTemplate:
    "{{.config.StoragePrefix}}_{{.config.BackendName}}_{{.volume.Namespace}}_{{.volume.RequestName}}"
labels:
  service: "nfs"
  perf: "standard"
```

9. Create storage class for standard NFS

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: ontap-nfs
  annotations:
    storageclass.kubernetes.io/is-default-class: "false"
provisioner: csi.trident.netapp.io
mountOptions:
  - vers=4.1
  - nconnect=16
parameters:
  backendType: ontap-nas
  provisioningType: thin
  selector: service=nfs
  snapshots: "true"
reclaimPolicy: Delete
volumeBindingMode: Immediate
allowVolumeExpansion: true
```

10. Verify the backends by running below command.

```
tridentctl get backend -n trident
```

NAME	STORAGE DRIVER	UUID	STATE	USER-STATE
vcn-0	ontap-nas	b97a4413-92fa-4358-8fe7-13ed70a2010b	online	normal
vcn-0	ontap-nas	c3325284-2513-4e13-8d76-9ef63f48a631	online	normal

Splunk Observability

This chapter contains the following:

- Prerequisites
- Install and Configure OTEL Collector Agents

This section provides the procedures to install and configure OTEL agents for various components of the solution to gain deep observability capabilities with Splunk Observability Cloud.

Prerequisites

The following prerequisites need to be met before installing and configuring Splunk OpenTelemetry Collector (OTEL):

1. To install Splunk OpenTelemetry Collector in a Kubernetes cluster environment, either of the destinations (splunkPlatform or splunkObservabilitycloud) to be configured. For this validation, Splunk Observability Cloud is pre-configured and required access to the observability cloud is provided.
2. Once access to observability cloud is enabled, generate Realm and Access token with which the OTEL collectors authenticates with Observability cloud and sends the metrics data. To create the Access token and realm, see: <https://help.splunk.com/en/splunk-observability-cloud/administer/authentication-and-security/authentication-tokens/org-access-tokens#admin-org-tokens> and https://dev.splunk.com/observability/docs/realms_in_endpoints/
3. Administrator Access to OpenShift cluster for deploying OTEL collector agents and sending the logs to the Observability cloud
4. Ensure Helm version 3 is installed on the Linux client machine

To install and configure OTEL collector agents, follow the steps below

Log into your OpenShift cluster with CLI and Install otel collector agents for OpenShift by running the following commands from the linux client machine.

1. Add helm repository of Splunk OTEL collector charts.

```
helm repo add splunk-otel-collector-chart https://signalfx.github.io/splunk-otel-collector-chart
```

2. Create a namespace and move to the project.

```
oc create namespace otel
oc project otel
```

3. Update the repo.

```
helm repo update
```

4. Install the Splunk otel collector using values.yaml file. Update the exporters, receivers and processors sections with appropriate values for your environment and run the following command to install the otel collector for your environment. Sample values.yaml file is provided below:

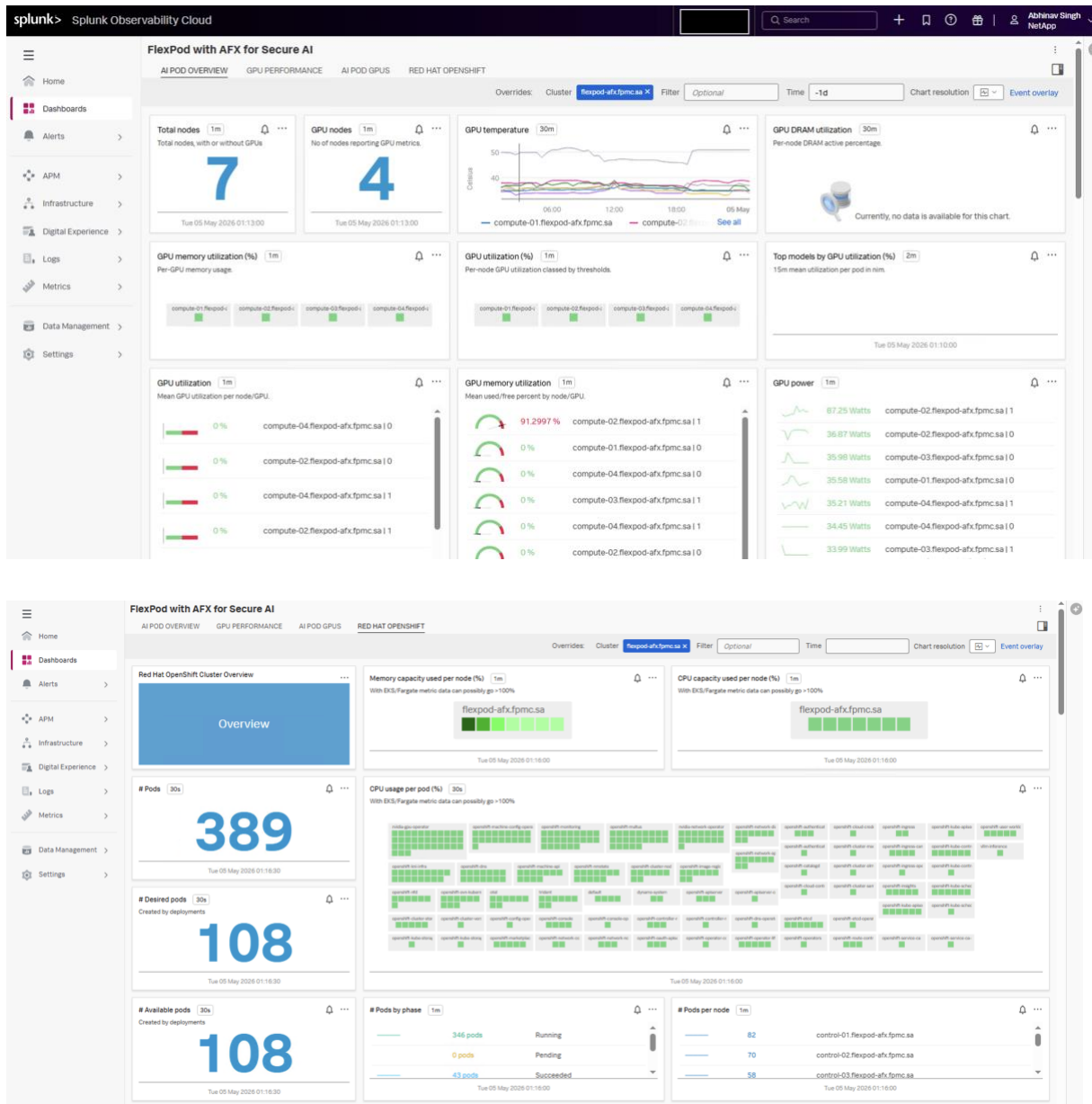
```
helm install ucs-otel-collector --
set="distribution=openshift,splunkObservability.accessToken=<*****>,clusterName=flexpod-afx.fpmc.sa,splunkObservability.realm=us1,gateway.enabled=false,environment=flexPod,operatorcrds.
install=false,operator.enabled=false,agent.discovery.enabled=true,splunkObservability.profilingEnabled=true" -f values.yaml splunk-otel-collector-chart/splunk-otel-collector
```

5. Verify the pod status.

```
oc get pod
```

NAME	READY	STATUS	RESTARTS	AGE
ucs-otel-collector-splunk-otel-collector-agent-2gdmr	1/1	Running	0	16m
ucs-otel-collector-splunk-otel-collector-agent-76btt	1/1	Running	0	16m
ucs-otel-collector-splunk-otel-collector-agent-bfkdg	1/1	Running	0	16m
ucs-otel-collector-splunk-otel-collector-agent-bvvff	1/1	Running	0	16m
ucs-otel-collector-splunk-otel-collector-agent-d5xrd	1/1	Running	0	16m
ucs-otel-collector-splunk-otel-collector-agent-dxctn	1/1	Running	0	16m
ucs-otel-collector-splunk-otel-collector-agent-zhp6g	1/1	Running	0	16m
ucs-otel-collector-splunk-otel-collector-k8s-cluster-recei4r64b	1/1	Running	0	14m

6. Below is the dashboard for Splunk for the infrastructure setup showing the nodes & pods



Solution Validation

This section showcases verified performance for AI workloads through tools such as Elbencho, NCCL tests, and GDSIO, ensuring consistent throughput, scalability, and efficient GPU data pipelines.

Running NCCL and GDSIO workloads requires architectures that can sustain high-bandwidth, low-latency GPU-to-GPU and GPU-to-storage communication, particularly for distributed training and GPUDirect Storage pipelines. While the Cisco UCS C240 M8 (2U PCIe-optimized) supports accelerated workloads,

the PCIe-based interconnect can constrain optimal inter-GPU communication patterns required for NCCL-driven collectives.

Additionally, NUMA alignment between GPUs and NICs requires careful consideration when sizing NCCL and GDS workloads for the platform, to maximize the RDMA-based data paths these workloads use. For such workloads, Cisco UCS C845A (MGX scale-out) and C885A/C880A (HGX with NVLink/NVSwitch) architectures are better suited, as they provide higher GPU density and direct GPU interconnect topologies, enabling efficient inter-GPU communication and maximizing performance for NCCL and GPUDirect Storage workloads.

Elbencho Benchmarking

Elbencho is a benchmark designed to measure I/O performance on modern and focuses on how fast and how consistently your storage stack can deliver data to compute and absorb writes such as checkpoints. The tool supports file systems, block devices as well S3 endpoints and runs in a single host or distributed runs across multiple clients to expose scale-out effects. Also supports buffering modes such as host-memory I/O and GPU-aware I/O, including optional NVIDIA GPUDirect Storage (GDS) via cuFile where supported. [Please refer the documentation for elbencho to install and configure.](#)

To validate the infrastructure setup Elbencho was used to saturate the 4-node AFX system for reads and writes ensuring the architecture was designed with end-to-end optimal parameters.

One FlexGroup volume was used to distribute data across the four AFX nodes and was created with the following option

```
vol create -volume elbencho1 -vserver svm1 -size 256t -auto-provision-as flexgroup
```

The command above provisions a 256TB FlexGroup volume distributed across four AFX nodes with 40 evenly balanced constituents (10 per node), ensuring optimal parallelism and load distribution. FlexGroup further accelerates large-file workloads by automatically chunking files larger than 10TB across all nodes. Four client servers (Cisco C240 M8) were used to drive the workloads, mounting the file system over NFS v4.2 with trunking enabled to maximize throughput.

Use the command example below to enable NFS trunking for a SVM on the AFX system.

```
vserver nfs modify -vserver svm1 -v4.1-trunking enabled
```

The NFS file system mount with trunking was done for all the clients using the command below.

```
mount -t nfs -o rw,vers=4.2,rsize=1048576,wsiz=1048576,trunkdiscovery,proto=tcp  
172.22.87.11:/elbencho1 /file1
```

A distributed elbencho test was used to validate both read and write performance at scale. Each client generated 80 large files (600 GB each), creating a highly parallel workload with 320 concurrent I/O streams. This approach was designed to simulate realistic, large-scale AI data pipelines and demonstrate the platform's ability to deliver consistent, high-throughput performance as demand scales.

```
elbencho --hosts=172.22.85.[115-118] -d -w -b 1M -t 80 --iodepth 1 --direct -s 600g /file1
```



The write workload achieved a sustained aggregate throughput of 40 GB/sec, evenly distributed across all four AFX nodes at approximately 10 GB/sec as shown below. The result demonstrates the linear scalability and consistent performance characteristics of the NetApp ONTAP disaggregated architecture under demanding, large-scale AI workloads.



The read benchmark delivered ~134 GB/sec of sustained aggregate throughput with balanced performance across all AFX nodes, confirming linear scalability and providing the bandwidth required to support large-scale, GPU-accelerated AI training pipelines.



RDMA Validation

This section details the validation methodology for the GPUDirect RDMA configuration deployed earlier.

1. Create a service account using the below .yaml file.

```
apiVersion: v1
kind: ServiceAccount
metadata:
  name: nvidiatools
  namespace: default
```

2. Assign privilege to the service account.

```
oc -n default adm policy add-scc-to-user privileged -z nvidiatools
```

3. Create a pod on one of the compute nodes using .yaml file.
4. Create a similar pod on another compute node.
5. ssh to the pod and verify the RDMA interfaces

```
sh-5.1# mst status -v
MST modules:
-----
MST PCI module is not loaded
MST PCI configuration module is not loaded
PCI devices:
-----
DEVICE_TYPE      MST      PCI      RDMA      NET
-----
NUMA
ConnectX7 (rev:0) NA      c6:00.1      1
ConnectX7 (rev:0) NA      0b:00.0  mlx5_0
0
GB100 (rev:0)    NA      8b:00.0      1
ConnectX7 (rev:0) NA      c6:00.0  mlx5_bond_0
1
ConnectX7 (rev:0) NA      0b:00.1  mlx5_1
0
GB100 (rev:0)    NA      46:00.0      0

sh-5.1# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
```

```

    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: eth0@if180: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 8900 qdisc noqueue state UP group default
    link/ether 0a:58:0a:82:02:aa brd ff:ff:ff:ff:ff:ff link-netnsid 0
    inet 10.130.2.170/23 brd 10.130.3.255 scope global eth0
        valid_lft forever preferred_lft forever
    inet6 fe80::858:aff:fe82:2aa/64 scope link
        valid_lft forever preferred_lft forever
3: net1@if23: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 9000 qdisc noqueue state UP group default
    qlen 1000
    link/ether 56:f8:61:11:8d:45 brd ff:ff:ff:ff:ff:ff link-netnsid 0
    inet 192.168.2.1/24 brd 192.168.2.255 scope global net1
        valid_lft forever preferred_lft forever
    inet6 fe80::54f8:61ff:fe11:8d45/64 scope link
        valid_lft forever preferred_lft forever
4: net2@if24: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 9000 qdisc noqueue state UP group default
    qlen 1000
    link/ether 6a:c4:9b:16:cb:20 brd ff:ff:ff:ff:ff:ff link-netnsid 0
    inet 192.168.2.2/24 brd 192.168.2.255 scope global net2
        valid_lft forever preferred_lft forever
    inet6 fe80::68c4:9bff:fe16:cb20/64 scope link
        valid_lft forever preferred_lft forever

```

6. Run the test on one of the pods

```

sh-5.1# ib_write_bw -R -T 41 -s 65536 -F -x 3 -m 4096 --report_gbits -q 16 -D 60 -d mlx5_0 -p
10000 --source_ip 192.168.2.1 --use_cuda=0 --use_cuda_dmabuf
WARNING: BW peak won't be measured in this run.
Perftest doesn't supports CUDA tests with inline messages: inline size set to 0

*****
* Waiting for client to connect... *
*****
initializing CUDA
Listing all CUDA devices in system:
CUDA device 0: PCIe address is 46:00
CUDA device 1: PCIe address is 8B:00

Picking device No. 0
[pid = 8942, dev = 0] device name = [NVIDIA RTX PRO 6000 Blackwell Server Edition]
creating CUDA Ctx
making it the current CUDA Ctx
CUDA device integrated: 0
using DMA-BUF for GPU buffer address at 0x7f87e3a00000 aligned at 0x7f87e3a00000 with aligned
size 2097152
allocated GPU buffer of a 2097152 address at 0xb93ac0 for type CUDA_MEM_DEVICE
Calling ibv_reg_dmabuf_mr(offset=0, size=2097152, addr=0x7f87e3a00000, fd=44) for QP #0
-----
RDMA_Write BW Test
Dual-port      : OFF      Device      : mlx5_0
Number of qps  : 16      Transport type : IB
Connection type : RC      Using SRQ    : OFF
PCIe relax order: ON      Lock-free    : OFF
ibv_wr* API     : ON      Using Enhanced Reorder : OFF
CQ Moderation   : 1
CQE Poll Batch  : Dynamic
Mtu             : 4096[B]
Link type       : Ethernet
GID index       : 4
Max inline data : 0[B]
rdma_cm QPs     : ON
Data_ex. method : rdma_cm      TOS      : 41
-----
Waiting for client rdma_cm QP to connect
Please run the same command with the IB/RoCE interface IP

```



```
-----
deallocating GPU buffer 00007f87e3a00000
destroying current CUDA Ctx
```

NCCL Validation

NCCL (NVIDIA Collective Communications Library) is a high-performance communication library designed to enable fast data exchange between GPUs—both within a server and across multiple servers. This test checks both the performance and the correctness of NCCL [NVIDIA Collective Communications Library] operations. For more information, refer [here](#). The test will run between the two pods deployed on two different compute nodes that have a secondary RDMA interface.

1. On one of the pods run the below commands to start NCCL test.

```
sh-5.1# source ./bashrc

mpirun --allow-run-as-root \
-H 192.168.2.1:1,192.168.2.3:1 \
-np 2 \
-bind-to none -map-by slot \
-mca btl ^openib \
-mca plm_rsh_args "-p 20024" \
-x NCCL_DEBUG=VERSION \
-x NCCL_DEBUG_SUBSYS=INIT,NET \
-x NCCL_SOCKET_IFNAME=net1 \
-x NCCL_IB_HCA=mlx5_0 \
all_reduce_perf -b 8 -e 16G -f2 -g 2
```

2. Output of the test between two Cisco UCS 240 M8 nodes using 1 GPU. The busBW for the test was 21.70 GBps.

```
mpirun --allow-run-as-root -H 192.168.2.1:1,192.168.2.3:1 -np 2 -bind-to none -map-by slot
-mca btl ^openib -mca plm_rsh_args "-p 20024" -x NCCL_DEBUG=VERSION -x
NCCL_DEBUG_SUBSYS=INIT,NET -x NCCL_SOCKET_IFNAME=net1 -x NCCL_IB_HCA=mlx5_0 all_reduce_perf
-b 8 -e 16G -f2 -g 1
Warning: Permanently added '[192.168.2.3]:20024' (ED25519) to the list of known hosts.
# nccl-tests version 2.18.3 nccl-headers=23004 nccl-library=23004
# Collective test starting: all_reduce_perf
# nThread 1 nGpus 1 minBytes 8 maxBytes 17179869184 step: 2(factor) warmup iters: 1 iters: 20 agg
iters: 1 validation: 1 graph: 0 unalign: 0
#
# Using devices
# Rank 0 Group 0 Pid 10594 on sample-be-workload-worker-0 device 0 [0000:46:00] NVIDIA RTX
PRO 6000 Blackwell Server Edition
# Rank 1 Group 0 Pid 10568 on sample-be-workload-worker-1 device 0 [0000:46:00] NVIDIA RTX
PRO 6000 Blackwell Server Edition
NCCL version 2.30.4+cuda13.2
#
#                                     out-of-place
in-place
#      size      count      type  redop  root    time  algbw  busbw  #wrong    time
#      (B)      (elements)              (us)  (GB/s)  (GB/s)              (us)
# (GB/s)  (GB/s)
      8          2    float    sum    -1    26.45    0.00    0.00      0    27.53
0.00    0.00      0
     16          4    float    sum    -1    28.24    0.00    0.00      0    30.77
0.00    0.00      0
     32          8    float    sum    -1    31.58    0.00    0.00      0    30.95
0.00    0.00      0
     64         16    float    sum    -1    29.32    0.00    0.00      0    30.01
0.00    0.00      0
    128         32    float    sum    -1    31.03    0.00    0.00      0    29.63
0.00    0.00      0
    256         64    float    sum    -1    29.53    0.01    0.01      0    28.54
0.01    0.01      0
```

0.02	512	128	float	sum	-1	29.48	0.02	0.02	0	30.48
	0.02	0								
0.03	1024	256	float	sum	-1	31.74	0.03	0.03	0	32.82
	0.03	0								
0.06	2048	512	float	sum	-1	34.04	0.06	0.06	0	32.03
	0.06	0								
0.14	4096	1024	float	sum	-1	36.95	0.11	0.11	0	29.45
	0.14	0								
0.27	8192	2048	float	sum	-1	33.23	0.25	0.25	0	30.23
	0.27	0								
0.53	16384	4096	float	sum	-1	30.52	0.54	0.54	0	30.68
	0.53	0								
1.04	32768	8192	float	sum	-1	31.50	1.04	1.04	0	31.51
	1.04	0								
1.66	65536	16384	float	sum	-1	38.64	1.70	1.70	0	39.37
	1.66	0								
2.47	131072	32768	float	sum	-1	53.25	2.46	2.46	0	52.99
	2.47	0								
3.08	262144	65536	float	sum	-1	88.18	2.97	2.97	0	85.13
	3.08	0								
3.27	524288	131072	float	sum	-1	158.80	3.30	3.30	0	160.25
	3.27	0								
9.79	1048576	262144	float	sum	-1	101.72	10.31	10.31	0	107.15
	9.79	0								
14.26	2097152	524288	float	sum	-1	154.78	13.55	13.55	0	147.10
	14.26	0								
16.69	4194304	1048576	float	sum	-1	259.40	16.17	16.17	0	251.37
	16.69	0								
18.08	8388608	2097152	float	sum	-1	463.83	18.09	18.09	0	463.94
	18.08	0								
18.57	16777216	4194304	float	sum	-1	901.81	18.60	18.60	0	903.43
	18.57	0								
18.81	33554432	8388608	float	sum	-1	1783.94	18.81	18.81	0	1783.68
	18.81	0								
18.91	67108864	16777216	float	sum	-1	3555.64	18.87	18.87	0	3549.55
	18.91	0								
19.01	134217728	33554432	float	sum	-1	7083.16	18.95	18.95	0	7061.26
	19.01	0								
19.14	268435456	67108864	float	sum	-1	13996.4	19.18	19.18	0	14023.5
	19.14	0								
19.50	536870912	134217728	float	sum	-1	28126.4	19.09	19.09	0	27534.0
	19.50	0								
20.16	1073741824	268435456	float	sum	-1	53552.5	20.05	20.05	0	53265.2
	20.16	0								
20.21	2147483648	536870912	float	sum	-1	105358	20.38	20.38	0	106254
	20.21	0								
21.02	4294967296	1073741824	float	sum	-1	204084	21.05	21.05	0	204290
	21.02	0								
21.52	8589934592	2147483648	float	sum	-1	400181	21.47	21.47	0	399102
	21.52	0								
21.64	17179869184	4294967296	float	sum	-1	791760	21.70	21.70	0	793903
	21.64	0								
# Out of bounds values : 0 OK										
# Avg bus bandwidth : 9.04148										
#										
# Collective test concluded: all_reduce_perf										
#										

3. Output of the test between two Cisco UCS 240 M8 nodes using 2 GPUs. The busBW for the test was 24.28 GBps.

```
[root@sample-be-workload-worker-0 ~]# mpirun --allow-run-as-root -H 192.168.2.1:1,192.168.2.3:1 -
np 2 -bind-to none -map-by slot -mca pml obl -mca btl ^openib -mca btl_tcp_if_include
192.168.2.0/24 -mca plm_rsh_args "-p 20024" -x NCCL_IB_DISABLE=1 -x NCCL_DEBUG=VERSION -x
NCCL_SOCKET_IFNAME=net1 -x NCCL_IB_HCA=mlx5_0 -x UCX_NET_DEVICES=net1 -x NCCL_NET_GDR_READ=1 -x
NCCL_ALGO=ring all_reduce_perf -b 8 -e 16G -f2 -g 2
```

```

Warning: Permanently added '[192.168.2.3]:20024' (ED25519) to the list of known hosts.
# nccl-tests version 2.18.3 nccl-headers=23004 nccl-library=23004
# Collective test starting: all_reduce_perf
# nThread 1 nGpus 2 minBytes 8 maxBytes 17179869184 step: 2(factor) warmup iters: 1 iters: 20 agg
iters: 1 validation: 1 graph: 0 unalign: 0
#
# Using devices
# Rank 0 Group 0 Pid 10613 on sample-be-workload-worker-0 device 0 [0000:46:00] NVIDIA RTX
PRO 6000 Blackwell Server Edition
# Rank 1 Group 0 Pid 10613 on sample-be-workload-worker-0 device 1 [0000:8b:00] NVIDIA RTX
PRO 6000 Blackwell Server Edition
# Rank 2 Group 0 Pid 10612 on sample-be-workload-worker-1 device 0 [0000:46:00] NVIDIA RTX
PRO 6000 Blackwell Server Edition
# Rank 3 Group 0 Pid 10612 on sample-be-workload-worker-1 device 1 [0000:8b:00] NVIDIA RTX
PRO 6000 Blackwell Server Edition
NCCL version 2.30.4+cuda13.2
#
#
# in-place out-of-place
# size count type redop root time algbw busbw #wrong time
# algbw busbw #wrong
# (B) (elements)
# (GB/s) (GB/s) (us) (GB/s) (GB/s) (us)
0.00 0.00 0 2 float sum -1 40.02 0.00 0.00 0 35.73
0.00 0.00 0 4 float sum -1 41.81 0.00 0.00 0 40.53
0.00 0.00 0 8 float sum -1 46.65 0.00 0.00 0 45.65
0.00 0.00 0 16 float sum -1 47.44 0.00 0.00 0 47.54
0.00 0.00 0 32 float sum -1 46.34 0.00 0.00 0 45.79
0.01 0.01 0 64 float sum -1 47.81 0.01 0.01 0 46.41
0.01 0.02 0 128 float sum -1 48.44 0.01 0.02 0 45.98
0.02 0.03 0 256 float sum -1 47.62 0.02 0.03 0 47.42
0.04 0.06 0 512 float sum -1 49.30 0.04 0.06 0 48.26
0.09 0.13 0 1024 float sum -1 48.43 0.08 0.13 0 46.01
0.19 0.28 0 2048 float sum -1 49.75 0.16 0.25 0 43.49
0.37 0.56 0 4096 float sum -1 49.55 0.33 0.50 0 44.07
0.65 0.97 0 8192 float sum -1 51.69 0.63 0.95 0 50.57
1.24 1.85 0 16384 float sum -1 55.13 1.19 1.78 0 53.06
1.36 2.04 0 32768 float sum -1 68.41 1.92 2.87 0 96.51
2.52 3.78 0 65536 float sum -1 106.18 2.47 3.70 0 103.94
2.82 4.23 0 131072 float sum -1 190.99 2.75 4.12 0 185.98
7.09 10.64 0 262144 float sum -1 142.39 7.36 11.05 0 147.82
10.94 16.41 0 524288 float sum -1 198.07 10.59 15.88 0 191.65
13.24 19.85 0 1048576 float sum -1 324.25 12.94 19.40 0 316.90
14.55 21.82 0 2097152 float sum -1 572.21 14.66 21.99 0 576.54
14.98 22.46 0 4194304 float sum -1 1106.73 15.16 22.74 0 1120.31

```

33554432	8388608	float	sum	-1	2151.33	15.60	23.40	0	2146.14
15.63	23.45	0							
67108864	16777216	float	sum	-1	4237.69	15.84	23.75	0	4238.53
15.83	23.75	0							
134217728	33554432	float	sum	-1	8407.72	15.96	23.95	0	8403.32
15.97	23.96	0							
268435456	67108864	float	sum	-1	16723.8	16.05	24.08	0	16736.1
16.04	24.06	0							
536870912	134217728	float	sum	-1	33428.0	16.06	24.09	0	33423.5
16.06	24.09	0							
1073741824	268435456	float	sum	-1	66805.3	16.07	24.11	0	66783.7
16.08	24.12	0							
2147483648	536870912	float	sum	-1	133655	16.07	24.10	0	134168
16.01	24.01	0							
4294967296	1073741824	float	sum	-1	268232	16.01	24.02	0	269255
15.95	23.93	0							
8589934592	2147483648	float	sum	-1	538116	15.96	23.94	0	538102
15.96	23.95	0							
17179869184	4294967296	float	sum	-1	1061252	16.19	24.28	0	1064914
16.13	24.20	0							
# Out of bounds values : 0 OK									
# Avg bus bandwidth : 10.7794									
# Collective test concluded: all_reduce_perf									

Validation of GDS and NFSoRDMA

NFS over RDMA and GDS can be tested using a rdma-tools container publicly available on quay.io.

1. Create two Persistent Volume Claims (PVCs) in the default namespace. In the OpenShift console, select Storage > PersistentVolumeClaims. Select the default Project. Click Create PersistentVolumeClaim > With Form and create the two PVCs as shown below.

Project: default ▾

Create PersistentVolumeClaim

[Edit YAML](#)

StorageClass
☐ ontap-nfs
 StorageClass for the new claim

PersistentVolumeClaim name *

 A unique name for the storage claim within the project

Access mode *
☐ Shared access (RWX)
 Access mode is set by StorageClass and cannot be changed

Size *

 Desired storage capacity
☐ Use label selectors to request storage
 PersistentVolume resources that match all label selectors will be considered for binding.

Volume mode *
☒ Filesystem ☐ Block

2. Verify the PVCs status.

Project: default ▾

PersistentVolumeClaims

★

Filter ▾ Name ▾ Search by name...

Name	Status	PersistentVolumes	Capacity	Used	StorageClass
PVC nfs-normal	Bound	pvc-23e0d9b2-2b63-43be-8507-93a698a23fa3	1TiB	-	ontap-nfs
PVC nfs-nfsa	Bound	pvc-23e0d9b2-2b63-43be-8507-93a698a23fa3	1TiB	-	ontap-nfs-nfsa

3. Create the .yaml file.

```
apiVersion: v1
kind: Pod
metadata:
  name: nvidiatools-30-workload-1
  namespace: default
spec:
  serviceAccountName: nvidiatools
  nodeSelector:
    kubernetes.io/hostname: compute-01.flexpod-afx.fpmc.sa
  volumes:
    - name: rdma-pv-storage
      persistentVolumeClaim:
        claimName: nfs-rdma
    - name: nordma-pv-storage
      persistentVolumeClaim:
        claimName: nfs-normal
  containers:
    - name: nvidiatools-30-workload
      image: quay.io/wabouham/ecosys-nvidia/rdma-tools:0.0.3
      imagePullPolicy: IfNotPresent
      securityContext:
        privileged: true
        capabilities:
          add: ["IPC_LOCK"]
      resources:
        limits:
          nvidia.com/gpu: 1
        requests:
          nvidia.com/gpu: 1
      volumeMounts:
        - name: rdma-pv-storage
          mountPath: /nfsfast
        - name: nordma-pv-storage
          mountPath: /nfsslow
```

Note: Service account was already created in the previous RDMA validation.

4. Verify mount on one of the worker nodes.

```
sh-5.1# mount | grep nfs
172.22.87.51:/trident_ocp_afx_rdma_nfs_default_nfs_rdma_1_54565 on /nfsfast type nfs4
(rw,noatime,nodiratime,vers=4.1,rsize=1048576,wsiz=1048576,namlen=255,hard,proto=rdma,port=20049
,timeo=600,retrans=2,sec=sys,clientaddr=172.22.87.102,local_lock=none,write=eager,addr=172.22.87.
51)

172.22.87.11:/trident_ocp_afx_nfs_default_nfs_normal_1_ed374 on /nfsslow type nfs4
(rw,relatime,vers=4.1,rsize=1048576,wsiz=1048576,namlen=255,hard,proto=tcp,nconnect=16,timeo=600
,retrans=2,sec=sys,clientaddr=172.22.87.102,local_lock=none,addr=172.22.87.11)
```

5. Wait for few minutes and run the below command to check the status.

```
sh-5.1 sh-5.1# /usr/local/cuda-12.8/gds/tools/gdscheck -p
GDS release version: 1.13.1.3
nvidia_fs version: 2.27 libcufile version: 2.12
Platform: x86_64
=====
ENVIRONMENT:
=====
=====
DRIVER CONFIGURATION:
=====
NVMe P2PDMA      : Unsupported
NVMe             : Unsupported
NVMeOF           : Unsupported
SCSI             : Unsupported
ScaleFlux CSD    : Unsupported
NVMesh           : Unsupported
DDN EXAScaler    : Unsupported
IBM Spectrum Scale : Unsupported
NFS              : Supported
BeeGFS           : Unsupported
WekaFS           : Unsupported
Userspace RDMA   : Unsupported
--Mellanox PeerDirect : Enabled
--rdma library     : Not Loaded (libcufile_rdma.so)
--rdma devices     : Not configured
--rdma_device_status : Up: 0 Down: 0
=====
CUFILE CONFIGURATION:
=====
properties.use_pci_p2pdma : false
properties.use_compat_mode : true
properties.force_compat_mode : false
properties.gds_rdma_write_support : true
properties.use_poll_mode : false
properties.poll_mode_max_size_kb : 4
properties.max_batch_io_size : 128
properties.max_batch_io_timeout_msecs : 5
properties.max_direct_io_size_kb : 16384
properties.max_device_cache_size_kb : 131072
properties.max_device_pinned_mem_size_kb : 33554432
properties.posix_pool_slab_size_kb : 4 1024 16384
properties.posix_pool_slab_count : 128 64 64
properties.rdma_peer_affinity_policy : RoundRobin
properties.rdma_dynamic_routing : 0
fs.generic.posix_unaligned_writes : false
fs.lustre.posix_gds_min_kb: 0
fs.beegfs.posix_gds_min_kb: 0
fs.weka.rdma_write_support: false
fs.gpfs.gds_write_support: false
fs.gpfs.gds_async_support: true
profile.nvtx : false
profile.cufile stats : 0
miscellaneous.api_check_aggressive : false
```

```

execution.max_io_threads : 4
execution.max_io_queue_depth : 128
execution.parallel_io : true
execution.min_io_threshold_size_kb : 8192
execution.max_request_parallelism : 4
properties.force_odirect_mode : false
properties.prefer_iouring : false
=====
GPU INFO:
=====
GPU index 0 NVIDIA RTX PRO 6000 Blackwell Server Edition bar:1 bar size (MiB):131072 supports
GDS, IOMMU State: Disabled
=====
PLATFORM INFO:
=====
IOMMU: disabled
Nvidia Driver Info Status: Supported(Nvidia Open Driver Installed)
Cuda Driver Version Installed: 13000
Platform: UCSC-C240-M8SX, Arch: x86_64(Linux 5.14.0-570.92.1.el9_6.x86_64)
Platform verification succeeded
sh-5.1#

```

Note: Make sure to use the correct GPU index, which has the NUMA-local NIC connecting to the storage.

Note: Due to the PCIe topology and locality between the GPUs and CX7 NICs, one GPU provides optimal affinity for RDMA operations, whereas the other is better positioned for GPUDirect Storage (GDS) workloads.

Results

Run a sample write workload on each mount. The NFS over RDMA mount with use GDS (-x 0) and the NFS over TCP mount will use Storage to CPU to GPU (-x 2).

```

#NFSoverRDMA#
sh-5.1# /usr/local/cuda-12.8/gds/tools/gdsio -D /nfsfast -d 0 -w 32 -s 5G -i 256K -x 0 -I 1 -T 120
IoType: WRITE XferType: GPUD Threads: 32 DataSetSize: 19571456/167772160(KiB) IOSize: 256(KiB)
Throughput: 1.897050 GiB/sec, Avg_Latency: 4119.704477 usecs ops: 76451 total_time 9.838853 secs
sh-5.1#

#Normal NFS
sh-5.1# /usr/local/cuda-12.8/gds/tools/gdsio -D /nfsslow -d 0 -w 32 -s 5G -i 256K -x 2 -I 1 -T 120
IoType: WRITE XferType: CPU_GPU Threads: 32 DataSetSize: 11590656/167772160(KiB) IOSize: 256(KiB)
Throughput: 1.016415 GiB/sec, Avg_Latency: 7687.446698 usecs ops: 45276 total_time 10.875197 secs
sh-5.1#

```

Run a sample read workload on each mount. The NFS over RDMA mount with use GDS (-x 0) and the NFS over TCP mount will use Storage to CPU to GPU (-x 2).

```

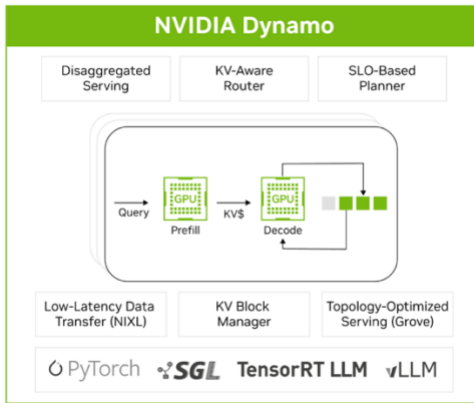
#NFSoverRDMA
sh-5.1# /usr/local/cuda-12.8/gds/tools/gdsio -D /nfsfast -d 0 -w 32 -s 5G -i 1M -x 0 -I 0 -T 120
IoType: READ XferType: GPUD Threads: 32 DataSetSize: 2151306240/167772160(KiB) IOSize: 1024(KiB)
Throughput: 17.149929 GiB/sec, Avg_Latency: 1822.433126 usecs ops: 2100885 total_time 119.629969
secssh-5.1#

#Normal NFS
sh-5.1# /usr/local/cuda-12.8/gds/tools/gdsio -D /nfsslow -d 0 -w 32 -s 5G -i 1M -x 2 -I 0 -T 120
IoType: READ XferType: CPU_GPU Threads: 32 DataSetSize: 1939303424/167772160(KiB) IOSize:
1024(KiB) Throughput: 15.524670 GiB/sec, Avg_Latency: 2014.403310 usecs ops: 1893851 total_time
119.130642 sec
ssh-5.1#

```

In both the tests, we can see that GDS/NFS0RDMA performance was better than normal NFS.

NVIDIA Dynamo



NVIDIA Dynamo is an open source, distributed inference-serving framework built to deploy models in multi-node environments at data center scale. It supports open-source inference engines—including SGLang, NVIDIA TensorRT™ LLM, and vLLM—and simplifies the complexities of distributed serving by disaggregating inference phases across different GPUs, intelligently routing requests to the appropriate GPU to avoid redundant computation and extending GPU memory through data caching to cost-effective storage tiers.

1. Follow the below steps to deploy NVIDIA Dynamo.

```
export RELEASE_VERSION=1.0.2 # Check https://github.com/ai-dynamo/dynamo/releases
export NAMESPACE=dynamo-system
export DOCKER_SERVER=nvcr.io/nvidia/ai-dynamo
export DOCKER_USERNAME='${oauthToken}'
export DOCKER_PASSWORD= <Your NGC API Key>
export HF_TOKEN=<Your Hugging Face Token>

oc new-project $NAMESPACE
```

2. Create secrets for both NGC and Hugging Face.

```
oc create secret docker-registry ngc-imagepullsecret --docker-server=${DOCKER_SERVER} --docker-username=${DOCKER_USERNAME} --docker-password=${DOCKER_PASSWORD} --namespace=${NAMESPACE}

oc create secret generic hf-token-secret --from-literal=HF_TOKEN=${HF_TOKEN} -n ${NAMESPACE}
```

3. Fetch the desired version and install it using helm.

```
helm fetch https://helm.ngc.nvidia.com/nvidia/ai-dynamo/charts/dynamo-platform-${RELEASE_VERSION}.tgz
helm install dynamo-platform dynamo-platform-${RELEASE_VERSION}.tgz --namespace ${NAMESPACE}
```

4. Verify the Dynamo CRD.

```
oc get crd | grep dynamo
dynamocheckpoints.nvidia.com                2026-04-24T10:13:04Z
dynamocomponentdeployments.nvidia.com        2026-04-24T10:13:04Z
dynamographdeploymentrequests.nvidia.com     2026-04-24T09:46:47Z
dynamographdeployments.nvidia.com            2026-04-24T10:13:05Z
dynamographdeploymentscalingadapters.nvidia.com 2026-04-24T10:13:05Z
dynamomodels.nvidia.com                     2026-04-24T10:13:05Z
dynamoworkermetadatas.nvidia.com             2026-04-24T10:13:05Z
```

5. Verify the pods status.

oc get pods -n \${NAMESPACE}				
NAME	READY	STATUS	RESTARTS	AGE
dynamo-platform-dynamo-operator-controller-manager-685c454kmz9l	1/1	Running	0	12m
dynamo-platform-nats-0	2/2	Running	0	3m48s

6. Apply agg.yaml file.

```
cat vllm-agg.yaml
apiVersion: nvidia.com/v1alpha1
kind: DynamoGraphDeployment
metadata:
  name: vllm-agg
spec:
  services:
    Frontend:
      envFromSecret: hf-token-secret
      componentType: frontend
      replicas: 1
      extraPodSpec:
        mainContainer:
          image: nvcr.io/nvidia/ai-dynamo/vllm-runtime:1.0.2
    VllmDecodeWorker:
      envFromSecret: hf-token-secret
      componentType: worker
      replicas: 1
      resources:
        limits:
          gpu: "1". #Change the GPU for the AIPerf test
        requests:
          custom:
            # Increase this value for larger models
            ephemeral-storage: "200Gi"
      extraPodSpec:
        mainContainer:
          image: nvcr.io/nvidia/ai-dynamo/vllm-runtime:1.0.2
          workingDir: /workspace/examples/backends/vllm
          command:
            - python3
            - -m
            - dynamo.vllm
          args:
            - --model
            - Qwen/Qwen3-8B
```

Note: We have used agg aggregated with vLLM. For more information on different backend types, refer [here](#).

7. Verify the pod status.

oc get pod			
NAME	READY	STATUS	RESTARTS
dynamo-platform-dynamo-operator-controller-manager-685c4544rb4c	1/1	Running	0
dynamo-platform-nats-0	2/2	Running	0
vllm-agg-frontend-6dc5f8fff-zhx48	1/1	Running	0
vllm-agg-vllmdecodeworker-20cfe8ed-8f94db7b6-8wr8l	1/1	Running	0

8. Create a route.

```
oc port-forward svc/vllm-agg-frontend 8000:8000 -n ${NAMESPACE} &
```

9. Send a query to the LLM.

```
# curl http://agg-k-dynamo-system.apps.flexpod-afx.fpmc.sa/v1/chat/completions -H "Content-Type: application/json" -d '{
  "model": "Qwen/Qwen3-8B",
  "messages": [{"role": "user", "content": "What is NVIDIA Dynamo?"}],
  "max_tokens": 200
}'

#Response
Handling connection for 8000
{"id":"chatcmpl-a49b46a6-1175-4c10-8343-4684874fe467","choices":[{"index":0,"message":{"content":"<think>\nOkay, I need to explain what NVIDIA Dynamo is. Let me start by recalling what I know. Dynamo is a software tool used for creating 3D models, right? I remember it's part of the Unity engine. So, maybe it's related to the Unity game engine? But wait, Unity is used for game development, not just 3D modeling. Hmm, maybe there's a confusion here.\n\nWait, I think there's another tool called Dynamo for 3D modeling. Let me check my memory. Yes, Dynamo is a visual programming tool used for creating 3D models, animations, and other design work. It's developed by Autodesk. So, if someone is asking about NVIDIA Dynamo, they might be referring to this specific tool. \n\nI should confirm if there's any confusion between Dynamo and other tools. For example, is there a Dynamo for Unity? Probably not, since Unity is a different engine. So, the answer would be that NVIDIA Dynamo is","role":"assistant","reasoning_content":null},"finish_reason":"length"}],"created":1777031256,"model":"Qwen/Qwen3-8B","object":"chat.completion","usage":{"prompt_tokens":13,"completion_tokens":200,"total_tokens":213},"nvext":{"timing":{"request_received_ms":1777031256627,"total_time_ms":426.19066}}}
```

NVIDIA AIPerf-LLM Metrics

This section shows how to benchmark Dynamo deployments using AIPerf, a comprehensive tool for measuring generative AI inference performance. AIPerf provides detailed metrics, real-time dashboards, and automatic visualization — you call it directly against your endpoints. To run Server-Side Benchmarking within the cluster- refer here. We ran the benchmark test by running a concurrency sweep on Qwen3-8B model. Each concurrency level sends enough requests for stable measurements (**max(c*3, 10)**):

```
MODEL="Qwen/Qwen3-8B"
URL="http://vllm-agg-frontend.dynamo-system.svc.cluster.local:8000"

for c in 50 100 200; do
  aiperf profile \
    --model "$MODEL" \
    --url "$URL" \
    --endpoint-type chat \
    --streaming \
    --concurrency $c \
    --request-count $(( c * 3 > 10 ? c * 3 : 10 )) \
    --synthetic-input-tokens-mean 2000 \
    --output-tokens-mean 256 \
    --artifact-dir "artifacts/deployment-a/c$c"
done
```

Note: Adjust concurrency levels to match your deployment's capacity. Very high concurrency on a small deployment (e.g., c250 on a single GPU) will cause server errors. Start with lower values and increase until you find the saturation point.

GPU=1 and Concurrency=50

NVIDIA AIPerf LLM Metrics							
Metric	avg	min	max	p99	p90	p50	std
Time to First Token (ms)	1,389.92	74.53	5,044.33	4,996.77	3,755.36	1,238.39	1,390.16
Time to Second Token (ms)	253.76	22.68	420.20	419.90	415.74	399.66	182.59
Time to First Output Token (ms)	1,389.92	74.53	5,044.33	4,996.77	3,755.36	1,238.39	1,390.16
Request Latency (ms)	9,938.21	6,659.06	15,365.41	14,956.04	13,703.09	10,399.04	2,695.16
Inter Token Latency (ms)	33.52	25.75	44.01	43.42	41.15	35.12	6.47
Output Token Throughput Per User (tokens/sec/user)	31.00	22.72	38.84	38.73	38.38	28.48	6.11
Output Sequence Length (tokens)	255.99	255.00	256.00	256.00	256.00	256.00	0.08
Input Sequence Length (tokens)	2,000.01	2,000.00	2,001.00	2,000.51	2,000.00	2,000.00	0.11
Output Token Throughput (tokens/sec)	1,282.69	N/A	N/A	N/A	N/A	N/A	N/A
Request Throughput (requests/sec)	5.01	N/A	N/A	N/A	N/A	N/A	N/A
Request Count (requests)	150.00	N/A	N/A	N/A	N/A	N/A	N/A

CLI Command: aiperf profile --model 'Qwen/Qwen3-8B' --url 'http://vllm-aggr-frontend.dynamo-system.svc.cluster.local:8000' --endpoint-type 'chat' --streaming --concurrency 50 --request-count 150 --synthetic-input-tokens-mean 2000 --output-tokens-mean 256 --artifact-dir 'artifacts/deployment-a/c50'

GPU=1 and Concurrency=100.

NVIDIA AIPerf LLM Metrics							
Metric	avg	min	max	p99	p90	p50	std
Time to First Token (ms)	1,904.94	111.97	10,534.62	10,525.99	7,461.76	142.72	3,052.02
Time to Second Token (ms)	165.75	28.47	456.28	445.93	430.59	52.58	174.83
Time to First Output Token (ms)	1,904.94	111.97	10,534.62	10,525.99	7,461.76	142.72	3,052.02
Request Latency (ms)	12,829.84	9,072.63	20,032.44	20,029.79	19,718.05	9,624.02	4,708.22
Inter Token Latency (ms)	42.84	35.01	72.42	72.33	62.77	37.21	10.57
Output Token Throughput Per User (tokens/sec/user)	24.45	13.81	28.56	28.53	27.83	26.87	4.52
Output Sequence Length (tokens)	256.01	256.00	257.00	256.00	256.00	256.00	0.08
Input Sequence Length (tokens)	2,000.00	2,000.00	2,000.00	2,000.00	2,000.00	2,000.00	0.00
Output Token Throughput (tokens/sec)	1,981.48	N/A	N/A	N/A	N/A	N/A	N/A
Request Throughput (requests/sec)	7.74	N/A	N/A	N/A	N/A	N/A	N/A
Request Count (requests)	300.00	N/A	N/A	N/A	N/A	N/A	N/A

CLI Command: aiperf profile --model 'Qwen/Qwen3-8B' --url 'http://vllm-aggr-frontend.dynamo-system.svc.cluster.local:8000' --endpoint-type 'chat' --streaming --concurrency 100 --request-count 300 --synthetic-input-tokens-mean 2000 --output-tokens-mean 256 --artifact-dir 'artifacts/deployment-a/c100'

GPU=1 and Concurrency=200.

NVIDIA AIPerf LLM Metrics							
Metric	avg	min	max	p99	p90	p50	std
Time to First Token (ms)	2,552.09	173.01	10,908.01	10,903.40	9,383.83	266.84	3,726.69
Time to Second Token (ms)	182.10	22.35	477.77	472.96	450.20	94.44	160.32
Time to First Output Token (ms)	2,552.09	173.01	10,908.01	10,903.40	9,383.83	266.84	3,726.69
Request Latency (ms)	15,614.00	11,294.88	22,795.73	22,784.22	22,466.69	13,034.42	4,556.72
Inter Token Latency (ms)	51.22	43.34	79.23	77.83	62.51	50.00	7.78
Output Token Throughput Per User (tokens/sec/user)	19.89	12.62	23.07	23.06	22.57	20.00	2.46
Output Sequence Length (tokens)	256.00	255.00	256.00	256.00	256.00	256.00	0.04
Input Sequence Length (tokens)	2,000.00	2,000.00	2,000.00	2,000.00	2,000.00	2,000.00	0.00
Output Token Throughput (tokens/sec)	3,255.55	N/A	N/A	N/A	N/A	N/A	N/A
Request Throughput (requests/sec)	12.72	N/A	N/A	N/A	N/A	N/A	N/A
Request Count (requests)	600.00	N/A	N/A	N/A	N/A	N/A	N/A

CLI Command: aiperf profile --model 'Qwen/Qwen3-8B' --url 'http://vllm-aggr-frontend.dynamo-system.svc.cluster.local:8000' --endpoint-type 'chat' --streaming --concurrency 200 --request-count 600 --synthetic-input-tokens-mean 2000 --output-tokens-mean 256 --artifact-dir 'artifacts/deployment-a/c200'

GPU=2 and Concurrency=50.

NVIDIA AIPerf LLM Metrics							
Metric	avg	min	max	p99	p90	p50	std
Time to First Token (ms)	1,370.80	70.94	5,010.60	5,010.18	3,857.95	1,232.49	1,386.14
Time to Second Token (ms)	257.28	18.72	417.24	416.75	413.80	401.42	181.85
Time to First Output Token (ms)	1,370.80	70.94	5,010.60	5,010.18	3,857.95	1,232.49	1,386.14
Request Latency (ms)	9,939.66	6,660.49	15,157.32	15,153.82	13,911.97	10,377.81	2,726.18
Inter Token Latency (ms)	33.60	25.80	43.70	43.55	40.81	34.37	6.53
Output Token Throughput Per User (tokens/sec/user)	30.95	22.88	38.75	38.75	38.40	29.09	6.15
Output Sequence Length (tokens)	256.02	256.00	259.00	256.00	256.00	256.00	0.24
Input Sequence Length (tokens)	2,000.00	2,000.00	2,000.00	2,000.00	2,000.00	2,000.00	0.00
Output Token Throughput (tokens/sec)	1,282.81	N/A	N/A	N/A	N/A	N/A	N/A
Request Throughput (requests/sec)	5.01	N/A	N/A	N/A	N/A	N/A	N/A
Request Count (requests)	150.00	N/A	N/A	N/A	N/A	N/A	N/A

```
CLI Command: aiperf profile --model 'Qwen/Qwen3-8B' --url 'http://vllm-agg-frontend.dynamo-system.svc.cluster.local:8000'
--endpoint-type 'chat' --streaming --concurrency 50 --request-count 150 --synthetic-input-tokens-mean 2000 --output-tokens-mean
256 --artifact-dir 'artifacts/deployment-a/c50'
```

GPU=2 and Concurrency=100.

NVIDIA AIPerf LLM Metrics							
Metric	avg	min	max	p99	p90	p50	std
Time to First Token (ms)	1,882.01	109.84	10,435.38	10,216.61	7,222.47	138.22	3,014.29
Time to Second Token (ms)	165.29	21.77	449.24	443.88	425.81	53.81	171.48
Time to First Output Token (ms)	1,882.01	109.84	10,435.38	10,216.61	7,222.47	138.22	3,014.29
Request Latency (ms)	12,793.14	9,062.75	19,940.68	19,894.40	19,579.10	9,625.55	4,653.17
Inter Token Latency (ms)	42.79	34.94	72.87	71.57	61.76	37.22	10.43
Output Token Throughput Per User (tokens/sec/user)	24.45	13.72	28.62	28.45	27.82	26.86	4.48
Output Sequence Length (tokens)	256.00	256.00	256.00	256.00	256.00	256.00	0.00
Input Sequence Length (tokens)	2,000.00	2,000.00	2,000.00	2,000.00	2,000.00	2,000.00	0.00
Output Token Throughput (tokens/sec)	1,986.67	N/A	N/A	N/A	N/A	N/A	N/A
Request Throughput (requests/sec)	7.76	N/A	N/A	N/A	N/A	N/A	N/A
Request Count (requests)	300.00	N/A	N/A	N/A	N/A	N/A	N/A

```
CLI Command: aiperf profile --model 'Qwen/Qwen3-8B' --url 'http://vllm-agg-frontend.dynamo-system.svc.cluster.local:8000'
--endpoint-type 'chat' --streaming --concurrency 100 --request-count 300 --synthetic-input-tokens-mean 2000 --output-tokens-mean
256 --artifact-dir 'artifacts/deployment-a/c100'
```

GPU=2 and Concurrency=200.

NVIDIA AIPerf LLM Metrics							
Metric	avg	min	max	p99	p90	p50	std
Time to First Token (ms)	2,566.82	176.25	10,701.62	10,697.87	9,823.74	273.49	3,762.81
Time to Second Token (ms)	170.22	24.44	469.85	464.95	438.37	92.11	155.99
Time to First Output Token (ms)	2,566.82	176.25	10,701.62	10,697.87	9,823.74	273.49	3,762.81
Request Latency (ms)	15,725.03	11,574.72	22,771.41	22,763.57	22,577.40	13,042.42	4,499.55
Inter Token Latency (ms)	51.60	44.19	79.55	77.88	63.97	50.03	7.64
Output Token Throughput Per User (tokens/sec/user)	19.72	12.57	22.63	22.59	22.13	19.99	2.33
Output Sequence Length (tokens)	256.00	256.00	256.00	256.00	256.00	256.00	0.00
Input Sequence Length (tokens)	2,000.00	2,000.00	2,000.00	2,000.00	2,000.00	2,000.00	0.00
Output Token Throughput (tokens/sec)	3,230.60	N/A	N/A	N/A	N/A	N/A	N/A
Request Throughput (requests/sec)	12.62	N/A	N/A	N/A	N/A	N/A	N/A
Request Count (requests)	600.00	N/A	N/A	N/A	N/A	N/A	N/A

```
CLI Command: aiperf profile --model 'Qwen/Qwen3-8B' --url 'http://vllm-agg-frontend.dynamo-system.svc.cluster.local:8000'
--endpoint-type 'chat' --streaming --concurrency 200 --request-count 600 --synthetic-input-tokens-mean 2000 --output-tokens-mean
256 --artifact-dir 'artifacts/deployment-a/c200'
```

GPU=4 and Concurrency=50.

NVIDIA AIPerf LLM Metrics							
Metric	avg	min	max	p99	p90	p50	std
Time to First Token (ms)	1,087.03	58.09	2,498.22	2,496.37	1,857.35	966.97	618.82
Time to Second Token (ms)	286.16	15.13	410.26	409.98	405.98	394.78	145.20
Time to First Output Token (ms)	1,087.03	58.09	2,498.22	2,496.37	1,857.35	966.97	618.82
Request Latency (ms)	7,010.08	5,396.75	8,580.14	8,578.60	7,762.81	7,335.34	713.95
Inter Token Latency (ms)	23.23	19.27	28.32	27.96	26.39	23.04	2.18
Output Token Throughput Per User (tokens/sec/user)	43.43	35.31	51.90	51.84	48.89	43.40	4.05
Output Sequence Length (tokens)	255.97	253.00	256.00	256.00	256.00	256.00	0.26
Input Sequence Length (tokens)	2,000.00	2,000.00	2,000.00	2,000.00	2,000.00	2,000.00	0.00
Output Token Throughput (tokens/sec)	1,813.60	N/A	N/A	N/A	N/A	N/A	N/A
Request Throughput (requests/sec)	7.09	N/A	N/A	N/A	N/A	N/A	N/A
Request Count (requests)	150.00	N/A	N/A	N/A	N/A	N/A	N/A

CLI Command: aiperf profile --model 'Qwen/Qwen3-8B' --url 'http://vllm-aggr-frontend.dynamo-system.svc.cluster.local:8000' --endpoint-type 'chat' --streaming --concurrency 50 --request-count 150 --synthetic-input-tokens-mean 2000 --output-tokens-mean 256 --artifact-dir 'artifacts/deployment-a/c50'

GPU=4 and Concurrency=100.

NVIDIA AIPerf LLM Metrics							
Metric	avg	min	max	p99	p90	p50	std
Time to First Token (ms)	1,223.06	60.96	5,121.38	5,092.39	3,826.21	577.97	1,409.51
Time to Second Token (ms)	207.08	18.63	429.60	423.12	413.49	205.58	161.67
Time to First Output Token (ms)	1,223.06	60.96	5,121.38	5,092.39	3,826.21	577.97	1,409.51
Request Latency (ms)	9,414.08	6,508.63	13,540.92	13,530.85	12,525.73	8,844.71	2,108.56
Inter Token Latency (ms)	32.12	25.14	44.16	42.71	38.39	31.95	4.48
Output Token Throughput Per User (tokens/sec/user)	31.71	22.64	39.78	39.40	36.55	31.29	4.22
Output Sequence Length (tokens)	256.00	256.00	256.00	256.00	256.00	256.00	0.00
Input Sequence Length (tokens)	2,000.00	2,000.00	2,000.00	2,000.00	2,000.00	2,000.00	0.00
Output Token Throughput (tokens/sec)	2,692.51	N/A	N/A	N/A	N/A	N/A	N/A
Request Throughput (requests/sec)	10.52	N/A	N/A	N/A	N/A	N/A	N/A
Request Count (requests)	300.00	N/A	N/A	N/A	N/A	N/A	N/A

CLI Command: aiperf profile --model 'Qwen/Qwen3-8B' --url 'http://vllm-aggr-frontend.dynamo-system.svc.cluster.local:8000' --endpoint-type 'chat' --streaming --concurrency 100 --request-count 300 --synthetic-input-tokens-mean 2000 --output-tokens-mean 256 --artifact-dir 'artifacts/deployment-a/c100'

GPU=4 and Concurrency=200.

NVIDIA AIPerf LLM Metrics							
Metric	avg	min	max	p99	p90	p50	std
Time to First Token (ms)	1,743.41	109.20	7,714.22	7,706.27	6,264.71	437.82	2,377.49
Time to Second Token (ms)	191.52	17.61	457.05	445.13	426.36	144.99	159.50
Time to First Output Token (ms)	1,743.41	109.20	7,714.22	7,706.27	6,264.71	437.82	2,377.49
Request Latency (ms)	11,784.28	8,141.24	17,965.65	17,959.66	17,230.09	9,915.15	3,487.17
Inter Token Latency (ms)	39.38	31.39	58.29	56.86	50.73	37.47	6.61
Output Token Throughput Per User (tokens/sec/user)	26.03	17.16	31.86	31.70	30.36	26.69	3.82
Output Sequence Length (tokens)	256.00	255.00	256.00	256.00	256.00	256.00	0.04
Input Sequence Length (tokens)	2,000.00	2,000.00	2,000.00	2,000.00	2,000.00	2,000.00	0.00
Output Token Throughput (tokens/sec)	4,306.14	N/A	N/A	N/A	N/A	N/A	N/A
Request Throughput (requests/sec)	16.82	N/A	N/A	N/A	N/A	N/A	N/A
Request Count (requests)	600.00	N/A	N/A	N/A	N/A	N/A	N/A

CLI Command: aiperf profile --model 'Qwen/Qwen3-8B' --url 'http://vllm-aggr-frontend.dynamo-system.svc.cluster.local:8000' --endpoint-type 'chat' --streaming --concurrency 200 --request-count 600 --synthetic-input-tokens-mean 2000 --output-tokens-mean 256 --artifact-dir 'artifacts/deployment-a/c200'

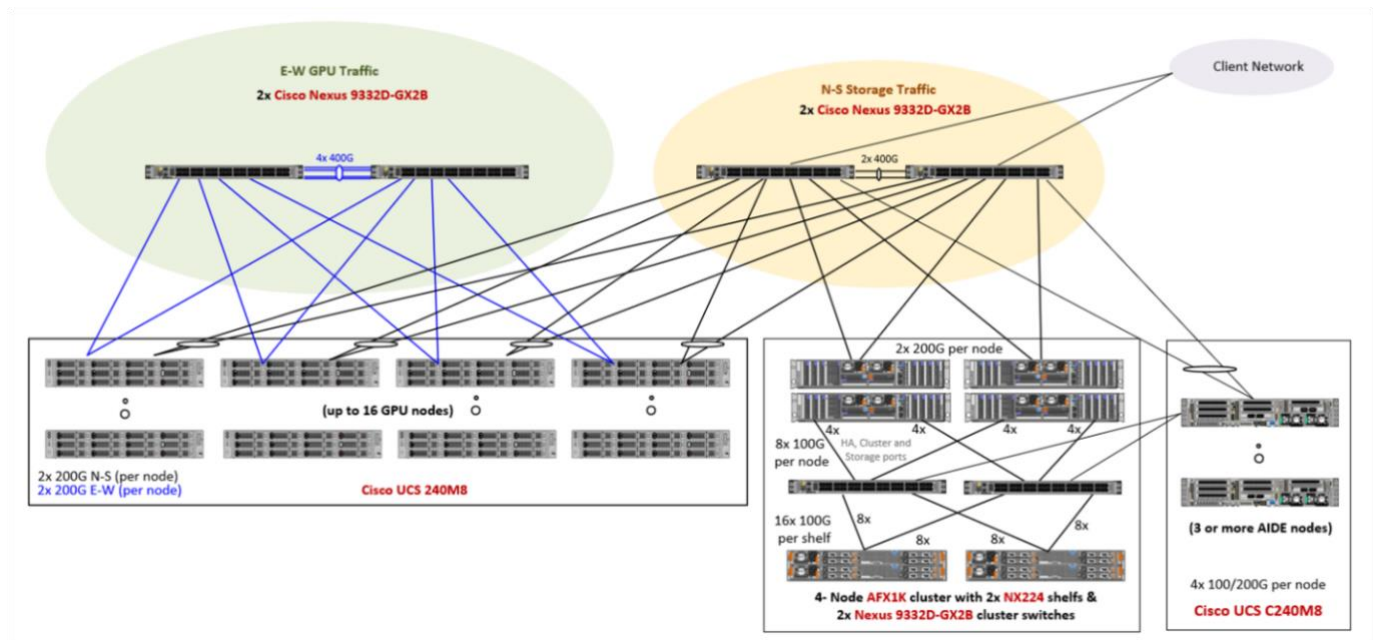
Solution Scaling Guidance

The Solution scaling is aligned to the NVIDIA Enterprise Reference Architecture (ERA) for MGX and HGX GPU servers with the focus on repeatable building blocks that can expand from a “single pod” to “multi-pod”, “multi-cluster” environments without redesign. Starting with validated MGX/HGX configurations for compute density, power/cooling, and high-speed networking, the approach standardizes Cisco fabric design (bandwidth, latency, segmentation, and resiliency) and integrates NetApp data management and data services to deliver predictable performance, high availability, and efficient data movement for training and inference workflows. As requirements grow, capacity and throughput scale by adding GPU nodes and fabric ports, and by scaling storage performance and namespace capabilities while preserving consistent operational practices, security controls, and observability enabling faster time to value and lower risk as the environment evolves.

- Inference Only - Cisco UCS C240 M8 Rack servers with NVIDIA RTX Pro 6000 GPUs and ConnectX-7 NICs for both E/W and N/S traffic
- Large Inference & Fine Tuning - Cisco UCS C845A M8 Rack servers with NVIDIA H200 GPUs and ConnectX-7 NIC for N/S and 2x NVIDIA BlueField-3 SuperNICs for E/W traffic aligned with 2-4-3-200 architecture
- Large Inference & Training - Cisco UCS C885A M8 Rack Servers with high-density NVIDIA HGX H200 GPUs and ConnectX-7 NIC, aligned with NVIDIA’s HGX-based 2-8-5-4200 architecture.

Cisco UCS C240 M8 configuration - AI Acceleration

Figure 8) Cisco UCS C240 M8 configuration



This can scale from 1 SU (Scale Unit) up to 4 SUs in a cluster. (Figure 8) [Error! Reference source not found.](#) shows storage scaling requirements for up to four scale units and 32 GPUs. Each scalable unit provides the following connectivity components: Table 7 below shows the individual components needed during scaling:

- For the E/W fabric: On 4 servers (SU), each equipped with ConnectX-7 NIC (2x 200GbE ports per server), providing a total of 8x 200Gb/s connections and an aggregate bandwidth of 1.6Tb/s
- For the N/S fabric: On 4 servers (SU), each equipped with ConnectX-7 NIC (2x 200GbE ports per server), providing a total of 8x 200Gb/s connections and an aggregate bandwidth of 1.6Tb/s
- For the storage side: Each NetApp AFX 1K node will be equipped with 2x X50131B dual port adapter. Eight storage ports are used for this configuration to connect to the frontend switches, each providing 200Gb/s bandwidth
- One pair of Cisco Nexus 9332D-GX2B 400GbE switches are used for E/W backend (B/E) connectivity and another pair of switches were used for N/S frontend (F/E) connectivity
- The filesystem mounts should be using NFSoRDMA and pNFS should be configured on the NetApp storage which provides consistent latency as solution is scaled
- The backend switches are configured with 4x 400GbE for ISL links and carries the backend VLAN. The C240 M8 servers connects to the backend switches over non-vPC links
- The front-end switches are configured as vPC pairs. 2x 200GbE links are used for vPC peer-link. The C240 M8 servers connect to the front-end switches over vPC. The storage controllers also connect to the frontend switches over non-vPC links

Table 7) Scaling on C240 MGX configuration

Nodes C240 M8	GPU	Storage Nodes - AFX 1K	Server Ports N/S+E/W 200GbE	Storage Ports 200GbE	Total switch ports N/S+E/W 400GbE	QDD-400G- SR8-S Transceivers	QSFP-200G- SR4-S Transceivers	Cables [MPO16 - 2 * MPO12]	Converged in E-W Nexus 9332D-GX2B	
									B/E Switch (E/W)	F/E Switch (N/S)
4	8	4	16	8	12	12	24	12	2	2
8	16	4	32	8	20	20	40	20	2	2
12	24	4	48	8	28	28	56	28	2	2
16	32	4	64	8	36	36	72	36	2	2

Cisco UCS C845A M8 Rack Server in 2-8-5-400 Configuration

A PCIe optimized NVIDIA certified compute node supporting eight GPUs (RTX Pro 6000, H200 NVL), five network adapters (1x NVIDIA ConnectX®-7 NIC for N/S and 4x NVIDIA BlueField®-3 SuperNICs), and two CPUs. This can scale from 1 SU up to 8 SUs in a cluster as shown in [Figure 9](#)

Figure 9) Scaling with C845A M8 in MGX up to 32 Nodes

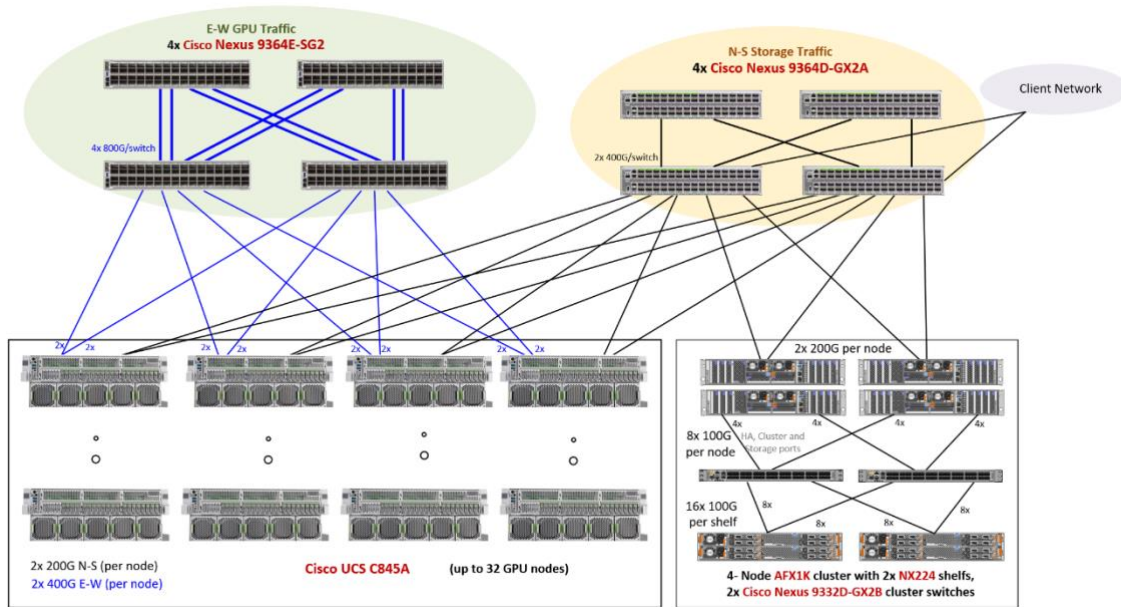


Table 8 shows storage scaling requirements for up to eight scale units and 256 GPUs. Each scalable unit provides the following connectivity components:

- For the E/W fabric: On 4 servers (SU), each equipped with 4x B3140H BlueField-3 SuperNICs (1x 400GbE port), providing a total of 16x 400Gb/s connections and an aggregate bandwidth of 6.4Tb/s
- For the Converged N/S fabric for storage: On 4 servers, each equipped with 1 dual port ConnectX-7 NIC (2x 200GbE ports per server), providing a total of 8x 200Gb/s connections and an aggregate bandwidth of 1.6Tb/s
- Cisco Nexus 9364E-SG2-O is used as spine and leaf for backend network. The front-end fabric uses Cisco Nexus 9364D-GX2A switches for spine and leaf
- For the storage side: Each NetApp AFX 1K node will be equipped with 2x X50131B dual port adapter

Table 8) MGX with Cisco UCS C845A M8 in 2-8-5-400 listing E/W Topology

Nodes Cisco C845 M8	GPUS	QDD-400G- DR4-S on the GPU node (400GbE)	Total switch ports E/W (800GbE)	OSFP-800G-DR8 (leaf + spine)	CB-M12-M12- SMF MPO-12 Cable	E/W Networking 9364E-SG2	
						Leaf	Spine
4	32	16	8	24	32	2	2
8	64	32	16	48	64	2	2
12	96	48	24	72	96	2	2
16	128	64	32	96	128	2	2
20	160	80	40	120	160	2	2
24	192	96	48	144	192	2	2
28	224	112	56	168	224	4	2
32	256	128	64	194	258	4	2

Table 9) MGX with UCS C845A M8 in 2-8-5-400 listing N/S Topology

GPU Nodes	GPU	Storage Nodes - AFX 1K	Server Ports N/S (200GbE)	Storage Ports (200GbE)	Total switch ports N/S (400GbE)	QSFP-200G-SR4-S	Cables [MPO16 - 2 * MPO12]	Transceivers for leaf and spine QDD-400G-SR8-S	Leaf to Spine Cables MPO16	N/S Networking 9364D-GX2A	
										Leaf	Spine
4	16	4	8	8	8	16	8	16	4	2	2
8	32	4	16	8	12	24	12	20	4	2	2
12	48	4	24	8	16	32	16	24	4	2	2
16	64	4	32	8	20	40	20	28	4	2	2
20	80	4	40	8	24	48	24	32	4	2	2
24	96	4	48	8	28	56	28	36	4	2	2
28	112	4	56	8	32	64	32	40	4	2	2
32	128	4	64	8	36	72	36	44	4	2	2

- Eight storage ports are used for 2-8-5-400 configuration, each port providing 200Gb/s bandwidth
- The GPU nodes and storage controllers connect to the front-end and backend switches over non-vPC links
- The filesystem mounts should be using NFSoRDMA and pNFS should be configured on the NetApp storage which provides consistent latency as solution is scaled
- [Table 9](#) shows the details of N/S topology which covers the compute to network to storage connections
- The bandwidth between the spine and leaf must equal the bandwidth coming into the leaf from the servers to have a fully non-blocking backend network

Cisco UCS C885A M8 Rack Server in 2-8-9-400 Configuration

2-8-9-400 is a [SXM](#) form factor configuration supporting eight GPUs, nine network adapters (1x NVIDIA ConnectX-7 NIC and 8 * BlueField-3 SuperNICs), and two CPUs. It scales from 1 SU to 8 SUs in a cluster. The topology of the 2-8-9-400 is shown in [Figure 10](#)

Figure 10) Scaling with C885A M8 in HGX up to 32 Nodes

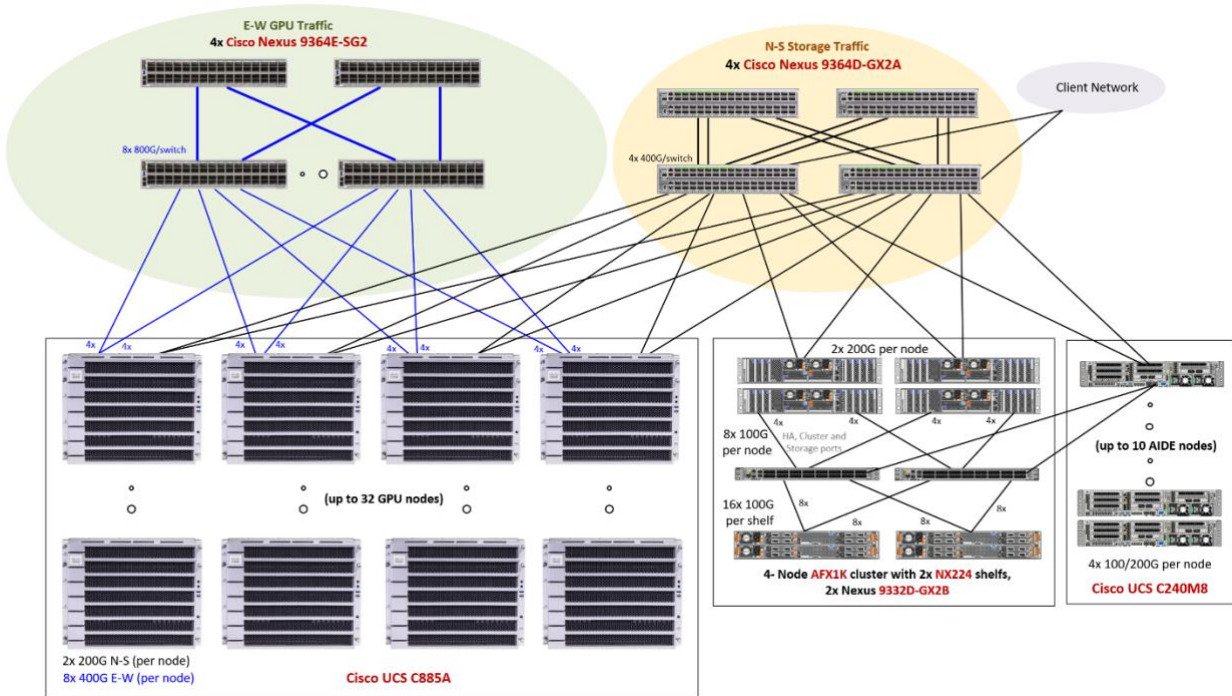


Table 10 shows storage scaling for up to eight scale units and 256 GPUs. Each scalable unit provides the following connectivity building blocks:

- For the E/W fabric: On 4 servers (SU), each equipped with 8x B3140H BlueField-3 SuperNICs (1x 400GbE port), providing a total of 32x 400Gb/s connections and an aggregate bandwidth of 12.8Tb/s
- For the Converged N/S fabric for storage: On 4 servers, each equipped with 1 dual port ConnectX-7 NIC (2x 200GbE ports per server), providing a total of 8x 200Gb/s connections and an aggregate bandwidth of 1.6Tb/s
- For the storage side: Each NetApp AFX 1K node will be equipped with 2x X50131B dual port adapter
- Cisco Nexus 9364E-SG2-O is used as spine and leaf for backend network. The front-end fabric uses Cisco Nexus 9364D-GX2A switches for spine and leaf
- The GPU nodes and storage controllers connect to the front-end and backend switches over non-vPC links
- The filesystem mounts should be using NFSoRDMA and pNFS should be configured on the NetApp storage which provides consistent latency as solution is scaled
- Table 11 shows the details of N/S topology which covers the compute to network to storage connections
- The bandwidth between the spine and leaf must equal the bandwidth coming into the leaf from the servers to have a fully non-blocking backend network

Table 10) HGX with Cisco UCS C885A M8 in 2-8-9-400 listing E/W Topology

Nodes Cisco C885A M8	GPU	Server Ports E/W (400GbE)	Total server facing switch ports (800GbE)	OSFP- 800G-DR8 OSFP (spine + leaf)	CB-M12-M12- SMF MPO-12 Cable	QDD-400G- DR4-S on the GPU node	E/W Networking 9364E-SG2	
							Leaf	Spine
4	32	32	16	48	64	32	2	2
8	64	64	32	96	128	64	2	2
12	96	96	48	144	196	96	2	2
16	128	128	64	192	256	128	4	2
20	160	160	80	240	320	160	4	2
24	192	192	96	288	384	192	4	2
28	224	224	112	336	448	224	4	2
32	256	256	128	384	512	256	6	2

Table 11) HGX with Cisco UCS C885A M8 in 2-8-9-400 listing N/S Topology

GPU Node s	GPU	Storage Nodes - AFX 1K	Server Ports N/S (200GbE)	Storage Ports (200Gb E)	Total switch ports N/S (400GbE)	QSFP-200G- SR4-S Transceivers for GPU node	Cables [MPO16 - 2 * MPO12	QDD-400G- SR8-S Transceivers (leaf + spine)	Leaf to Spine Cables MPO16	N/S Networking 9364D-GX2A	
										Leaf	Spine
4	32	4	8	8	8	16	8	24	8	2	2
8	64	4	16	8	12	24	12	28	8	2	2
12	96	4	24	8	16	32	16	32	8	2	2
16	128	4	32	8	20	40	20	36	8	2	2
20	160	4	40	8	24	48	24	40	8	2	2
24	192	4	48	8	28	56	28	44	8	2	2
28	224	4	56	8	32	64	32	48	8	2	2
32	256	4	64	8	36	72	36	52	8	2	2

Licensing

Licensing information for the various components of the solution stack is available here:

- [Cisco Intersight](#)
- [Cisco Nexus NX-OS licensing options](#)
- [NVIDIA licensing](#)
- [Red Hat OpenShift licensing](#)
- [ONTAP One Licensing](#)
- [AIDE Licensing](#)
- [NVIDIA Spectrum Licensing](#)

Conclusion

This NetApp Verified Architecture demonstrates how enterprises can confidently move from AI experimentation to production-grade AI factories using a secure, scalable, and validated FlexPod® AI platform. By combining Cisco AI-accelerated servers based on NVIDIA MGX and HGX designs, high-bandwidth Cisco networking, and NetApp® AFX storage and NetApp AIDE, the solution delivers the performance, agility, and governance required for modern AI workloads across training, RAG pipelines, and inference.

The architecture has been validated end-to-end with real-world benchmarks, advanced networking (RDMA and GPUDirect Storage), and production-class inferencing workflows on Red Hat® OpenShift, proving its ability to scale predictably while maintaining operational consistency and data protection. With a modular AI POD based design and repeatable building blocks, this solution enables customers to start small and expand seamlessly as AI demands grow reducing risk, accelerating time to value, and establishing a future-ready foundation for enterprise AI innovation. The same infrastructure can be extended to support OpenShift Virtualization for the co-existence of containers and Virtual machines.

Acknowledgments

Dr. Neto from Brazil, Eric Lewandoski, Joe Scott, Chris Gebhardt, Justin Parisi, Reese Lloyd – NetApp
 John George, Bill Shields, Chris O'Brien, Archana Sharma – Cisco
 Andy Siegel, John Fragalla - NVIDIA

Bill of Materials

Per Cisco C240 M8

Part Number	Description	Qty
UCS-M8-MLB	UCS M8 RACK MLB	1
UCSC-C240-M8SX	UCS C240 M8 Rack w/o CPU, mem, drives, 2U w SFF backplane	1
ISM-MANAGED	Deployment mode for C Series Servers in Standalone mode	1
UCSC-GPUAD-240M8	GPU AIR DUCT FOR C240M8	1
UCS-TPM-002D-D	TPM 2.0 TCG FIPS140-2 CC+ Cert M7 Intel MSW2022 Compliant	1
UCSC-RAIL-D	Ball Bearing Rail Kit for C220 & C240 M7/M8 rack servers	1
UCSC-BZL-C240-D	C240 M7/M8 Security Bezel	1
N20-BKVM-D	KVM local IO cable for UCS servers console port	1
CIMC-LATEST-D	IMC SW (Recommended) latest release for C-Series Servers.	1
UCSC-HSLP-C220M8	Heatsink for C220M8, C240M8L and C240M8 w/GPU	2
UCS-DDR5-BLK	UCS DDR5 DIMM Blanks	16
UCSC-BBLKD-M7	UCS C-Series M7 SFF drive blanking panel	24
CBL-NVME-240M8-P4	C240M8SFF NVMe cable for drive 1-4 (P4)	1
CBL-NVME-240M8-P2	C240M8SFF NVMe cable for drive 21-24 (P2)	1
UCS-CPU-I6530P	Intel I6530P 2.3GHz/225W 32C/144MB DDR5 6400MT/s	2
UCSC-PSU1-2300W-D	Cisco UCS 2300W AC Power Supply for Rack Servers Titanium	2
CAB-C19-CBN	Cabinet Jumper Power Cord, 250 VAC 16A, C20-C19 Connectors	2
UCSC-M-V5Q50GV2-D	Cisco VIC 15427 4x 10/25/50G mLOM C-Series w/Secure Boot	1
UCSC-M2RR-240M8	C240M8 Boot Raid Controller for Hot-swap M.2s by riser 3	1
UCS-M2-960G-D	960GB M.2 SATA Micron G2 SSD	1
UCS-MRX48G1RF5	48GB DDR5-6400 RDIMM 1Rx4 (24Gb)	16
UCSC-RIS1C-240M8	UCS C240 M8 Riser 1C PCIe Gen5 (2x16)	1
UCSC-RIS2C-240M8	UCS C240 M8 Riser 2C PCIe Gen5 (2x16)(CPU2)	1
UCSC-RIS3C-240M8	UCS C240 M8 Riser 3C PCIe Gen5 (x16)(CPU2)	1
UCSC-P-N7D200GFO	NVIDIA OEM MCX755106AS-HEAT 2x200GbE QSFP112 PCIe Gen5 NIC	2
UCSC-RISAB-24XM7	UCS C-Series M7 2U Air Blocker GPU only	2
RTX Pro 6000		
CAI-GPU-RTXP6000=	NVIDIA RTX Pro 6000: 600W, 96GB, 2-Slot FHFL GPU	2
NVIDIA-LIC-OPTOUT	NVIDIA SW LICENSE OPTOUT	2
GPU Kit		

CBL-G5GPU-C240M7=	C240M7 PCIe CEM compliant 12VHPWR power cable(upto 450W)	2
-------------------	--	---

Network Switches – Client Connectivity to AFX, 4 switches

Part Number	Description	Qty
N9K-C9332D-GX2B	Nexus 9300 Series, 32p 400G Switch	4
MODE-NXOS	Mode selection between ACI and NXOS	4
NXK-AF-PE	Dummy PID for Airflow Selection Port-side Exhaust	4
NXOS-CS-10.5.3F	Nexus 9300, 9500, 9800 NX-OS SW 10.5.3 (64bit) Cisco Silicon	4
NXK-ACC-KIT-1RU	Nexus 3K/9K Fixed Accessory Kit, 1RU front and rear removal	4
NXA-SFAN-35CFM-PE	Nexus Fan, 35CFM, port side exhaust airflow /w EEPROM	24
NXA-PAC-1500W-PE	Nexus 1500W PSU port-side Exhaust	8
CAB-C15-CBN	Cabinet Jumper Power Cord, 250 VAC 13A, C14-C15 Connectors	8
DCN-OTHER	Select if this product will NOT be used for AI Applications	4
NXOS-SLP-INFO-9K	Info PID for Smart Licensing using Policy for N9K	4

AFX 1K + Cluster Backend switches

Part Number	Description	Qty
AFX-1K-100-C	AFX 1K Node,-C	4
SW-ONTAP-ONE-LIC-AFX-E-C	SW,ONTAP One License, AFX, Encryption,-C	4
X50200A-C	Power Supply,2000W Titanium,-C	8
X50300A-C	Blanking Panel, IO Module	12
HA-MISC-HW	HA-MISC-HW	2
X50130B-C	IO Module,2PT,100GbE,B,-C	16
X50131B-C	IO Module,2PT,100/200/400GbE,B,-C	8
X66310A-5-C	Cable,Q-DD-4xQSFP,SMF,MPO-12 APC-4xLC-LC,5M-C	8
X66320A-5-C	Cable,400Gb,SMF,MPO-12 APC to MPO-12 APC,5m-C	2
X66200-2-N-C	Cable,Cntrlr-Switch OM4,MPO/MPO,2m,-C	16
X50210A-C	QSFP,100GBASE-DR1,LC-LC,SMF,500m,-C	32
X50220A-C	QSFP-DD,400GBASE-DR4,MPO-12,SMF, 500m,-C	12
X50205A-C	QSFP,Optical,Transceiver,200GbE,SR,-C	16
X-02659-00-N-C	Rail Kit,4-Post,Rnd/Sq-Hole,Adj,24-32,-C	4
X800-42U-R6-C	Jumper Crd,In-Cab,C13-C14,-C	12
SVC-G1A-ADVISOR	Advisor, HW Support	1
SSP-G1A-ADVISOR	Advisor, SW Support	1
CS-4HR-REPLACEMENT-A	4hr Parts Replacement	1
PS-DEPLOY-STD-ONTAP-H	PS Deployment ONTAP, Standard, High	1
PS-DEPLOY-ADV-ONTAP-H	PS Deployment ONTAP, Advanced, High	1
CS-SOL-FIELD-FLEXPOD	Solution Field ID, FlexPod	1
NX224	Shelf	
NX224-0-24-A-C	NVMe SHLF,2U24,Empty,NSM140,-C	2
X4035A-2-A-C	Drive Pack 2X7.6TB,NVMe4,SED,-C	24
SW-ONTAP-ONE-1K-C	SW,ONTAP One NX,Per TB,1K,-C	364
X60135A-C	IO Module,Encr,2P,40G/100G,QSFP,-C	16
X66310A-5-C	Cable,Q-DD-4xQSFP,SMF,MPO-12 APC-4xLC-LC,5M-C	8
X50210A-C	QSFP,100GBASE-DR1,LC-LC,SMF,500m,-C	32
X50220A-C	QSFP-DD,400GBASE-DR4,MPO-12,SMF, 500m,-C	8
X5532A-N-C	Rail,4-Post,Thin,Rnd/Sq-Hole,Sm,Adj,24-32,-C	2
X50200A-C	Power Supply,2000W Titanium,-C	4
X800-42U-R6-C	Jumper Crd,In-Cab,C13-C14,-C	4
CS-SOL-FIELD-FLEXPOD	Solution Field ID, FlexPod	1
CS-G1A-SE-ADVISOR	SupportEdge Advisor	1
SVC-G1A-ADVISOR	Advisor, HW Support	1
SSP-G1A-ADVISOR	Advisor, SW Support	1

CS-4HR-REPLACEMENT-A	4hr Parts Replacement	1
Cisco 9332	400G Storage Backend Switches	
SVC-CISCO-ENTITLEMENT	Entitlement, Cisco Switch NOW Site	2
X190300-CS-PE	Switch, Cisco C9332D 32Pt 400G,PTSX	2
SVCVN-OSP-N9KC9D3X-3-5	Service X190300,C9332D Onsite 24x7x4hr,3-5Yr	2
DOC-CISCO-CONTRACT	Cisco Switch Service Contract	2

Additional Information

To learn more about the information that is described in this document, review the following documents and/or websites:

NetApp Documentation

- [NetApp Documentation](#)
- [NetApp AI Solutions Documentation](#)
- [NetApp Install and Maintain AFX Systems](#)
- [NFS over RDMA](#)
- [What is pNFS](#)
- [NetApp Trident](#)
- [NetApp Interoperability Matrix](#)

Cisco UCS M8 Rack Servers & Management Software

- [Cisco UCS C885A M8 Rack Server Data Sheet](#)
- [Cisco UCS C845A M8 Rack Server Spec Sheet](#)
- [Cisco UCS C240 M8 Rack Server Spec Sheet](#)
- [Cisco Intersight Data Sheet](#)
- [Intersight At-a-Glance](#)
- [Cisco UCS C845A M8 Rack Server at a Glance](#)
- [Cisco UCS C885A M8 Rack Server at a Glance](#)

Cisco Nexus Switches

- [Cisco Nexus 9332D-GX2B and Nexus 9364D-GX2A Switch Data Sheet](#)
- [Cisco Nexus 9364E-SG2 Switch Data Sheet](#)
- [Nexus Dashboard, Release 4.2.x User Content](#)
- [Cisco Nexus Dashboard Data Sheet](#)
- [Cisco Data Center Networking \(DCN\) Licensing Ordering Guide](#)

NVIDIA Documentation

- [NVIDIA NGC Catalog](#)

Red Hat OpenShift

- [Red Hat OpenShift Operator](#)
- [Red Hat OpenShift AI-Self Managed](#)

Transceiver

To identify the supported transceivers for any component in the Cisco AI POD stack, including interoperability across components, refer to Cisco's Transceiver Matrix Group site. The links below include the main site and the relevant subsites.

- [Cisco Optics-to-Device Compatibility Matrix](#)
- [Cisco Optics Product Information](#)
- [Cisco Optics Selector](#)
- [Transceiver data sheets](#)

Version history

As an option, use the NetApp Table style to create a Version History table. Do not add a table number or caption.

Version	Date	Document version history
Version 1.0	May 2026	Initial release

Refer to the [Interoperability Matrix Tool \(IMT\)](#) on the NetApp Support site to validate that the exact product and feature versions described in this document are supported for your specific environment. The NetApp IMT defines the product components and versions that can be used to construct configurations that are supported by NetApp. Specific results depend on each customer's installation in accordance with published specifications.

Copyright information

Copyright © 2026 NetApp, Inc. All rights reserved. Printed in the U.S. No part of this document covered by copyright may be reproduced in any form or by any means—graphic, electronic, or mechanical, including photocopying, recording, taping, or storage in an electronic retrieval system—without prior written permission of the copyright owner.

Software derived from copyrighted NetApp material is subject to the following license and disclaimer:

THIS SOFTWARE IS PROVIDED BY NETAPP "AS IS" AND WITHOUT ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, WHICH ARE HEREBY DISCLAIMED. IN NO EVENT SHALL NETAPP BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

NetApp reserves the right to change any products described herein at any time, and without notice. NetApp assumes no responsibility or liability arising from the use of products described herein, except as expressly agreed to in writing by NetApp. The use or purchase of this product does not convey a license under any patent rights, trademark rights, or any other intellectual property rights of NetApp.

The product described in this manual may be protected by one or more U.S. patents, foreign patents, or pending applications.

LIMITED RIGHTS LEGEND: Use, duplication, or disclosure by the government is subject to restrictions as set forth in subparagraph (b)(3) of the Rights in Technical Data Non-commercial Items at DFARS 252.227-7013 (FEB 2014) and FAR 52.227-19 (DEC 2007).

Data contained herein pertains to a commercial product and/or commercial service (as defined in FAR 2.101) and is proprietary to NetApp, Inc. All NetApp technical data and computer software provided under this Agreement is commercial in nature and developed solely at private expense. The U.S. Government has a non-exclusive, non-transferrable, non-sublicensable, worldwide, limited irrevocable license to use the Data only in connection with and in support of the U.S. Government contract under which the Data was delivered. Except as provided herein, the Data may not be used, disclosed, reproduced, modified, performed, or displayed without the prior written approval of NetApp, Inc. United States Government license rights for the Department of Defense are limited to those rights identified in DFARS clause 252.227-7015(b) (FEB 2014).

Trademark information

NETAPP, the NETAPP logo, and the marks listed at <http://www.netapp.com/TM> are trademarks of NetApp, Inc. Other company and product names may be trademarks of their respective owners.