Multi-Tenancy In AI Clouds

Network Isolation with Netris®

Whitepaper

Abstract

This paper presents an architectural model for implementing hard multi-tenancy in AI factory environments built on NVIDIA® DGX™ systems. It outlines how enterprises and NVIDIA Cloud Partners (NCPs) can establish consistent tenant isolation across heterogeneous connectivity domains, including East–West fabrics based on NVIDIA Spectrum-X™ or InfiniBand™, Ethernet-based North–South fabrics, NVLink® Multi-Node rack-scale fabrics, and optional DPU-augmented domains.

The document describes an architectural model in which tenant intent is defined once and consistently applied across heterogeneous fabrics. It illustrates how Netris® orchestrates tenant-related network configuration within this model using supported NVIDIA management APIs. The paper examines the technical considerations involved in aligning Ethernet VRFs, InfiniBand PKeys, and NVLink partition boundaries; coordinating lifecycle updates across fabrics; and maintaining operational consistency through topology-aware validation and configuration verification.

This guidance is intended for CTOs, AI/Cloud architects, NCP operators, and enterprise infrastructure teams deploying shared DGX-based clusters that require predictable tenant isolation and consistent network behavior.

4701 Patrick Henry Drive, Bldg. 25
Santa Clara, California, 95054, USA

Copyright Notice, Authorization to Use and Disclaimers

Document History

Version	Date	Authors	Description of Change
1.0	02/20/2026	Alex Saroyan (Netris) Andrey Khomyakov (Netris)	Initial Version

Executive Summary

Netris is the coordination layer that enables hard multi-tenancy in NVIDIA DGX-based AI factories by translating tenant intent into consistent isolation boundaries across Ethernet, InfiniBand, NVLink multi-node, and DPU-enabled networking domains.

In this architecture, tenant intent is defined once using a high-level abstraction and translated into the isolation constructs required by each fabric, including Ethernet VRFs/VXLANs, InfiniBand partitions (PKeys), and NVLink GPU partitions.

Netris implements this model by automatically generating and applying configuration across switches, DPUs, and integrated fabric management systems while continuously validating the resulting network state.

This approach allows operators to manage tenant boundaries consistently across heterogeneous fabrics, enabling scalable and predictable multi-tenant AI infrastructure while allowing compute and storage orchestration systems to evolve independently.

In One Sentence

Netris enables hard multi-tenancy in NVIDIA DGX-based AI factories by defining tenant intent once and automatically aligning isolation boundaries across Ethernet, InfiniBand, NVLink, and DPU-enabled networking domains.

Architecture Principles

The architecture described in this paper is based on three principles:

1. Define tenant intent once
   Tenant isolation is expressed using a high-level abstraction representing the intended isolation domain.
2. Translate intent across fabrics.
   Netris converts tenant intent into the isolation constructs required by each participating fabric, including Ethernet VRFs/VXLANs, InfiniBand partitions (PKeys), NVLink GPU partitions, and host networking policies.
3. Continuously validate isolation state.
   Netris continuously validates topology and configuration state to ensure tenant boundaries remain consistent as infrastructure evolves.

Table of Contents

The Multi-Tenancy Model In AI Cloud

AI infrastructure is increasingly being operated as a shared platform across enterprises and NVIDIA Cloud Providers (NCPs). These AI factories host many independent teams or customers on the same physical resources. Hard multi-tenancy (network hardware-enforced tenant isolation), therefore, becomes a core architectural requirement: each tenant must receive predictable performance and strict isolation while participating in shared infrastructure.

Tenant Isolation Across Heterogeneous Fabrics

AI factories and NCPs may incorporate several types of connectivity fabrics, each defining tenant boundaries using its own isolation construct:

- **VRF / VXLAN** on Ethernet fabrics,
- **Partition Keys (PKeys)** on InfiniBand fabrics,
- **NVL partitions** on systems equipped with an NVLink Multi-Node fabric, such as NVL72.

In shared AI factories and NCPs, properly aligned tenant boundaries across all participating connectivity fabrics reduce operational and security risks to the business stemming from error-prone, manual configuration and coordination processes or inadvertent exposure of one tenant's data to another.

Fabric Domains in Multi-Tenant AI Factories

AI factories and NCPs built with NVIDIA technologies commonly include:

- **East–West fabric, which provides a high-bandwidth, low-latency interconnect that supports RDMA-based collective operations and inter-GPU communication within the GPU cluster.** This fabric may be implemented using InfiniBand (native RDMA) or NVIDIA Spectrum-X™ Ethernet configured for lossless transport and RDMA over Ethernet (RoCE).
- **North–South fabric, which provides external connectivity for ingress, egress, storage access, and service traffic between the GPU cluster and other parts of the data center, cloud environments, and the public Internet.** It is implemented using general-purpose Ethernet. Optional DPUs may participate in this fabric to enable concurrent multi-tenancy through host-level segmentation. The North–South domain also supports edge connectivity services, including routing, NAT, and firewalls.
- **NVLink Multi-Node rack-scale fabrics, such as NVL72 (or NVL144), provide the high-bandwidth, low-latency fabric that forms a unified GPU collective for tightly**

coupled AI workloads. These systems can be partitioned to define one or more GPU collectives.

- **Management Network:** Often implemented as a logically isolated domain within the North–South fabric; in some deployments, provided as a separate management fabric.

Each fabric applies its own isolation mechanism. In large-scale multi-tenant AI factories, this creates a requirement for an architectural model that treats these connectivity fabrics as parallel isolation planes, together defining the tenant's complete operational scope.

Purpose of the Multi-Tenancy Model

This model establishes the foundation for multi-tenancy in DGX-based AI factories and helps operators to:

- Maximize utilization of shared DGX-based infrastructure
- Provide secure and isolated environments for internal teams or external customers
- Avoid prohibitively time-consuming and error-prone manual reconfiguration
- Support rapid tenant onboarding and lifecycle in scale-out for AI fabrics

The next section examines the technical challenges involved in implementing this model across heterogeneous fabrics.

Multi-tenancy Challenges in AI Infrastructure

Coordinating tenant boundaries across multiple independent connectivity fabrics—Ethernet, InfiniBand, and NVLink Multi-Node—is central to operating multi-tenant AI factories and NCPs while maintaining the desired level of security. Still, several technical challenges arise when these fabrics must operate together.

Coordinating Many Independent Configurations Across Multiple Fabrics

AI factories and NCPs routinely create, expand, shrink, or retire tenant environments. Each lifecycle event must be reflected across all participating fabrics to maintain alignment. Common day-2 operations include:

- Adding or removing servers from a tenant
- Updating InfiniBand PKey membership
- Adding or removing GPU UIDs in NVLink partitions
- Modifying VRF/VXLAN routing domains
- Updating DPU or edge policies

Tenant isolation requires changes to VRFs, VXLANs, IP ranges, switch ports, RoCE, InfiniBand PKeys and HCA GUIDs, NVLink partitions and GPU UIDs, DPUs, edge policies, and many other highly specialized parameters. Every tenant change must be applied consistently across Ethernet, InfiniBand, and NVLink fabrics.

Manually applying configuration changes across Ethernet, InfiniBand, and NVLink fabrics introduces specialized, complex, time-consuming, and error-prone workflows. The required domain expertise is difficult to staff. Building in-house automation is costly and time-consuming. Furthermore, sustaining internally developed cross-fabric automation introduces a significant ongoing operational burden. In environments that rely on manual processes, field observations have reported unintended service impacts during on-the-fly tenant updates in approximately 20% of cases.

Security and Operational Challenges

Misaligned tenant boundaries can expose one tenant's traffic to another, disrupt workloads, or violate expected isolation.

These issues create operational, security, and compliance risks, including higher remediation costs, reduced customer confidence, regulatory scrutiny, financial penalties, and loss of usable GPU capacity.

Other challenges:

1. **Limited Visibility Across Fabrics**
   Operators cannot easily see how Ethernet, InfiniBand, and NVLink tenant boundaries relate to one another, making misconfigurations difficult to detect until they cause an incident.
2. **Inconsistent Reconfiguration Timing**
   Updates often apply at different speeds across fabrics, creating short windows where tenant isolation is incomplete or misleading.
3. **Rollback Difficulty After Failures**
   Undoing a failed tenant change requires coordinated reversions across all fabrics, and any mismatch can prolong outages or expose traffic.
4. **Validation Complexity Before Changes**
   It is difficult to determine, in advance, whether a change to one fabric will preserve alignment with the others.
5. **Configuration Drift Across Toolchains**
   Independent automation systems or manual workflows can introduce subtle, compounding differences across fabrics over time.

6. **Undocumented or Tribal Knowledge Workflows**
   Many cross-fabric workflows rely on informal knowledge rather than standardized, repeatable procedures.
7. **Limited Automation Standards**
   There is no shared, cross-fabric representation of tenant intent at the hardware or fabric-management layer, leading operators to rely on custom tooling when higher-level orchestration is required.
8. **Scaling Pressure From Rapid Tenant Turnover**
   Fast-changing or short-lived tenant environments increase the frequency and volume of multi-fabric updates.

Together, these impacts underscore the importance of maintaining consistent tenant boundaries across all fabrics in a shared AI infrastructure.

Recommendations for Implementing a Multi-Tenant Environment

At the AI-factory scale, operators benefit from a unified tenant abstraction, such as a Virtual Private Cloud (VPC), to help reason consistently about isolation and membership across heterogeneous fabrics. The following architectural practices support this objective.

1. Establish a Unified Tenant Abstraction

Define the tenant as a high-level abstraction that includes an isolated communication boundary along with the set of workloads assigned to it. The nature of the workloads may evolve, but the tenant abstraction remains constant, providing a stable way to reason about isolation and membership regardless of the connectivity fabrics in the AI factory.

2. Translate the High-Level Abstraction into Fabric Configurations

Programmatically translate this tenant abstraction into the appropriate configurations for each fabric, including:

- VRFs and VXLANs on Ethernet,
- PKeys on InfiniBand,
- and NVLink partition membership when NVLink Multi-Node fabrics are present.

Deriving these configurations from a high-level abstraction reduces the need for operators to define and manage fabric-specific and switch-specific configurations.

3. Apply Tenant Configurations Across Fabrics Without Disruption

Tenant creation, resizing, and retirement actions must be applied across all fabrics without causing configuration conflicts and preserving the operational and isolation boundaries of all existing tenants.

4. Validate Topology and Isolation Continuously

Continuous validation helps detect divergences between operator intent and the running state of all fabrics. This validation should confirm that partition membership, routing boundaries, and GPU collectives conform to the tenant abstraction. Proactive validation helps prevent outages and reduce the risk of data loss.

5. Provide an API for Upstream Platforms

AI factories and NCPs frequently integrate networking with higher-level orchestration layers such as IaaS and PaaS systems. Exposing a consistent high-level abstract networking API allows these systems to integrate with the networking layer without requiring specialized fabric-specific knowledge across multiple control planes.

The following sections describe one implementation of the architectural principles described above.

Architecture of a Multi-Tenant AI Cloud with Netris

Netris allows users to define, in high-level terms, tenant isolation and other parameters through an intuitive GUI or API in a declarative way (defining the desired outcome) – and then Netris algorithms automatically configure the relevant network elements to align with the desired outcome continuously.

In this role, Netris serves as a foundational coordination layer of AI networking infrastructure, aligning tenant-related configuration across the participating fabrics while integrating with NVIDIA fabric management systems.

In the most basic sense, the IaaS or PaaS operator requests an action of creating, modifying, or retiring a tenant using a Netris API endpoint, and Netris algorithm translates this request into the appropriate configurations and applies changes for Ethernet, InfiniBand, and NVLink Multi-Node fabrics, then applies it consistently and securely across the environment.

Figure 1 illustrates the high-level representation of this model.

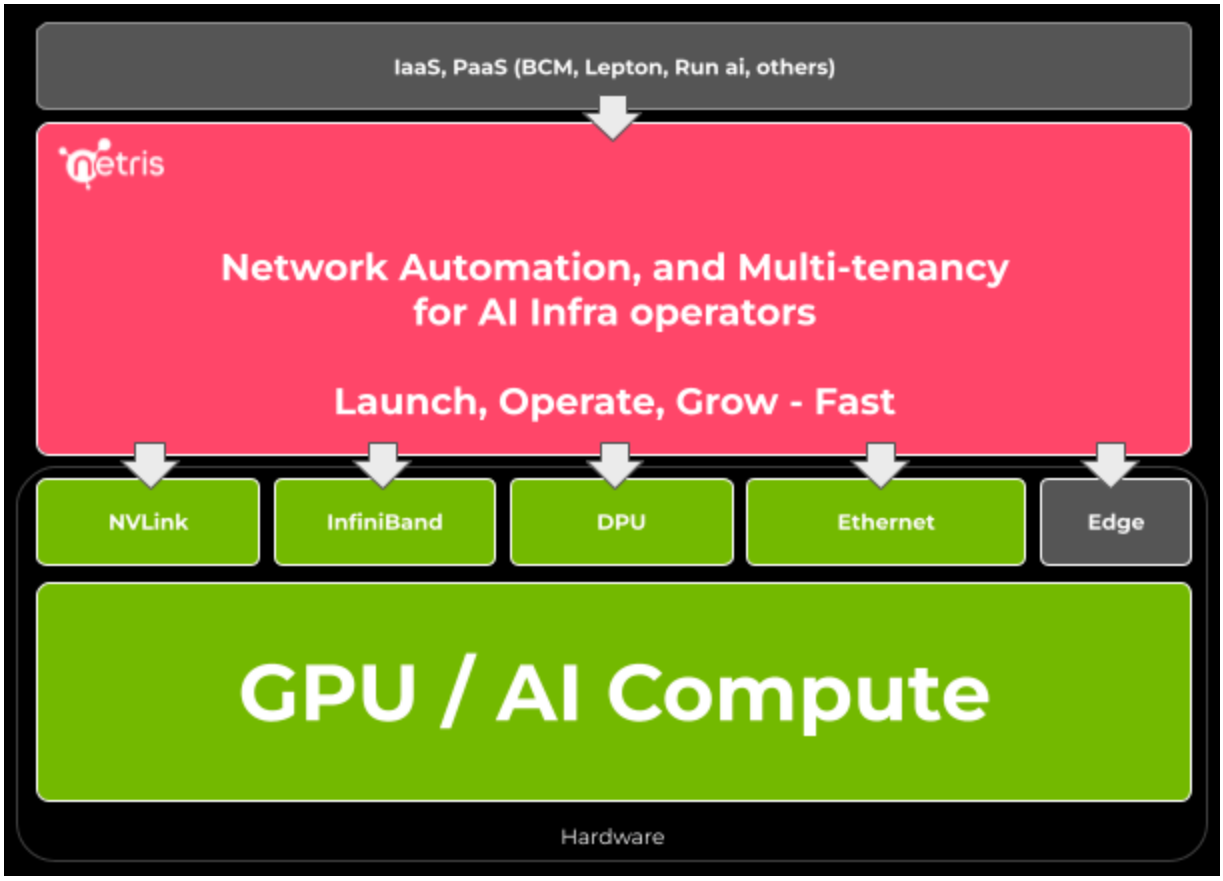

Figure 1. Integration of Infrastructure and Platform Layers Using Netris within the NVIDIA AI Ecosystem

Interpreting Operator's Intent and Supporting Alignment Across Fabrics

Operators or upstream platforms define a tenant using a high-level abstraction, such as a Server Cluster—essentially a set of endpoints allocated to a tenant. Netris algorithm automatically translates this high-level definition into the fabric-specific configurations required by the participating connectivity fabrics:

- **Spectrum-X East–West Ethernet:** Netris automatically generates tenant VRFs and L3VPN VXLAN configurations, and then assigns switch ports to the appropriate VXLANs. Netris functions as the fabric manager for this Ethernet fabric. Based on the Ethernet switch operating system version, Netris implements fabric-wide configurations (switches and BlueField-3 SuperNICs) compliant with the appropriate version of the NVIDIA Spectrum-X Deployment Guide for lossless inter-GPU communication.

- **InfiniBand East-West:** Netris translates the tenant definitions into InfiniBand partitions (PKeys) and associated HCA membership, then applies them through the NVIDIA UFM API.
- **NVLink Multi-Node Fabrics:** When NVLink Multi-Node fabrics, such as NVL72 or NVL144, are present, Netris creates, modifies, and deletes NVLink partitions as appropriate and assigns the relevant GPU UIDs to these NVLink partitions using the NVIDIA NMX-C gRPC API.
- **North–South Ethernet Fabric:** Netris configures the tenant VXLAN-based routing domains for ingress, egress, storage access, and service communication. Optional DPU configurations inherit the same tenant definition. Netris functions as the Ethernet fabric manager for the North-South fabric, applies the relevant configurations consistently across switches and DPUs, and provides the automation that aligns North–South connectivity with the rest of the fabrics in the AI factory.
- **Edge Connectivity:** When deployed, Netris SoftGate can provide multi-tenant NAT, Layer-4 Load Balancing, and other edge services automatically aligned within the tenant boundaries.

Figure 2 demonstrates how Netris fits into the operational architecture of an AI factory or an NCP.

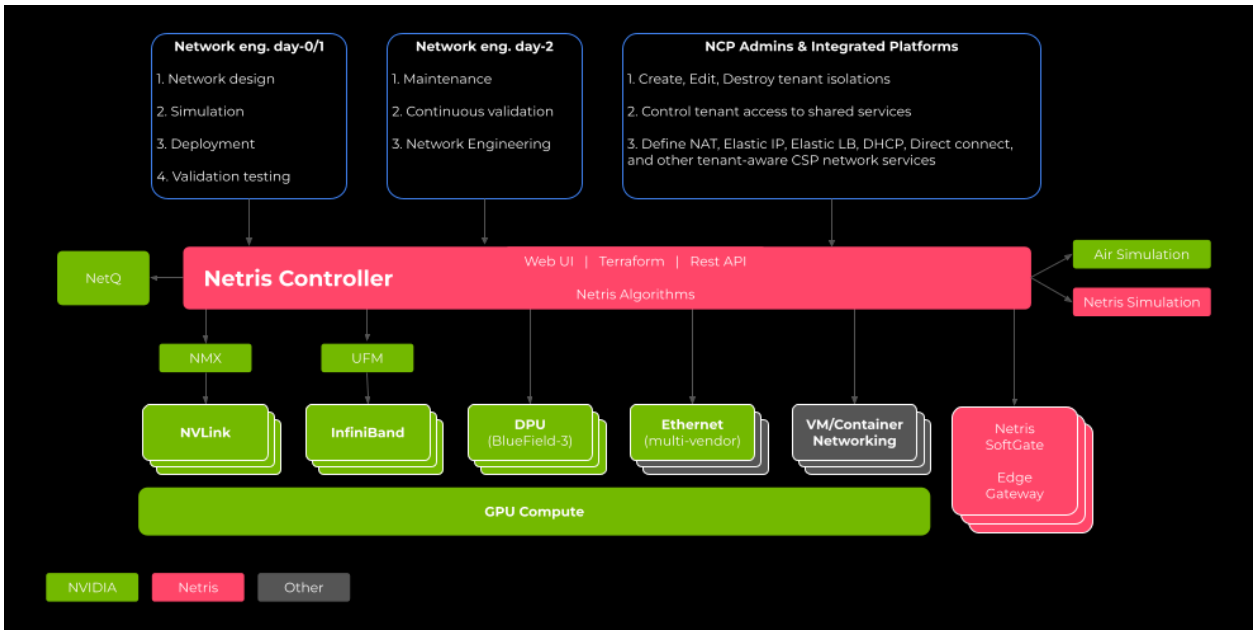

Figure 2. Operational Architecture of Netris in an NVIDIA-Based AI Infrastructure

This unified intent-driven model supports the implementation of dynamic, multi-tenant clouds on NVIDIA DGX-based architectures by interpreting the intent expressed by

upstream platforms in one place and automatically and dynamically applying the relevant configurations consistently across all connectivity fabrics participating in the AI factory.

Upstream Platform Integration

AI factories typically operate within a broader platform stack that includes user portals, billing systems, PaaS layers, and administrative tooling. These upstream systems are responsible for user interaction, policy, and workload orchestration, and are often custom-built by the operator.

Netris integrates as one of several infrastructure management components consumed by the platform. It focuses specifically on network-related tenant isolation and connectivity across the underlying fabrics, while remaining independent from compute and storage management systems.

Figure 3 illustrates this integration model. Upstream platforms coordinate tenant lifecycle actions and interact independently with networking, compute, and storage management components. Netris operates alongside compute and storage managers, translating tenant intent into network configuration and ensuring that network isolation remains aligned with the overall platform architecture.

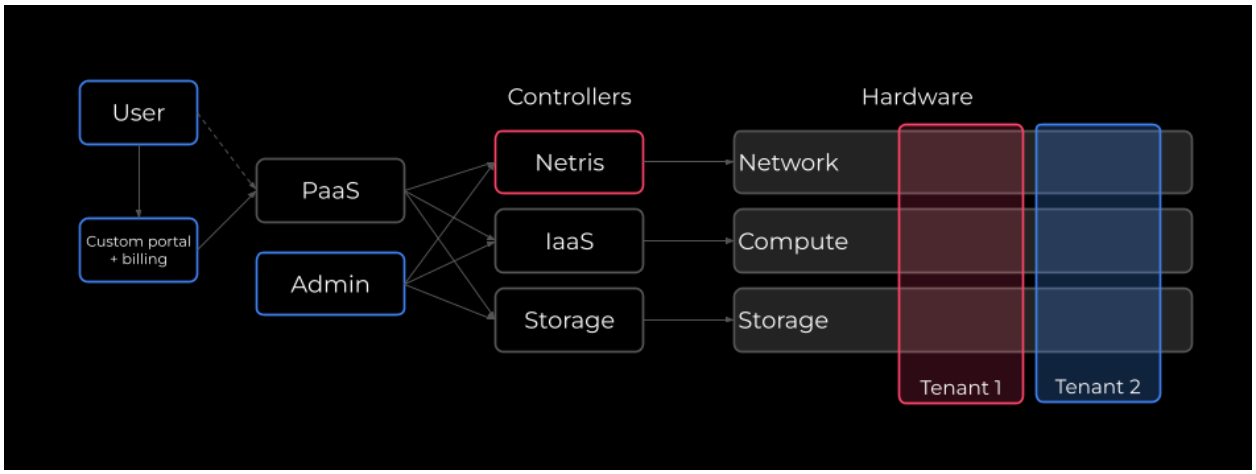

Figure 3. Netris platform integration architecture

Upstream platforms, such as PaaS layers or administrative tools, typically interact with Netris through a stable API to express tenant lifecycle intent—for example, creating, expanding, or retiring a tenant. This interaction does not require the upstream platforms to directly manage fabric-specific details or communicate with individual network control planes.

In parallel, the same upstream platforms interact with infrastructure services responsible for compute and storage. Each of these systems—networking, compute, and storage—retains responsibility for its own domains. Netris does not replace upstream orchestration platforms or infrastructure managers; it provides a consistent integration point for applying tenant-related network isolation across Ethernet, InfiniBand, and NVLink fabrics.

Once tenant isolation is established at the network level, compute and storage systems operate within those boundaries using their existing mechanisms. In this way, tenant environments span network, compute, and storage domains, while each domain continues to be managed by the systems designed for that purpose.

Simulation

Many operators use simulation environments to explore network designs, tenant boundaries, and planned changes before touching a live AI factory.

Netris supports this by integrating with NVIDIA simulation tools such as NVIDIA Air. Operators can load a representation of their current production environment into a simulation and use it as a safe space to experiment, test ideas, or rehearse changes without risking production workloads.

Simulation is commonly used for learning the platform, proof-of-concept work, and pre-deployment testing. It complements production operations by giving teams a practical way to explore “what if” scenarios before changes are applied to real infrastructure.

Separation of Responsibilities

Netris architecture helps operators maintain a clear division of responsibilities:

- **Network engineering teams:** design and validate connectivity fabrics using Netris, which implements the orchestration model, integrates with NVIDIA management systems, and serves as the Ethernet fabric manager.
- **Platform operators:** manage tenant lifecycle and express intent through Netris northbound API.
- **NVIDIA management systems (UFM, NMX):** maintain inter-GPU network fabric health and GPU operational behavior.

This architectural approach supports multi-tenancy at scale, consistent tenant security boundaries, and complex distributed operations.

An Example Network Isolation Implementation With Netris

This section illustrates how Netris orchestrates network provisioning and tenant isolation across the different fabric domains of an NVIDIA-based AI infrastructure. The examples highlight how multi-fabric configuration and tenant alignment are automated through the Netris Server Cluster construct, supporting consistent behavior across InfiniBand, Ethernet, NVLink Multi-Node, and management networks.

Network Provisioning with Netris

A Server Cluster is the primary mechanism Netris uses to define tenant boundaries; in its most basic form, it is a list of servers assigned to a tenant. From this list of servers and using the topology information, GPU UIDs, HCA GUIDs, and other metadata in the controller, the Netris algorithm generates the following:

- which switch ports participate in the tenant's VRF and VXLAN,
- which HCAs belong to the tenant's InfiniBand partition (PKey),
- and which GPU identifiers (UID) are included in the tenant's NVLink partition.

Netris then distributes the appropriate configuration to NVIDIA UFM, NMX, and directly to the Ethernet components, including SuperNICs and DPUs, when present.

This approach aligns tenant isolation across Ethernet, InfiniBand, NVLink Multi-Node, and North–South domains without requiring operators to configure each fabric directly.

In this example, tenant "Alpha" is used as a symbolic example; vrf-alpha, vni-alpha, switchport-set-alpha, pkey-alpha, hca-set-alpha, nvl-alpha, and gpu-set-alpha represent abstract configuration constructs derived from the Server Cluster definition.

East–West Fabric

The East–West fabric carries inter-GPU communication and collective operations traffic within a GPU cluster.

If East–West is implemented using Spectrum-X Ethernet

Tenant Alpha's East–West Ethernet boundary is represented through:

- **VRF:** vrf-alpha

- **VNI:** vni-alpha
- **Switch Port Set:** switchport-set-alpha (list of switch ports)

These constructs follow the isolation and transport requirements described in the NVIDIA Spectrum-X deployment guide. Netris configures these VRFs, VNIs, and inserts the appropriate switch ports into the correct VXLANs (VNI) on every Ethernet switch in the East-West fabric, so that the routing domain for inter-GPU communication remains consistent with the intended tenant boundary.

Netris also applies the host-side settings on BlueField-3 SuperNICs, helping maintain alignment with the Spectrum-X lossless transport model as defined in the NVIDIA Spectrum-X Deployment Guide.

If East–West is implemented using InfiniBand

For InfiniBand-based East–West fabrics, Tenant Alpha's communication boundary is defined by an InfiniBand partition and its associated host channel adapters (HCAs):

- **InfiniBand Partition Key (PKey)**: pkey-alpha
- **HCA membership set**: hca-set-alpha (list of HCA GUIDs)

Netris integrates with NVIDIA UFM to collect and maintain an inventory of servers and their corresponding HCA GUIDs. This data is stored in the Netris controller and used to derive InfiniBand partition membership from the Server Cluster definition.

When a tenant is created, Netris creates the corresponding InfiniBand PKey in UFM and associates the required HCA GUIDs with that partition. When the tenant is resized, Netris updates the membership of the existing PKey it previously created. When the tenant is deleted, Netris removes the associated PKey from UFM. In this way, the lifecycle of the InfiniBand partition is aligned with the tenant lifecycle defined in Netris.

The assigned PKey defines the communication domain for HCAs belonging to the tenant. When in-network collective acceleration using SHArP is enabled, the same PKey boundary is used to scope SHArP participation. Netris sets up SHArP reservations consistent with the tenant's InfiniBand partition, ensuring that in-network aggregation behavior remains aligned with the tenant's isolation boundaries.

North–South Fabric

The North–South fabric supports ingress, egress, storage access, and service connectivity for each tenant. Tenant Alpha's routing boundary is configured using:

- **VRF:** vrf-alpha
- **VNI:** vni-alpha
- **Switch Port Set:** switchport-set-alpha (list of switch ports)

This domain remains consistent whether East–West is implemented via Spectrum-X Ethernet or InfiniBand. Netris derives routing, segmentation, and forwarding behavior using vrf-alpha, vni-alpha, and switchport-set-alpha, and applies these semantics across the North–South connectivity layer.

Optional DPUs on the North-South fabric interpret the same constructs to align tenant routing and service policy at the host edge.

At the North-South fabric perimeter, SoftGate can provide routing, NAT, and related ingress/egress services, promoting consistent isolation across ingress and service paths.

NVLink Multi-Node Fabrics (NVL72/NVL144)

When NVLink Multi-Node fabrics are present, Tenant Alpha's GPU grouping is represented as:

- **NVLink Partition Identifier:** nvl-alpha
- **GPU Identifier Set:** gpu-set-alpha (list of GPU UIDs)

The NVLink partition binds the GPUs allocated to the tenant into a single NVLink GPU collective domain. Netris maintains an inventory of servers and their corresponding GPU UIDs and maps the Server Cluster's GPU membership to the NVLink partition. Netris then communicates this grouping to NVIDIA NMX-C using the supported NMX-C gRPC API. Aligning nvl-alpha with constructs such as vrf-alpha and pkey-alpha coordinates the GPU-side collective behavior with routing and transport-level boundaries across the AI factory.

Management Networks (Out-of-Band)

AI infrastructure typically includes a dedicated management network to provision, monitor, and operate infrastructure components independently of the tenants' data-plane traffic. This network provides access to server management interfaces, DPU management interfaces, and other control endpoints and is intentionally isolated from tenant workloads. In some cases, the operator may choose to grant tenants access to this network.

In some architectures, this management network is referred to as out-of-band (OOB), a term historically used to allude to management connectivity that is isolated from tenant networks and often implemented as a physically separate network. In practice,

management connectivity may be deployed either as a logically isolated domain within the North–South Ethernet fabric or as a physically separate Ethernet fabric dedicated to management traffic. Both deployment models are common in DGX-based AI factories and are supported by Netris.

Regardless of physical topology, management connectivity is carried over Ethernet and uses the same isolation primitives applied elsewhere in the architecture. A management network is configured using:

- a dedicated routing context (VRF),
- an associated VXLAN overlay identifier (VNI), and
- a defined set of switch ports that provide connectivity to management interfaces.

These primitives are used consistently whether the management network is implemented on the shared North–South fabric or on a standalone management fabric.

Netris uses topology information and metadata (e.g., interface labels) to ensure that interfaces connected to management endpoints—such as server management ports—are consistently placed into the management network. This allows management connectivity to remain stable and isolated as servers are added, removed, or reassigned across tenant environments, without requiring manual reconfiguration.

Network Services for Tenants

Multi-tenant AI environments often require network services at the tenant boundary, interconnections between tenant VPCs, and reachability to and from external networks. These services are distinct from the core fabric routing and isolation functions and are commonly shared across tenants.

VPC-peering (Shared storage and more)

VPC peering provides controlled Layer-3 connectivity between two or more VPCs within the same Netris-managed fabric. A peering relationship allows selected IP prefixes to be exchanged between VPCs while preserving each VPC's independent routing domain and isolation boundaries.

Conceptually, a VPC peering relationship can be viewed as a routed boundary between two tenant networks. Each VPC retains its own routing context, and only explicitly permitted IP prefixes are advertised across the peering relationship. Prefix-lists define which routes are exchanged. Layer-2 domains are never extended across the peering relationship. The

operator should ensure that the IP address space of the peered VPCs doesn't overlap to avoid reachability conflicts.

Netris implements VPC peering using the VRF route leaking feature.

Although VPC peering is technically feasible wherever routed connectivity exists, it is primarily intended for use on the North–South fabric. In practice, VPC peering is most commonly used to provide controlled access to shared services—such as storage systems or platform components—without collapsing tenant isolation or introducing direct connectivity between tenant VPCs.

Shared Storage Access via VPC Peering

Storage systems in AI factories are frequently shared across multiple tenants. Unlike compute resources, storage endpoints are often not dedicated on a per-tenant basis and may have practical limits on the number of logical interfaces they can expose. As a result, storage isolation and sharing are typically achieved through a combination of network segmentation and storage-platform controls rather than dedicated hardware per tenant.

A common and scalable pattern is the Shared Services VPC model, in which storage endpoints are placed into a dedicated VPC. Tenant VPCs establish VPC peering relationships with this shared services VPC, allowing tenants to access storage through controlled, routed connectivity.

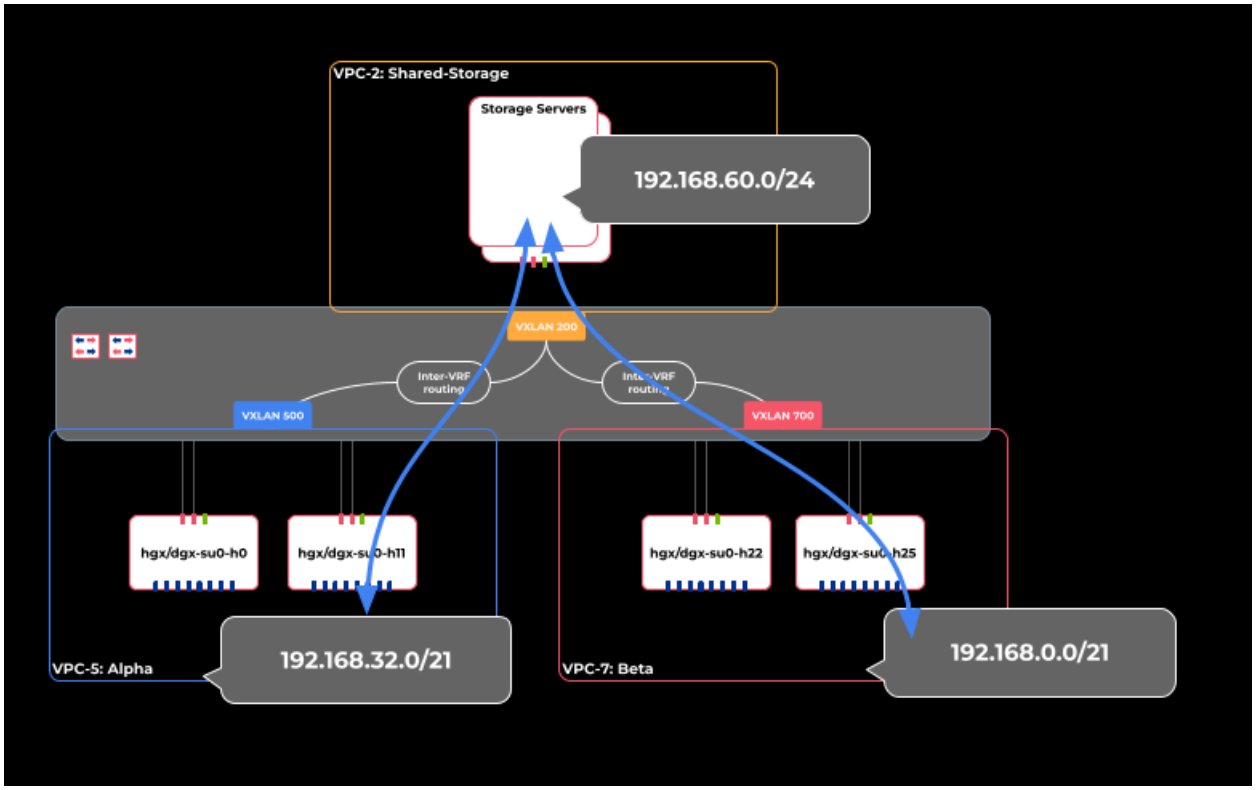

Figure 4. Shared storage access with VPC peering

This approach:

- Preserves strict isolation between tenant VPCs
- Avoids per-tenant scaling limits on storage-side interfaces
- Centralizes storage access control and lifecycle management
- Simplifies onboarding and offboarding of tenants

Each tenant's access to storage is governed by the VPC peering relationship and the associated prefix-lists, ensuring that only explicitly permitted prefixes are reachable.

Alternative: Shared Storage Access via VLAN Subinterfaces

Some storage platforms support exposing logical interfaces—such as subinterfaces or virtual ports—directly into each tenant's VPC. In this model, the storage system participates directly in each tenant's North–South routing domain, eliminating the need for VPC peering to access storage.

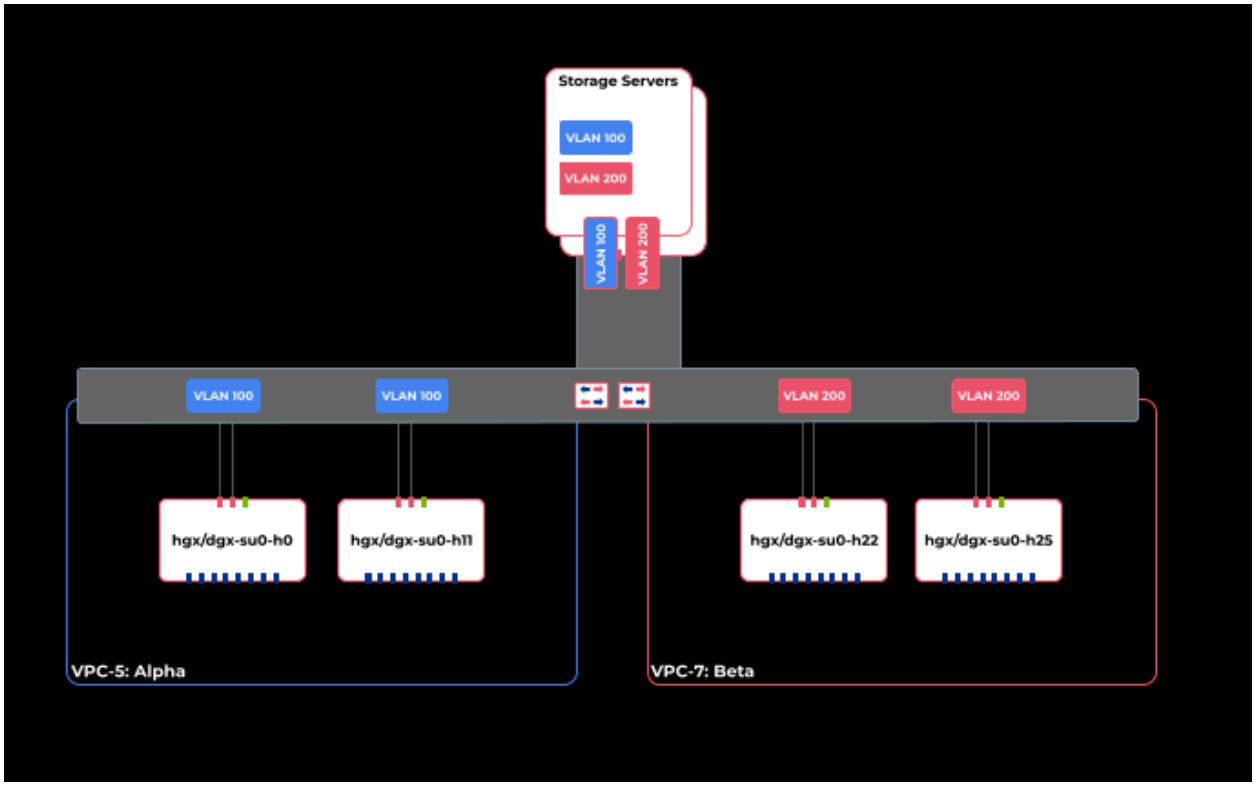

Figure 5. Shared storage access with VLAN subinterfaces

While this approach can provide clear per-tenant visibility and straightforward routing semantics, its scalability is constrained by the storage platform's ability to support a large number of logical interfaces. Operators should account for these limits when designing environments.

VPC Connect

VPC Connect provides a mechanism for connecting tenant VPCs with external networks, including non-Netris-managed data center networks, WAN routers, and security appliances.

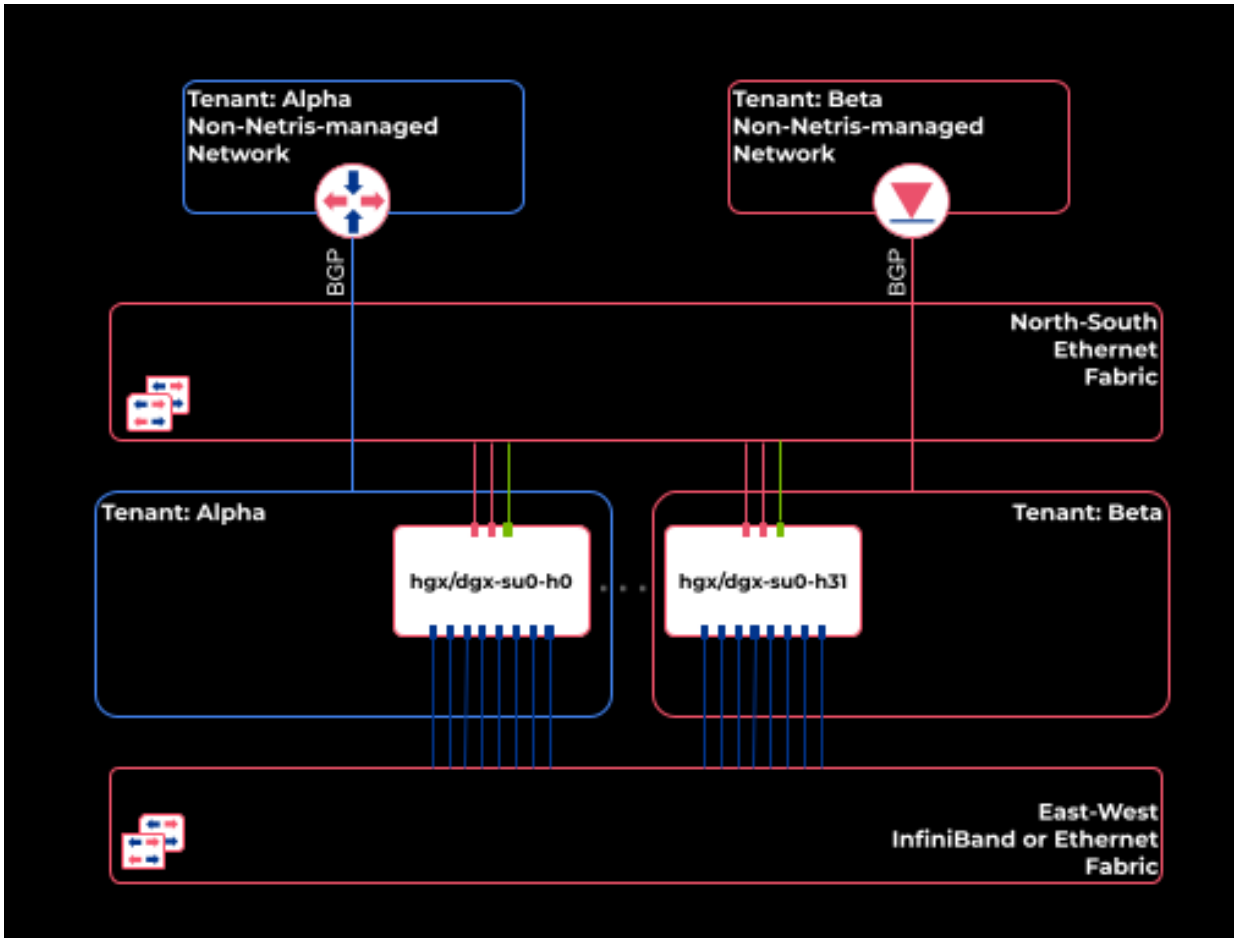

Figure 6. Netris VPC Connect

In this model, Netris exposes a switch port on the North–South fabric. When a single-tenant VPC is in scope, the port is configured as a routed port, and the eBGP neighbor relationship terminates directly on that switch port. When multiple tenant VPCs are in scope, the port is configured as IEEE 802.1Q tagged, with each tenant mapped to a dedicated VLAN, and eBGP neighbor relationships terminate on corresponding SVIs (Switch Virtual Interfaces) established on the switch or elsewhere in the fabric, as determined by Netris.

Route exchange between tenant VPCs and the external network is explicitly controlled using route-maps and prefix-lists, allowing operators to define which prefixes are advertised in each direction while preserving clear routing boundaries.

The external endpoint—such as a router, firewall, or other network transport device—is typically owned and operated by the operator's networking team.

Edge Services (NATing, Layer-4 LB)

Typically, AI workloads need Internet access. Inferencing workloads need to be accessed from the Internet. Additionally, some workloads need access to the Internet for integrations with 3rd-party services and other administrative use cases (downloading packages, performing DNS lookups, etc.).

In practice, two classes of services are foundational at the VPC boundary.

- Tenant-aware (VPC-aware) Network Address Translation (NAT) enables controlled ingress and egress, allowing workloads inside a VPC to initiate outbound connections and, where required, be accessed from external networks without exposing internal addressing.
- Tenant-aware (VPC-aware) Layer-4 load balancing that provides a stable service endpoint (frontend public IP address) for scalable inference and application traffic, distributing TCP and UDP flows across backend workloads within a tenant's routing domain.

Netris SoftGate is an optional, horizontally scalable edge services layer that operates as an integrated extension of the Netris-managed North–South fabric. Deployed as a cluster of dedicated bare-metal nodes running the SoftGate dataplane and agent, it delivers VPC-aware network edge services for controlled ingress and egress while remaining consistent with tenant isolation boundaries defined in Netris.

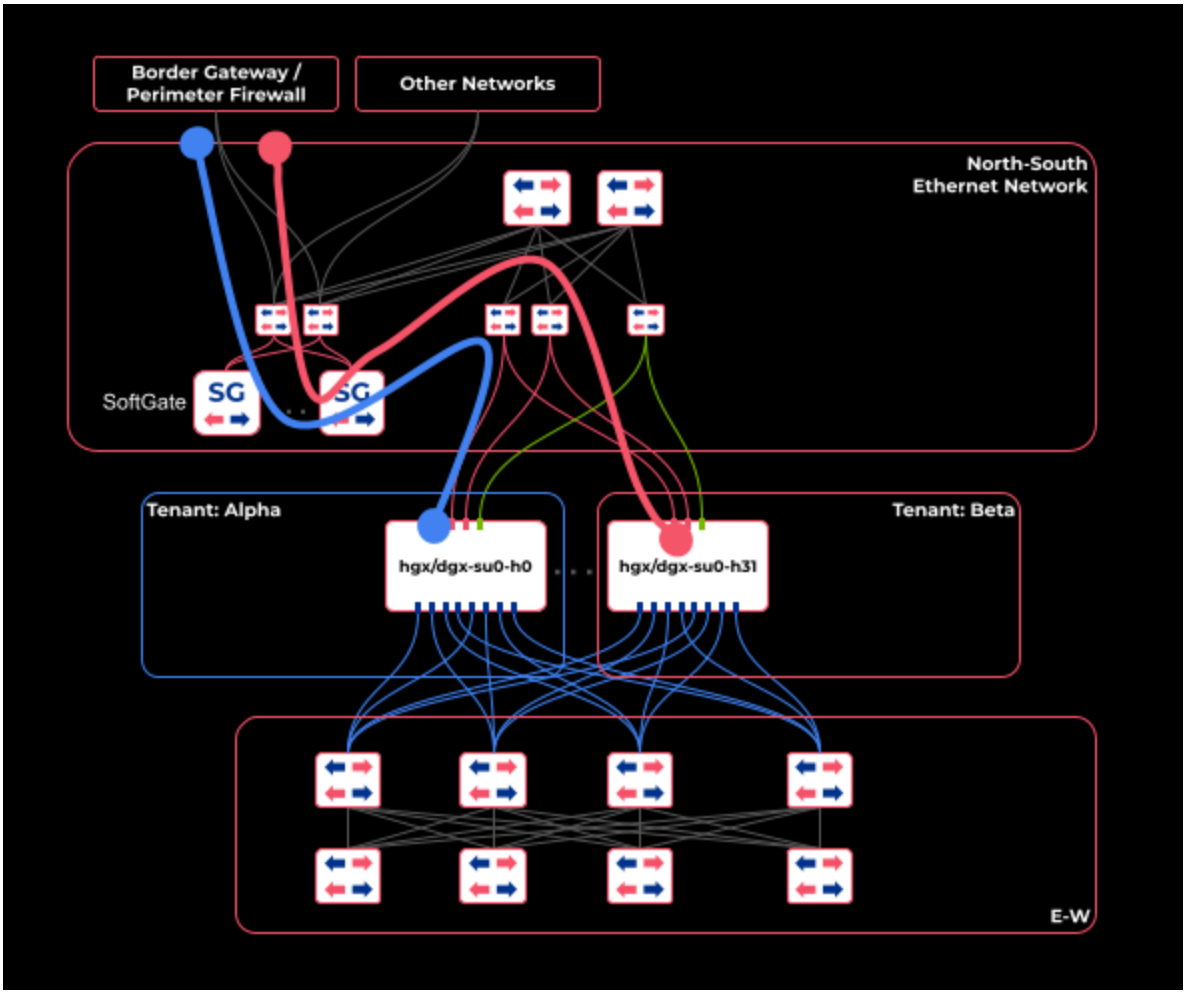

Figure 7. VPC edge services provided by Netris SoftGate

The primary services provided by Softgate include:

- Network Address Translation (NAT) for controlled ingress and egress
- Layer-4 load balancing for TCP and UDP traffic, including health checks
- DHCP services, where required

Access Control Lists

Access Control Lists (ACLs) in Netris are expressed as a high-level policy associated with a tenant VPC, rather than as a configuration bound to a specific device or service. Operators define the desired access-control intent without specifying where that policy must be enforced within the infrastructure.

The Netris algorithm automatically determines the appropriate enforcement point based on topology, traffic direction, and participating network elements. Depending on the deployment, access control policies may be enforced on fabric switches, DPUs, host-based enforcement points, or edge service components. Enforcement placement is derived automatically and does not require the operator's intervention.

This approach reduces cognitive load and misconfiguration risk by eliminating the need to reason about device-level policy placement while ensuring consistent enforcement aligned with tenant network boundaries. Access Control Lists operate independently of NAT, load balancing, and DHCP services and do not require deployment of Netris Softgate.

Concurrent Multi-Tenancy / Host-Level Segmentation

So far, this document has described a multi-tenancy use case in which the atomic unit of tenancy is the whole bare-metal server. More specifically, this means that any given bare-metal server in the deployment, and thus all of its GPUs, is assigned to only one tenant at any given time.

Operators may choose to enable concurrent multi-tenancy (scenarios in which bare-metal servers are shared across multiple tenants) in their environments to improve hardware utilization across some or all of their AI clusters. In this model, isolation boundaries extend beyond the network fabric and into the hosts themselves. Host-level segmentation enables a single physical server to be partitioned so that multiple tenants can safely share GPUs, connectivity, and services simultaneously.

This approach changes how tenant isolation must be enforced. Network boundaries can no longer rely solely on server membership; they must extend inside the server and apply consistently to GPUs, host networking, and shared services.

There are two mechanisms available to operators for deploying concurrent multi-tenancy in NVIDIA-based AI infrastructures:

- DPU/hardware-accelerated segmentation.
- EVPN-on-Host software-based segmentation.

DPU-based/Hardware-Accelerated Host Segmentation (HBN)

A BlueField DPU (Data Processing Unit) is a system-on-a-chip, an Ethernet-switch-like device installed inside the server, intended to offload data center networking, storage, and security functions from the general-purpose host CPU to dedicated hardware.

NVIDIA DPUs can operate in one of 3 modes: regular NIC mode, DPU mode, and Zero-Trust mode.

- In NIC mode, the BlueField card behaves like a traditional network adapter from the host's perspective.
- In DPU mode, the embedded ARM subsystem owns and controls the NIC resources and data path. It means that the on-chip ARM cores manage the network controller functions and traffic before anything reaches the parent host.
- Zero-Trust mode is an enhanced security variant of DPU mode. In Zero-Trust mode, the host admin can't use typical host paths to access the DPU hardware for administration. DPU management is performed via the embedded ARM and/or out-of-band BMC interfaces instead.

In the concurrent multi-tenancy use case, DPU is typically configured in Zero-Trust mode. Host-Based Networking (HBN), which is a DOCA service that runs on the DPU in Zero-Trust mode, is deployed to implement server-side routing and network services using standard Layer-3 protocols.

Zero-Trust DPU mode with Host-Based Network enables hardware-based tenant isolation to be securely enforced outside the host operating system, even when multiple tenants share a single physical host.

Each DPU typically includes:

- A dedicated management port used for access to the DPU operating system and the DPU's BMC
- One or more high-speed uplink ports connected to the North–South Ethernet fabric

When operating in DPU mode, the DPU presents Virtual Functions (VFs) to the host. These VFs appear to the host operating system as network interface cards (NICs) and can be independently configured. From the host's perspective, they behave like regular NICs. From the platform's perspective, they represent isolated, hardware-enforced networking endpoints, much like a traditional switch port.

By assigning different VFs to different tenants, the operator can partition a single server and support multiple tenants concurrently on the North–South network. Each tenant's

traffic is isolated and enforced by the DPU, even though the workloads execute on the same physical host.

Netris, with the help of the DOCA Platform Framework (DPF), deploys a Netris management container on the DPU to manage the DPU's configuration. Netris configures the DPU as a VTEP on the North–South Ethernet fabric and applies the required tenant-specific networking configuration.

In this model, the DPU becomes a first-class participant in the EVPN fabric. Netris maps tenant definitions directly to DPU-managed VFs and enforces isolation consistently with the rest of the network.

DPU-accelerated segmentation is one mechanism for enabling concurrent multi-tenancy. In environments without DPUs, similar isolation semantics can be achieved using host-based techniques described in the following section.

Software-Based Host Segmentation (EVPN on Host)

In software-based host segmentation, tenant isolation is enforced with the host operating system itself, allowing a shared server to participate directly in tenant-specific network domains without dedicated DPU hardware.

This model is commonly used for shared CPU-based infrastructure, such as control-plane nodes for managed Kubernetes services, shared service hosts, or other environments where multiple tenants' resources must coexist on the same server while maintaining strict network isolation.

In this architecture, the host becomes a VTEP in the North–South EVPN/VXLAN fabric. Netris supports this model by deploying the Netris EVPN-on-Host agent to servers in scope for concurrent multi-tenancy. The agent configures the host networking stack to implement the same network constructs used across the rest of the North–South fabric, including:

- Tenant-specific VRFs
- VXLAN bridges and VNI mappings
- Routing tables aligned with tenant VPCs

From the platform's perspective, a host configured with EVPN-on-Host behaves like any other EVPN endpoint in the fabric. Virtual machines or containers running on the server can be connected directly to tenant VPCs (i.e., VXLANs) by attaching them to the appropriate Linux bridges established by the Netris agent. This approach enables secure communication between containers, virtual machines, GPU resources (whether dedicated

or shared), and external infrastructure endpoints—such as storage systems or proprietary appliances—while preserving hard, network-enforced isolation between tenants.

Node Provisioning with Base Command Manager (BCM)

Shared Services (Warehouse) Model

The NVIDIA Base Command Manager (BCM) is responsible for bootstrapping and lifecycle management of DGX nodes. To continuously manage a node, BCM must maintain network connectivity to that system. This requirement influences how DGX nodes are integrated into a multi-tenant network architecture.

In practice, operators commonly deploy BCM within a dedicated management VPC. This VPC contains the BCM instance and associated management components and is isolated from tenant workload VPCs.

Tenant VPCs are then selectively peered with the BCM VPC using controlled Layer-3 connectivity. This peering establishes a constrained communication path that allows DGX nodes allocated to a tenant to reach BCM for provisioning and lifecycle operations. At the same time, routing policies prevent lateral connectivity between tenant VPCs and prevent BCM from serving as a transit path between tenants.

In this model, DGX nodes remain associated with tenant isolation constructs at the networking layer (e.g., VPC boundaries, InfiniBand partitions, NVLink partitions), while maintaining the required management-plane reachability to BCM. The result is a separation of concerns: BCM retains lifecycle control of the nodes, and tenant isolation is preserved through network-level segmentation and controlled route exchange.

Network Validation in Multi-Tenant AI Infrastructure

Operating a multi-tenant AI infrastructure requires coordinating multiple independent systems—network fabrics, accelerators, and control planes—that were not designed to collectively validate tenant intent. Each system may appear healthy in isolation, while the overall tenant configuration is incorrect.

Network validation must therefore be system-aware, topology-aware, and explicitly tied to tenant intent rather than to individual device state.

Netris maintains a canonical tenant definition using high-level constructs such as Server Clusters. Based on this definition, Netris algorithms automatically generate the appropriate configuration for Ethernet fabrics, and the appropriate partitioning information for InfiniBand and NVLink fabrics.

Topology-Aware Validation

Validation must account for network topology. It must be clear how switches and servers are connected and whether this connectivity matches the operator's declared design.

Netris acts as the fabric manager for Ethernet fabrics. It maintains the expected Ethernet topology as defined by the operator's declared configuration.

Netris continuously verifies that the live Ethernet fabric topology matches the declared topology maintained in the topology manager. This validation covers both switch-to-switch links and switch-to-server links. For each declared connection, Netris confirms that the corresponding physical interfaces are connected to the expected peer device and port. When the observed physical connectivity does not match the declared topology, Netris raises a notification event to indicate the mismatch.

Detection of Partial States

Tenant lifecycle changes are applied across multiple systems. Validation must show whether these changes were fully applied.

Netris tracks the status of the configuration application during tenant operations. It surfaces cases where the configuration was not applied, failed to apply, or was only applied to part of the environment.

This allows operators to identify incomplete changes directly.

Ongoing Validation and Drift Detection

Over time, the deployed configuration can diverge from the tenant definition due to manual changes or system updates. Validation must therefore be ongoing.

Netris treats the declared tenant intent stored in the controller as authoritative. Switch agents continuously retrieve the relevant intent from the controller, locally generate the

corresponding device configuration, and apply it to the switch. This control loop operates continuously, ensuring that the applied configuration remains aligned with the declared intent.

If the agent observes repeated configuration changes within a short period—indicating that the applied configuration is not stable—it raises a notification event to alert the operator.

Data-Plane Reachability Validation in EVPN Fabrics

In EVPN VXLAN-based Ethernet fabrics, correct operation requires VTEP-to-VTEP reachability across the fabric.

Netris tracks reachability between all VTEPs in each managed EVPN fabric. Netris verifies that every VTEP can reach every other VTEP and surfaces issues if reachability is lost.

Validation That Scales With Change

As tenant onboarding and the rate of changes increase, validation must scale with them.

By centralizing tenant definitions, exposing cross-fabric membership through APIs, maintaining Ethernet topology consistency, tracking configuration status, and monitoring EVPN reachability, Netris helps support repeatable validation as environments grow.

Summary and Architectural Outlook

As AI factories mature into long-lived, shared platforms, the networking layer increasingly defines the operational limits of multi-tenancy. In DGX-based environments, tenant isolation results from the coordinated behavior of multiple independent fabrics—Ethernet, InfiniBand, NVLink Multi-Node, and management networks—each with distinct control planes, operational semantics, and failure modes.

This document describes an architectural model in which tenant intent is defined once and consistently expressed across all participating connectivity domains. By aligning Ethernet routing domains, InfiniBand partitions, and NVLink GPU collectives, operators can maintain predictable isolation while supporting tenant onboarding, resizing, and retirement without relying on tightly coupled, fabric-specific workflows.

Netris serves as the authoritative source of the desired state for fabric-wide Ethernet configuration and coordinates the appropriate partitioning across InfiniBand and NVLink

Multi-Node fabrics. As NVIDIA continues to evolve its multi-tenancy guidance across connectivity domains, Netris, as an independent software vendor within the NVIDIA ecosystem, incorporates this guidance into a consistent, multi-tenant network architecture using supported integration interfaces.

By maintaining a consistent tenant intent representation and ensuring consistent Ethernet fabric behavior, Netris enables upstream platforms (IaaS/PaaS) to evolve independently of the underlying network implementations while preserving a clear separation of responsibilities.

As AI infrastructure continues to incorporate new compute and networking platforms—including DPUs, evolving GPU architectures, new interconnect technologies, and deeper platform integrations—tenant abstractions are also expected to evolve—from server-based allocation toward finer-grained GPU allocation models. Architectures that centralize tenant intent while decoupling it from individual fabric control planes offer greater durability over time. A unified, intent-driven networking layer—combined with continuous, cross-fabric validation—establishes a foundation that can accommodate these shifts without requiring fundamental changes to the tenant model or operational workflow.