DECEMBER 2022

VMWARE NSX ® REFERENCE DESIGN GUIDE

Software Version 3.2

VMware NSX Reference Design Guide

Table of Contents

1 Introduction 8
1.1 How to Use This Document and Provide Feedback 8
1.2 Networking and Security Today 8
1.3 NSX Architecture Value and Scope 9
1.3.1 Containers and Cloud Native Application Integrations with NSX 11
1.3.2 Role of NSX in the VMware multi-cloud platform 13
2 NSX Architecture Components 16
2.1 Management Plane and Control Plane 16
2.1.1 Management Plane 16
2.1.2 Control Plane 17
2.1.3 NSX Manager Appliance 17
2.2 Data Plane 18
2.3 NSX Consumption Model 19
2.3.1 NSX Policy API Framework 19
2.3.2 Policy API Usage Example 1- Templatize and Deploy 3-Tier Application Topology 20
2.3.3 Policy API Usage Example 2- Application Security Policy Lifecycle Management 20
2.3.4 When to use Policy vs Manager UI/API 21
2.3.5 NSX Logical Object Naming relationship between manager and policy mode 22
3 NSX Logical Switching 24
3.1 The NSX Virtual Switch 24
3.1.1 Segments and Transport Zones 24
3.1.2 Uplink vs. pNIC 26
3.1.3 Teaming Policy 27
3.1.4 Uplink Profile 30
3.1.5 Transport Node Profile 32
3.1.6 Network I/O Control 32

3.1.7 Enhanced Data Path NSX virtual switch 33
3.2 Logical Switching 35
3.2.1 Overlay Backed Segments 35
3.2.2 Flooded Traffic 36
3.2.3 Unicast Traffic 39
3.2.4 Data Plane Learning 40
3.2.5 Tables Maintained by the NSX Controller 41
3.2.6 Overlay Encapsulation 43
3.3 Bridging Overlay to VLAN with the Edge Bridge 44
3.3.1 Overview of the Capabilities 45
4 NSX Logical Routing 50
4.1 Single Tier Routing 51
4.1.1 Distributed Router (DR) 51
4.1.2 Services Router 54
4.2 Two-Tier Routing 59
4.2.1 Interface Types on Tier-1 and Tier-0 Gateway 60
4.2.2 Route Types on Tier-0 and Tier-1 Gateways 61
4.2.3 Fully Distributed Two Tier Routing 63
4.3 Routing Capabilities 65
4.3.1 Static routing 65
4.3.2 Border Gateway Protocol (BGP) 66
4.3.3 Open Shortest Path First version 2 (OSPF v2) 68
4.3.4 Routing protocol recommendation in the data center. 85
4.4 IPv6 Routing Capabilities 87
4.5 Services High Availability 89
4.5.1 Active/Active 89
4.5.2 Active/Standby 92
4.5.3 High availability failover triggers 94
4.6 VRF Lite 98
4.6.1 VRF Lite Generalities 98
4.6.2 VRF Lite HA 103
4.6.3 Connecting Tier-1 gateways to VRFs 107
4.6.4 VRF route leaking 108
4.6.5 VRFs shared Internet access over a common VLAN 111
4.7 Multi-Protocol BGP EVPN - VXLAN 111

4.7.1 MP-BGP EVPN for the Control Plane 113
4.8 Multicast routing within NSX 119
4.8.1 Multicast in a virtual environment 119
4.8.2 NSX multicast capabilities 120
4.8.3 Technical overview 122
4.8.4 NSX multicast design recommendations 132
4.8.5 Static RP vs. BSR 136
4.9 Edge Node 137
4.9.1 Types of Edge Nodes 138
4.9.2 Multi-TEP support on Edge Node 139
4.9.3 Bare Metal Edge Node 140
4.9.4 VM Edge Node 141
4.9.5 Edge Cluster 142
4.9.6 Failure Domain 143
4.10 Other Network Services 144
4.10.1 Unicast Reverse Path Forwarding (uRPF) 144
4.10.2 Network Address Translation 145
4.10.3 DHCP Services 147
4.10.4 Metadata Proxy Service 147
4.10.5 Gateway Firewall Service 147
4.10.6 Proxy ARP 147
4.11 Topology Consideration 150
4.11.1 Supported Topologies 150
4.11.2 Unsupported Topologies 154
5 NSX Security 155
5.1 NSX Security Use Cases 156
5.2 NSX DFW Architecture and Components 158
5.2.1 Management Plane 158
5.2.2 Control Plane 158
5.2.3 Data Plane 159
5.3 NSX Data Plane Implementation - ESXi vs. KVM Hosts 159
5.3.1 ESXi Hosts- Data Plane Components 159
5.3.2 NSX DFW Policy Lookup and Packet Flow 161
5.4 NSX Security Policy - Plan, Design and Implement 162
5.4.1 Security Policy Methodology 163

5.4.2 Security Rule Model 164
5.4.3 Security Policy - Consumption Model 166
5.5 Intrusion Detection 174
5.6 Service Insertion 175
5.7 Additional Security Features 175
5.8 NSX Security Enforcement – Agnostic to Network Isolation 176
5.8.1 NSX Distributed Firewall for VLAN Backed workloads 176
5.8.2 NSX Distributed Firewall for Mix of VLAN and Overlay backed workloads 177
5.8.3 NSX Distributed Firewall for Overlay Backed workloads 178
5.9 Gateway Firewall 178
5.9.1 Consumption 179
5.9.2 Implementation 179
5.9.3 Deployment Scenarios 179
5.10 Endpoint Protection with NSX 182
5.11 Recommendation for Security Deployments 184
5.11.1 A Practical Approach to Building Micro-Segmentation Policy 184
5.12 NSX Firewall- For All Deployment Scenario 193
6 NSX Load Balancer 195
6.1 NSX Load Balancer Roadmap 195
6.2 Load Balancing Overview 195
6.3 NSX Load Balancing Architecture 197
6.4 NSX Load Balancing deployment modes 198
6.4.1 In-line load balancing 198
6.4.2 One-arm load balancing 199
6.5 NSX load-balancing technical details 202
6.5.1 Load-balancer high-availability 202
6.5.2 Load-balancer monitor 204
6.5.3 Load-balancer Layer4 or Layer7 load balancing 204
6.5.4 Load-balancer IPv6 207
6.5.5 Load-balancer traffic flows 207
6.5.6 Load-balancing combined with SR services (NAT and Firewall) 210
7 NSX Design Considerations 212
7.1 NSX Deployment Models 212
7.1.1 Role of overlay networking in the modern datacenter 215
7.1.2 Distributed security model, dvpgs vs NSX VLAN segments 218

7.2 Physical Infrastructure of the Data Center 220
7.2.1 Physical Network Requirements 220
7.2.2 Physical fabric considerations 221
7.3 NSX Infrastructure Component Connectivity 231
7.3.1 Overview 231
7.3.2 NSX Manager Node Availability and Hypervisor interaction 232
7.3.3 Physical placement considerations for the NSX Manager Nodes 233
7.3.4 Deployment Options for NSX Management Cluster 237
7.4 Compute Cluster Design 242
7.4.1 Running a VDS prepared for NSX on ESXi hosts 243
7.4.2 ESXi-Based Compute Hypervisor with two pNICs 247
7.4.3 ESXi-Based Compute Hypervisor with Four (or more) pNICs 251
7.4.4 KVM-Based Compute Hypervisor with two pNICs 255
7.5 Edge Node and Services Design 258
7.5.1 Bridging Use Case 259
7.5.2 Edge Connectivity Guidelines for Layer 3 Peering Use Case 274
7.5.3 Edge Connectivity Guidelines for Services Only Use Case 289
7.5.4 NSX Edge Resources Design 292
7.6 NSX Platform Design Consideration 307
7.6.1 NSX domains design guidelines 308
7.6.2 Network Topologies and Logical Design Guidelines 317
7.6.3 NSX Components Placement in a vSphere Deployment 324
8 NSX Performance & Optimization 344
8.1 Typical Data Center Workloads 344
8.2 Next Generation Encapsulation - Geneve 344
8.3 Performance Considerations 345
8.3.1 pNIC Feature Support 346
8.3.2 Number of pNICs 358
8.3.3 vNIC Queues 359
8.3.4 vCPU Count 359
8.4 Jumbo MTU for Higher Throughput 359
8.4.1 Checking MTU on an ESXi host 361
8.5 Performance Factors for NSX Edges 362
8.5.1 Data Plane Development Kit (DPDK) 362
8.5.2 SSL Offload 362

8.6 Key Performance Factors 363
8.6.1 Compute Node Performance Factors 363
8.6.2 VM Edge Node Performance Factors 364
8.6.3 Bare Metal Edge Node Performance Factors 370
8.6.4 Summary of Performance – NSX Components to NIC Features 370
8.7 Results 371
8.8 NFV: Raw Packet Processing Performance 372
8.8.1 N-VDS Enhanced Data Path 372
8.9 Acceleration with the N-VDS in Enhanced Datapath Mode 373
8.9.1 Poll Mode Driver 373
8.9.2 CPU Affinity and Optimization 373
8.9.3 Buffer Management 373
8.9.4 Flow Cache 373
8.9.5 Checking whether a NIC is N-VDS Enhanced Data Path Capable 374
8.10 Benchmarking Tools 375
8.10.1 Compute 375
8.11 Edges 375
8.11.1 VM Edge 375
8.11.2 Bare Metal Edge 375
8.12 Conclusion 376

Intended Audience

This document is targeted toward virtualization and network architects interested in deploying VMware NSX® network virtualization solutions in a variety of on premise solutions.

Revision History

Version	Updates	Comments
1.0	None	NSX 2.0
2.0	Completely Revised	NSX 2.5
3.0	Completely Revised	NSX 3.0
3.2	Completely Revised	NSX 3.2

1 Introduction

This document provides guidance and best practices for designing environments that leverage the capabilities of VMware NSX®. It is targeted at virtualization and network architects interested in deploying NSX solutions.

1.1 How to Use This Document and Provide Feedback

This document is organized into several chapters. Chapter 2 to 6 explain the architectural building blocks of NSX as a full stack solution, providing a detail functioning of NSX components, features and scope. They also describe components and functionality utilized for security use cases. These chapters lay the groundwork to help understand and implement the design guidance described in the design chapter.

The design chapter (Chapter 7) examines detailed use cases of network virtualization and recommendations of either best practices or leading practices based on the type of use case or design form factor. It offers guidance for a variety of factors including physical infrastructure considerations, compute node requirements, and variably sized environments from small to enterprise scale.

The performance chapter (Chapter 8) aims at clarifying the myths vs facts about NSX based SDN performances.

This document does not cover installation, and operational monitoring and troubleshooting. For further details, review the complete NSX INSTALLATION AND ADMINISTRATION GUIDES.

Finally starting with this design guide, readers are encouraged to send a feedback to NSXDesignFeedback_AT_groups_vmware_com (convert to email format).

1.2 Networking and Security Today

In the digital transformation era, organizations are increasingly building custom applications to drive core business and gain competitive advantages. The speed with which development teams deliver new applications and capabilities directly impacts the organization's success and bottom line. This exerts increasing pressure on organizations to innovate quickly and makes developers central to this critical mission. As a result, the way developers create apps, and the way IT provides services for those apps, are evolving.

Application Proliferation

With applications quickly emerging as the new business model, developers are under immense pressure to deliver apps in a record time. This increasing need to deliver more apps in a less time can drive developers to use public clouds or open source technologies. These solutions allow them to write and provision apps in a fraction of the time required with traditional methods.

Heterogeneity

Application proliferation has given rise to heterogeneous environments, with application workloads being run inside VMs, containers, clouds, and bare metal servers. IT departments must maintain governance, security, and visibility for

application workloads regardless of whether they reside on premises, in public clouds, or in clouds managed by third-parties.

Cloud-centric Architectures

Cloud-centric architectures and approaches to building and managing applications are increasingly common because of their efficient development environments and fast delivery of applications. These cloud architectures can put pressure on networking and security infrastructure to integrate with private and public clouds. Logical networking and security must be highly extensible to adapt and keep pace with ongoing change.

Against this backdrop of increasing application needs, greater heterogeneity, and the complexity of environments, IT must still protect applications and data while addressing the reality of an attack surface that is continuously expanding.

1.3 NSX Architecture Value and Scope

VMware NSX is designed to address application frameworks and architectures that have heterogeneous endpoints and technology stacks. In addition to vSphere, and VMware public clouds, these environments may include other hypervisors, containers, bare metal operating systems. NSX allows IT and development teams to choose the technologies best suited for their applications. NSX is also designed for management, operations, and consumption by development organizations in addition to IT.

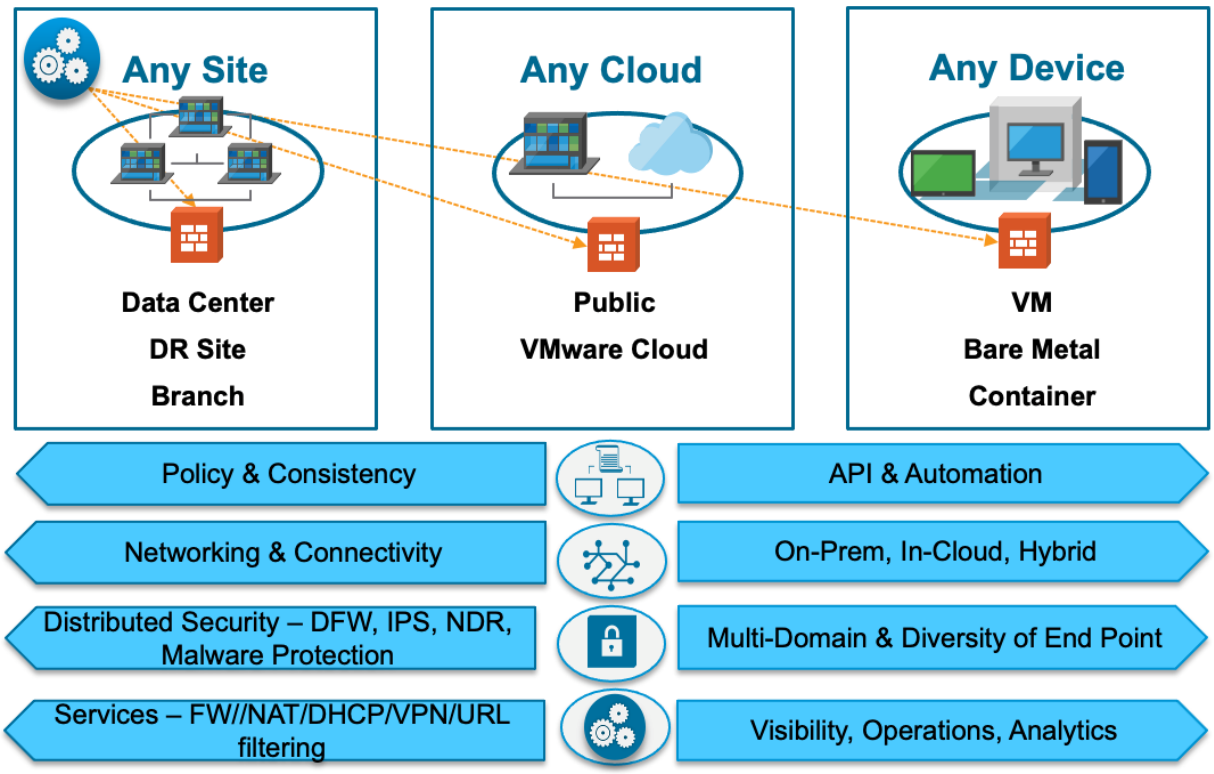


Figure 1-1: NSX Anywhere Architecture

The NSX architecture is designed around four fundamental attributes. FIGURE 1-1: NSX ANYWHERE ARCHITECTURE depicts the universality of those attributes that spans from any site, to any cloud, and to any endpoint device. This enables greater decoupling, not just at the infrastructure level (e.g., hardware, hypervisor), but also at the public cloud and container level; all while maintaining the four key attributes of platform implemented across the domains. NSX architectural value and characteristics of NSX architecture include:

- Policy and Consistency: Allows policy definition once and realizable end state via RESTful API, addressing requirements of today's automated environments. NSX maintains unique and multiple inventories and controls to enumerate desired outcomes across diverse domains.
- Networking and Connectivity: Allows consistent logical switching and distributed routing without being tied to a single compute manager/domain (e.g. vCenter server). The connectivity is further extended across containers and clouds via domain specific implementation while still providing connectivity across heterogeneous endpoints.
- Security and Services: Allows a unified security policy model as with networking connectivity. This enables implementation of services such as load balancer, Edge (Gateway) Firewall, Distributed Firewall, and Network Address Translation cross multiple compute domains. Providing consistent security between VMs and container workloads in private and public clouds is essential to assuring the integrity of the overall framework set forth by security operations.
- Visibility: Allows consistent monitoring, metric collection, and flow tracing via a common toolset across compute domains and clouds. Visibility is essential for operationalizing mixed workloads – VM and container-centric – typically both have drastically different tools for completing similar tasks.

These attributes enable the heterogeneity, app-alignment, and extensibility required to support diverse requirements. Additionally, NSX supports DPDK libraries that offer line-rate stateful services.

Heterogeneity

In order to meet the needs of heterogeneous environments, a fundamental requirement of NSX is to be compute-manager agnostic. As this approach mandates support for multi-hypervisor and/or multi-workloads, a single NSX manager's manageability domain can span multiple vCenters. When designing the management plane, control plane, and data plane components of NSX, special considerations were taken to enable flexibility, scalability, and performance.

The management plane was designed to be independent of any compute manager, including vSphere. The VMware NSX® Manager™ is fully independent; management of the NSX based network functions are accesses directly – either programmatically or through the GUI.

The control plane architecture is separated into two components – a centralized cluster and an endpoint-specific local component. This separation allows the control plane to scale as the localized implementation – both data plane implementation and security enforcement – is more efficient and allows for heterogeneous environments.

The data plane was designed to be normalized across various environments. NSX introduces a host switch that normalizes connectivity among various compute domains, including multiple VMware vCenter® instances, KVM, containers, bare metal servers, and other off premises or cloud implementations. This switch is referred as N-VDS. The functionality of the N-VDS switch was fully implemented in the ESXi VDS 7.0 and later, which allows ESXi customers to take advantage of full NSX functionality without having to change VDS. Regardless of implementation, data plane connectivity is normalized across all platforms, allowing for a consistent experience.

App-aligned

NSX was built with the application as the key construct. Regardless of whether the app was built in a traditional monolithic model or developed in a newer microservices application framework, NSX treats networking and security consistently. This consistency extends across containers and multi-hypervisors on premises, then further into the public cloud.

1.3.1 Containers and Cloud Native Application Integrations with NSX

In an age where a new application is directly tied to business gains, delays in application deployment translate to lost revenue or business opportunity. The current era of digital transformation challenges IT in addressing directives to normalize applications and data security, increase the speed of delivery, and improve application availability. IT administrators realize that a new approach must be taken to support business needs and meet timelines. Architecturally solving the problem by explicitly defining connectivity, security, and policy as a part of the application lifecycle is essential. Programmatic and automatic creation of network and switching segments based on application-driven infrastructure is the only way to meet the requirements of these newer architectures.

NSX is designed to address the needs of these emerging application frameworks and architectures with heterogeneous endpoints and technology stacks. NSX allows IT and development teams to choose the technologies best suited for their particular applications or use cases without compromising consistent security and operations. NSX provides a common framework to manage and increase the visibility of environments that contain everything from physical servers to VMs and containers. As developers embrace newer technologies like containers and the percentage of workloads running in public clouds increases, network virtualization must expand to offer a full range of networking and security services (e.g., LB, NAT, DFW, etc.) native in these environments. NSX solutions provide networking, security, visibility, and Load Balancing services for container-based application platforms based on Kubernetes such as Tanzu Kubernetes Grid (TKG) and Openshift, as well as managed Kubernetes offerings running in the public cloud (Amazon's Elastic Kubernetes Service – EKS, Amazon Kubernetes Service – AKS, and Google Kubernetes Engine GKE).

The NSX portfolio includes multiple products with tight integration to deliver end-to-end Kubernetes network and security services.

Antrea is an open-source Kubernetes-native networking and security solution that can be installed in clusters running in private or public clouds and bare metal servers. The Antrea data plane implementation is based on Open vSwitch. This choice makes it highly portable across Linux and Windows operating systems and allows hardware offloading. Antrea provides a comprehensive security policy

model that builds upon Kubernetes network policies by introducing the concepts of policy tiering, rule priorities, and cluster-level policies. Antrea includes troubleshooting and monitoring tools for visibility and diagnostic capabilities such as packet tracing, policy analysis, and flow inspection. Antrea instances running on multiple clusters can be integrated with NSX to provide a consistent policy model and centralized visibility across clusters, clouds, and workload form factors (containers, VM, bare metal). Antrea is the default Container Network Interface (CNI) for Tanzu guest clusters and TKG.

VMware NSX Advanced Load Balancer (formerly Avi) dispenses Kubernetes load balancing and ingress capabilities. The NSX ALB Kubernetes Ingress Services is optimized for North-South traffic management, local and global server load balancing (GSLB), performance monitoring, application security (WAF), and DNS/IPAM management. The NSX ALB Kubernetes Ingress Services provides operational consistency regardless of which on-prem, private-cloud, or public-cloud environment the Kubernetes cluster is running on.

The NSX Container Plug-in (NCP) provides direct integration with several vSphere based private cloud environments where containerized applications could reside. The NSX Container Plugin leverages the Container Network Interface (CNI) to interface with the container and allows NSX to directly orchestrate networking, policy, and load balancing. NCP is frequently used with application platforms and enterprise distributions of k8s, notably VCF with Tanzu, Tanzu Application Services, and RedHat Open Shift.

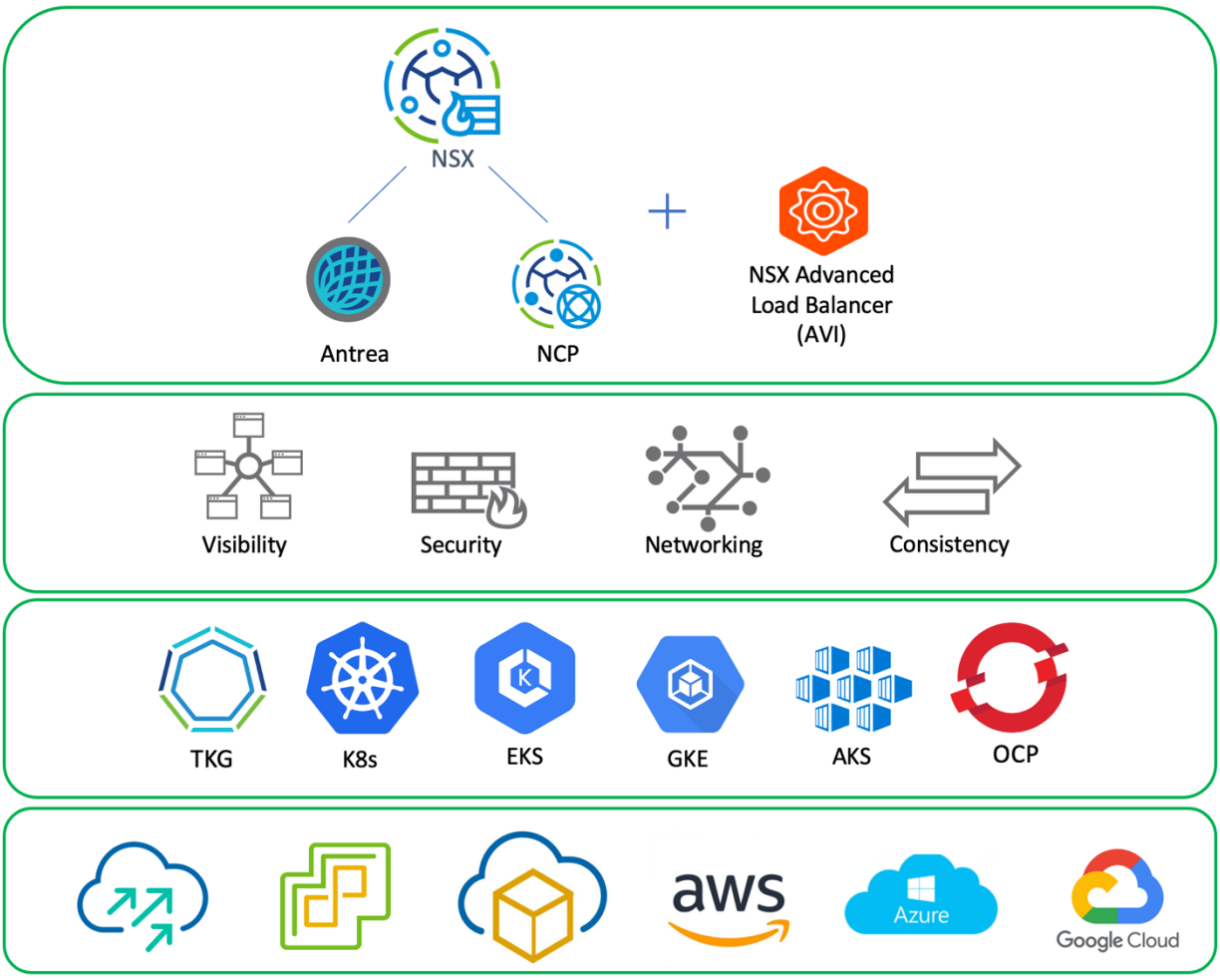


Figure 1-2: Programmatic Integration with Various PaaS and CaaS

1.3.2 Role of NSX in the VMware multi-cloud platform

A multi-cloud deployment model relies on the use of more than one public or private cloud service provider for compute, network, and storage resources. VMware delivers a software defined infrastructure, Platform-as-a-Service (PaaS) and management stack that can be layered on top of any physical hardware layer on any cloud or data center. The software stack is based on VMware Cloud Foundation (VCF) and includes vSphere, VSAN, and NSX as its core components. It provides a unified approach to building, running and managing traditional and modern apps on any cloud. This unique architectural approach provides a single platform that can function across all application types and multiple cloud environments. NSX is a key strategic asset for the VMware multi-cloud platform. Limited and fragmented public cloud native network and security services are augmented by rich and uniform enterprise-grade capabilities across any cloud.

The cloud operating model cuts across the traditional silos – networking, network security, load balancing and endpoint protection solutions are designed, deployed

and managed by an increasingly integrated team or a set of integrated processes in the form of automation and pipelines. In this world network and security become software that is defined in advance and is integrated into the customer CI/CD pipeline. It is accessed programmatically by high level, declarative or intent based APIs, and it is oriented to serving the needs of the application. The NSX APIs allow customers to embrace the cloud operation model, where an entire workload, including all network and security services, is launched with no human touch, while not sacrificing key enterprise functionalities.

Virtualized networking separates the logical connectivity policies from the physical transport layer. In a multi-cloud world, the transport layer is increasingly less available because it is embedded into another cloud or running on the public Internet. Thus, virtual networking is essential for making the VMware stack run on any provider's hardware, for seamlessly connecting the heterogeneous clouds of a modern enterprise and present a uniform consumption model across different providers.

Organizations are looking for a multi-cloud platform that delivers best in class workload security. Thanks to NSX, VMware clouds can transparently insert services, to protect, manage and operationalize applications at scale, and to have an intimate understanding of the end user and the application context. This allows for unique features such as "Virtual patching" via the NSX distributed IPS that protects individual vulnerable workloads before the application of a security patch.

While all the VMware clouds run the same code base, some go a step further in term of simplification and application alignment. The NSX that is presented to VMware Cloud customers is much simpler than the premise-based version because the underlying topology is controlled by VMware and all that is required is specify application-level policies. Since the same software stack is deployed on any VMware cloud, operations such as the vMotion of a workload and its attached firewall/security policy between private and public clouds are available.

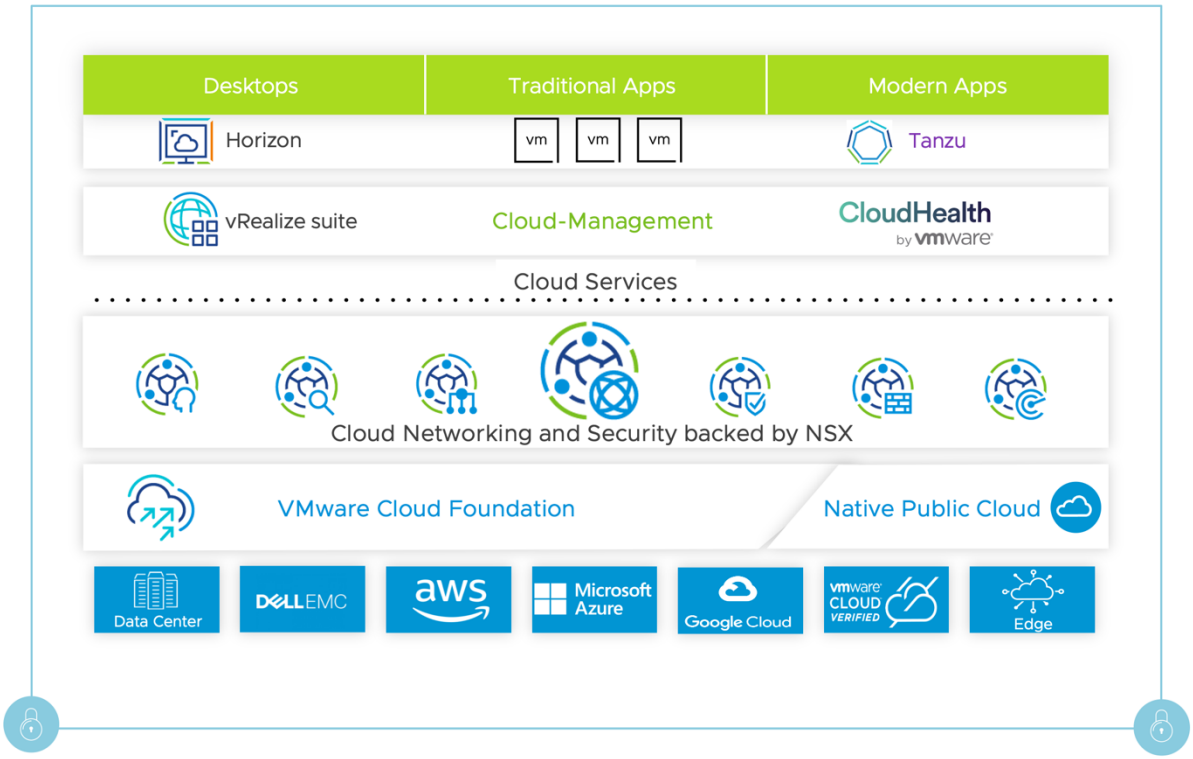


Figure 1-3: NSX role in the VMware Multi-cloud platform

2 NSX Architecture Components

NSX reproduces the complete set of networking services (e.g., switching, routing, firewalling, load balancing, QoS) in software. These services can be programmatically assembled in arbitrary combinations to produce unique, isolated virtual networks in a matter of seconds. NSX works by implementing three separate but integrated planes: management, control, and data. The three planes are implemented as sets of processes, modules, and agents residing on two types of nodes: manager appliance and transport.

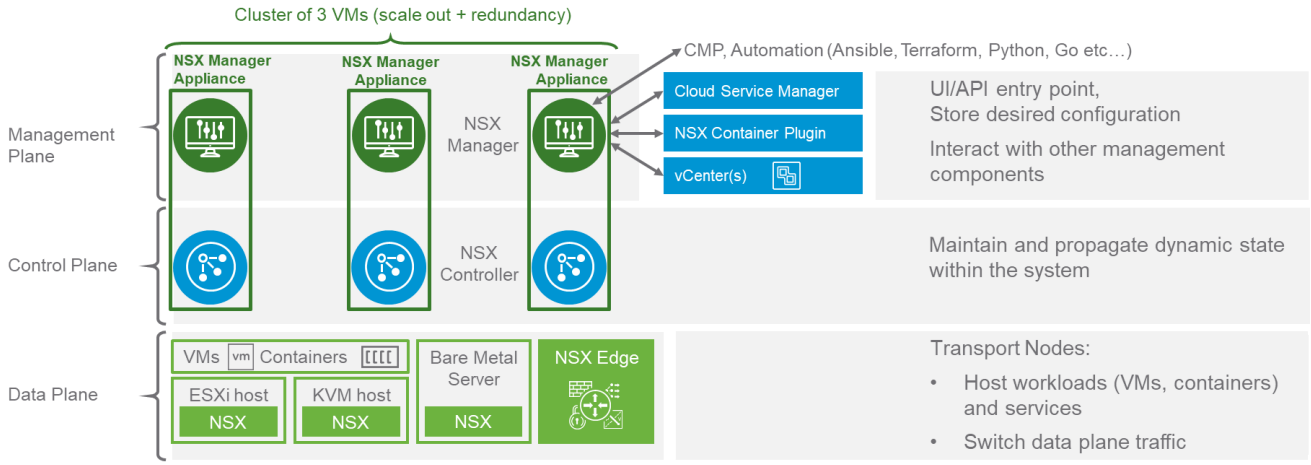


Figure 2-1: NSX Architecture and Components

2.1 Management Plane and Control Plane

2.1.1 Management Plane

The management plane provides an entry point to the system for API as well NSX graphical user interface. It is responsible for maintaining user configuration, handling user queries, and performing operational tasks on all management, control, and data plane nodes.

The NSX Manager implements the management plane for the NSX ecosystem. It provides an aggregated system view and is the centralized network management component of NSX. NSX Manager provides the following functionality:

- Serves as a unique entry point for user configuration via RESTful API (CMP, automation) or NSX user interface.
- Responsible for storing desired configuration in its database. The NSX Manager stores the final configuration request by the user for the system. This configuration will be pushed by the NSX Manager to the control plane to become a realized configuration (i.e., a configuration effective in the data plane).
- Retrieves the desired configuration in addition to system information (e.g., statistics).

- Provides ubiquitous connectivity, consistent enforcement of security and operational visibility via object management and inventory collection and for multiple compute domains – up to 16 vCenter servers, container orchestrators (TAS/TKGI & OpenShift) and clouds (AWS and Azure)

Data plane components or transport node run a management plane agent (MPA) that connects them to the NSX Manager.

2.1.2 Control Plane

The control plane computes the runtime state of the system based on configuration from the management plane. It is also responsible for disseminating topology information reported by the data plane elements and pushing stateless configuration to forwarding engines.

NSX splits the control plane into two parts:

- Central Control Plane (CCP) – The CCP is implemented as a cluster of virtual machines called CCP nodes. The cluster form factor provides both redundancy and scalability of resources. The CCP is logically separated from all data plane traffic, meaning any failure in the control plane does not affect existing data plane operations. User traffic does not pass through the CCP Cluster.
- Local Control Plane (LCP) – The LCP runs on transport nodes. It is adjacent to the data plane it controls and is connected to the CCP. The LCP is responsible for programing the forwarding entries and firewall rules of the data plane.

2.1.3 NSX Manager Appliance

Instances of the NSX Manager and NSX Controller are bundled in a virtual machine called the NSX Manager Appliance. In the releases prior to 2.4, there were separate appliances based on the roles, one management appliance and 3 controller appliances, so total four appliances to be deployed and managed for NSX. Starting 2.4, the NSX manager, NSX policy manager and NSX controller elements will co-exist within a common VM. Three unique NSX appliance VMs are required for cluster availability. NSX relies on a cluster of three such NSX Manager Appliances for scaling out and for redundancy. This three-node cluster relies on a majority of the appliances in the cluster to be available and as such, attention should be paid to placement of these appliances for availability The NSX Manager stores all of its information in an in-memory database immediately which is synchronized across the cluster and written to disk, configuration or read operations can be performed on any appliance. The performance requirements of the NSX Manager Appliance's disk is significantly reduced as most operations occur in memory.

The benefits with this converged manager appliance include less management overhead with reduced appliances to manage. And a potential reduction in the total amount of resources (CPU, memory and disk). With the converged manager appliance, one only need to consider the appliance sizing once.

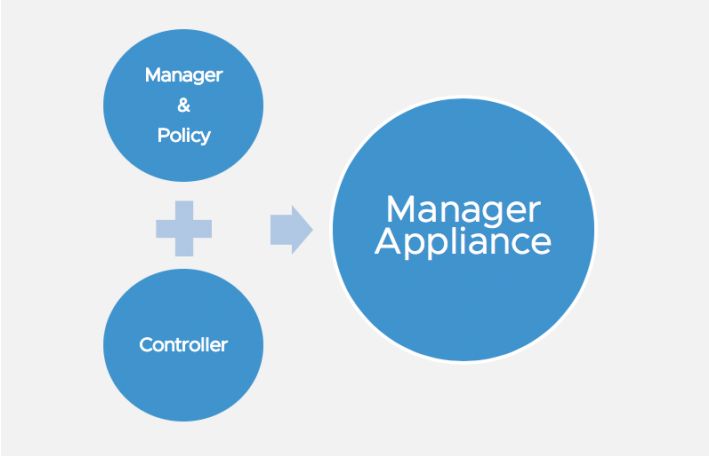


Figure 2-2: NSX Manager and Controller Consolidation

Each appliance has a dedicated IP address and its manager process can be accessed directly or through a load balancer. Optionally, the three appliances can be configured to maintain a virtual IP address which will be serviced by one appliance selected among the three. The design consideration of NSX Manager appliance is further discussed in CHAPTER 7.

2.2 Data Plane

The data plane performs stateless forwarding or transformation of packets based on tables populated by the control plane. It reports topology information to the control plane and maintains packet level statistics. The hosts running the local control plane daemons and forwarding engines implementing the NSX data plane are called transport nodes. Transport nodes are running an instance of the NSX virtual switch called the NSX Virtual Distributed Switch, or N-VDS.

On ESXi platforms, the N-VDS is built on the top of the vSphere Distributed Switch (VDS). In fact, the N-VDS is so close to the VDS that NSX 3.0 introduced the capability of installing NSX directly on the top of a VDS on ESXi hosts. For all other kinds of transport node, the N-VDS is based on the platform independent Open vSwitch (OVS) and serves as the foundation for the implementation of NSX in other environments (e.g., cloud, containers, etc.).

As represented in FIGURE 2-1: NSX ARCHITECTURE AND COMPONENTS, there are two main types of transport nodes in NSX:

- **Hypervisor Transport Nodes**: Hypervisor transport nodes are hypervisors prepared and configured for NSX. NSX provides network services to the virtual machines running on those hypervisors. NSX currently supports VMware ESXi and KVM hypervisors.
- **Edge Nodes**: VMware NSX Edge nodes are service appliances dedicated to running centralized network services that cannot be distributed to the hypervisors. They can be instantiated as a bare metal appliance or in virtual machine form factor. They are grouped in one or several clusters, representing a pool of capacity. It is important to remember that an Edge

Node does not represent a service itself but just a pool of capacity that one or more services can consume.

2.3 NSX Consumption Model

A user can interact with the NSX platform through the Graphical User Interface or the REST API framework. Two GUI & REST API options are available to interact with the NSX Manager:

1) Policy Mode
   - Default UI mode
   - Any new deployment should use the Policy Mode
   - API accessed via URI which start with /policy/api
2) Manager Mode
   - Manager UI is disabled by default
   - Deprecated as of NSX version 3.2
   - Temporarily address deployments created via manager mode and upgraded from previous releases
   - NSX version 3.2 introduces the Manager to Policy promotion tool
   - API accessed via URI which start with /api

2.3.1 NSX Policy API Framework

NSX Policy API framework provide an outcome driven config option. This allows a single API call to configure multiple NSX networking & security objects for an application deployment. This is more applicable for customers using automation and for CMP plugins. Some of the main benefits of declarative API framework are:

- Outcome driven: Reduces the number of configuration steps by allowing a user to describe desired end-goal (the "what"), and letting the system figure out "how" to achieve it. This allows users to utilize user-specified names, not system generated IDs
- Order Independent: create/update/delete in any order and always arrive at the same consistent result
- Prescriptive: reduces potential for user error with built-in dependency checks
- Policy Life Cycle Management: Simpler with single API call. Toggle marked-to-delete flag in the JSON request body to manage life cycle of entire application topology.

The NSX API documentation can be accessible directly from the NSX Manager UI, under Policy section within API documentation, or it can be accessed from CODE.VMWARE.COM.

The following examples walks you through the policy API examples for two of the customer scenarios:

2.3.2 Policy API Usage Example 1- Templatize and Deploy 3-Tier Application Topology

This example provides how Policy API helps user to create the reusable code template for deploying a 3-Tier APP shown in the figure, which includes Networking, Security & Services needed for the application.

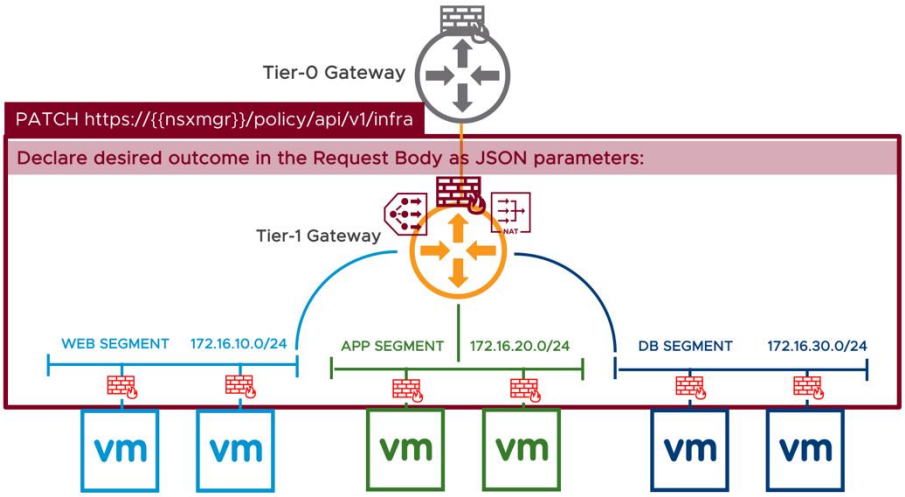


Figure 2-3: NSX Policy API - Infra

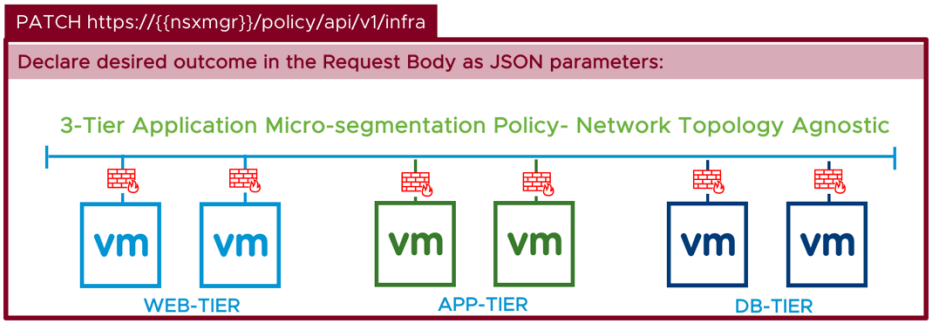
The desired outcome for deploying the application, as shown in the figure above, can be defined using JSON. Once JSON request body is defined to reflect the desired outcome, then API & JSON request body can be leveraged to automate following operational workflows:

- Deploy entire topology with single API and JSON request body.
- The same API/JSON can be further leveraged to templatize and reuse to deploy same application in different environment (PROD, TEST and DEV).
- Handle life cycle management of entire application topology by toggling the "marked_for_delete" flag in the JSON body to true or false.

See details of this at EXAMPLE 1.

2.3.3 Policy API Usage Example 2- Application Security Policy Lifecycle Management

This example demonstrates how a security admin can leverage declarative API to manage the life cycle of security configuration, grouping, and micro-segmentation policy for a given 3-tier application. The following figure depicts the entire application topology and the desired outcome to provide zero trust security model for an application.



The policy model can define the desired outcome by specifying grouping and micro-segmentation polices using JSON. It uses single API call with a JSON request body to automate following operational workflows:

- Deploy white-list security policy with single API and JSON request body.
- The same API/JSON can further leveraged to templatize and reuse to secure same application in different environment (PROD, TEST and DEV).
- Handle life cycle management of entire application topology by toggling the "marked_for_delete" flag in the JSON body to true or false.

2.3.4 When to use Policy vs Manager UI/API

VMware recommendation is to use NSX Policy UI going forward as all the new features are implemented only on Policy UI/API and the Manager UI/API has been deprecated.

The following table further highlights the feature transition map to the Policy UI/API model.

Policy Mode	Manager Mode
All new deployments should use Policy mode.	Deployments which were created using the advanced interface, for example, upgrades from versions where policy mode was not available. Customers should plan the transition to Policy Mode, possibly via the manager to policy promotion tool.
New deployments which integrate with cloud automation platforms, automation tools, or the NSX Container plug-in (NCP). All the northbound integrations between VMware	Deployments which in previous versions were integrated with cloud automation platforms, NCP, or custom orchestration tools via the manager api. Customers should plan the transition to Policy Mode. Consult the specific guidance for NCP and VMware

products and NSX now support policy mode.	Integrated Openstack (VIO). Migration of manager api objects managed by vRealize Automation is not yet supported.
Networking features available in Policy mode only: ● DNS Services and DNS Zones ● VPN ● Federation	Networking features available in Manager mode only: ● N/A
Security features available in Policy mode only: ● Context Profiles ○ L7 applications ○ FQDN ● Identity Firewall ● TLS Inspection ● IDS/IPS ● New Distributed Firewall and Gateway Firewall Layout ○ Categories ○ Auto service rules ○ Drafts ● Endpoint Protection ● Network Introspection (East-West Service Insertion)	Security features available in Manager mode only: ● Bridge Firewall

It is recommended that whichever mode is used to create objects (Policy or Manager) be the only mode used (if the Manager Mode objects are required, create all objects in Manager mode). Do not alternate use of the modes or there will be unpredictable results. Note that the default mode for the NSX Manager is Policy mode. When working in an installation where all objects are new and created in Policy mode, the Manager mode option will not be visible in the UI. For details on switching between modes, please see the NSX DOCUMENTATION.

2.3.5 NSX Logical Object Naming relationship between manager and policy mode

The name of some of the networking and security logical objects in the Manager API/Data model have changed in the new policy model. The table below provides the before and after naming side by side for those NSX Logical objects. This change only affects the name for the given NSX object, but conceptually and functionally it is the same as before.

	Manager API/UI Object	Policy API Object

Networking		
	Logical switch	Segment
	T0 Logical Router	Tier-0 Gateway
	T1 Logical Router	Tier-1 Gateway
	Centralized Service Port	Service Interface
Security	Manager API/UI Object	Policy API Object
	NSGroup	Group
	IP Sets, MAC Sets	Group
	Firewall Section	Security-Policy
	Edge Firewall	Gateway Firewall

3 NSX Logical Switching

This chapter details how NSX creates virtual Layer 2 networks, called segments, to provide connectivity between its services and the different virtual machines in the environment.

3.1 The NSX Virtual Switch

A transport node is, by definition, a device implementing the NSX data plane. The software component running this data plane is a virtual switch, responsible for forwarding traffic between logical and physical ports on the device. The NSX virtual switch is called the NSX Virtual Distributed Switch, or N-VDS. On ESXi hosts, the N-VDS implementation is derived from VMware vSphere® Distributed Switch™ (VDS). With any other kind of transport node (KVM hypervisors, Edges, bare metal servers, cloud VMs etc.) the N-VDS implementation is derived from the Open vSwitch (OVS).

NSX 3.0 introduced a new model for its ESXi transport nodes where the NSX software components can be directly installed on the top of an existing VDS. This has several benefits such as using existing model of VDS for non-overlay traffic and avoiding the migration of VMkernel interfaces to N-VDS.

Representations of the NSX virtual switch in this document will be labeled "N-VDS or VDS with NSX" for each scenario where the transport node represented can be an ESXi host. Note that while the new VDS-based model greatly simplifies NSX consumption on the ESXi platform, it has very little if any impact on NSX-based designs: both N-VDS and VDS running NSX provide the same NSX capabilities, only with a different representation in vCenter. Operational details on how to run NSX on VDS are out of scope of this document, but simplification in term of VMkernel interface management that this new model brings will be called out in the design section. We thus recommend deploying NSX on the top of a VDS on ESXi hosts instead of using an N-VDS for greenfield deployments starting with supported ESXi and vSphere versions. The N-VDS is still the only virtual switch available on platforms other than ESXi.

3.1.1 Segments and Transport Zones

In NSX, virtual layer 2 domains are called segments. There are two kinds of segments:

- VLAN backed segments
- Overlay backed segments

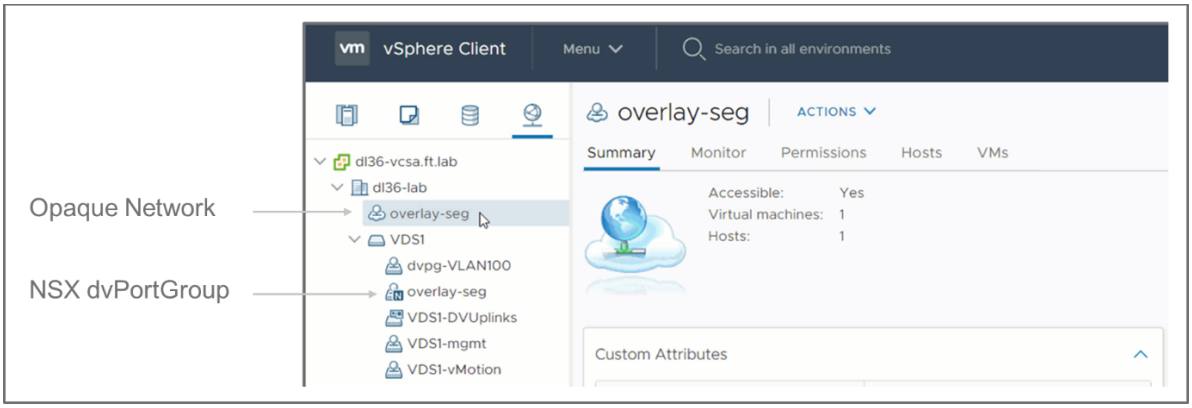
A VLAN backed segment is a layer 2 broadcast domain that is implemented as a traditional VLAN in the physical infrastructure. That means that traffic between two VMs on two different hosts but attached to the same VLAN backed segment will be carried over a VLAN between the two hosts in native IEEE encapsulation. The resulting constraint is that an appropriate VLAN needs to be provisioned in the physical infrastructure for those two VMs to communicate at layer 2 over a VLAN backed segment.

On the other hand, two VMs on different hosts and attached to the same overlay backed segment will have their layer 2 traffic carried by tunnel between their hosts. This IP tunnel is instantiated and maintained by NSX without the need for any segment specific configuration in the physical infrastructure, thus decoupling NSX virtual networking from this physical infrastructure.

Starting in NSX version 3.2, security only deployments can have VMs connected to traditional vSphere dvpg too. The security functionalities available are equivalent to those on NSX VLAN backed segments, but overlay backed segment cannot be deployed on the same ESXi host where NSX security features have been enabled for vSphere dvpgs.

Note: representation of NSX segments in vCenter

This design document will only use the term "segment" when referring to the NSX virtual Layer broadcast domain. Note however that in the vCenter UI, those segments will appear as "opaque networks" on host configured with an N-VDS, and as NSX dvportgroups on host configured with a VDS. Below is a screenshot representing both possible representation:



Check the following KB article for more information on the impact of this difference in representation: https://kb.vmware.com/s/article/79872

Segments are created as part of an NSX object called a transport zone. There are VLAN transport zones and overlay transport zones. A segment created in a VLAN transport zone will be a VLAN backed segment, while, as you can guess, a segment created in an overlay transport zone will be an overlay backed segment. NSX transport nodes attach to one or more transport zones, and as a result, they gain access to the segments created in those transport zones. Transport zones can thus be seen as objects defining the scope of the virtual network because they provide access to groups of segments to the hosts that attach to them, as illustrated in FIGURE 3-1 below:

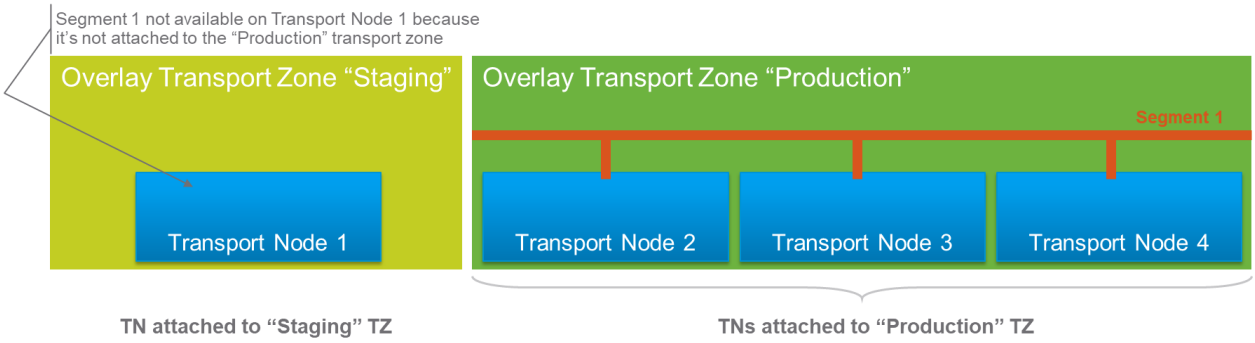


Figure 3-1: NSX Transport Zone

In above diagram, transport node 1 is attached to transport zone "Staging", while transport nodes 2-4 are attached to transport zone "Production". If one creates a segment 1 in transport zone "Production", each transport node in the "Production" transport zone immediately gain access to it. However, this segment 1 does not extend to transport node 1. The span of segment 1 is thus defined by the transport zone "Production" it belongs to.

Few additional points related to transport zones and transport nodes:

- Multiple virtual switches, N-VDS, VDS (with or without NSX) or VSS, can coexist on a ESXi transport node; however, a given pNIC can only be associated with a single virtual switch. This behavior is specific to the VMware virtual switch model, not to NSX.
- An NSX virtual switch (N-VDS or VDS with NSX) can attach to a single overlay transport zone and multiple VLAN transport zones at the same time.
- A transport node can have multiple NSX virtual switches. A transport node can thus attach to multiple overlays and VLAN transport zones.
- A transport zone can only be attached to a single NSX virtual switch on a given transport node. In other words, two NSX virtual switches on the same transport node cannot be attached to the same transport zone.
- Edge transport node-specific points:
  - An edge transport node can only have one N-VDS attached to an overlay transport zone.
  - If multiple VLAN segments are backed by the same VLAN ID (across all the VLAN transport zones of an edge N-VDS), only one of those segments will be "realized" (i.e. working effectively).

Please see additional consideration at RUNNING A VDS PREPARED FOR NSX ON ESXI HOSTS.

3.1.2 Uplink vs. pNIC

NSX introduces a clean differentiation between the physical uplinks of the host (aka pNICs, or vmnics on ESXi hosts) and the uplinks of the NSX virtual switch. The uplinks of the NSX virtual switch are logical constructs that can be mapped to one

pNIC or multiple pNICs bundled into a link aggregation group (LAG).FIGURE 3-2 illustrates the difference between an uplink and a pNIC:

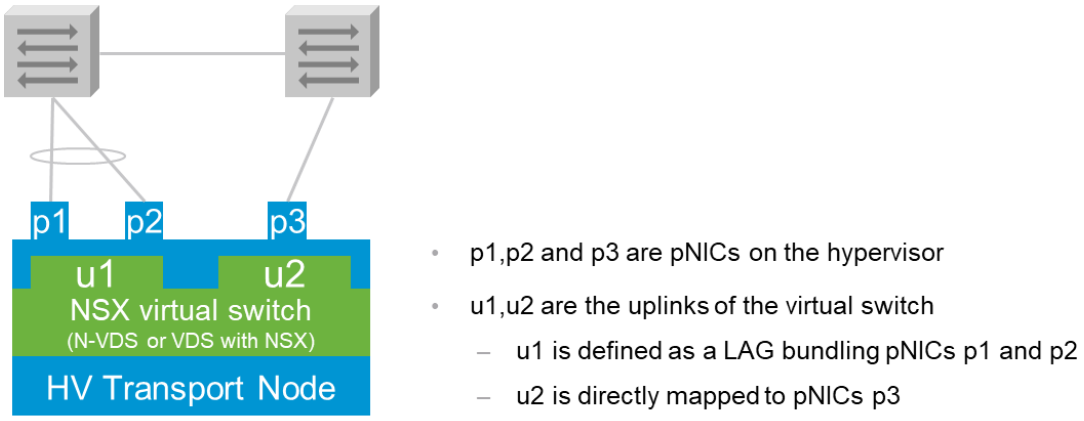


Figure 3-2: N-VDS Uplinks vs. Hypervisor pNICs

In this example, a single virtual switch with two uplinks is defined on the hypervisor transport node. One of the uplinks is a LAG, bundling physical port p1 and p2, while the other uplink is only backed by a single physical port p3. Both uplinks look the same from the perspective of the virtual switch; there is no functional difference between the two.

Note that the example represented in this picture is by no means a design recommendation, it's just illustrating the difference between the virtual switch uplinks and the host physical uplinks.

3.1.3 Teaming Policy

The teaming policy defines how the NSX virtual switch uses its uplinks for redundancy and traffic load balancing. There are two main options for teaming policy configuration:

- Failover Order – An active uplink is specified along with an optional list of standby uplinks. Should the active uplink fail, the next available uplink in the standby list takes its place immediately. This policy results in an active/standby use of the uplinks.
- Load Balanced Source Port/Load Balance Source Mac Address – Traffic is distributed across a specified list of active/active uplinks.
  - The "Load Balanced Source Port" policy maps a VM's virtual interface to an uplink of the host. Traffic sent by this virtual interface will leave the host through this uplink only, and traffic destined to this virtual interface will necessarily enter the host via this uplink.
  - The "Load Balanced Source Mac Address" goes a little bit further in term of granularity for rare scenario where a virtual interface that can source traffic from different mac addresses. Here, two frames sent by the same virtual interface could be associated to different uplinks based on their source mac address.

The teaming policy only defines how the NSX virtual switch balances traffic across its uplinks. The uplinks can in turn be individual pNICs or LAGs (as seen in the previous section.) Note that a LAG uplink has its own hashing options, however, those hashing options only define how traffic is distributed across the physical members of the LAG uplink, whereas the teaming policy define how traffic is distributed between NSX virtual switch uplinks.

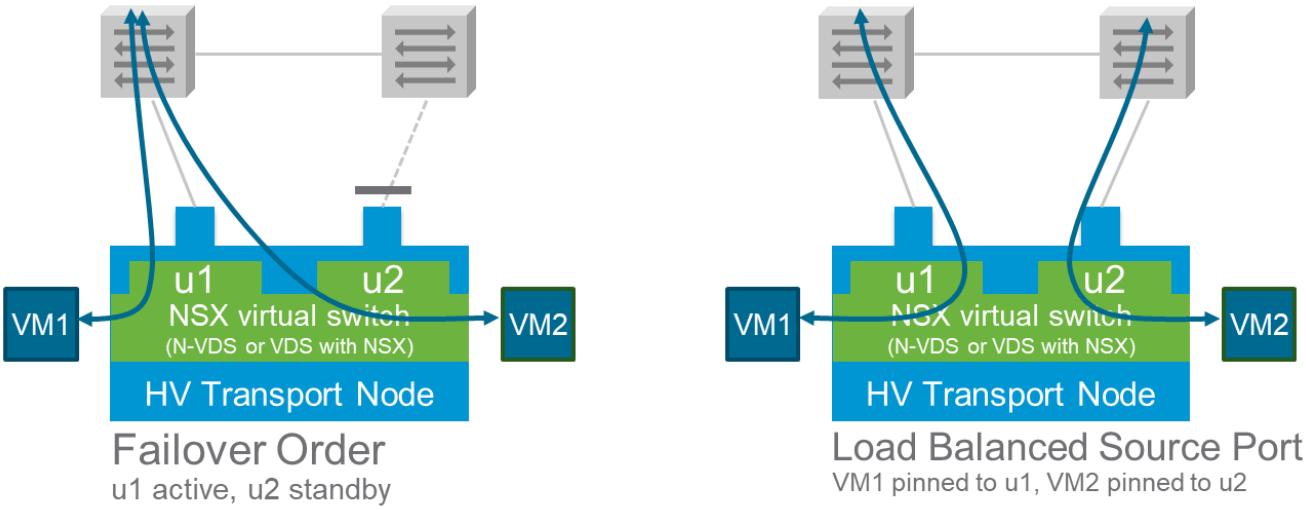


Figure 3-3: N-VDS Teaming Policies

FIGURE 3-3 presents an example of the failover order and source teaming policy options, illustrating how the traffic from two different VMs in the same segment is distributed across uplinks. The uplinks of the virtual switch could be any combination of single pNICs or LAGs; whether the uplinks are pNICs or LAGs has no impact on the way traffic is balanced between uplinks. When an uplink is a LAG, it is only considered down when all the physical members of the LAG are down. When defining a transport node, the user must specify a default teaming policy that will be applicable by default to the segments available to this transport node.

3.1.3.1 ESXi Hypervisor-specific Teaming Policy

ESXi hypervisor transport nodes allow defining more specific teaming policies, identified by a name, on top of the default teaming policy. It's called "named teaming policies" which can override the default teaming policy for some specific VLAN backed segments. Overlay backed segments always follow the default teaming policy. This capability is typically used to steer precisely infrastructure traffic from the host to specific uplinks.

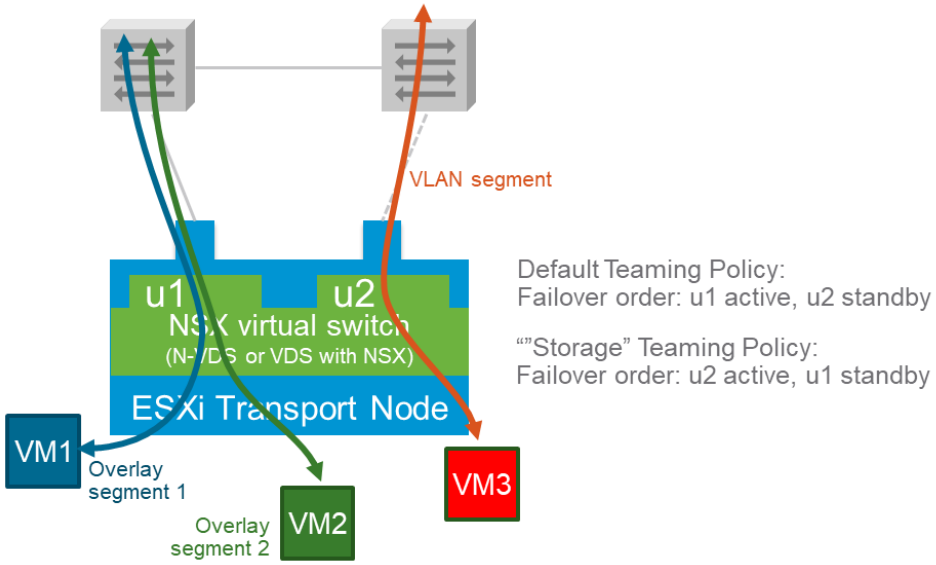


Figure 3-4: Named Teaming Policy

In the above FIGURE 3-4, the default failover order teaming policy specifies u1 as the active uplink and u2 as the standby uplink. By default, all the segments are thus going to send and receive traffic on u1. However, an additional failover order teaming policy called "Storage" has been added, where u2 is active and u1 standby. The VLAN segment where VM3 is attached can be mapped to the "Storage" teaming policy, thus overriding the default teaming policy for the VLAN traffic consumed by this VM3. Sometimes, it might be desirable to only send overlay traffic on a limited set of uplinks. This can also be achieved with named teaming policies for VLAN backed segment, as represented in the FIGURE 3-5 below:

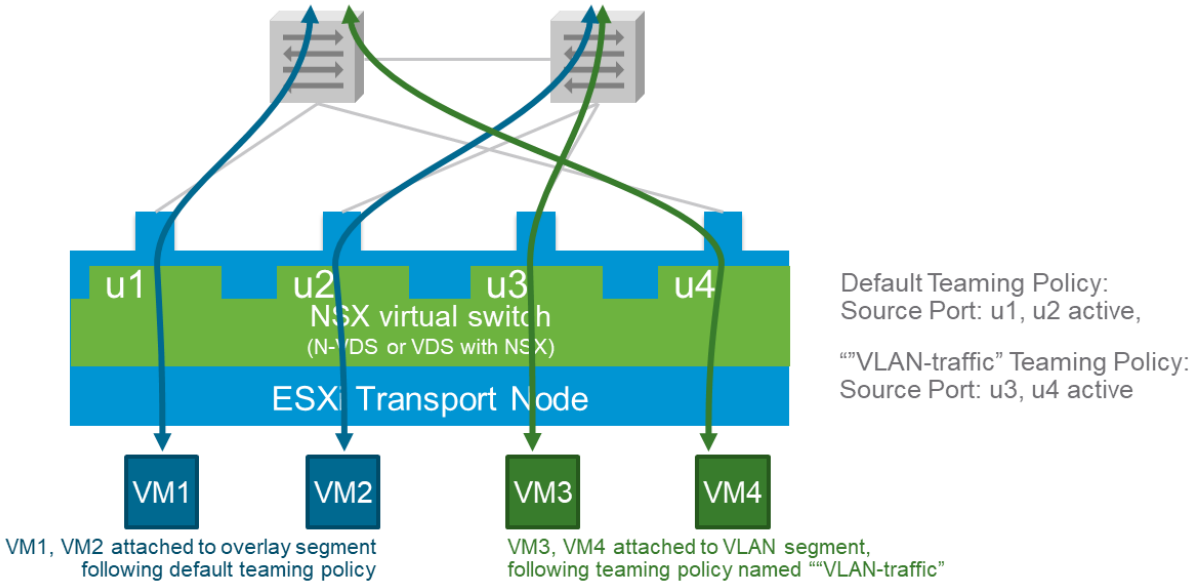


Figure 3-5: Other named teaming policy use case

Here, the default teaming policy only includes uplinks u1 and u2. As a result, overlay traffic is constrained to those uplinks. However, an additional teaming policy named "VLAN-traffic" is configured for load balancing traffic on uplink u3 and u4. By mapping VLAN segments to this teaming policy, overlay and VLAN traffic are segregated.

3.1.3.2 KVM Hypervisor teaming policy capabilities

KVM hypervisor transport nodes can only have a single LAG and only support the failover order default teaming policy; the load balance source teaming policies and named teaming policies are not available for KVM. A LAG must be configured for more than one physical uplink to be active on an N-VDS on a KVM hypervisor.

3.1.4 Uplink Profile

As mentioned earlier, a transport node includes at least one NSX virtual switch, implementing the NSX data plane. It is common for multiple transport nodes to share the exact same NSX virtual switch configuration. It is also very difficult from an operational standpoint to configure (and maintain) multiple parameters consistently across many devices. For this purpose, NSX defines a separate object called an uplink profile that acts as a template for the configuration of a virtual switch. The administrator can this way create multiple transport nodes with similar virtual switches by simply pointing to a common uplink profile. Even better, when the administrator modifies a parameter in the uplink profile, it is automatically updated in all the transport nodes following this uplink profile.

The following parameters are defined in an uplink profile:

- The transport VLAN used for overlay traffic. Overlay traffic will be tagged with the VLAN ID specified in this field.

- The MTU of the uplinks. NSX will assume that it can send overlay traffic with this MTU on the physical uplinks of the transport node without any fragmentation by the physical infrastructure.
- The name of the uplinks and the LAGs used by the virtual switch. LAGs are optional of course, but if you want to define some, you can give them a name, specify the number of links and the hash algorithm they will use.
- The teaming policies applied to the uplinks (default and named teaming policies)

The virtual switch uplinks defined in the uplink profile must be mapped to real, physical uplinks on the device becoming a transport node.

FIGURE 3-6 shows how a transport node "TN1" is created using the uplink profile "UP1".

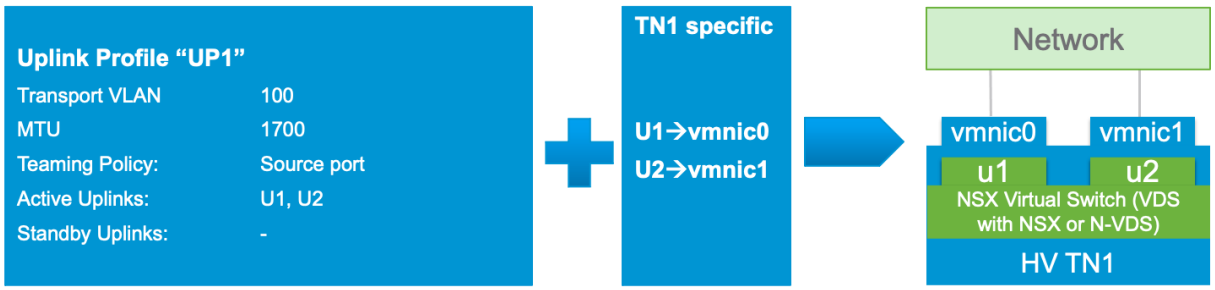


Figure 3-6: Transport Node Creation with Uplink Profile

The uplinks U1 and U2 listed in the teaming policy of the uplink profile UP1 are just variable names. When transport node TN1 is created, some physical uplinks available on the host are mapped to those variables. Here, we're mapping vmnic0 to U1 and vmnic1 to U2. If the uplink profile defined LAGs, physical ports on the host being prepared as a transport node would have to be mapped to the member ports of the LAGs defined in the uplink profile.

The benefit of this model is that we can create an arbitrary number of transport nodes following the configuration of the same uplink profile. There might be local differences in the way virtual switch uplinks are mapped to physical ports. For example, one could create a transport node TN2 still using the same UP1 uplink profile, but mapping U1 to vmnic3 and U2 to vmnic0. Then, it's possible to change the teaming policy of UP1 to failover order and setting U1 as active and U2 as standby. On TN1, this would lead vmnic0 as active and vmnic1 as standby, while TN2 would use vmnic3 as active and vmnic0 as standby.

If uplink profiles allow configuring the virtual switches of multiple transport nodes in a centralized fashion, they also allow for very granular configuration if needed. Suppose now that we want to turn a mix of ESXi host and KVM hosts into transport nodes. UP1 defined above cannot be applied to KVM hosts because those only support the failover order policy. The administrator can simply create an uplink profile specific to KVM hosts, with a failover order teaming policy, while keeping an uplink profile with a source teaming policy for ESXi hosts, as represented in FIGURE 3-7 below:

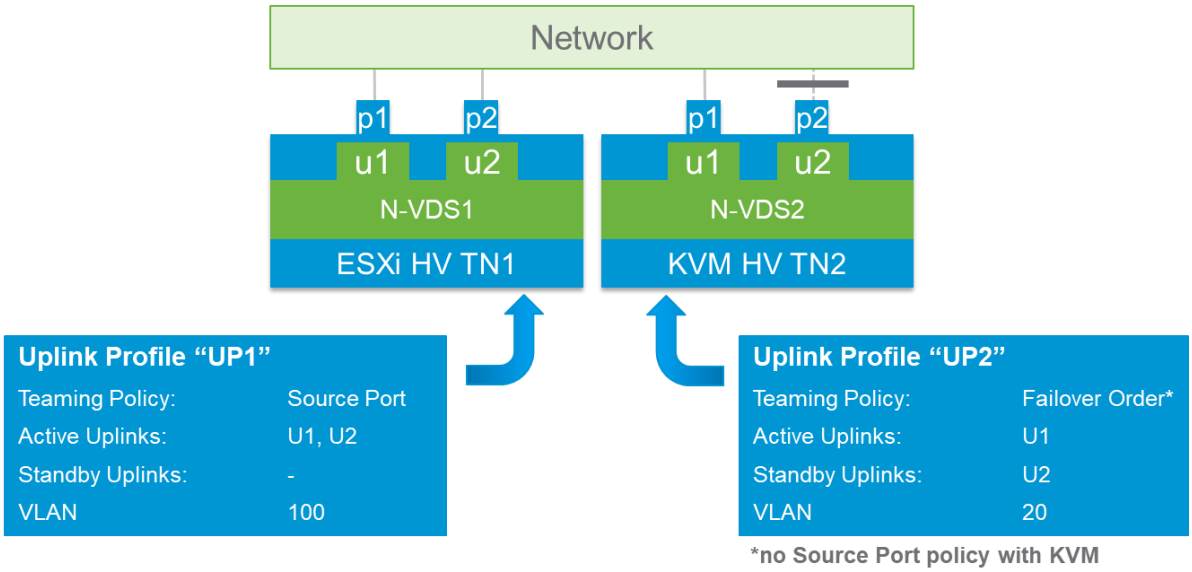


Figure 3-7: Leveraging Different Uplink Profiles

If NSX had a single centralized configuration for all the hosts, we would have been forced to fall back to the lowest common denominator failover order teaming policy for all the hosts.

The uplink profile model also allows for different transport VLANs on different hosts. This can be useful when the same VLAN ID is not available everywhere in the network, for example, the case for migration, reallocation of VLANs based on topology or geo-location change.

When running NSX on VDS, the LAG definition and the MTU fields of the uplink profile are now directly defined on the VDS, controlled by vCenter. It is still possible to associate transport node based on N-VDS and transport nodes based on VDS to the same uplink profile. It's just that the LAG definition and the MTU will be ignored on the VDS-based transport node.

3.1.5 Transport Node Profile

The Tansport Node Profile (in short TNP) is a template for creating a transport node that can be applied to a group of hosts in a single shot. Just assume that there is a cluster with several hosts with the same configuration as the one represented in FIGURE 3-6 above. The TNP would capture the association between p1→port1, p2→port2 and so on. This TNP could then be applied to the cluster, thus turning all its hosts into transport nodes in a single configuration step. Further, configuration changes are kept in sync across all the hosts, leading to easier cluster management.

3.1.6 Network I/O Control

Network I/O Control, or NIOC, is the implementation in NSX of vSphere's Network I/O Control v3. This feature allows managing traffic contention on the uplinks of an

ESXi hypervisor. NIOC allows the creation of shares, limits and bandwidth reservation for the different kinds of ESXi infrastructure traffic.

- Shares: Shares, from 1 to 100, reflect the relative priority of a traffic type against the other traffic types that are active on the same physical adapter.
- Reservation: The minimum bandwidth that must be guaranteed on a single physical adapter. Reserved bandwidth for system traffic that is unused becomes available to other types of system traffic. Unused system traffic reservations do NOT become available to VM traffic.
- Limit: The maximum bandwidth that a traffic type can consume on a single physical adapter.

The pre-determined types of ESXi infrastructure traffic are:

- Management Traffic is for host management
- Fault Tolerance (FT) is for sync and recovery.
- NFS Traffic is traffic related to a file transfer in the network file system.
- vSAN traffic is generated by virtual storage area network.
- vMotion traffic is for computing resource migration.
- vSphere replication traffic is for replication.
- vSphere Data Protection Backup traffic is generated by backup of data.
- Virtual Machine traffic is generated by virtual machines workload
- iSCSI traffic is for Internet Small Computer System Interface storage

When using an N-VDS on the transport node, the NIOC parameters are specified as a profile that is provided as part of the Uplink Profile during the ESXi Transport Node creation. If the transport node is running NSX on top of a VDS, the NIOC configuration takes place directly in vCenter. In addition to system traffic parameters, NIOC provides an additional level of granularity for the VM traffic category: share, reservation and limits can also be applied at the Virtual Machine vNIC level. This configuration is still done with vSphere, by editing the vNIC properties of the VMs. For more details, see the vSphere documentation.

3.1.7 Enhanced Data Path NSX virtual switch

When creating an ESXi Transport Node, the administrator must choose between two types of NSX virtual switch: standard or Enhanced Data Path (EDP). This option is available irrespective of whether NSX is installed using an N-VDS or a VDS. The Enhanced Data Path virtual switch is optimized for the Network Function Virtualization, where the workloads typically perform networking functions with very demanding requirements in term of latency and packet rate. In order to accommodate this use case, the Enhanced Data Path virtual switch has an optimized data path, with a different resource allocation model on the host. The specifics of this virtual switch are outside the scope of this document. The important points to remember regarding this switch are:

- It can only be instantiated on an ESXi hypervisor.

- Its uses case is very specific to NFV.

The two kinds of virtual switches can however coexist on the same hypervisor. It's not recommended for common enterprise or cloud use cases.

For the further understanding of enhanced data path N-VDS refer to following resources. For the performance related understanding refer to NFV: Raw Packet Processing Performance.

3.2 Logical Switching

This section on logical switching focuses on overlay backed segments due to their ability to create isolated logical L2 networks with the same flexibility and agility that exists with virtual machines. This decoupling of logical switching from the physical network infrastructure is one of the main benefits of adopting NSX.

3.2.1 Overlay Backed Segments

FIGURE 3-8 presents logical and physical network views of a logical switching deployment.

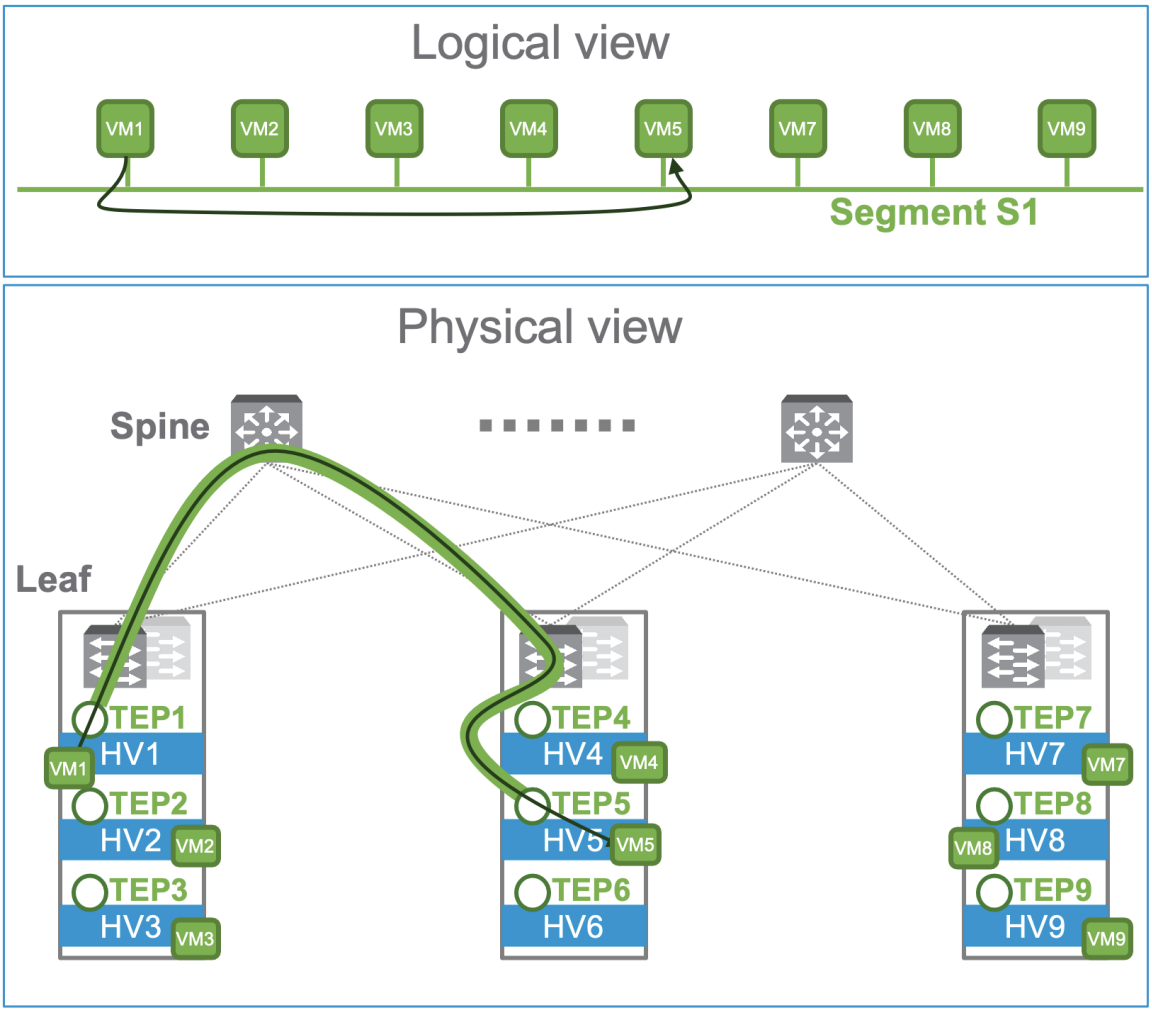


Figure 3-8: Overlay Networking – Logical and Physical View

In the upper part of the diagram, the logical view consists of five virtual machines that are attached to the same segment, forming a virtual broadcast domain. The physical representation, at the bottom, shows that the five virtual machines are running on hypervisors spread across three racks in a data center. Each hypervisor is an NSX transport node equipped with a tunnel endpoint (TEP). The TEPs are configured with IP addresses, and the physical network infrastructure just need to provide IP connectivity between them. Whether the TEPs are L2

adjacent in the same subnet or spread in different subnets does not matter. The VMware® NSX Controller™ (not pictured) distributes the IP addresses of the TEPs across the transport nodes so they can set up tunnels with their peers. The example shows “VM1” sending a frame to “VM5”. In the physical representation, this frame is transported via an IP point-to-point tunnel between transport nodes “HV1” to “HV5”.

The benefit of this NSX overlay model is that it allows direct connectivity between transport nodes irrespective of the specific underlay inter-rack (or even inter-datacenter) connectivity (i.e., L2 or L3). Segments can also be created dynamically without any configuration of the physical network infrastructure.

3.2.2 Flooded Traffic

The NSX segment behaves like a LAN, providing the capability of flooding traffic to all the devices attached to this segment; this is a cornerstone capability of layer 2. NSX does not differentiate between the different kinds of frames replicated to multiple destinations. Broadcast, unknown unicast, or multicast traffic will be flooded in a similar fashion across a segment. In the overlay model, the replication of a frame to be flooded on a segment is orchestrated by the different NSX components. NSX provides two different methods for flooding traffic described in the following sections. They can be selected on a per segment basis.

3.2.2.1 Head-End Replication Mode

In the head end replication mode, the transport node at the origin of the frame to be flooded sends a copy to each other transport node that is connected to this segment.

FIGURE 3-9 offers an example of virtual machine “VM1” on hypervisor “HV1” attached to segment “S1”. “VM1” sends a broadcast frame on “S1”. “HV1” floods the frame to the logical ports local to “HV1”, then determines that there are remote transport nodes part of “S1”. The NSX Controller advertised the TEPs of those remote interested transport nodes, so “HV1” will send a tunneled copy of the frame to each of them.

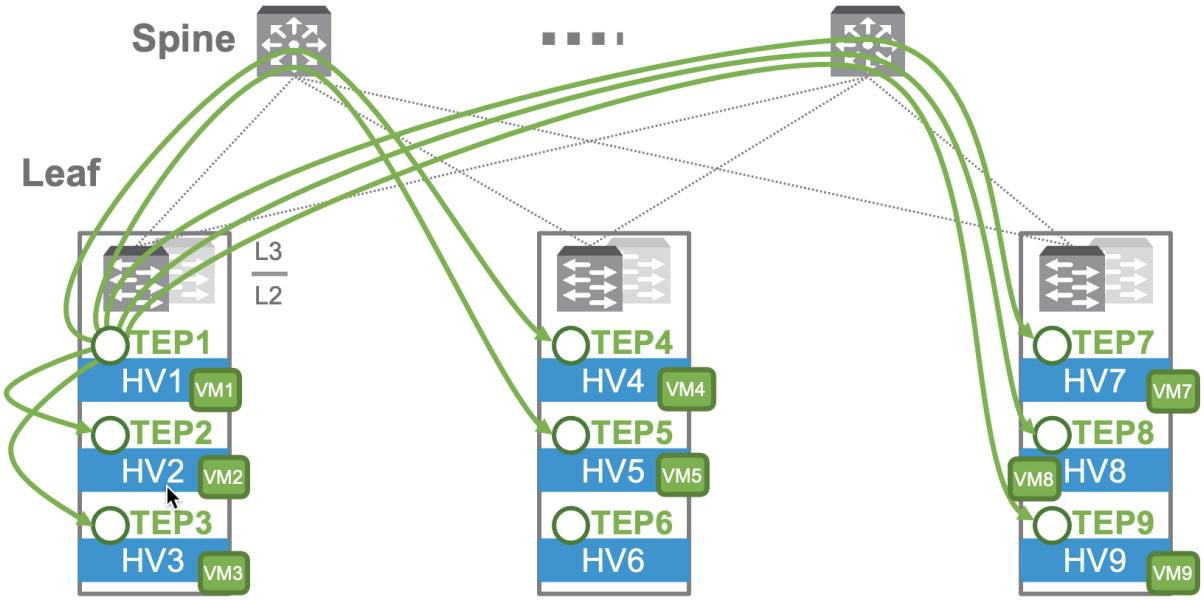


Figure 3-9: Head-end Replication Mode

The diagram illustrates the flooding process from the hypervisor transport node where "VM1" is located. "HV1" sends a copy of the frame that needs to be flooded to every peer that is interested in receiving this traffic. Each green arrow represents the path of a point-to-point tunnel through which the frame is forwarded. In this example, hypervisor "HV6" does not receive a copy of the frame. This is because the NSX Controller has determined that there is no recipient for this frame on that hypervisor.

In this mode, the burden of the replication rests entirely on source hypervisor. Seven copies of the tunnel packet carrying the frame are sent over the uplink of "HV1". This should be considered when provisioning the bandwidth on this uplink.

3.2.2.2 Two-tier Hierarchical Mode

In the two-tier hierarchical mode, transport nodes are grouped according to the subnet of the IP address of their TEP. Transport nodes in the same rack typically share the same subnet for their TEP IPs, though this is not mandatory. Based on this assumption, FIGURE 3-10 shows hypervisor transport nodes classified in three groups: subnet 10.0.0.0, subnet 20.0.0.0 and subnet 30.0.0.0. In this example, the IP subnet have been chosen to be easily readable; they are not public IPs.

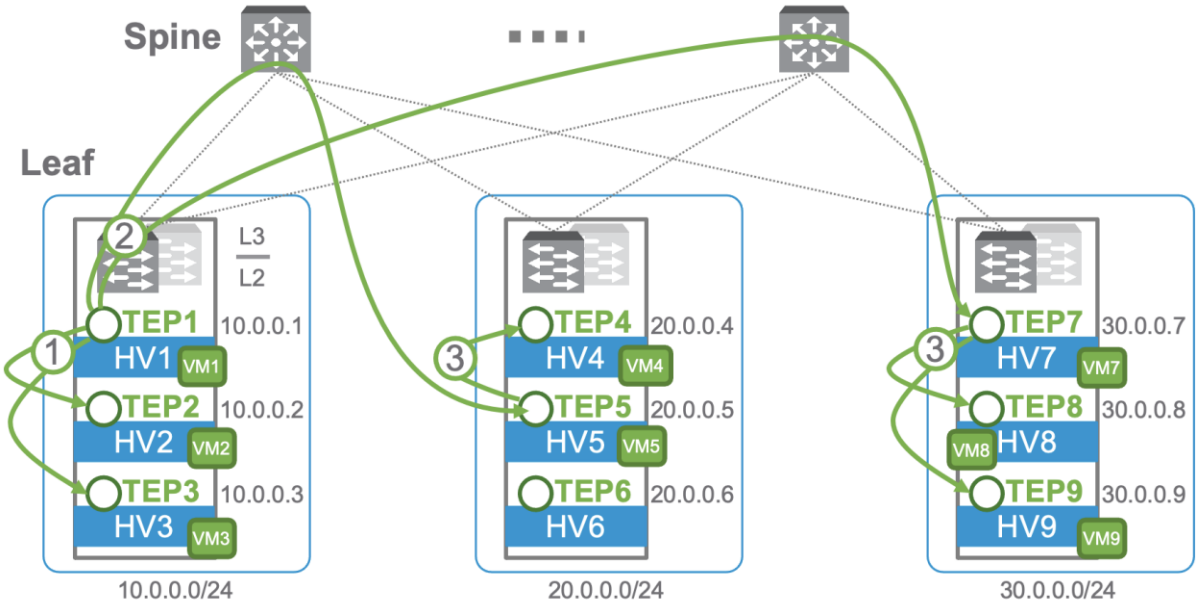


Figure 3-10: Two-tier Hierarchical Mode

Assume that "VM1" on "HV1" needs to send the same broadcast on "S1" as in the previous section on head-end replication. Instead of sending an encapsulated copy of the frame to each remote transport node attached to "S1", the following process occurs:

1. "HV1" sends a copy of the frame to all the transport nodes within its group (i.e., with a TEP in the same subnet as its TEP). In this case, "HV1" sends a copy of the frame to "HV2" and "HV3".
2. "HV1" sends a copy to a single transport node on each of the remote groups. For the two remote groups - subnet 20.0.0.0 and subnet 30.0.0.0 – "HV1" selects an arbitrary member of those groups and sends a copy of the packet there a bit set to indicate the need for local replication. In this example, "HV1" selected "HV5" and "HV7".
3. Transport nodes in the remote groups perform local replication within their respective groups. "HV5" relays a copy of the frame to "HV4" while "HV7" sends the frame to "HV8" and "HV9". Note that "HV5" does not relay to "HV6" as it is not interested in traffic from "LS1".

The source hypervisor transport node knows about the groups based on the information it has received from the NSX Controller. It does not matter which transport node is selected to perform replication in the remote groups so long as the remote transport node is up and available. If this were not the case (e.g., "HV7" was down), the NSX Controller would update all transport nodes attached to "S1". "HV1" would then choose "HV8" or "HV9" to perform the replication local to group 30.0.0.0.

In this mode, as with head end replication example, seven copies of the flooded frame have been made in software, though the cost of the replication has been spread across several transport nodes. It is also interesting to understand the traffic pattern on the physical infrastructure. The benefit of the two-tier hierarchical mode is that only two tunnel packets (compared to the headend

mode of five packets) were sent between racks, one for each remote group. This is a significant improvement in the network inter-rack (or inter-datacenter) fabric utilization - where available bandwidth is typically less than within a rack. That number that could be higher still if there were more transport nodes interested in flooded traffic for "S1" on the remote racks. In the case where the TEPs are in another data center, the savings could be significant. Note also that this benefit in term of traffic optimization provided by the two-tier hierarchical mode only applies to environments where TEPs have their IP addresses in different subnets. In a flat Layer 2 network, where all the TEPs have their IP addresses in the same subnet, the two-tier hierarchical replication mode would lead to the same traffic pattern as the source replication mode.

The default two-tier hierarchical flooding mode is recommended as a best practice as it typically performs better in terms of physical uplink bandwidth utilization.

3.2.3 Unicast Traffic

When a frame is destined to an unknown MAC address, it is flooded in the network. Switches typically implement a MAC address table, or filtering database (FDB), that associates MAC addresses to ports in order to prevent flooding. When a frame is destined to a unicast MAC address known in the MAC address table, it is only forwarded by the switch to the corresponding port.

The NSX virtual switch maintains such a table for each segment/logical switch it is attached to. A MAC address can be associated with either a virtual NIC (vNIC) of a locally attached VM or a remote TEP (when the MAC address is located on a remote transport node reached via the tunnel identified by that TEP).

FIGURE 3-11 illustrates virtual machine "Web3" sending a unicast frame to another virtual machine "Web1" on a remote hypervisor transport node. In this example, the NSX virtual switch on both the source and destination hypervisor transport nodes are fully populated.

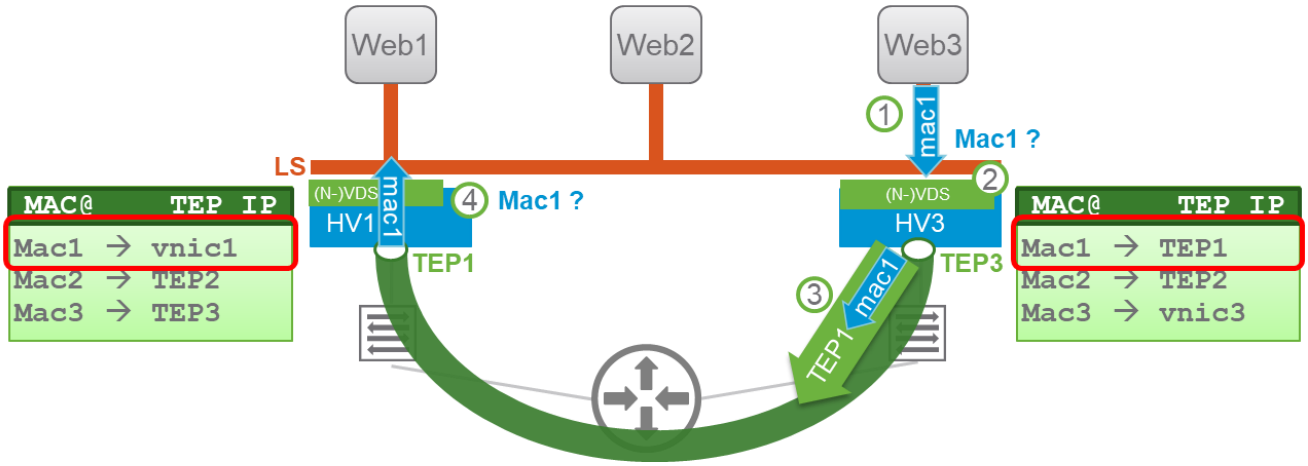


Figure 3-11: Unicast Traffic between VMs

1. "Web3" sends a frame to "Mac1", the MAC address of the vNIC of "Web1".

2. "HV3" receives the frame and performs a lookup for the destination MAC address in its MAC address table. There is a hit. "Mac1" is associated to the "TEP1" on "HV1".
3. "HV3" encapsulates the frame and sends it to "TEP1".
4. "HV1" receives the tunnel packet, addressed to itself and decapsulates it. TEP1 then performs a lookup for the destination MAC of the original frame. "Mac1" is also a hit there, pointing to the vNIC of "VM1". The frame is then delivered to its final destination.

This mechanism is relatively straightforward because at layer 2 in the overlay network, all the known MAC addresses are either local or directly reachable through a point-to-point tunnel.

In NSX, the MAC address tables can be populated by the NSX Controller or by learning from the data plane. The benefit of data plane learning, further described in the next section, is that it is immediate and does not depend on the availability of the control plane.

3.2.4 Data Plane Learning

In a traditional layer 2 switch, MAC address tables are populated by associating the source MAC addresses of frames received with the ports where they were received. In the overlay model, instead of a port, MAC addresses reachable through a tunnel are associated with the TEP for the remote end of this tunnel. Data plane learning is a matter of associating source MAC addresses with source TEPs. Ideally data plane learning would occur through the NSX virtual switch associating the source MAC address of received encapsulated frames with the source IP of the tunnel packet. But this common method used in overlay networking would not work for NSX with the two-tier replication model. Indeed, as shown in section TWO-TIER HIERARCHICAL MODE, it is possible that flooded traffic gets replicated by an intermediate transport node. In that case, the source IP address of the received tunneled traffic represents the intermediate transport node instead of the transport node that originated the traffic. FIGURE 3-12 below illustrates this problem by focusing on the flooding of a frame from VM1 on HV1 using the two-tier replication model (similar to what was described earlier in FIGURE 3-10: TWO-TIER HIERARCHICAL MODE) When intermediate transport node HV5 relays the flooded traffic from HV1 to HV4, it is actually decapsulating the original tunnel traffic and re-encapsulating it, using its own TEP IP address as a source.

Figure 3-12: Data Plane Learning Using Tunnel Source IP Address

The problem is thus that, if the NSX virtual switch on "HV4" was using the source tunnel IP address to identify the origin of the tunneled traffic, it would wrongly associate Mac1 to TEP5.

To solve this problem, upon re-encapsulation, TEP 5 inserts an identifier for the source TEP as NSX metadata in the tunnel header. Metadata is a piece of information that is carried along with the payload of the tunnel. FIGURE 3-13 displays the same tunneled frame from "Web1" on "HV1", this time carried with a metadata field identifying "TEP1" as the origin.

Figure 3-13: Data Plane Learning Leveraging Metadata

With this additional piece of information, "HV4" can correctly identify the origin of the tunneled traffic on replicated traffic.

3.2.5 Tables Maintained by the NSX Controller

While NSX can populate the filtering database of a segment/logical switch from the data plane just like traditional physical networking devices, the NSX Controller is also building a central repository for some tables that enhances the behavior of the system. These tables include:

- Global MAC address to TEP table
- Global ARP table, associating MAC addresses to IP addresses

3.2.5.1 MAC Address to TEP Tables

When the vNIC of a VM is attached to a segment/logical switch, the NSX Controller is notified of the MAC address as well as the TEP by which this MAC address is reachable. Unlike individual transport nodes that only learn MAC addresses corresponding to received traffic, the NSX Controller has a global view of all MAC addresses declared in the NSX environment.

The global MAC address table can proactively populate the local MAC address table of the different transport nodes before they receive any traffic. Also, in the rare case when transport node receives a frame from a VM destined to an unknown MAC address, it will send a request to look up this MAC address in the global table of the NSX Controller while simultaneously flooding the frame.

Not all the MAC addresses present in the data plane tables are reported to the NSX Controller. If a VM is allowed to send traffic on a segment/logical switch from several source MAC addresses, those secondary MAC addresses are not pushed to the NSX Controller. Similarly, the NSX Controller is not notified of MAC addresses learned from an Edge bridge connected to a physical layer 2 network. This behavior was implemented in order to protect the NSX Controller from an injection of an arbitrarily large number of MAC addresses into in the network.

3.2.5.2 ARP Tables

The NSX Controller also maintains an ARP table in order to help implement an ARP suppression mechanism. The NSX virtual switch snoops DHCP and ARP traffic to learn MAC address to IP associations. Those associations are then reported to the NSX Controller. An example of the process is summarized in FIGURE 3-14: ARP SUPPRESSION.

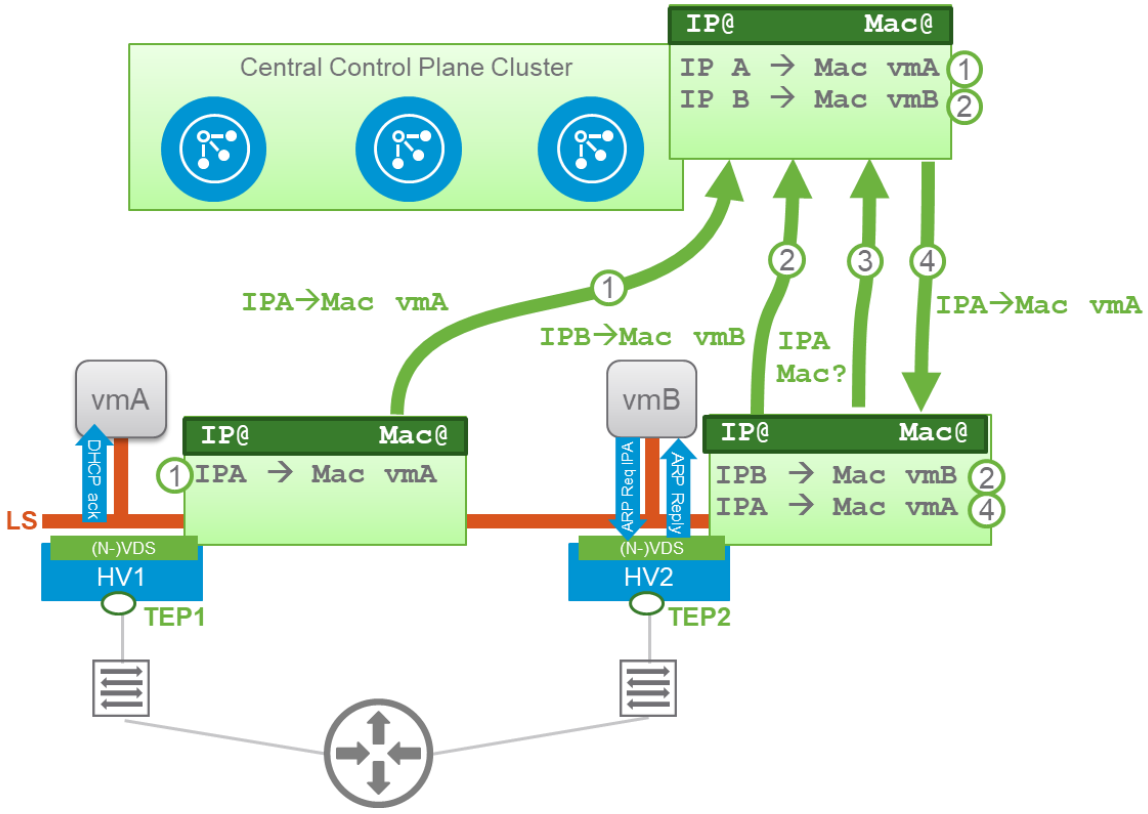


Figure 3-14: ARP Suppression

1. Virtual machine "vmA" has just finished a DHCP request sequence and been assigned IP address "IPA". The NSX virtual switch on "HV1" reports the association of the MAC address of virtual machine "vmA" to "IPA" to the NSX Controller.
2. Next, a new virtual machine "vmB" comes up on "HV2" that must communicate with "vmA", but its IP address has not been assigned by DHCP and, as a result, there has been no DHCP snooping. The virtual switch will be able to learn this IP address by snooping ARP traffic coming from "vmB". Either "vmB" will send a gratuitous ARP when coming up or it will send an ARP request for the MAC address of "vmA". The virtual switch then can derive the IP address "IPB" associated to "vmB". The association (vmB -> IPB) is then pushed to the NSX Controller.
3. The NSX virtual switch also holds the ARP request initiated by "vmB" and queries the NSX Controller for the MAC address of "vmA".

4. Because the MAC address of “vmA” has already been reported to the NSX Controller, the NSX Controller can answer the request coming from the virtual switch, which can now send an ARP reply directly to “vmB” on the behalf of “vmA”. Thanks to this mechanism, the expensive flooding of an ARP request has been eliminated. Note that if the NSX Controller did not know about the MAC address of “vmA” or if the NSX Controller were down, the ARP request from “vmB” would still be flooded by the virtual switch.

3.2.6 Overlay Encapsulation

NSX uses Generic Network Virtualization Encapsulation (Geneve) for its overlay model. Geneve is currently an IETF Internet Draft that builds on the top of VXLAN concepts to provide enhanced flexibility in term of data plane extensibility.

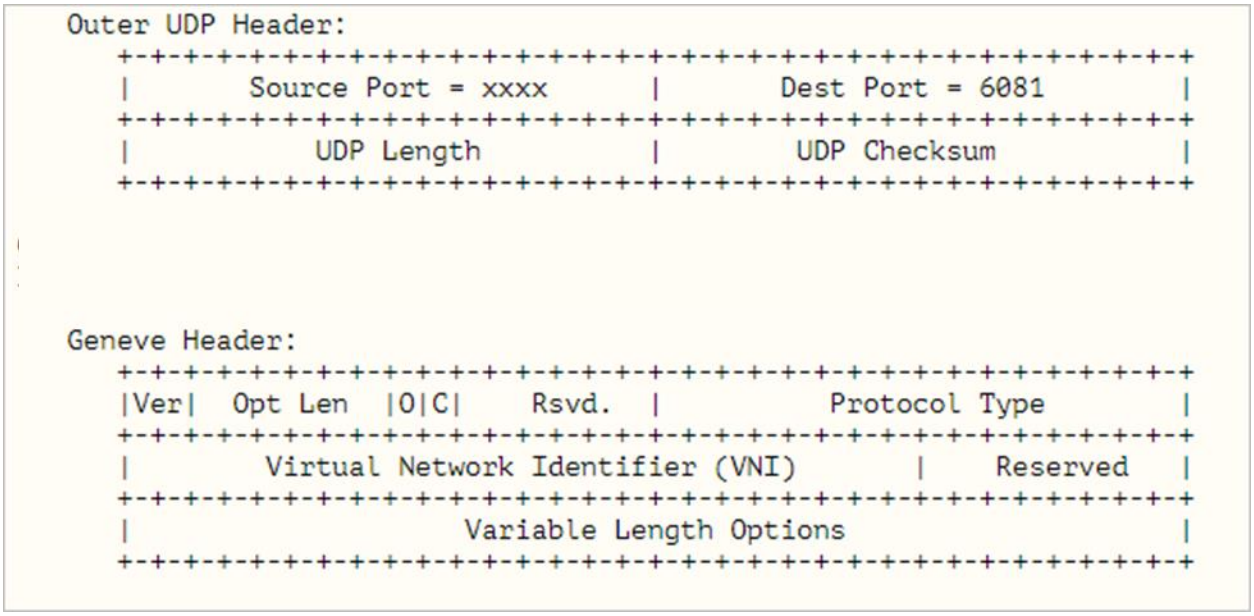


Figure 3-15: Geneve Encapsulation (from IETF Draft)

VXLAN has static fields while Geneve offers flexible field. This capability can be used by anyone to adjust the need of typical workload and overlay fabric, thus NSX tunnels are only setup between NSX transport nodes. NSX only needs efficient support for the Geneve encapsulation by the NIC hardware; most NIC vendors support the same hardware offload for Geneve as they would for VXLAN.

Network virtualization is all about developing a model of deployment that is applicable to a variety of physical networks and diversity of compute domains. New networking features are developed in software and implemented without worry of support on the physical infrastructure. For example, the data plane learning section described how NSX relies on metadata inserted in the tunnel header to identify the source TEP of a forwarded frame. This metadata could not have been added to a VXLAN tunnel without either hijacking existing bits in the VXLAN header or making a revision to the VXLAN specification. Geneve allows any vendor to add its own metadata in the tunnel header with a simple Type-Length-Value (TLV) model. NSX defines a single TLV, with fields for:

- Identifying the TEP that sourced a tunnel packet
- A version bit used during the intermediate state of an upgrade
- A bit indicating whether the encapsulated frame is to be traced
- A bit for implementing the two-tier hierarchical flooding mechanism. When a transport node receives a tunneled frame with this bit set, it knows that it must perform local replication to its peers
- Two bits identifying the type of the source TEP

These fields are part of a VMware specific TLV. This TLV can be changed or enlarged by VMware independent of any other vendors. Similarly, other vendors or partners can insert their own TLVs. Geneve benefits from the same NIC offloads as VXLAN (the capability is advertised in the VMware compatibility list for different NIC models.) Because overlay tunnels are only setup between NSX transport nodes, there is no need for any hardware or software third party vendor to decapsulate or look into NSX Geneve overlay packets. Thus, networking feature adoption can be done in the overlay, isolated from underlay hardware refresh cycles.

3.3 Bridging Overlay to VLAN with the Edge Bridge

Even in highly virtualized environments, customers often have workloads that cannot be virtualized, because of licensing or application-specific reasons. Even for the virtualized workload some applications have embedded IP that cannot be changed or legacy application that requires layer 2 connectivity. Those VLAN backed workloads typically communicate with overlay backed VMs at layer 3, through gateways (Tier-0 or Tier-1) instantiated on the NSX Edges. However, there are some scenarios where layer 2 connectivity is required between VMs and physical devices. For this functionality, NSX introduces the NSX Bridge, a service that can be instantiated on an Edge for the purpose of connecting an NSX logical segment with a traditional VLAN at layer 2.

The most common use cases for this feature are:

- Physical to virtual/virtual to virtual migration. This is generally a temporary scenario where a VLAN backed environment is being virtualized to an overlay backed NSX. The NSX Edge Bridge is a simple way to maintain connectivity between the different components during the intermediate stages of the migration process.

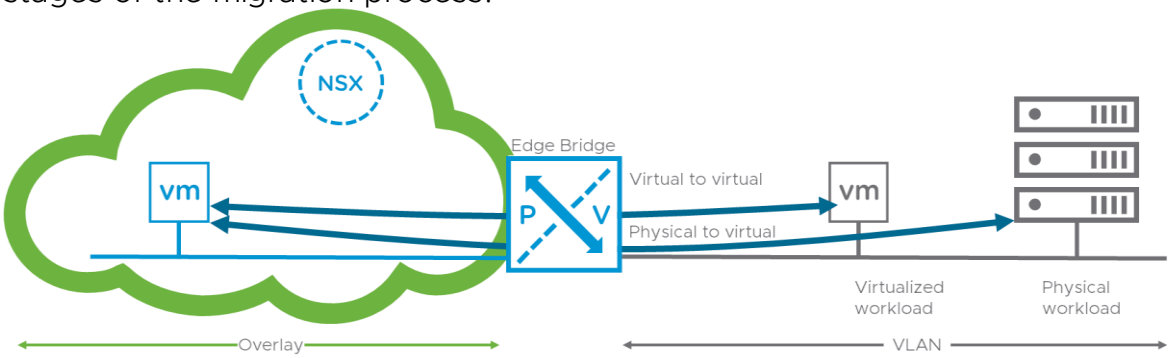


Figure 3-16: physical to virtual or virtual to virtual migration use case

- Integration of physical, non-virtualized appliances that require L2 connectivity to the virtualized environment. The most common example is a database server that requires L2 connectivity, typically because L3 connectivity has not been validated and is not supported by the vendor. This could also be the case of a service appliance that need to be inserted inline, like a physical firewall or load balancer.

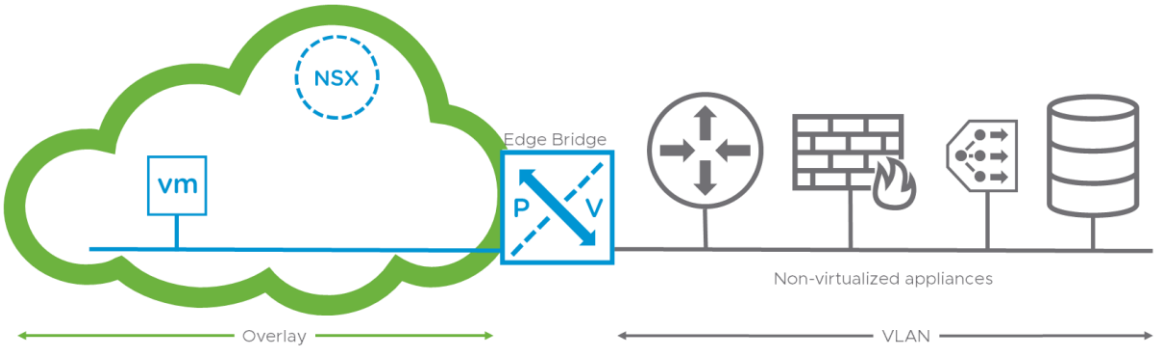


Figure 3-17: integration of non-virtualized appliances use case

Whether it is for migration purposes or for integration of non-virtualized appliances, if L2 adjacency is not needed, leveraging a gateway on the Edges (L3 connectivity) is typically more efficient, as routing allows for Equal Cost Multi Pathing, which results in higher bandwidth and a better redundancy model. A common misconception exists regarding the usage of the edge bridge, from the fact that modern SDN based adoption must not use bridging. In fact, that is not the case, the Edge Bridge can be conceived as a permanent solution for extending overlay-backed segments into VLANs. The use case of having a permeant bridging for set of workloads exist due to variety of reasons such as older application cannot change IP address, end of life gear does not allow any change, regulation, third party connectivity and span of control on those topologies or devices. However, as an architect if one desired to enable such use case must consider some level of dedicated resources and planning that ensue, such as bandwidth, operational control and protection of bridged topologies.

3.3.1 Overview of the Capabilities

The following sections present the capabilities of the NSX Edge Bridge.

3.3.1.1 DPDK-based performance

One of the main benefits of running a Bridge on the NSX Edge is the data plane performance. Indeed, the NSX Edge is leveraging the Data Plane Development Kit (DPDK), providing low latency, high bandwidth and scalable traffic forwarding performance.

3.3.1.2 Extend an Overlay-backed Segment/Logical Switch to a VLAN

In its most simple representation, the only thing the NSX Edge Bridge achieves is to convert an Ethernet frame between two different L2 representations: overlay and VLAN. In the overlay representation, the L2 frame and its payload are encapsulated in an IP-based format (as described above, NSX currently leverages Geneve, Generic Network Virtualization Encapsulation). In the VLAN

representation, the L2 frame may include an 802.1Q trunk VLAN tag or be an IEE 802.3 frame, depending on the desired connectivity model. The Edge Bridge is currently capable of making a one-to-one association between an overlay-backed Segment (identified by a Virtual Network Identifier in the overlay header) and a specific VLAN ID.

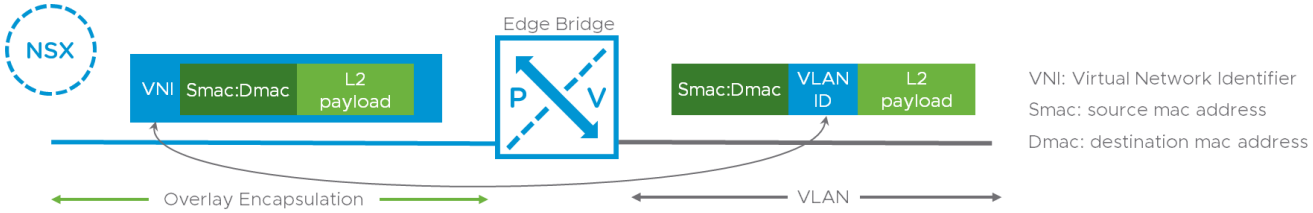


Figure 3-18: One-to-one association between segment and VLAN ID

As of NSX 2.4, a segment can be connected to only a single Edge Bridge. That means that L2 traffic can enter and leave the NSX overlay in a single location, thus preventing the possibility of a loop between a VLAN and the overlay. It is however possible to bridge several different segments to the same VLAN ID, if those different bridging instances are leveraging separate Edge uplinks.

Starting NSX 2.5, the same segment can be attached to several bridges on different Edges. This allows certain bare metal topologies to be connected with overlay segment and bridging to VLANs that can exist in separate rack without depending on physical overlay. With NSX 3.0, the Edge Bridge supports bridging 802.1Q tagged traffic carried in an overlay backed segment (Guest VLAN Tagging.) For more information about this feature, see the BRIDGING WHITE PAPER on the VMware communities website.

3.3.1.3 High Availability with Bridge Instances

The Edge Bridge operates as an active/standby service. The Edge bridge active in the data path is backed by a unique, pre-determined standby bridge on a different Edge. NSX Edges are deployed in a pool called an Edge Cluster. Within an Edge Cluster, the user can create a Bridge Profile, which essentially designates two Edges as the potential hosts for a pair of redundant Bridges. The Bridge Profile specifies which Edge would be primary (i.e. the preferred host for the active Bridge) and backup (the Edge that will host the backup Bridge). At the time of the creation of the Bridge Profile, no Bridge is instantiated yet. The Bridge Profile is just a template for the creation of one or several Bridge pairs.

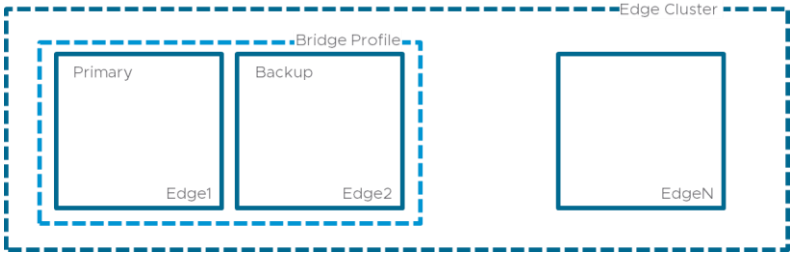


Figure 3-19: Bridge Profile, defining a redundant Edge Bridge (primary and backup)

Once a Bridge Profile is created, the user can attach a segment to it. By doing so, an active Bridge instance is created on the primary Edge, while a standby Bridge

is provisioned on the backup Edge. NSX creates a Bridge Endpoint object, which represents this pair of Bridges. The attachment of the segment to the Bridge Endpoint is represented by a dedicated Logical Port, as shown in the diagram below:

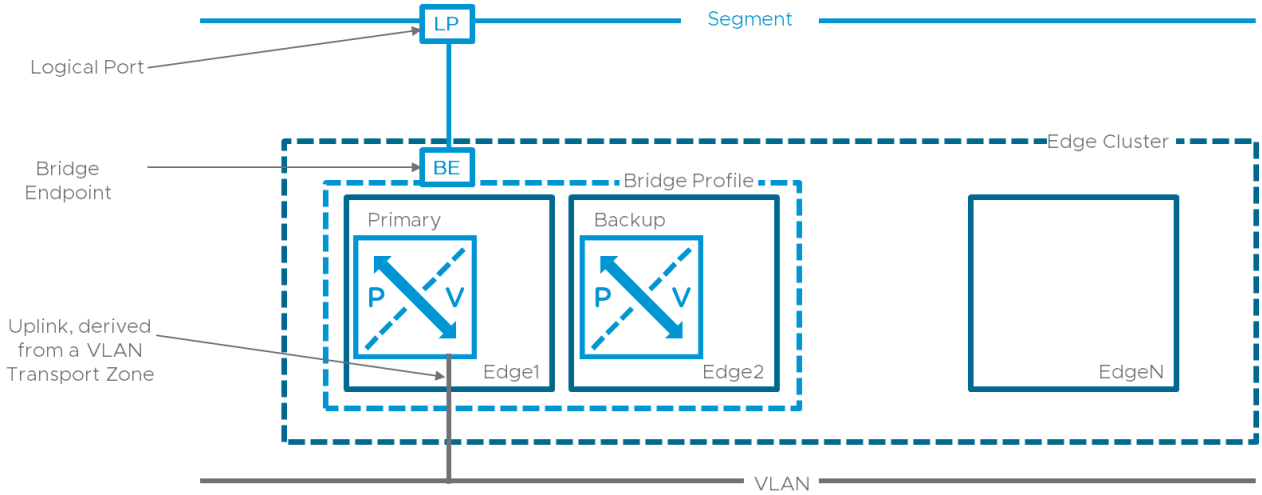


Figure 3-20: Primary Edge Bridge forwarding traffic between segment and VLAN

When associating a segment to a Bridge Profile, the user can specify the VLAN ID for the VLAN traffic as well as the physical port that will be used on the Edge for sending/receiving this VLAN traffic. At the time of the creation of the Bridge Profile, the user can also select the failover mode. In the preemptive mode, the Bridge on the primary Edge will always become the active bridge forwarding traffic between overlay and VLAN as soon as it is available, usurping the function from an active backup. In the non-preemptive mode, the Bridge on the primary Edge will remain standby should it become available when the Bridge on the backup Edge is already active.

3.3.1.4 Edge Bridge Firewall

The traffic leaving and entering a segment via a Bridge is subject to the Bridge Firewall. Rules are defined on a per-segment basis and are defined for the Bridge as a whole, i.e. they apply to the active Bridge instance, irrespective of the Edge on which it is running.

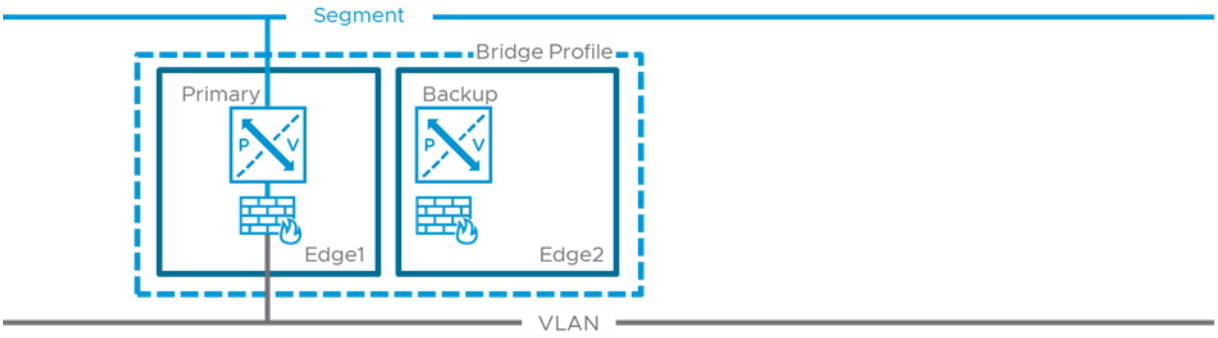


Figure 3-21: Edge Bridge Firewall

The firewall rules can leverage existing NSX grouping constructs, and there is currently a single firewall section available for those rules.

3.3.1.5 Edge Bridge Teaming Policies

In a bridge configuration, when the NVDS carrying the VLAN traffic has more than one uplink, it is possible to associate a named teaming policy to the bridging instance to control which uplink will be used. The uplink policy directing the bridge VLAN traffic should have a single active uplink and no standby uplinks. Any other teaming configuration is not supported and may lead to traffic loss (SEE KB ARTICLE). The default teaming policy will govern the overlay traffic. When load balancing based on source port ID is selected for the default teaming policy, the NVDS will load balance the bridge overlay traffic per segment. Each segment will be associated with one of the TEPs on the NVDS based on a hash value.

The figure below shows an example of a bare metal edge where named teaming policies are applied to the VLAN traffic while the overlay traffic follows the default teaming policy. Load balancing of the VLAN traffic over different pNICs is achieved by associating different named teaming policies to the two bridging instances for the X and Y overlays. More details on the recommended design for bare metal and edge node VMs are provided in chapter 7 (BRIDGING USE CASE).

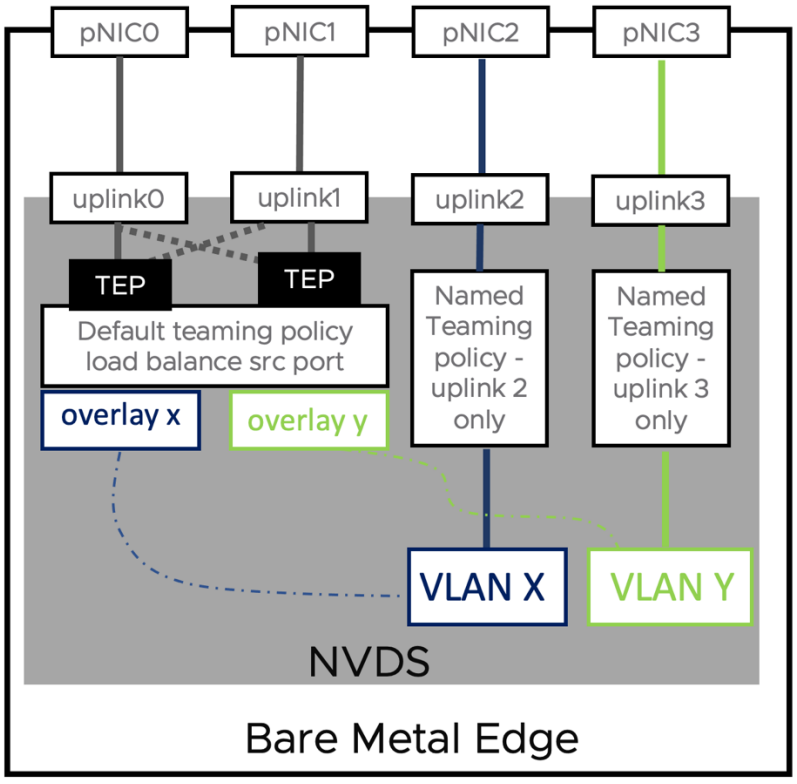


Figure 3-22: Edge Bridge Teaming Policies

3.3.1.6 Seamless integration with NSX gateways

This part requires understanding of Tier-0 and Tier-1 gateway and refer to the LOGICAL ROUTING chapter for further understanding about Tier-0 and Tier-1 gateways.

Routing and bridging seamlessly integrate. Distributed routing is available to segments extended to VLAN by a Bridge. The following diagram is a logical representation of a possible configuration leveraging T0 and T1 gateways along with Edge Bridges.

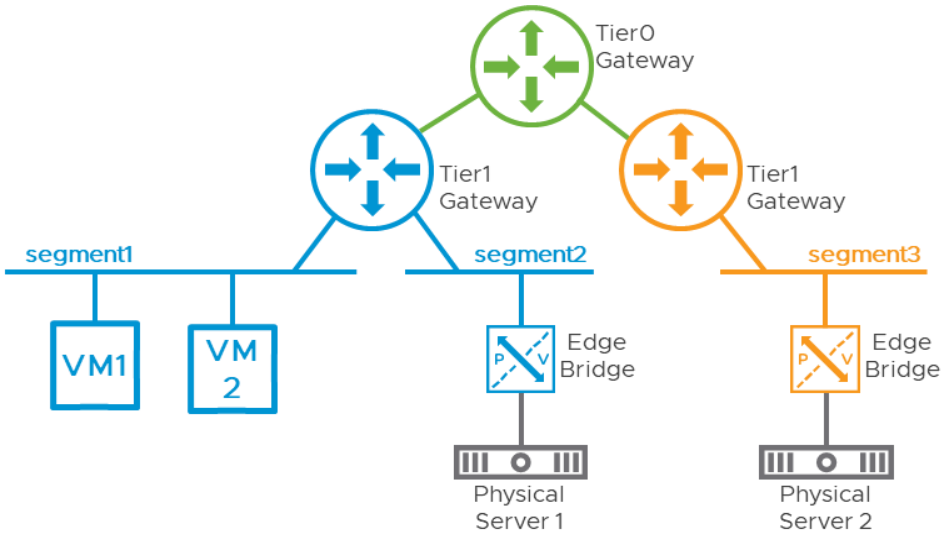


Figure 3-23: Integration with routing

In this above example, VM1, VM2, Physical Servers 1 and 2 have IP connectivity. Remarkably, through the Edge Bridges, Tier-1 or Tier-0 gateways can act as default gateways for physical devices. Note also that the distributed nature of NSX routing is not affected by the introduction of an Edge Bridge. ARP requests from physical workload for the IP address of an NSX router acting as a default gateway will be answered by the local distributed router on the Edge where the Bridge is active.

4 NSX Logical Routing

The logical routing capability in the NSX platform provides the ability to interconnect both virtual and physical workloads deployed in different logical L2 networks. NSX enables the creation of network elements like segments (Layer 2 broadcast domains) and gateways (routers) in software as logical constructs and embeds them in the hypervisor layer, abstracted from the underlying physical hardware. Since these network elements are logical entities, multiple gateways can be created in an automated and agile fashion.

The previous chapter showed how to create segments; this chapter focuses on how gateways provide connectivity between different logical L2 networks. FIGURE shows both logical and physical view of a routed topology connecting segments/logical switches on multiple hypervisors. Virtual machines "Web1" and "Web2" are connected to "overlay Segment 1" while "App1" and "App2" are connected to "overlay Segment 2".

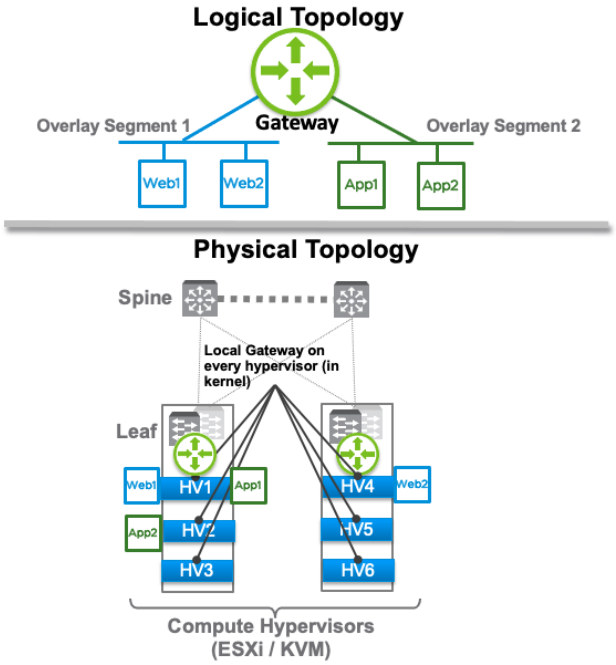


Figure 4-1: Logical and Physical View of Routing Services

In a data center, traffic is categorized as East-West (E-W) or North-South (N-S) based on the origin and destination of the flow. When virtual or physical workloads in a data center communicate with the devices external to the data center (e.g., WAN, Internet), the traffic is referred to as North-South traffic. The traffic between workloads confined within the data center is referred to as East-West traffic. In modern data centers, more than 70% of the traffic is East-West.

For a multi-tiered application where the web tier needs to talk to the app tier and the app tier needs to talk to the database tier and, these different tiers sit in different subnets. Every time a routing decision is made, the packet is sent to the router. Traditionally, a centralized router would provide routing for these different tiers. With VMs that are hosted on same the ESXi or KVM hypervisor, traffic will

leave the hypervisor multiple times to go to the centralized router for a routing decision, then return to the same hypervisor; this is not optimal.

NSX is uniquely positioned to solve these challenges as it can bring networking closest to the workload. Configuring a Gateway via NSX Manager instantiates a local distributed gateway on each hypervisor. For the VMs hosted (e.g., "Web 1", "App 1") on the same hypervisor, the E-W traffic does not need to leave the hypervisor for routing.

4.1 Single Tier Routing

NSX Gateway provides optimized distributed routing as well as centralized routing and services like NAT, AVI Load balancer, DHCP server etc. A single tier routing topology implies that a Gateway is connected to segments southbound providing E-W routing and is also connected to physical infrastructure to provide N-S connectivity. This gateway is referred to as Tier-0 Gateway.

Tier-0 Gateway consists of two components: distributed routing component (DR) and centralized services routing component (SR).

4.1.1 Distributed Router (DR)

A DR is essentially a router with logical interfaces (LIFs) connected to multiple subnets. It runs as a kernel module and is distributed in hypervisors across all transport nodes, including Edge nodes. The traditional data plane functionality of routing and ARP lookups is performed by the logical interfaces connecting to the different segments. Each LIF has a vMAC address and an IP address being the default IP gateway for its connected segment. The IP address is unique per LIF and remains the same everywhere the segment exists. The vMAC associated with each LIF remains constant in each hypervisor, allowing the default gateway IP and MAC addresses to remain the same during vMotion.

The left side of FIGURE 4-2 shows a logical topology with two segments, "Web Segment" with a default gateway of 172.16.10.1/24 and "App Segment" with a default gateway of 172.16.20.1/24 are attached to Tier-0 Gateway. In the physical topology view on the right, VMs are shown on two hypervisors, "HV1" and "HV2". A distributed routing (DR) component for this Tier-0 Gateway is instantiated as a kernel module and will act as a local gateway or first hop router for the workloads connected to the segments. Please note that the DR is not a VM and the DR on both hypervisors has the same IP addresses.

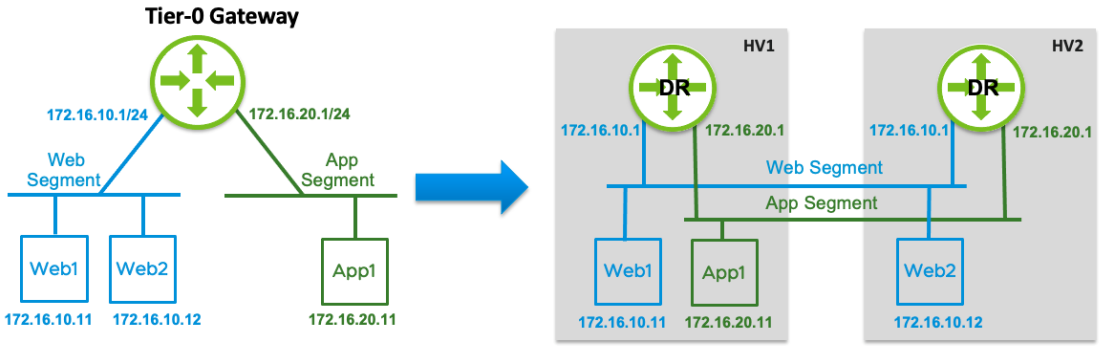


Figure 4-2: E-W Routing with Workloads on the same Hypervisor

East-West Routing - Distributed Routing with Workloads on the Same Hypervisor

In this example, "Web1" VM is connected to "Web Segment" and "App1" is connected to "App-Segment" and both VMs are hosted on the same hypervisor. Since "Web1" and "App1" are both hosted on hypervisor "HV1", routing between them happens on the DR located on that same hypervisor.

FIGURE 4-3 presents the logical packet flow between two VMs on the same hypervisor.

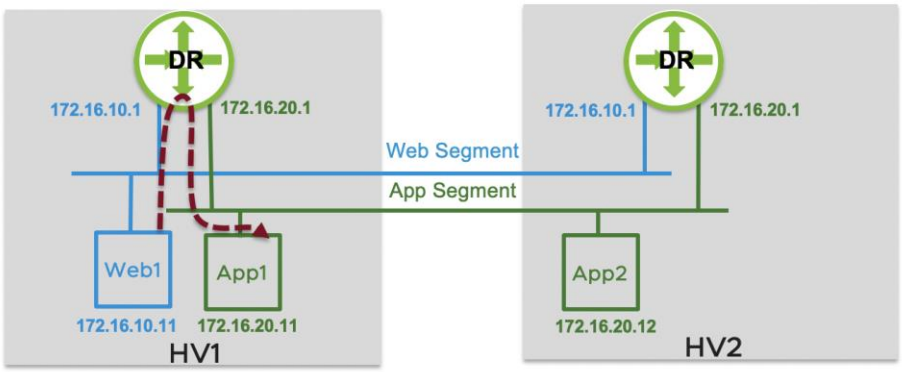


Figure 4-3: Packet Flow between two VMs on same Hypervisor

1. "Web1" (172.16.10.11) sends a packet to "App1" (172.16.20.11). The packet is sent to the default gateway interface (172.16.10.1) for "Web1" located on the local DR.
2. The DR on "HV1" performs a routing lookup which determines that the destination subnet 172.16.20.0/24 is a directly connected subnet on "LIF2". A lookup is performed in the "LIF2" ARP table to determine the MAC address associated with the IP address for "App1". If the ARP entry does not exist, the controller is queried. If there is no response from controller, an ARP request is flooded to learn the MAC address of "App1".
3. Once the MAC address of "App1" is learned, the L2 lookup is performed in the local MAC table to determine how to reach "App1" and the packet is delivered to the App1 VM.

4. The return packet from “App1” follows the same process and routing would happen again on the local DR.

In this example, neither the initial packet from “Web1” to “App1” nor the return packet from “App1” to “Web1” left the hypervisor.

East-West Routing - Distributed Routing with Workloads on Different Hypervisor

In this example, the target workload “App2” differs as it rests on a hypervisor named “HV2”. If “Web1” needs to communicate with “App2”, the traffic would have to leave the hypervisor “HV1” as these VMs are hosted on two different hypervisors. FIGURE shows a logical view of topology, highlighting the routing decisions taken by the DR on “HV1” and the DR on “HV2”.

When “Web1” sends traffic to “App2”, routing is done by the DR on “HV1”. The reverse traffic from “App2” to “Web1” is routed by DR on “HV2”. Routing is performed on the hypervisor attached to the source VM.

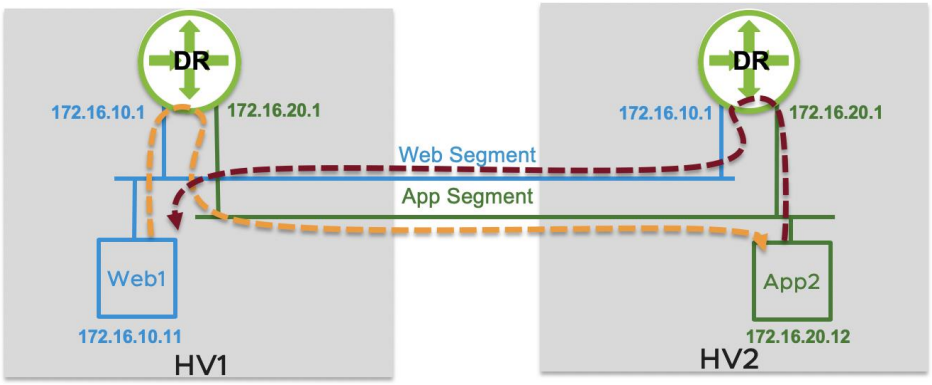


Figure 4-4: E-W Packet Flow between two Hypervisors

FIGURE shows the corresponding physical topology and packet walk from “Web1” to “App2”.

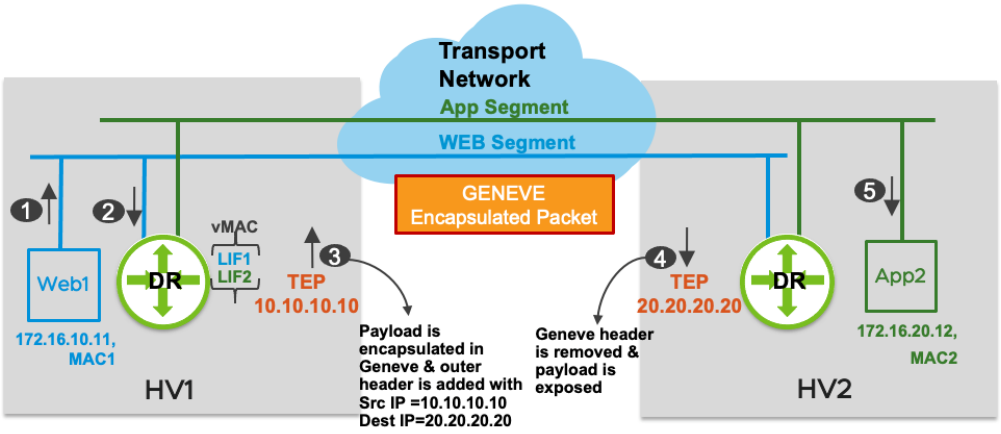


Figure 4-5: End-to-end E-W Packet Flow

1. "Web1" (172.16.10.11) sends a packet to "App2" (172.16.20.12). The packet is sent to the default gateway interface (172.16.10.1) for "Web1" located on the local DR. Its L2 header has the source MAC as "MAC1" and destination MAC as the vMAC of the DR. This vMAC will be the same for all LIFs.
2. The routing lookup happens on the HV1 DR, which determines that the destination subnet 172.16.20.0/24 is a directly connected subnet on "LIF2". A lookup is performed in the "LIF2" ARP table to determine the MAC address associated with the IP address for "App2". This destination MAC, "MAC2", is learned via the remote HV2 TEP 20.20.20.20.
3. HV1 TEP encapsulates the original packet and sends it to the HV2 TEP with a source IP address of 10.10.10.10 and destinations IP address of 20.20.20.20 for the encapsulating packet. The destination virtual network identifier (VNI) in the Geneve encapsulated packet belongs to "App Segment".
4. HV2 TEP 20.20.20.20 decapsulates the packet, removing the outer header upon reception. It performs an L2 lookup in the local MAC table associated with "LIF2".
5. Packet is delivered to "App2" VM.

The return packet from "App2" destined for "Web1" goes through the same process. For the return traffic, the routing lookup happens on the HV2 DR. This represents the normal behavior of the DR, which is to always perform routing on the DR instance running in the kernel of the hypervisor hosting the workload that initiates the communication. After the routing lookup, the packet is encapsulated by the HV2 TEP and sent to the remote HV1 TEP. The HV1 decapsulates the Geneve packet and delivers the encapsulated frame from "App2" to "Web1".

4.1.2 Services Router

East-West routing is completely distributed in the hypervisor, with each hypervisor in the transport zone running a DR in its kernel. However, some services of NSX are not distributed, due to its locality or stateful nature such as:

- Physical infrastructure connectivity (BGP Routing with Address Families – VRF lite)
- NAT
- DHCP server
- VPN
- Gateway Firewall
- Bridging
- Service Interface
- Metadata Proxy for OpenStack

A services router (SR) is instantiated on an edge cluster when a service is enabled that cannot be distributed on a gateway.

A centralized pool of capacity is required to run these services in a highly available and scaled-out fashion. The appliances where the centralized services or SR instances are hosted are called Edge nodes. An Edge node is the appliance that provides connectivity to the physical infrastructure.

Left side of FIGURE 4-6 shows the logical view of a Tier-0 Gateway showing both DR and SR components when connected to a physical router. Right side of FIGURE 4-6 shows how the components of Tier-0 Gateway are realized on Compute hypervisor and Edge node. Note that the compute host (i.e. HV1) has just the DR component and the Edge node shown on the right has both the SR and DR components. SR/DR forwarding table merge has been done to address future use-cases. SR and DR functionality remains the same after SR/DR merge in NSX 2.4 release, but with this change SR has direct visibility into the overlay segments. Notice that all the overlay segments are attached to the SR as well.

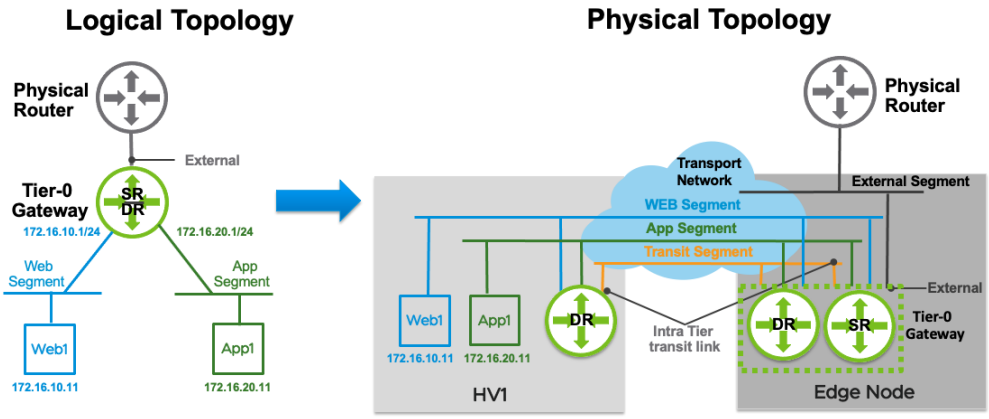


Figure 4-6: Logical Router Components and Interconnection

A Tier-0 Gateway can have following interfaces:

- External Interface – Interface connecting to the physical infrastructure/router. Static routing and BGP are supported on this interface. This interface was referred to as uplink interface in previous releases. This interface can also be used to extend a VRF (Virtual routing and forwarding instance) from the physical networking fabric into the NSX domain.
- Service Interface: Interface connecting VLAN segments to provide connectivity and services to VLAN backed physical or virtual workloads. Service interface can also be connected to overlay segments for Tier-1 standalone load balancer use-cases explained in Load balancer Chapter 6 . This interface was referred to as centralized service port (CSP) in previous releases. Note that a gateway must have a SR component to realize service interface. NSX 3.0 does not support dynamic routing over this interface. Static routing is supported
- Linked Segments – Interface connecting to an overlay segment. This interface was referred to as downlink interface in previous releases. Static routing is supported over that interface
- Intra-Tier Transit Link (Internal link between the DR and SR). A transit overlay segment is auto plumbed between DR and SR and each end gets an

IP address assigned in 169.254.0.0/24 subnet by default. This address range is configurable only when creating the Tier-0 gateway.

As mentioned previously, connectivity between DR on the compute host and SR on the Edge node is auto plumbed by the system. Both the DR and SR get an IP address assigned in 169.254.0.0/24 subnet by default. The management plane also configures a default route on the DR with the next hop IP address of the SR's intra-tier transit link IP. This allows the DR to take care of E-W routing while the SR provides N-S connectivity.

North-South Routing by SR Hosted on Edge Node

From a physical topology perspective, workloads are hosted on hypervisors and N-S connectivity is provided by Edge nodes. If a device external to the data center needs to communicate with a virtual workload hosted on one of the hypervisors, the traffic would have to come to the Edge nodes first. This traffic will then be sent on an overlay network to the hypervisor hosting the workload. FIGURE 4-7 shows the traffic flow from a VM in the data center to an external physical infrastructure.

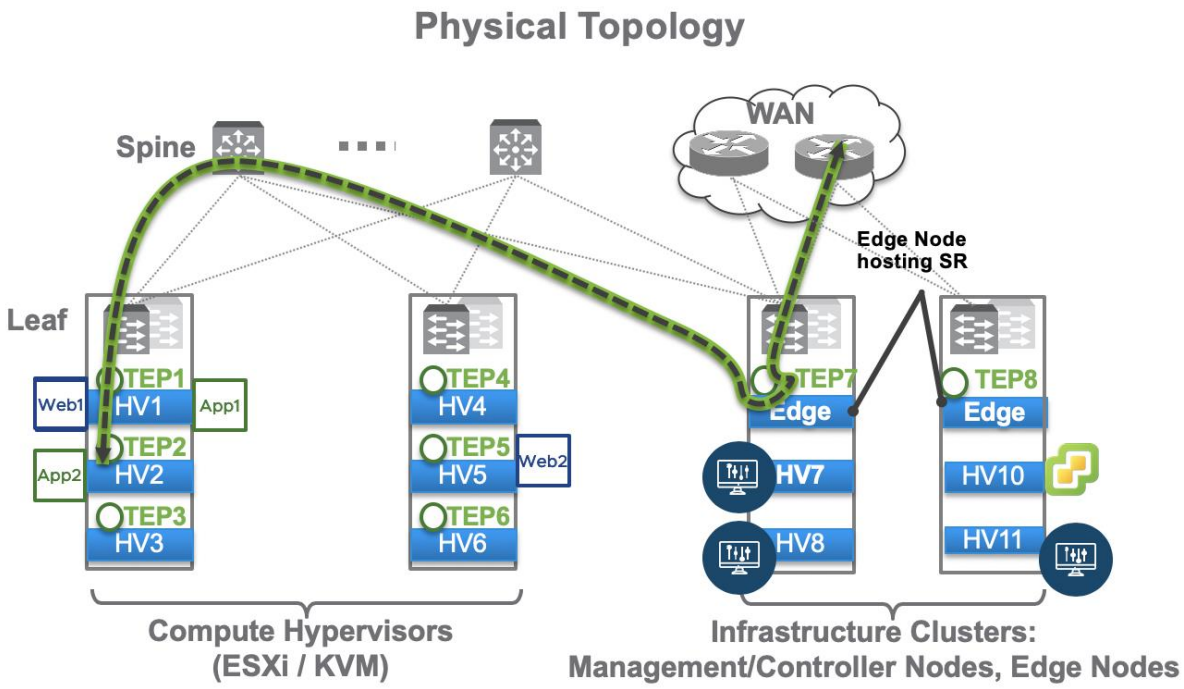


Figure 4-7: N-S Routing Packet Flow

FIGURE 4-8 shows a detailed packet walk from data center VM "Web1" to a device on the L3 physical infrastructure. As discussed in the E-W routing section, routing always happens closest to the source. In this example, eBGP peering has been established between the physical router interface with the IP address 192.168.240.1 and the Tier-0 Gateway SR component hosted on the Edge node with an external interface IP address of 192.168.240.3. Tier-0 Gateway SR has a BGP route for 192.168.100.0/24 prefix with a next hop of 192.168.240.1 and the

physical router has a BGP route for 172.16.10.0/24 with a next hop of 192.168.240.3.

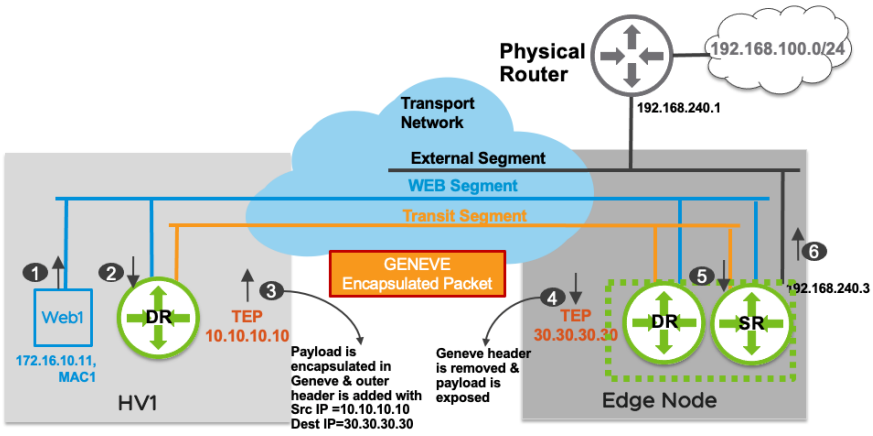


Figure 4-8: End-to-end Packet Flow – Application "Web1" to External

1. "Web1" (172.16.10.11) sends a packet to 192.168.100.10. The packet is sent to the "Web1" default gateway interface (172.16.10.1) located on the local DR.
2. The packet is received on the local DR. DR doesn't have a specific connected route for 192.168.100.0/24 prefix. The DR has a default route with the next hop as its corresponding SR, which is hosted on the Edge node.
3. The HV1 TEP encapsulates the original packet and sends it to the Edge node TEP with a source IP address of 10.10.10.10 and destination IP address of 30.30.30.30.
4. The Edge node is also a transport node. It will encapsulate/decapsulate the traffic sent to or received from compute hypervisors. The Edge node TEP decapsulates the packet, removing the outer header prior to sending it to the SR.
5. The SR performs a routing lookup and determines that the route 192.168.100.0/24 is learned via external interface with a next hop IP address 192.168.240.1.
6. The packet is sent on the VLAN segment to the physical router and is finally delivered to 192.168.100.10.

Observe that routing and ARP lookup happens on the DR hosted on the HV1 hypervisor to determine that the packet must be sent to the SR. On the edge node, the packet is directly sent to the SR after the tunnel encapsulation has been removed.

FIGURE 4-9 follows the packet walk for the reverse traffic from an external device to "Web1".

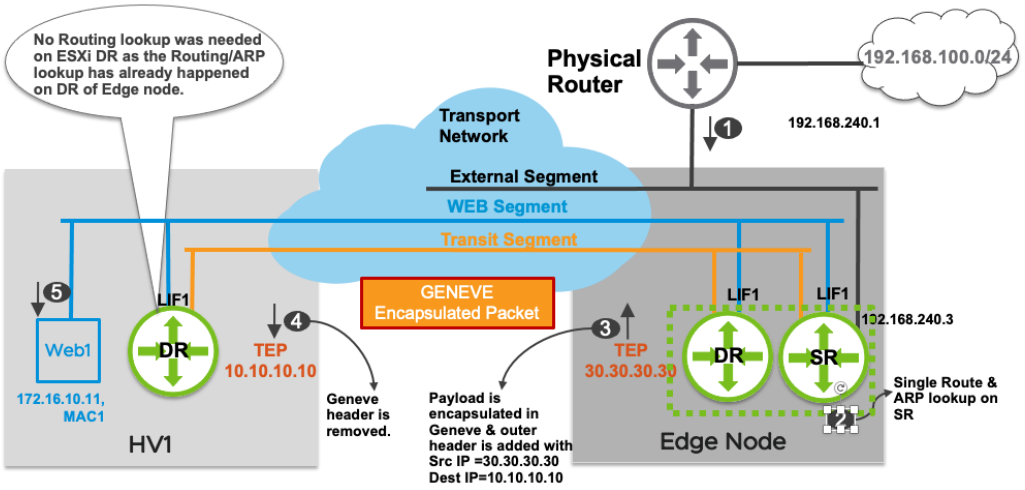


Figure 4-9: End-to-end Packet Flow – External to Application "Web1"

1. An external device (192.168.100.10) sends a packet to "Web1" (172.16.10.11). The packet is routed by the physical router and sent to the external interface of Tier-0 Gateway hosted on Edge node.
2. A single routing lookup happens on the Tier-0 Gateway SR which determines that 172.16.10.0/24 is a directly connected subnet on "LIF1". A lookup is performed in the "LIF1" ARP table to determine the MAC address associated with the IP address for "Web1". This destination MAC "MAC1" is learned via the remote TEP (10.10.10.10), which is the "HV1" host where "Web1" is located.
3. The Edge TEP encapsulates the original packet and sends it to the remote TEP with an outer packet source IP address of 30.30.30.30 and destination IP address of 10.10.10.10. The destination VNI in this Geneve encapsulated packet is of "Web Segment".
4. The HV1 host decapsulates the packet and removes the outer header upon receiving the packet. An L2 lookup is performed in the local MAC table associated with "LIF1".
5. The packet is delivered to Web1.

This time routing and ARP lookup happened on the merged SR/DR hosted on the Edge node. No such lookup was required on the DR hosted on the HV1 hypervisor, and packet was sent directly to the VM after removing the tunnel encapsulation header.

FIGURE 4-9 showed a Tier-0 gateway with one external interface that leverages Edge node to connect to physical infrastructure. If this Edge node goes down, N-S connectivity along with other centralized services running on Edge node will go down as well.

To provide redundancy for centralized services and N-S connectivity, it is recommended to deploy a minimum of two edge nodes. High availability modes are discussed in the section SERVICES HIGH AVAILABILITY.

4.2 Two-Tier Routing

In addition to providing optimized distributed and centralized routing functions, NSX supports a multi-tiered routing model with logical separation between different gateways within the NSX infrastructure. The top-tier gateway is referred to as a Tier-0 gateway while the bottom-tier gateway is a Tier-1 gateway. This structure gives complete control and flexibility over services and policies. Various stateful services can be hosted on the Tier-1 while the Tier-0 can operate in an active-active manner.

Configuring two tier routing is not mandatory. It can be single tiered as shown in the previous section. FIGURE 4-10 presents an NSX two-tier routing architecture.

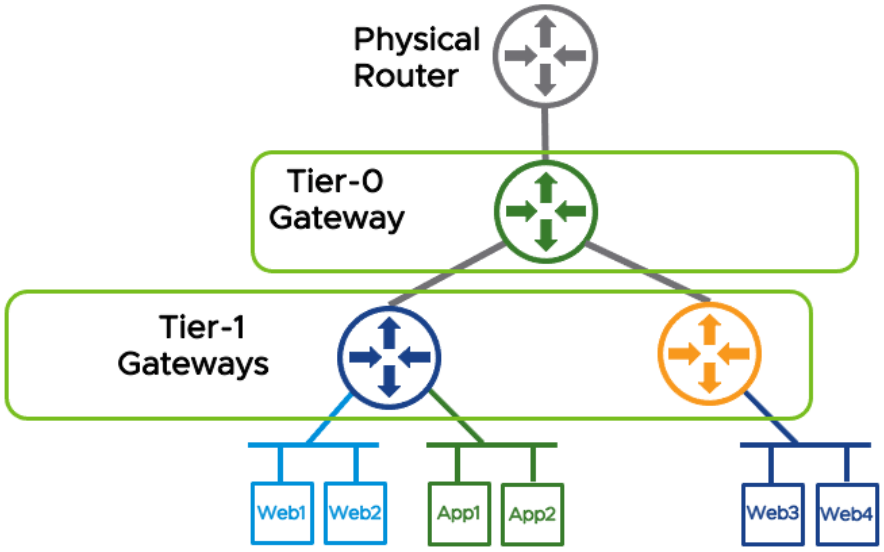


Figure 4-10: Two Tier Routing and Scope of Provisioning

Northbound, the Tier-0 gateway connects to one or more physical routers/L3 switches and serves as an on/off ramp to the physical infrastructure. Southbound, the Tier-0 gateway connects to one or more Tier-1 gateways or directly to one or more segments as shown in North-South routing section. Northbound, the Tier-1 gateway connects to a Tier-0 gateway using a RouterLink port. Southbound, it connects to one or more segments using downlink interfaces.

Concepts of DR/SR discussed in the SECTION 4.1 remain the same for multi-tiered routing. Like Tier-0 gateway, when a Tier-1 gateway is created, a distributed component (DR) of the Tier-1 gateway is intelligently instantiated on the hypervisors and Edge nodes. Before enabling a centralized service on a Tier-0 or Tier-1 gateway, an edge cluster must be configured on this gateway. Configuring an edge cluster on a Tier-1 gateway, instantiates a corresponding Tier-1 services component (SR) on two Edge nodes part of this edge cluster. Configuring an Edge cluster on a Tier-0 gateway does not automatically instantiate a Tier-0 service component (SR), the service component (SR) will only be created on a specific edge node along with the external interface creation.

Unlike the Tier-0 gateway, the Tier-1 gateway does not support northbound connectivity to the physical infrastructure. A Tier-1 gateway can only connect northbound to:

- a Tier-0 gateway
- a service port, this is used to connect a one-arm load-balancer to a segment. More details are available in CHAPTER 6.

Note that connecting Tier-1 to Tier-0 is a one click configuration or one API call configuration regardless of components instantiated (DR and SR) for that gateway.

4.2.1 Interface Types on Tier-1 and Tier-0 Gateway

External and Service interfaces were previously introduced in the services router section. Figure 4-11 shows these interfaces types along with a new "RouterLink" interface in a two-tiered topology.

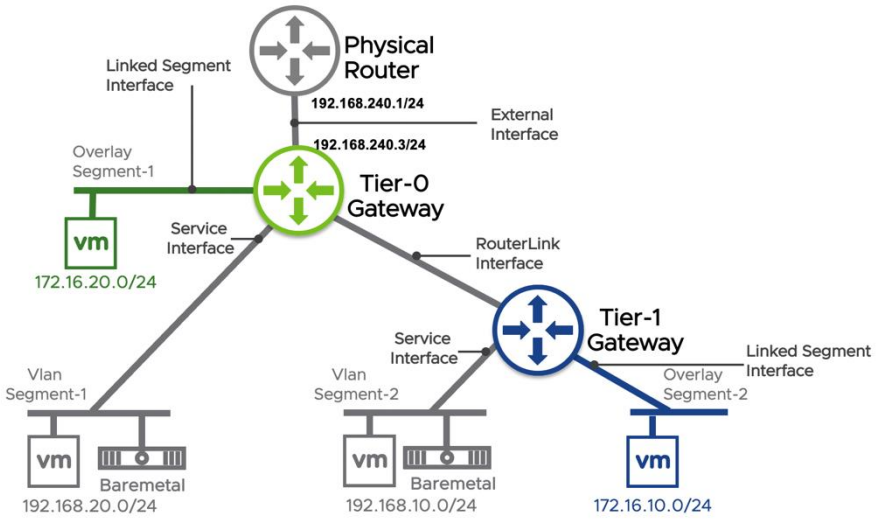


Figure 4-11: Anatomy of Components with Logical Routing

- External Interface: Interface connecting to the physical infrastructure/router. Static routing and BGP are supported on this interface. This interface only exists on Tier-0 gateway. This interface was referred to as Uplink interface in previous releases. This interface type will also be used to extend a VRF (Virtual Routing and Forwarding) from the physical networking fabric into the NSX domain.
- Router Link Interface/Linked Port: Interface connecting Tier-0 and Tier-1 gateways. Each Tier-0-to-Tier-1 peer connection is provided a /31 subnet within the 100.64.0.0/16 reserved address space (RFC6598). This link is created automatically when the Tier-0 and Tier-1 gateways are connected. This subnet can be changed when the Tier-0 gateway is being created. It is not possible to change it afterward.

- Service Interface: Interface connecting VLAN segments to provide connectivity to VLAN backed physical or virtual workloads. Service interface can also be connected to overlay/VLAN segments for standalone load balancer use cases explained in load balancer Chapter 6. Service Interface supports static. It is supported on both Tier-0 and Tier-1 gateways configured in Active/Standby high-availability configuration mode explained in section 4.6.2. Note that a Tier-0 or Tier-1 gateway must have an SR component to realize service interfaces. This interface was referred to as centralized service interface in previous releases. Dynamic Routing is not supported on Service Interfaces.
- Loopback: Tier-0 gateway supports the loopback interfaces. A Loopback interface is a virtual interface, and it can be redistributed into a routing protocol.

4.2.2 Route Types on Tier-0 and Tier-1 Gateways

There is no dynamic routing between Tier-0 and Tier-1 gateways. The NSX platform takes care of the auto-plumbing between Tier-0 and Tier-1 gateways. The following list details route types on Tier-0 and Tier-1 gateways.

- Tier-0 Gateway:
  - Connected – Connected routes on Tier-0 include external interface subnets, service interface subnets, loopback and segment subnets connected to Tier-0. In FIGURE 4-12, 172.16.20.0/24 (Connected segment), 192.168.20.0/24 (Service Interface) and 192.168.240.0/24 (External interface) are connected routes for the Tier-0 gateway.
  - Static – User configured static routes on Tier-0.
  - NAT IP – NAT IP addresses owned by the Tier-0 gateway discovered from NAT rules configured on Tier-0 Gateway.
  - BGP – Routes learned via BGP neighbors.
  - IPsec Local IP – Local IPsec endpoint IP address for establishing VPN sessions.
  - DNS Forwarder IP – Listener IP for DNS queries from clients. Also used as the source IP to forward DNS queries to the upstream DNS server.
  - Inter SR. SRs of a same Tier-0 gateway in the same edge cluster will create an automatic iBGP peering adjacency between them to exchange routing information. This topology is only supported with Active/Active topologies and with NSX Federation.
- Tier-1 Gateway
  - Connected – Connected routes on Tier-1 include segment subnets connected to Tier-1 and service interface subnets configured on Tier-1 gateway.
  - In FIGURE 4-12, 172.16.10.0/24 (Connected segment) and 192.168.10.0/24 (Service Interface) are connected routes for Tier-1 gateway.
  - Static– User configured static routes on Tier-1 gateway.

- NAT IP – NAT IP addresses owned by the Tier-1 gateway discovered from NAT rules configured on the Tier-1 gateway.
- LB VIP – IP address of load balancing virtual server.
- LB SNAT – IP address or a range of IP addresses used for Source NAT by load balancer.
- IPsec Local IP – Local IPsec endpoint IP address for establishing VPN sessions.
- DNS Forwarder IP – Listener IP for DNS queries from clients. Also used as the source IP to forward DNS queries to the upstream DNS server.

Route Advertisement on the Tier-1 and Tier-0 Logical Router

The Tier-0 gateway could use static routing or dynamic routing to connect to the physical routers. The Tier-1 gateway cannot connect to physical routers directly; it must connect to a Tier-0 gateway to provide N-S connectivity to the subnets attached to it. When a Tier-1 gateway is connected to a Tier-0 gateway, a default route is automatically created on the Tier-1. That default route is pointing to the RouterLink IP address that is owned by the Tier-0.

FIGURE 4-12 explains the route advertisement on both the Tier-1 and Tier-0 gateway.

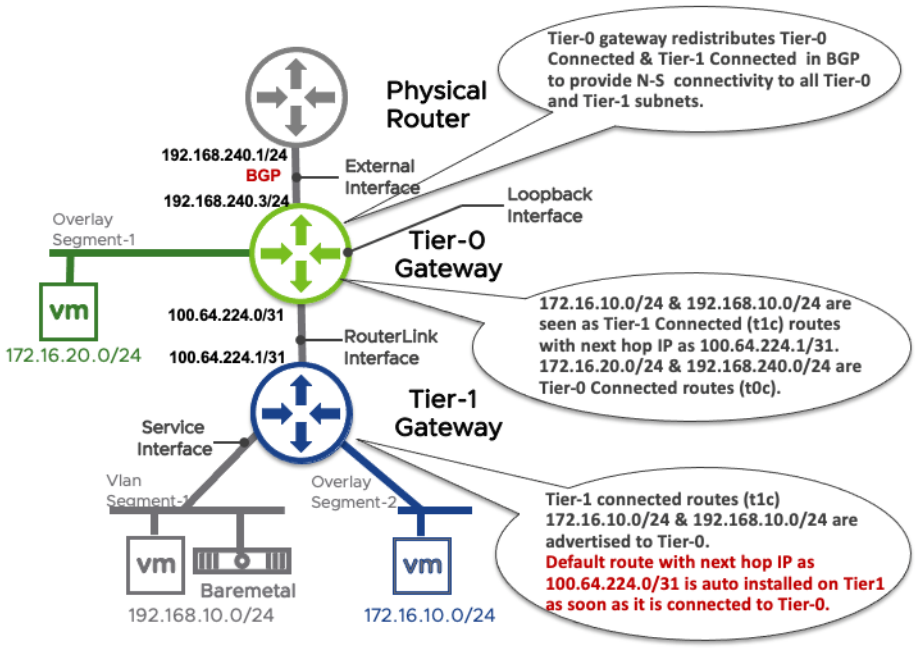


Figure 4-12: Routing Advertisement

"Tier-1 Gateway" advertises connected routes to Tier-0 Gateway. Figure 4-12 shows an example of connected routes (172.16.10.0/24 and 192.168.10.0/24). If there are other route types, like NAT IP etc. as discussed in section 4.2.2, a user can advertise those route types as well. As soon as "Tier-1 Gateway" is connected

to “Tier-0 Gateway”, the management plane configures a default route on “Tier-1 Gateway” with next hop IP address as RouterLink interface IP of “Tier-0 Gateway” i.e. 100.64.224.0/31 in the example above.

Tier-0 Gateway sees 172.16.10.0/24 and 192.168.10.1/24 as Tier-1 Connected routes (t1c) with a next hop of 100.64.224.1/31. Tier-0 Gateway also has Tier-0 “Connected” routes (172.16.20.0/24) in Figure 4-12.

Northbound, “Tier-0 Gateway” redistributes the Tier-0 connected and Tier-1 connected routes in BGP and advertises these routes to its BGP neighbor, the physical router.

4.2.3 Fully Distributed Two Tier Routing

NSX provides a fully distributed routing architecture. The motivation is to provide routing functionality closest to the source. NSX leverages the same distributed routing architecture discussed in distributed router section and extends that to multiple tiers.

FIGURE 4-13 shows both logical and per transport node views of two Tier-1 gateways serving two different tenants and a Tier-0 gateway. Per transport node view shows that the distributed component (DR) for Tier-0 and the Tier-1 gateways have been instantiated on two hypervisors.

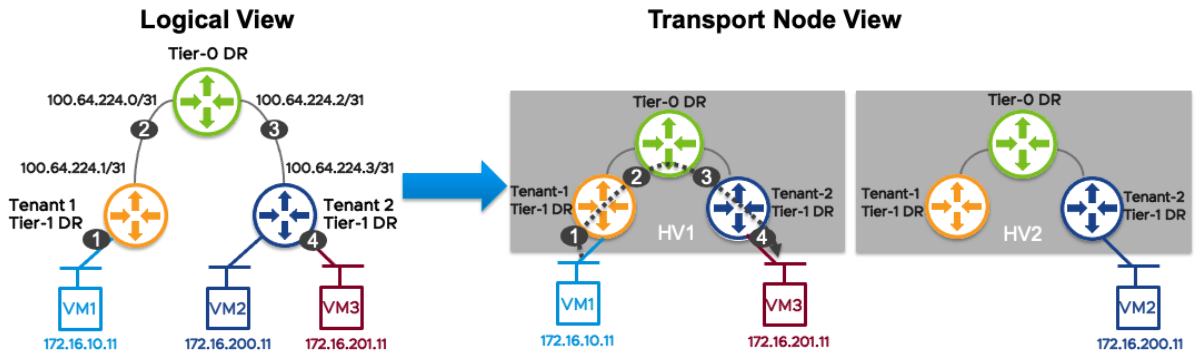


Figure 4-13: Logical Routing Instances

If “VM1” in tenant 1 needs to communicate with “VM3” in tenant 2, routing happens locally on hypervisor “HV1”. This eliminates the need to route of traffic to a centralized location to route between different tenants or environments.

Multi-Tier Distributed Routing with Workloads on the same Hypervisor

The following list provides a detailed packet walk between workloads residing in different tenants but hosted on the same hypervisor.

1. “VM1” (172.16.10.11) in tenant 1 sends a packet to “VM3” (172.16.201.11) in tenant 2. The packet is sent to its default gateway interface located on tenant 1, the local Tier-1 DR.

2. Routing lookup happens on the tenant 1 Tier-1 DR and the packet is routed to the Tier-0 DR following the default route. This default route has the RouterLink interface IP address (100.64.224.0/31) as a next hop.
3. Routing lookup happens on the Tier-0 DR. It determines that the 172.16.201.0/24 subnet is learned from the tenant 2 Tier-1 DR (100.64.224.3/31) and the packet is routed there.
4. Routing lookup happens on the tenant 2 Tier-1 DR. This determines that the 172.16.201.0/24 subnet is directly connected. L2 lookup is performed in the local MAC table to determine how to reach "VM3" and the packet is sent.

The reverse traffic from "VM3" follows the similar process. A packet from "VM3" to destination 172.16.10.11 is sent to the tenant-2 Tier-1 DR, then follows the default route to the Tier-0 DR. The Tier-0 DR routes this packet to the tenant 1 Tier-1 DR and the packet is delivered to "VM1". During this process, the packet never left the hypervisor to be routed between tenants.

Multi-Tier Distributed Routing with Workloads on different Hypervisors

Figure 4-14 shows the packet flow between workloads in different tenants which are also located on different hypervisors.

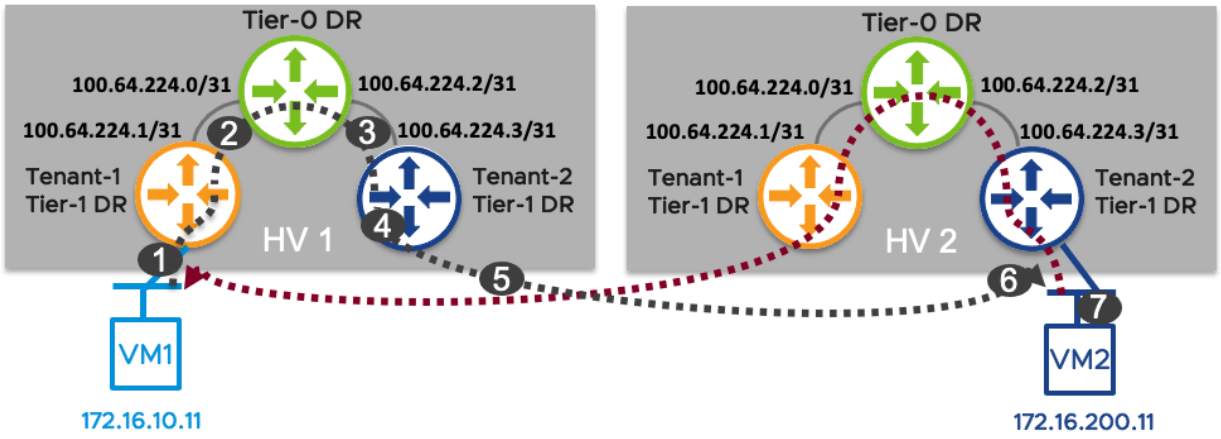


Figure 4-14: Logical routing end-to-end packet Flow between hypervisor

The following list provides a detailed packet walk between workloads residing in different tenants and hosted on the different hypervisors.

1. "VM1" (172.16.10.11) in tenant 1 sends a packet to "VM2" (172.16.200.11) in tenant 2. VM1 sends the packet to its default gateway interface located on the local Tier-1 DR in HV1.
2. Routing lookup happens on the tenant 1 Tier-1 DR and the packet follows the default route to the Tier-0 DR with a next hop IP of 100.64.224.0/31.
3. Routing lookup happens on the Tier-0 DR which determines that the 172.16.200.0/24 subnet is learned via the tenant 2 Tier-1 DR (100.64.224.3/31) and the packet is routed accordingly.
4. Routing lookup happens on the tenant 2 Tier-1 DR which determines that the 172.16.200.0/24 subnet is a directly connected subnet. A lookup is performed in ARP table to determine the MAC address associated with the

"VM2" IP address. This destination MAC is learned via the remote TEP on hypervisor "HV2".

5. The "HV1" TEP encapsulates the packet and sends it to the "HV2" TEP, finally leaving the host.
6. The "HV2" TEP decapsulates the packet and recognize the VNI in the Geneve header. A L2 lookup is performed in the local MAC table associated to the LIF where "VM2" is connected.
7. The packet is delivered to "VM2".

The return packet follows the same process. A packet from "VM2" gets routed to the local hypervisor Tier-1 DR and is sent to the Tier-0 DR. The Tier-0 DR routes this packet to tenant 1 Tier-1 DR which performs the L2 lookup to find out that the MAC associated with "VM1" is on remote hypervisor "HV1". The packet is encapsulated by "HV2" and sent to "HV1", where this packet is decapsulated and delivered to "VM1". It is important to notice that in this use case, routing is performed locally on the hypervisor hosting the VM sourcing the traffic.

4.3 Routing Capabilities

NSX supports static routing as well dynamic routing protocols to provide connectivity to the IPv4 and IPv6 workloads.

4.3.1 Static routing

In a multi-tier routing architecture and as described previously, NSX automatically creates the RouterLink segment, ports and static routes to interconnect a Tier-0 gateway with one or several Tier-1 gateways as depicted previously.

By default, the RouterLink segment is in the 100.64.0.0/16 IPv4 address range which is a shared IPv4 address space that is compliant with the RFC 6598. A default static route pointing to the Tier-0 gateway DR RouterLink port is automatically installed on the Tier-1 gateway routing table.

Northbound, static routes can be configured on Tier-1 gateways with the next hop IP as the RouterLink IP of the Tier-0 gateway (100.64.0.0/16 range or a range defined by user for RouterLink interface when the Tier-0 gateway is being created). Southbound, static routes can also be configured on Tier-1 gateway with a next hop as a layer 3 device reachable via a service or downlink interface.

Tier-0 gateways can be configured with a static route toward external subnets with a next hop IP of the physical router. Southbound, static routes can be configured on Tier-0 gateways with a next hop of a layer 3 device reachable via Service interface.

ECMP is supported with static routes to provide load balancing, increased bandwidth, and fault tolerance for failed paths or Edge nodes. Figure 4-4-15 shows a Tier-0 gateway with two external interfaces leveraging Edge node, EN1 and EN2 connected to two physical routers. Two equal cost static default routes configured for ECMP on Tier-0 Gateway. Up to eight paths are supported in ECMP.

Tier-0 northbound static routes can be protected via BFD. BFD timers depend on the Edge node type. Bare metal Edge supports a minimum of 50ms TX/RX BFD

keep alive timer while the VM form factor Edge supports a minimum of 500ms TX/RX BFD keep alive timer.

It is recommended to always implement BFD when configuring static routing between the Tier-0 gateway and the physical network. BFD will detect an upstream physical network failures in the absence of a dynamic routing protocol.

First hop redundancy protocols on the physical network and a virtual IP (VIP) on the Tier-0 Gateway can also be implemented to provide high availability, but they are less reliable and slower than BFD, cannot provide ECMP, and VIP only works with active/standby Tier-0 gateways. They should only be considered if enabling BFD was not an option.

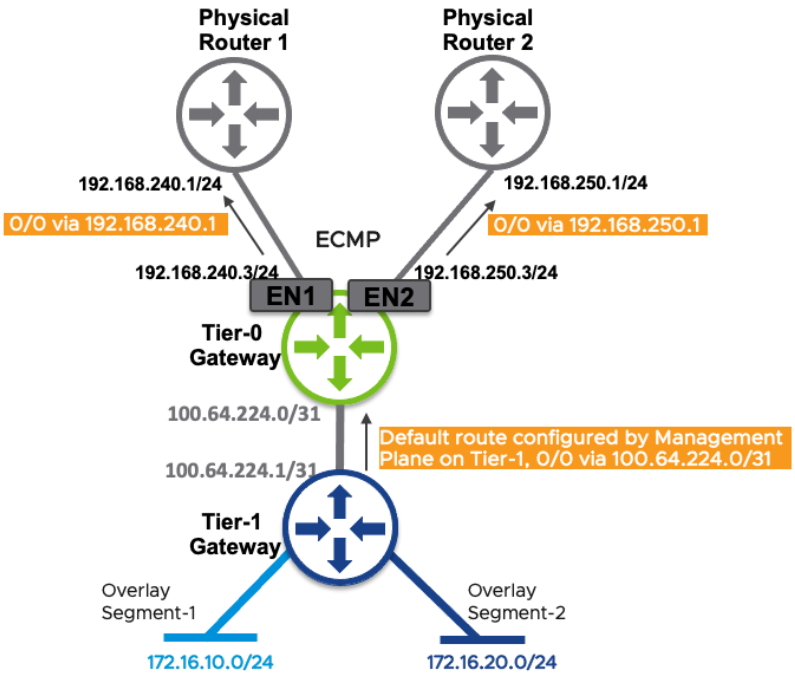


Figure 4-15: Static Routing Configuration

4.3.2Border Gateway Protocol (BGP)

BGP is the de facto protocol on the WAN and in most modern data centers. A typical leaf-spine topology has eBGP running between leaf switches and spine switches.

Tier-0 gateways support eBGP and iBGP on the external interfaces with physical routers. BFD can also be enabled per BGP neighbor for faster failover. BFD timers depend on the Edge node type. Bare metal Edge supports a minimum of 50ms TX/RX BFD keep alive timer while the VM form factor Edge supports a minimum of 500ms TX/RX BFD keep alive timer.

With NSX 3.0 release, the following BGP features are supported:

- Two and four bytes AS numbers in *asplain, asdot* and *asdot+* format.

- eBGP multi-hop support, allowing eBGP peering to be established on loopback interfaces.
- iBGP
- eBGP multi-hop BFD
- ECMP support with BGP neighbors in same or different AS numbers. (Multi-path relax)
- BGP Allow AS in
- BGP route aggregation support with the flexibility of advertising a summary route only to the BGP peer or advertise the summary route along with specific routes. A more specific route must be present in the routing table to advertise a summary route.
- Route redistribution in BGP to advertise Tier-0 and Tier-1 Gateway internal routes as mentioned in section 4.2.2.
- Inbound/outbound route filtering with BGP peer using prefix-lists or route-maps.
- Influencing BGP path selection by setting Weight, Local preference, AS Path Prepend, or MED.
- Standard, Extended and Large BGP community support.
- BGP well-known community names (e.g., no-advertise, no-export, no-export-subconfed) can also be included in the BGP route updates to the BGP peer.
- BGP communities can be set in a route-map to facilitate matching of communities at the upstream router.
- Graceful restart (Full and Helper mode) in BGP.
- BGP peering authentication using plaintext or MD5.
- MP-BGP as the control plane protocol for VXLAN overlays or IPv6 address families.

Active/active ECMP services supports up to eight paths. The ECMP hash algorithm is 5-tuple northbound of Tier-0 SR. ECMP hash is based on the source IP address, destination IP address, source port, destination port and IP protocol. The hashing algorithm determines how incoming traffic is forwarded to the next-hop device when there are multiple paths. ECMP hashing algorithm from DR to multiple SRs is 2-tuple and is based on the source IP address and destination IP address.

Graceful Restart (GR)

Graceful restart in BGP allows a BGP speaker to preserve its forwarding table while the control plane restarts. It is recommended to enable BGP Graceful restart when the BGP peer has multiple supervisors. A BGP control plane restart could happen due to a supervisor switchover in a dual supervisor hardware, planned maintenance, or active routing engine crash. As soon as a GR-enabled router restarts (control plane failure), it preserves its forwarding table, marks the routes as stale, and sets a grace period restart timer for the BGP session to reestablish. If

the BGP session reestablishes during this grace period, route revalidation is done, and the forwarding table is updated. If the BGP session does not reestablish within this grace period, the router flushes the stale routes.

The BGP session will not be GR capable if only one of the peers advertises it in the BGP OPEN message; GR needs to be configured on both ends. GR can be enabled/disabled per Tier-0 gateway. The GR restart timer is 180 seconds by default and cannot be change after a BGP peering adjacency is in the established state, otherwise the peering needs to be negotiated again.

4.3.3Open Shortest Path First version 2 (OSPF v2)

OSPFv2 is a supported dynamic routing protocol supported in NSX starting with version 3.1.1. Based on RFC 2328, OSPFv2 provides fast convergence, scalability and allows NSX for vSphere domain to be migrated seamlessly to NSX.

OSPF is considered as an Interior Gateway Protocol (IGP) as routes are advertised within an autonomous system. E-BGP on the other side is considered as an Exterior Gateway Protocol since the routes are exchanged between different autonomous systems.

OSPF is a routing protocol where routers exchange their link status and information using specific messages called Link State Updates (LSU) to all neighbors on an OSPF enabled segment. The data structure of the LSU contains Link State Advertisements (LSA).

The routers in the OSPF domain flood the LSAs throughout the area and inject them into a Link State Database (LSDb). Each LSDb in the area is identical and the routers compute the SPF calculation using the Dijkstra algorithm.

OSPFv2 is currently not supported on the following architectures:

- Tier-0 VRF
- Federation
- EVPN
- Tier-0 or Tier-1 Service Interfaces.

OSPFv3 is currently not supported (no IPv6 support in the current implementation).

4.3.3.1 OSPF Adjacency

Routers need to establish an OSPF adjacency to exchange LSAs. An OSPF Router-ID (RID) is a 32 bits dotted decimal value that is selected when the protocol is enabled to identify itself in the OSPF domain. A Router ID must be unique within the OSPF domain.

In NSX 3.1.1, the RID is identical between BGP and OSPF.

In NSX 3.1.2, choosing the RID follows the following logic:

- If no loopback is configured, RID will be equal to the highest numeric interface IPv4 address.
- If a loopback is configured the RID will be equal to the highest numeric loopback IPv4 address.

NSX does not support to manually configure an OSPF router-ID. There is no preemption when it comes to calculating the Router ID otherwise that will trigger a reset in the OSPF adjacency and disrupt traffic forwarding.

To change the RID, the OSPF process must be restarted either using the UI or an API call. An NSX edge reboot will not change the RID. If the NSX edge is being redeployed, the OSPF RID will be recalculated.

Once the OSPF RID is chosen, OSPF hello messages are sent on the OSPF enabled external uplinks using the multicast address "224.0.0.5" as represented on FIGURE 4-16.

Figure 4-16 - OSPF Adjacency between the Tier-0 SR and the physical network fabric

OSPF hellos perform multiple duties in the OSPF process:

- Discover the neighbors on a segment and try to establish an adjacency if one OSPF router is detected.
- Check the OSPF configuration between neighbors.
- Monitor the adjacency.

OSPF can be enabled on 2 external uplinks interfaces per Tier-0 Service Router as demonstrated on FIGURE 4-17.

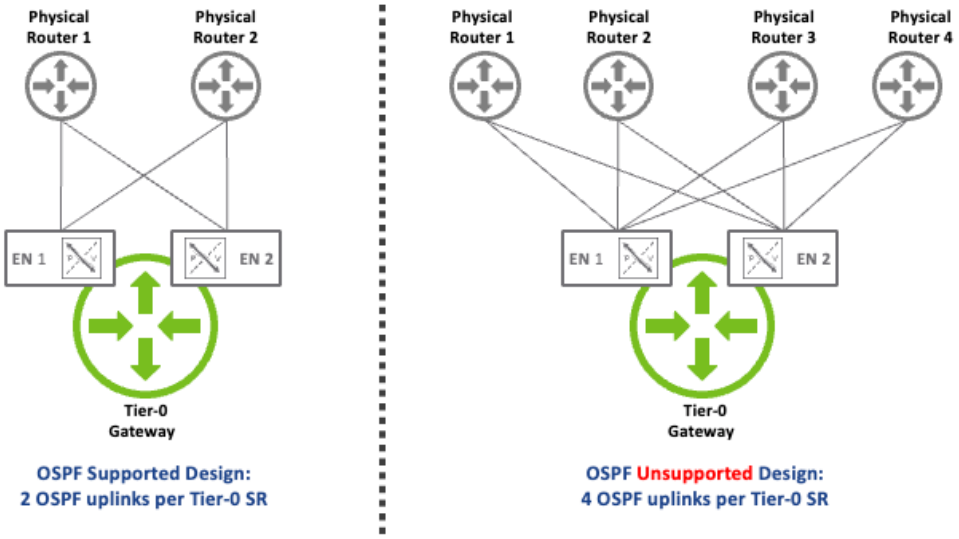


Figure 4-17: OSPF enabled on uplinks

As explained earlier, Hello messages are sent on the OSPF enabled external uplink to the multicast address 224.0.0.5 IP address (ALL SPF Routers). The hello message will check the following parameters before establishing the adjacency:

- Interfaces between the OSPF routers must be in the same subnet.
- Interfaces between the OSPF routers must belong to the same OSPF area.
- Interfaces between the OSPF routers must have the same OSPF area type.
- Router ID must be unique between the OSPF Routers.
- OSPF timers (HelloInterval and RouterDeadInterval) must match between the OSPF routers.
- Authentication process must be validated.

OSPF Authentication

The Tier-0 gateway supports the following authentication method:

- None
- Password: Password is sent in clear text over the external uplink interface
- MD5: Password is a message digest and sent over the external uplink interface

NSX supports area-wide authentication and does not support authentication per interface.

VMware recommends authenticating every dynamic routing protocol adjacency using the MD5 method to exchange routes in a secure way.

NSX currently supports up to 8 characters for the authentication password

Maximum Transmission Unit (MTU) and NSX

Even though the MTU is not verified by the OSPF Hello process, interfaces involved in an OSPF adjacency must have the same MTU value configured to exchange LSUs. While an adjacency is being established, the database description is exchanged between peers using DB Description packets. MTU is verified in the DB Description packet and if the MTU values do not match, the OSPF adjacency is brought down. NSX does not support the "ignore MTU mismatch" feature.

OSPF Timers

Hello packets will be sent on OSPF enabled uplink interface at a specific and configurable value (HelloInterval). It acts as a keepalive to make sure that the neighbors are still present on the segment and can exchange their network topology with each other. A neighbor will be declared down if a router fails to receive OSPF hellos during the time specified by the DeadInterval timer.

By default, in OSPF, HelloInterval and DeadInterval are respectively 10s and 40s on both Point-To-Point and Broadcast OSPF network type interface.

OSPF Network Type	Hello Interval	Dead Interval
Point to Point	Default: 10 seconds Minimum: 1 second	Default: 40 seconds Minimum: 3 seconds
Broadcast	Default: 10 seconds Minimum: 1 second	Default: 40 seconds Minimum: 3 seconds

Figure 4-18: OSPF Timers

Bidirectional Forwarding Detection (BFD) and NSX

BFD can also be enabled per OSPF neighbor for faster failover. BFD timers depend on the Edge node type. Starting with NSX version 3.0, bare metal Edge supports a minimum of 50ms TX/RX BFD keep alive timer while the VM form factor Edge supports a minimum of 500ms TX/RX BFD keep alive timer. VMware strongly recommends configuring BFD on top of every OSPF interfaces to minimize north-south traffic disruption.

FIGURE 4-19 represents the BFD timers supported by NSX:

Edge Node Form Factor	BFD Hello Packet Interval	BFD Dead Multiple	Detection failure
Bare Metal	Minimum supported: 50ms	Minimum supported: 3	150ms
Virtual Machine	Minimum supported: 500ms	Minimum supported: 3	1500ms

Figure 4-19: BFD Timers for OSPF

OSPF Adjacency States

An OSPF router can have multiple OSPF enabled interfaces and therefore different neighbors identified by their OSPF Router-ID (ID). The routers need to keep track of each adjacency state and data structure as specified in RFC 2328 – SECTION 10.1:

- Down State:
  - Initial OSPF state. No information has been received from the neighbor.
- Init State:
  - Hello packet has been seen from the neighbor and his Router ID has been identified. (Router has not seen its own router ID in the OSPF Hello received from the neighbor).
- 2-Way:
  - The OSPF router has seen its own OSPF Router ID in the received Hello packet. Bidirectional communication between the two peers is established. It is critical to understand that on an OSPF Broadcast network type adjacency, OSPF DROther routers can stay in this state indefinitely since the adjacency will be fully established with the Designated Router (DR) and Backup Designated Router (BDR).
- ExStart:
  - In this state, the OSPF routers will try to elect an OSPF Designated Router and Backup Designated Router. The router with the highest OSPF Router-ID will be elected DR. The NSX Tier-0 will never be elected DR or BDR since its OSPF Priority is equal to "0". The Tier-0 SR is considered as an OSPF DROther router.
- Exchange:
  - Each router is sending a complete description of its LSDb by sending Database Description packets. LSAs request based on the current knowledge is requested at this step. The Link State Update flooding process can start after the requests have been made.
- Loading:
  - Link State Request and Link-State-Update packets are exchanged between OSPF neighbors so that they can have the same knowledge of the OSPF topology.
- Full:
  - Routers have exchanged their LSDb and are fully adjacent.

After both routers have exchanged their LSDb, they run the Dijkstra algorithm. Since their LSDb is identical they possess the same knowledge of the OSPF topology. The Dijkstra algorithm is run so that each router can calculate their own best way to reach every destination.

Graceful Restart (GR)

Graceful restart in OSPF allows a neighbor to preserve its forwarding table while the control plane restarts. It is recommended to enable OSPF Graceful restart when the OSPF peer has multiple supervisors. An OSPF control plane restart could happen due to a supervisor switchover in a dual supervisor hardware, planned maintenance, or active routing engine crash. As soon as a GR-enabled router restarts (control plane failure), it preserves its forwarding table, marks the

routes as stale, and sets a grace period restart timer for the OSPF adjacency to reestablish. If the OSPF adjacency reestablishes during this grace period, route revalidation is done, and the forwarding table is updated. If the OSPF adjacency does not reestablish within this grace period, the router flushes the stale routes.

OSPF Graceful restart helper mode is supported.

4.3.3.2 OSPF Network Types

NSX supports both Point to Point and Broadcast OSPF network types.

Point to multipoint, Point to Multipoint Non-Broadcast and Non-Broadcast network types are considered deprecated in any modern Data Center architecture and are not supported.

Figure 4-20represents the different OSPF network types supported in NSX 3.1.1

OSPF Network Type	DR / BDR on the segment	Number of OSPF Routers on the segment
Point-to-Point	No Both SPF Routers are considered "DROther"	2 maximum
Broadcast	Yes Tier-0 is never a DR or BDR	Multiple but adjacency is fully established only with the DR or BDR. DROther routers will see each other and will stay in a "2-way" state.

Figure 4-20: Supported OSPF Network types

OSPF Point to Point Network Type:

OSPF Point-to-Point network type is the recommended network type for most OSPF deployment. It simplifies the overall OSPF peering topology. On a given ethernet segment, the maximum number of routers fully establishing an OSPF adjacency using the Point-to-Point network type is two. No DR or BDR election is performed using this OSPF network type. The Tier-0 SR OSPF Routers are considered "DROther" (OSPF priority of 0).

It is recommended to configure the Tier-0 SR uplink interfaces using a /31 network mask as depicted on FIGURE 4-20:

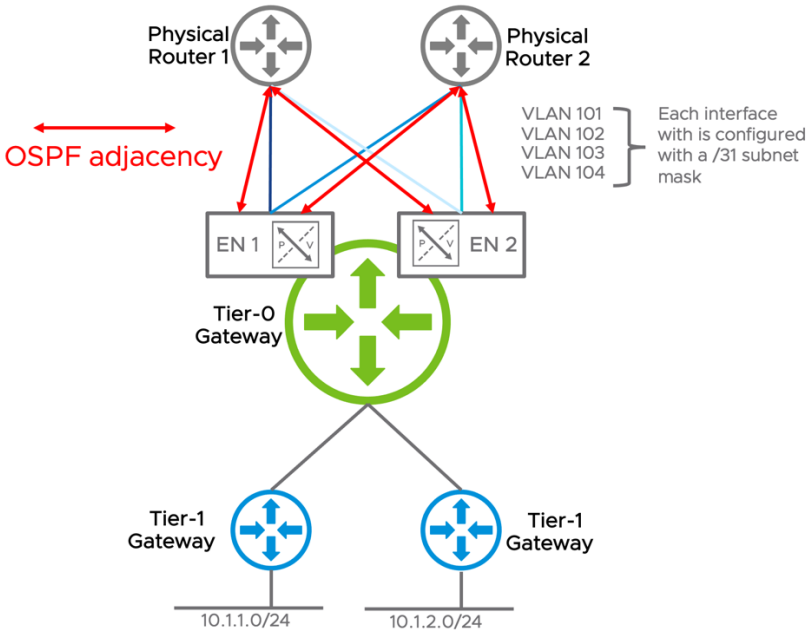


Figure 4-20: OSPF Point to Point network type topology

In this example, the Tier-0 SR hosted on both the Edge Node 01 and Edge Node 02 will establish an adjacency with both the physical router 1 and the physical router 2.

From a global Tier-0 perspective, all four adjacencies to the physical routers will be in the "Full" state.

OSPF Broadcast Network Type:

On an ethernet segment using the OSPF broadcast type, more than two OSPF router can exchange their LSAs. To reduce the LSAs flooding on that segment, the OSPF process will elect a Designated Router and a Backup Designated Routers. The router with the highest OSPF priority (1-255) will be elected DR. The router with the second highest OSPF priority will be elected BDR. All other routers on the segment will be considered OSPF DROthers.

If the priorities for either the DR or BDR role are equal, the router with the highest OSPF RouterID will be elected DR.

All DROther routers will establish an adjacency with both the DR and the BDR as demonstrated on FIGURE 4-21. These adjacencies will be in the "Full" state and LSUs can be exchanged. The DR and BDR also establish an adjacency between themselves.

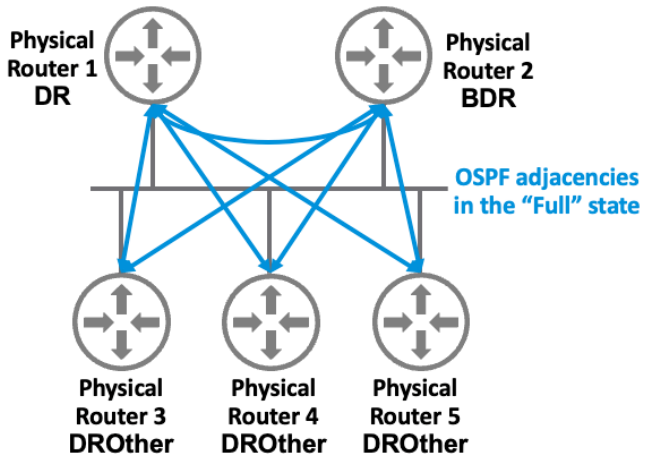


Figure 4-21: OSPF adjacency in the Full State – Broadcast network type

The DROthers routers will see each other's in the 2-Way state but will not exchange their LSAs directly between themselves, the DR and BDR will perform that function.

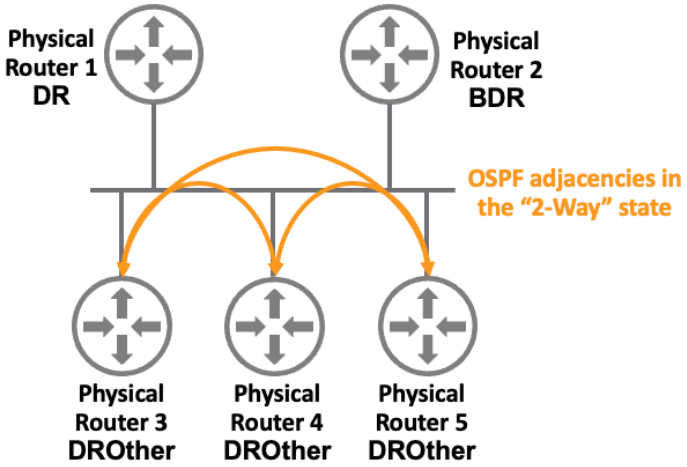


Figure 4-22: OSPF adjacency in the "2-Way" State – Broadcast network type

In this example, each router will have 2 adjacencies in the Full State and 2 adjacencies in the "2-Way" state.

From an NSX perspective, the Tier-0 gateway is always a DROther as its OSPF priority is hard-coded to 0. It is not possible to change that priority.

FIGURE 4-23 demonstrates the OSPF adjacencies in the "Full" state between the physical networking fabric and the Tier-0 Service Routers

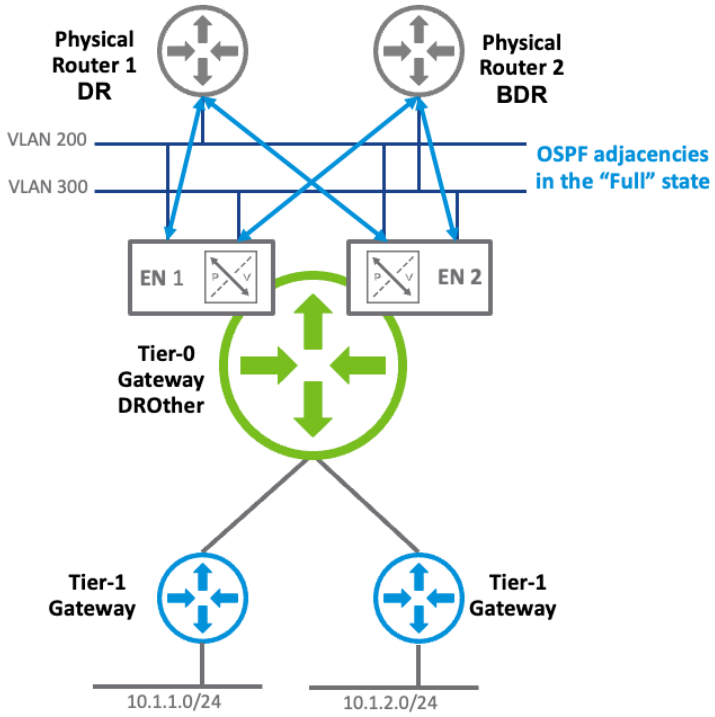


Figure 4-23: OSPF adjacency in the Full State – Broadcast network type

FIGURE 4-24 demonstrates the OSPF adjacencies in the "2-Way" state between the Tier-0 Service Routers.

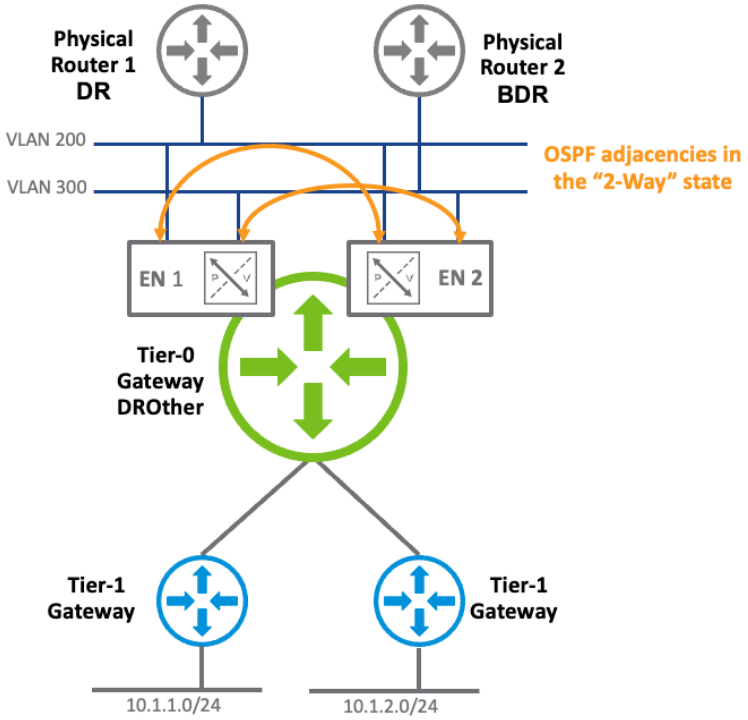


Figure 4-24: OSPF adjacencies in the "2-Way" state

When OSPF uses the broadcast network type, the Database Description packets are sent from the Tier-0 SR (DROthers) to the Designated Router in the networking fabric using the multicast address 224.0.0.6 (ALL OSPF DR Router multicast address). The DR sends its Database Description packets using the 224.0.0.5 multicast address (ALL OSPF Router multicast address).

The number of adjacencies is increased using this network type which can increase the complexity of the OSPF topology.

The OSPF LSDb is identical throughout the entire area therefore, there is no need to establish an adjacency between the Tier-0 SR.

As explained earlier, VMware recommends using the Point-to-Point OSPF network type over the Broadcast network type to simplify the routing topology.

4.3.3.3 OSPF Areas

As mentioned previously, all the routers in an OSPF area must have the same knowledge of the networking topology. Therefore, their LSDb must be identical. It contains all the LSA information flooded by all the OSPF routers in the area. When there is a network change in the area (e.g link failure or cost changed), the router that experienced the change will send newer LSAs to update all the OSPF routers. This will trigger all the OSPF routers in the area to re-run the Dijkstra algorithm and recalculate the best path to reach the destinations.

This can be problematic in a very large-scale environment where it would be unnecessary for all routers to recompute the Dijkstra algorithm when a networking factor has changed.

OSPF routers and links can be compartmented into different areas to limit the LSA flooding and reduce the time the SPF algorithm is run.

Routers with links in both a backbone area and a standard area are considered as Autonomous Border Router. The Tier-0 gateway cannot perform ABR's duties since only one area per Tier-0 is supported.

FIGURE 4-25 represents the OSPF LSAs that are supported in an NSX OSPF architecture.

OSPF LSA Type	Name	Description
1	Router	Collected states of the router interface
2	Network	Represents the set of routers attached to an ethernet segment.
3	Summary	Describes inter-area routes. Originated by an ABR
4	Summary (ASBR)	Describes routes to an ASBR. Originated by an ABR
5	External	Describes routes redistributed into the OSPF domain. Originated by an ASBR
7	External (NSSA)	Describes routes redistributed into the OSPF domain using a special "Not So Stubby Area" Originated by an ASBR These LSAs are flooded only within the originating NSSA.

Figure 4-25: OSPF LSAs support

The OSPF areas supported by NSX are listed in the FIGURE 4-26 and represented in FIGURE 4-27.

OSPF Area	Description and LSA Type Support
Standard	Non backbone area that is connected to a backbone area using an ABR. Support LSA Type : 1 – 2 – 3 – 4 – 5
Backbone	Has the knowledge of the entire topology. Inter-Area traffic must flow through it. Support LSA Type : 1 – 2 – 3 – 4 – 5
Not So Stubby	Does not allow "External LSA – Type 5" to be injected but will rather use "Not so Stubby LSA - Type 7". Support LSA Type : 1 – 2 – 3 – 7 (Single LSA type 3 is supported and represented by a default route)

Figure 4-26: OSPF Area support table

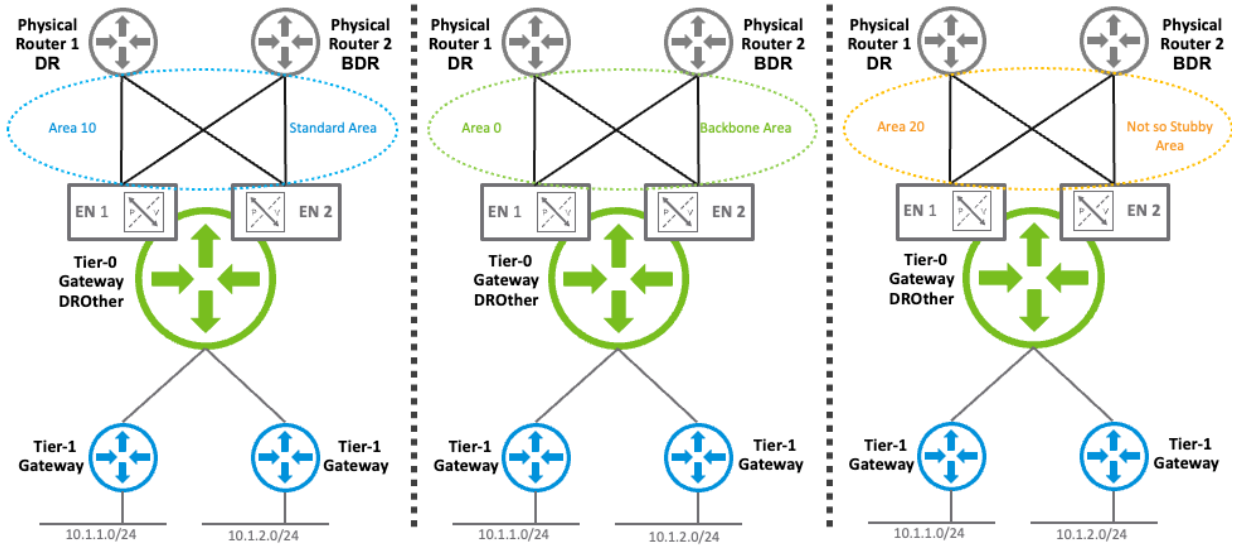


Figure 4-27: OSPF Area support representation

Since a Tier-0 service router redistributes the routes into the OSPF domain, it can be considered as an Autonomous System Boundary Router. An ABR in the OSPF topology will inject LSA type 4 in other areas to make sure the OSPF routers know how to reach the ASBR. When a Tier-0 SR redistributes a prefix into OSPF, it uses an external metric of type 2:

- Total cost to reach the prefix is always the redistributed metric. Internal cost to reach the ASBR is ignored.

The OSPF External type 1 is not supported by NSX when a prefix is redistributed.

FIGURE 4-28 represents the different external metric types used by the Tier-0 when different area types are used:

- Standard and Backbone areas: Routes are redistributed using LSAs type 5 with an external type of E2 and a cost of 20.
- When the Tier-0 gateway is connected to a not so stubby area, it redistributes its routes using LSAs type 7 with an external type of N2 and a cost of 20.

The OSPF metric cannot be changed in NSX and will have a hard coded value of 20 throughout the OSPF domain.

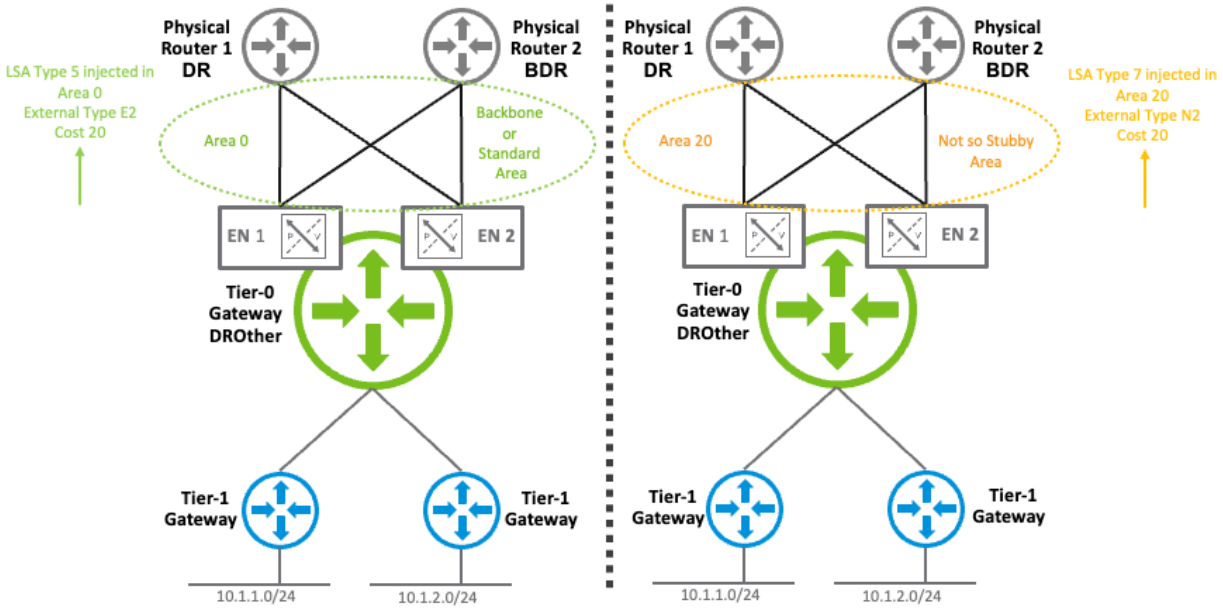


Figure 4-28: OSPF LSA Types injected by a Tier-0 gateway

Not So Stubby Areas are recommended for very large topologies. Instead of redistributing the prefixes using LSA type 5 in the OSPF domain, LSA type 7 will be used. This type of LSA is originated by the ASBR and flooded in the NSSA only. Another ABR in the OSPF domain will translate that LSA type 7 into an LSA type 5 that will be transmitted throughout the OSPF domain.

With the advent of OSPF support in NSX 3.1.1, it is now possible to choose which protocol the routes can be redistributed into. It is possible to redistribute prefixes in BGP or OSPF.

It is possible to use OSPF to learn BGP peers IP addresses in the case of E-BGP multi-hop topology.

FIGURE 4-29 represents an example where OSPF and BGP can be used on the same Tier-0 gateway. OSPF is used as an IGP to provide connectivity between the loopback interfaces.

BGP can be setup between the physical routers' loopback interfaces and the Tier-0 loopback interfaces. This solution should only be considered when direct single hop eBGP peering is not an option. Chapter 7 provides guidance on the recommended routing configuration for BGP peering on directly connected segments.

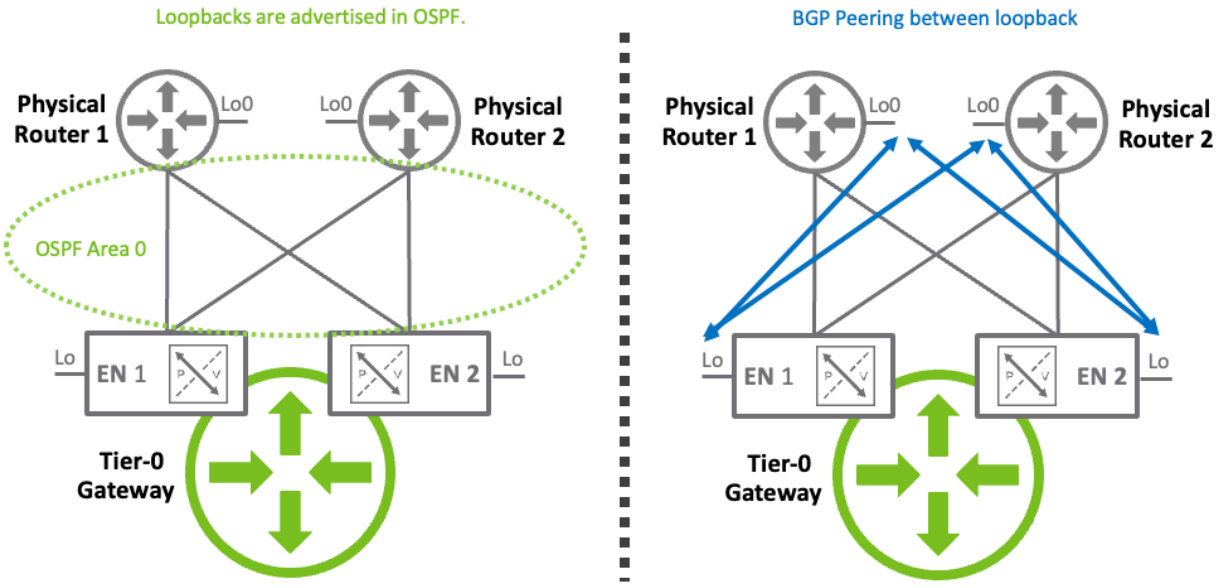


Figure 4-29: OSPF used to learn Loopback reachability to establish a E-BGP multi-hop peering

4.3.3.4 OSPF Topologies and ECMP

NSX supports OSPF on both the Active/Active and Active/Standby topologies.

Equal cost multipath is supported on OSPF towards the physical fabric on both the Active/Active and Active/Standby topologies.

To leverage ECMP between the NSX Tier-0 gateway and the physical networking fabric, OSPF routes must be learned with the same metric. When this conditions is met, the Tier-0 SR can install all the routes as "multipath" in the routing table if ECMP is enabled in the routing protocol configuration.

With BGP and static routes, each Tier-0 SR can leverage up to 8 different links and next-hop to load balance efficiently the traffic. 2 uplinks can be enabled for OSPF per Tier-0 SR. The number of ECMP routes that can be installed per Tier-0 for each prefix learned by OSPF is 2 as demonstrated in FIGURE 4-27.

The edge node network kernel module support hashing with 5 tuples (Source IP – Destination IP – IP protocol – L4 Source Port – L4 Destination Port).

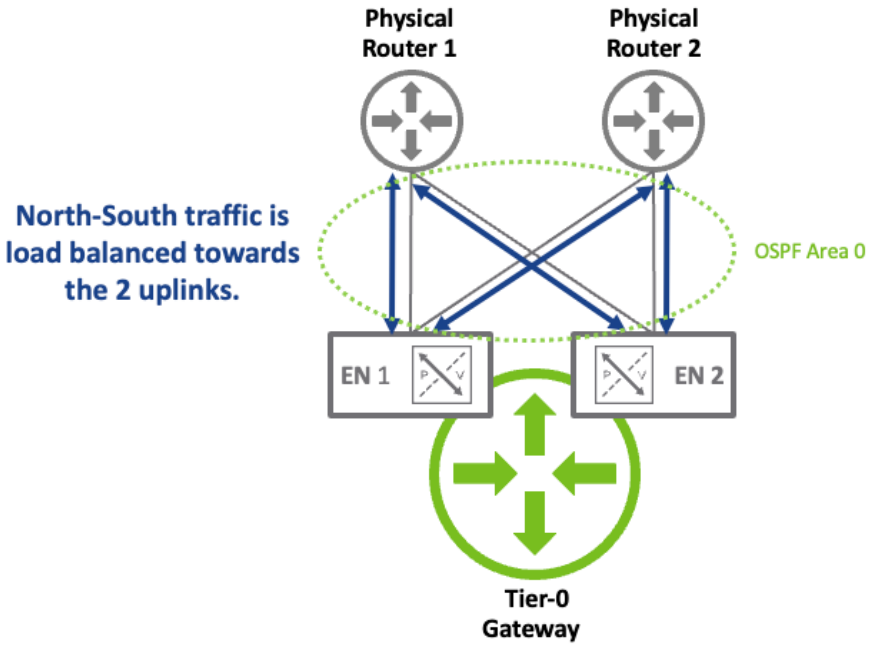


Figure 4-27: OSPF and ECMP

OSPF in an Active / Standby topology

In the active/standby topology represented in FIGURE 4-28, the Tier-0 gateway running in standby mode is running the OSPF process and establishes an adjacency with the top of rack switches. The interface between the Standby Tier-0 SR and Tier-0 DR is not active and as a consequence, the DR will forward all the traffic to the SR on the active Edge Node (EN1 in the example provided on FIGURE 4-28).

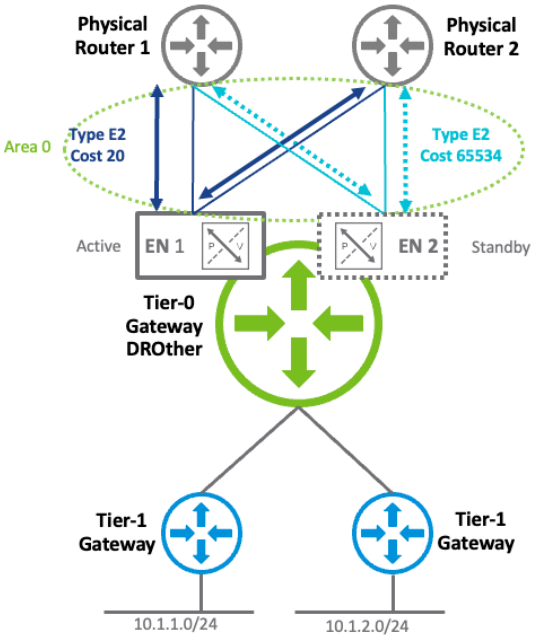


Figure 4-28: OSPF Architecture in Active/Standby Topology

Since the internal transit interface on the standby Tier-0 SR is administratively shutdown, the physical fabric needs to send the traffic to the NSX domain via the active Tier-0 SR.

The standby Tier-0 SR will redistribute the internal NSX prefixes in OSPF and advertise them with a cost of 65534 (External Type E2 or N2).

The active Tier-0 SR redistributes the same prefixes with a cost of 20, the networking fabric will logically prefer the prefixes with a lower cost and therefore use the links via the active Tier-0 SR.

In this topology, ECMP with the top of rack switches can still be leveraged as demonstrated on FIGURE 4-29. There is no ECMP between the Tier-0 DR and the Tier-0 SR.

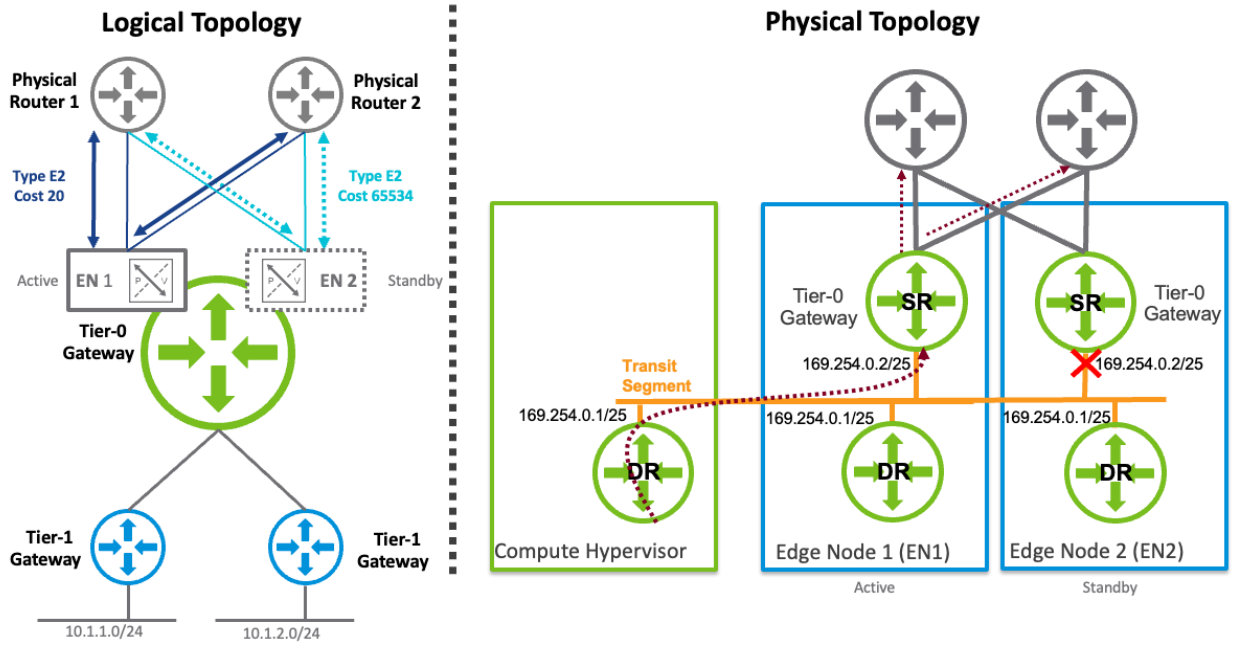


Figure 4-29: ECMP between the physical fabric and the Active Tier-0 SR

OSPF in an Active / Active topology

When no stateful services are running on the Tier-0 gateway, the Active/Active high availability mode can be selected. ECMP can be leveraged between the top of rack switches and the Tier-0 gateway as well as between the Tier-0 DR and the Tier-0 SR in a single tier routing architecture.

The topology demonstrated in FIGURE 4-30 shows the different level of ECMP available in this topology:

- ECMP between the Tier-0 DR and the Tier-0 SR. The Tier-0 DR has two default routes pointing to both internal transit segment interfaces on the Tier-SRs.
- ECMP between the Tier-0 SR and the physical networking fabric. OSPF can support up to two ECMP paths.

This topology is recommended when no services are running on the Tier-0 gateway or when predictable traffic flow is preferred.

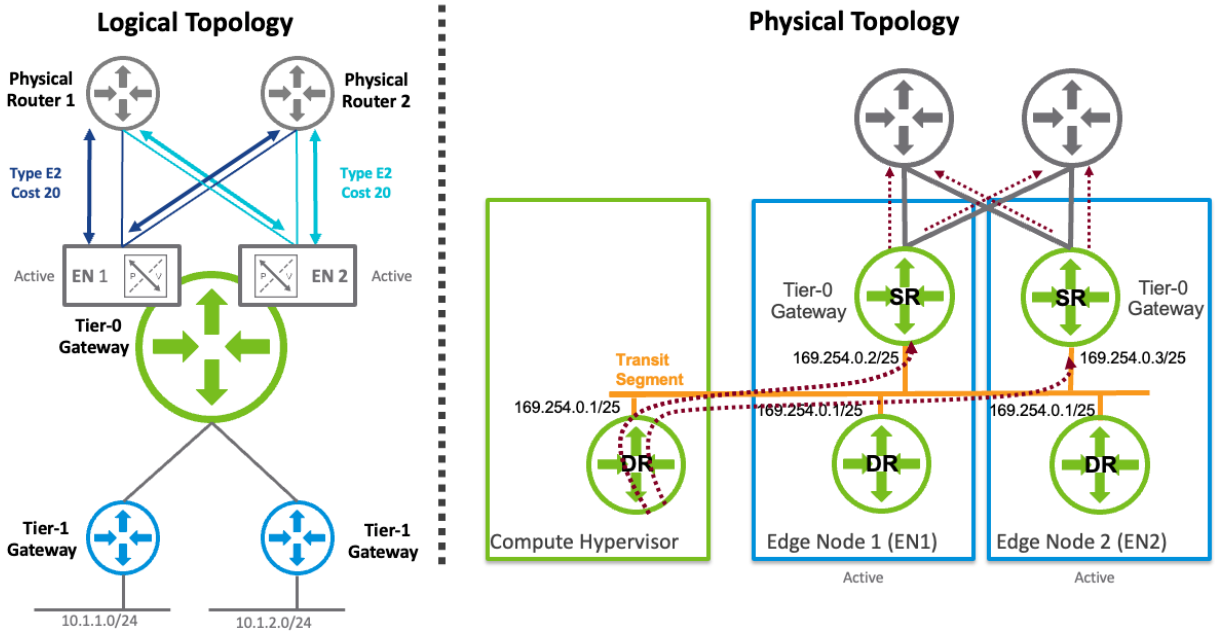


Figure 4-30:ECMP architecture in Active/Active topology

4.3.3.5 OSPF Summarization

The Tier-0 Service Routers is considered as an ASBR as it redistributes routes into the OSPF domain. In OSPF, Summarization at the ABR and ASBR level is possible.

In networking, summarization usually solve some scalability issues as it decreases the number of routes installed in the routing table.

There are other benefits to summarization directly related to OSPF as if one network disappears in one area, OSPF routers in subsequent area will not have to remove that prefix from the routing table. This rationale will lower the number of CPU cycles needed for OSPF to run efficiently.

Lowering the number of routes in the routing table has some beneficial effects on the physical resources consumed by the networking appliances (CPU, Memory)

In FIGURE 4-31, a large number of Tier-1 gateways are instantiated in the NSX domain. Each gateway contains a single /24 network for each tenant. Without summarization, each Tier-0 SR will advertise the Tier-1 connected segments with 256x Type 5 LSAs. Each Type 5 LSA will advertise a /24 network to the physical fabric.

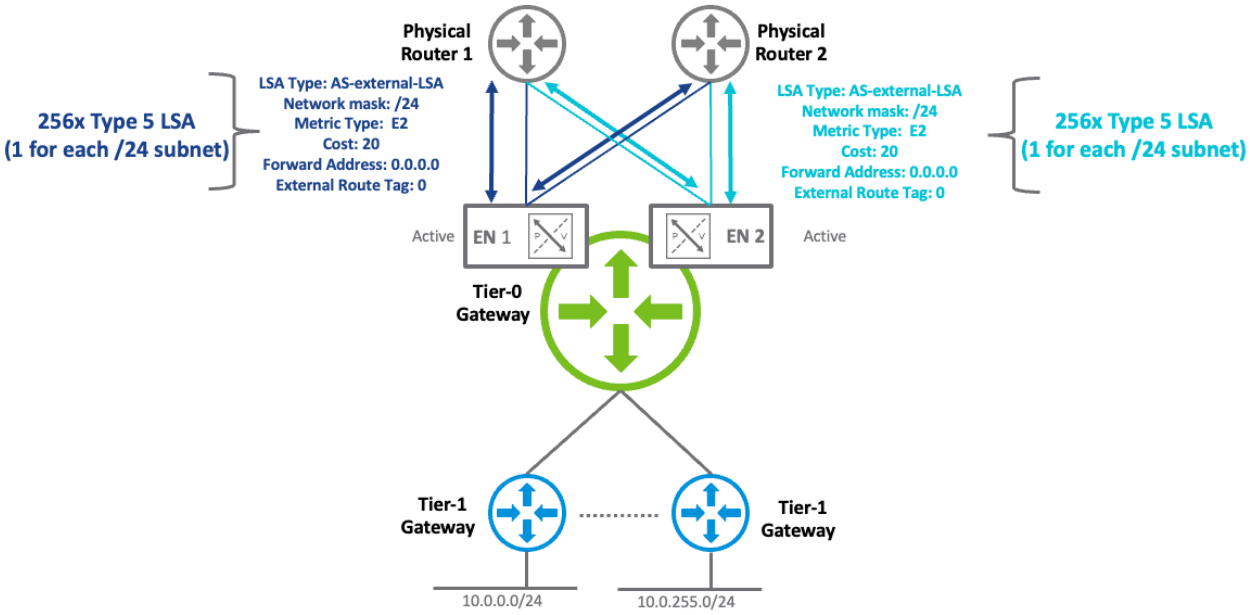


Figure 4-31: LSA type 5 advertisement without summarization

If the IP Addressing schema has been designed properly for summarization (contiguous subnets), these large number of Type 5 LSAs can be advertised as a single summary prefix embedded in a single LSA Type 5 as depicted in figure 4-32.

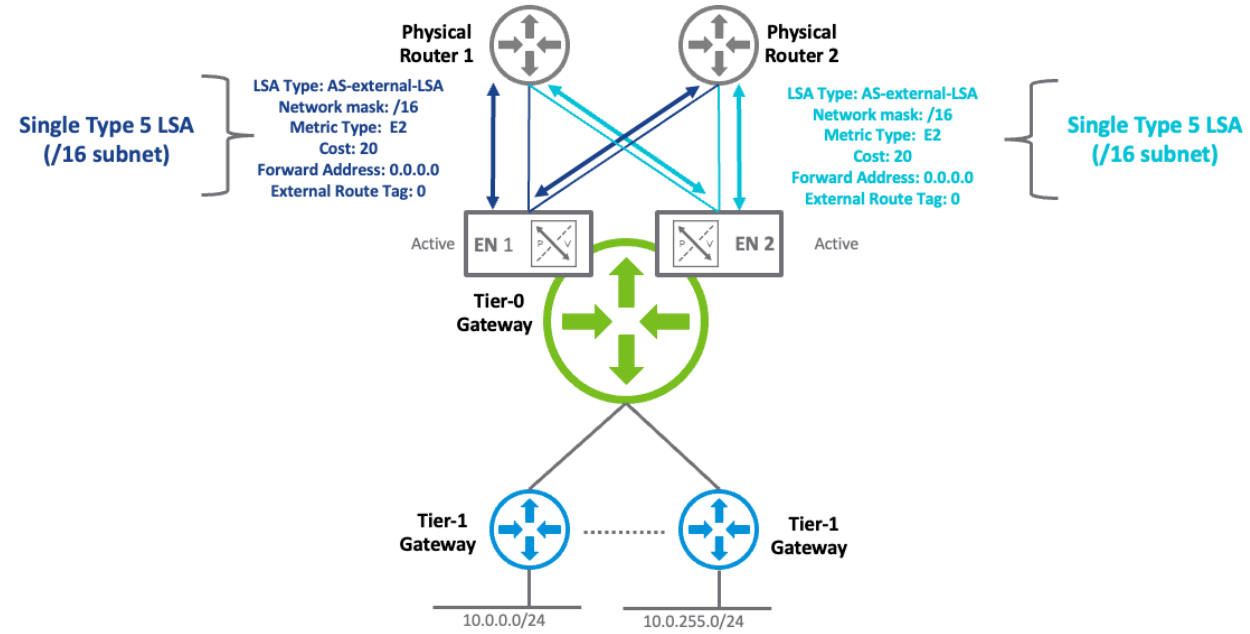


Figure 4-32: LSA type 5 advertisement with route summarization

When the area connected to the Tier-0 gateway is an NSSA, route summarization is supported and the type of LSA injected in the area for summarization is an LSA Type 7.

4.3.3.6 OSPF Redistribution

The administrators have the choice to use route maps to redistribute certain prefixes when performing routes redistribution into OSPF. Routes maps in OSPF are used to allow or filter specific prefixes and it is not possible to set any action in these route maps, such as alter the metric, set the metric-type, or apply a custom tag. The T0 attaches to the redistributed routes a system generated tag that cannot be modified.

Redistributing prefixes between OSPF and BGP is currently not supported.

4.3.3.7 OSPF Default information originate

Advertising a default route from the Tier-0 gateway is supported as demonstrated in FIGURE 4-32.

Enabling the default information originate feature on a Tier-0 gateway is not a common requirement because few designs require NSX to connect the data center to external resources. When required, it should be performed carefully as it may impact the entire data center traffic.

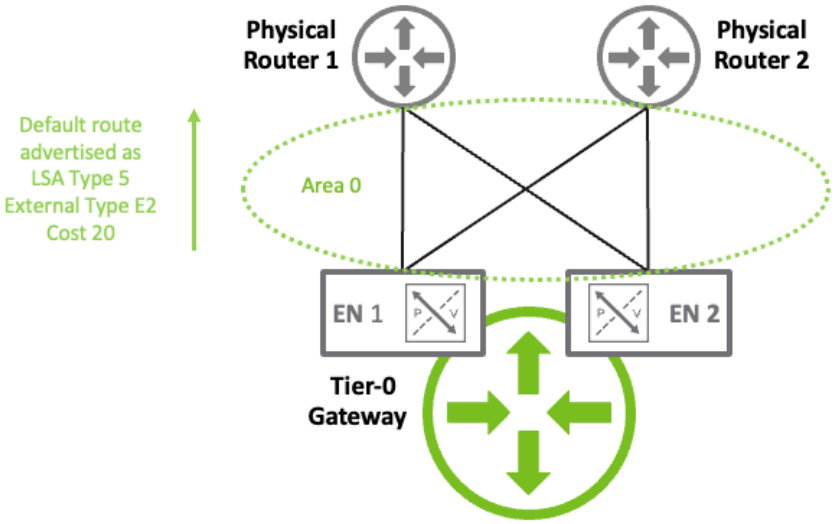


Figure 4-32: Originating a default route into OSPF

4.3.4 Routing protocol recommendation in the data center.

While NSX support static routing, HA VIP and Proxy ARP, VMware recommends the use of a dynamic routing protocol to provide scalability and better redundancy planning in the data center.

The data center will be able to grow and adapt to any change automatically using a dynamic routing protocol. Statically configuring the routing components in the data center will complexify the overall architecture as the data center scale increase.

From a dynamic routing perspective, there are two options to when it comes to designing a modern data center routing architecture with NSX:

- OSPFv2
- BGP

BGP is well known in the networking industry to be the best routing protocol when it comes to interoperability as it exchanges prefixes between autonomous systems on the internet.

Modern and scalable data centers architecture rely on BGP to provide connectivity as multiprotocol support is not possible with OSPFv2. OSPFv2 implementation require a different routing protocol or static routing to route IPv6 packets.

VMware recommends using BGP over OSPF as it provides more flexibility, had a better feature set and support address families.

OSPF should be considered when it is the only routing protocol running in the physical fabric and no feature available only with BGP is required. In such a scenario OSPF avoids the implementation of routing redistribution leading to simpler and more scalable design.

A comparison of features between OSPF and BGP is described in the Table 4-6.

OSPFv2	BGP
Link State routing protocol (LSA Flooding)	Path Vector routing protocol
IPv6 not supported by the protocol	MP-BGP Address families supported (IPv4 – IPv6 - L2EVPN)
Federation – VRF – EVPN not supported	Federation – VRF – EVPN supported
BFD and Graceful Restart supported	BFD and Graceful Restart supported
Troubleshooting and protocol complicated	Simpler troubleshooting.
Summarization and Filtering supported at the ASBR level	Summarization and Filtering supported
Route maps supported (no set actions)	Route maps supported (set actions supported and provide better flexibility in routing policy)

Table 4-6: OSPF vs BGP comparison in the Data Center.

4.4 IPv6 Routing Capabilities

NSX also supports dual stack for the interfaces on a Tier-0 or Tier-1 Gateway. Users can leverage distributed services like distributed routing and distributed firewall for East-West traffic in a single tier topology or multi-tiered topology for IPv6 workloads now. Users can also leverage centralized services like Gateway Firewall for North-South traffic.

NSX Datacenter supports the following unicast IPv6 addresses:

- Global Unicast: Globally unique IPv6 address and internet routable
- Link-Local: Link specific IPv6 address and used as next hop for IPv6 routing protocols
- Unique local: Site specific unique IPv6 addresses used for inter-site communication but not routable on internet. Based on RFC4193.

The following table shows a summarized view of supported IPv6 unicast and multicast address types on NSX Datacenter components.

NSX-T Component	Unicast Addresses Supported	Multicast Addresses Supported
Tier-0 or Tier-1 Gateway Distributed Router (DR)	Global, Unique Local, Link Local	All node address (FF02::1) All Routers address (FF02::2) Solicited-node multicast address (FF02::1:FF:0/104)
Tier-0 or Tier-1 Gateway Services Router (SR)	Global, Unique Local, Link Local	All node address (FF02::1) All Routers address (FF02::2) Solicited-node multicast address (FF02::1:FF:0/104)
Inter-Tier Transit Link (Router link)	Global, Unique Local, Link Local	All node address (FF02::1) All Routers address (FF02::2)
Intra-Tier Transit Link (SR-DR Link)	Link Local	All node address (FF02::1) All Routers address (FF02::2)

Figure 4-33: Type of IPv6 addresses supported on Tier-0 and Tier-1 Gateway components

FIGURE 4-34 shows a single tiered routing topology on the left side with a Tier-0 Gateway supporting dual stack on all interfaces and a multi-tiered routing topology on the right side with a Tier-0 Gateway and Tier-1 Gateway supporting dual stack on all interfaces. A user can either assign static IPv6 addresses to the workloads or use a DHCPv6 relay supported on gateway interfaces to get dynamic IPv6 addresses from an external DHCPv6 server.

For a multi-tier IPv6 routing topology, each Tier-0-to-Tier-1 peer connection is provided a /64 unique local IPv6 address from a pool i.e. fc5f:2b61:bd01::/48. A user has the flexibility to change this subnet range and use another subnet if desired. Similar to IPv4, this IPv6 address is auto plumbed by system in background.

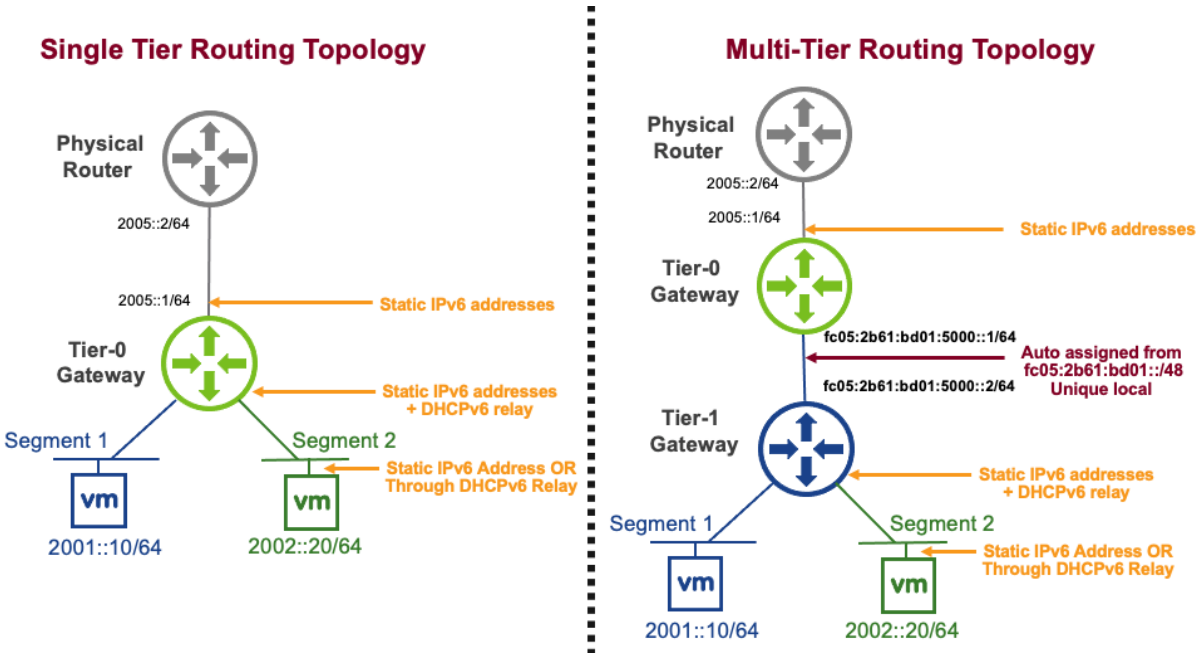


Figure 4-34: Single tier and Multi-tier IPv6 routing topology

Tier-0 Gateway supports following IPv6 routing features:

- Static routes with IPv6 Next-hop
- MP-eBGP with IPv4 and IPv6 address families
- Multi-hop eBGP
- IBGP
- ECMP support with static routes, EBGP and IBGP
- Outbound and Inbound route influencing using Weight, Local Pref, AS Path prepend and MED.
- IPv6 Route Redistribution
- IPv6 Route Aggregation
- IPv6 Prefix List and Route map
- IPv6 Loopback Interfaces

Tier-1 Gateway supports following IPv6 routing features:

- Static routes with IPv6 Next-hop

IPv6 routing between Tier-0 and Tier-1 Gateway is auto plumbed similar to IPv4 routing. As soon as Tier-1 Gateway is connected to Tier-0 Gateway, the management plane configures a default route (::/0) on Tier-1 Gateway with next hop IPv6 address as Router link IP of Tier-0 Gateway (fc05:2b61:bd01:5000::1/64,

as shown in FIGURE 4-35). To provide reachability to subnets connected to the Tier-1 Gateway, the Management Plane (MP) configures routes on the Tier-0 Gateway for all the LIFs connected to Tier-1 Gateway with a next hop IPv6 address as Tier-1 Gateway Router link IP (fc05:2b61:bd01:5000::2/64, as shown in figure 4-32). 2001::/64 & 2002:/64 are seen as "Tier-1 Connected" routes on Tier-0.

Northbound, Tier-0 Gateway redistributes the Tier-0 connected and Tier-1 Connected routes in BGP and advertises these to its eBGP neighbor, the physical router.

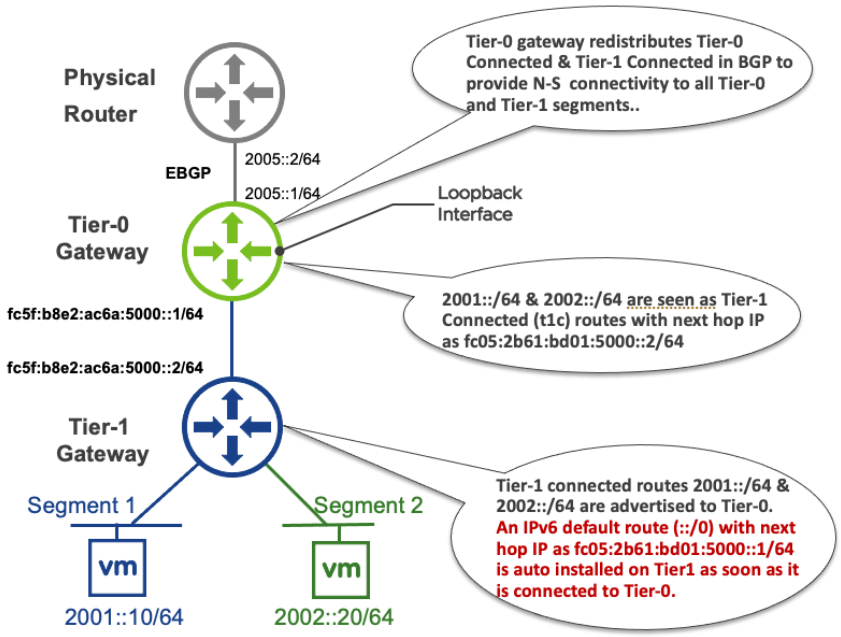


Figure 4-35: IPv6 Routing in a Multi-tier topology

4.5 Services High Availability

NSX Edge nodes run in an Edge cluster, hosting centralized services, and providing connectivity to the physical infrastructure. Since the services are run on the SR component of a Tier-0 or Tier-1 gateway, the following concept is relevant to SR. This SR service runs on an Edge node and has two modes of operation – active/active or active/standby. When a Tier-1 gateway is configured to be hosted on an Edge cluster, an SR is automatically instantiated even if no services are configured or running on the Tier-1. When a Tier-1 SR is instantiated, the Tier-0 DR is removed from the hypervisors and located on the edge nodes only.

4.5.1 Active/Active

Active/Active – This is a high availability mode where SRs hosted on Edge nodes act as active forwarders. Stateless services such as layer 3 forwarding are IP based, so it does not matter which Edge node receives and forwards the traffic. All the SRs configured in active/active configuration mode are active forwarders. This high availability mode is only available on Tier-0 gateway.

Stateful services typically require tracking of connection state (e.g., sequence number check, connection state), thus traffic for a given session needs to go through the same Edge node. As of NSX 3.0, active/active HA mode does not support stateful services such as Gateway Firewall or stateful NAT. Stateless services, including reflexive NAT and stateless firewall, can leverage the active/active HA model.

Left side of FIGURE 4-36 shows a Tier-0 gateway (configured in active/active high availability mode) with two external interfaces leveraging two different Edge nodes, EN1 and EN2. Right side of the diagram shows that the services router component (SR) of this Tier-0 gateway instantiated on both Edge nodes, EN1 and EN2. A Compute host, ESXi is also shown in the diagram that only has distributed component (DR) of Tier-0 gateway.

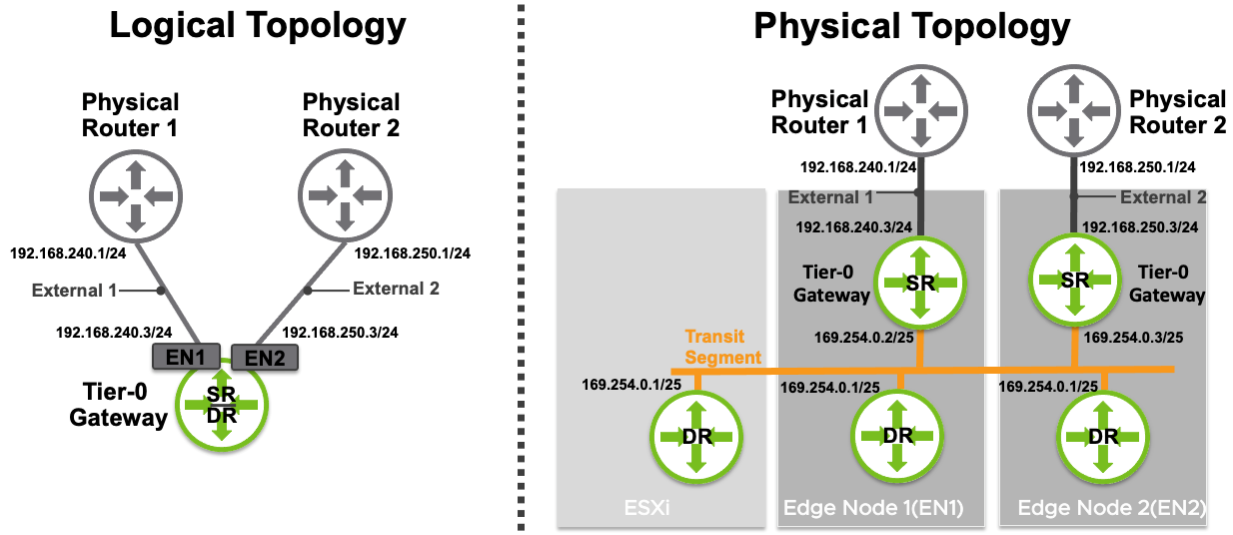


Figure 4-36: Tier-0 gateway configured in Active/Active HA mode

Note that Tier-0 SR on Edge nodes, EN1 and EN2 have different IP addresses northbound toward physical routers and different IP addresses southbound towards Tier-0 DR. Management plane configures two default routes on Tier-0 DR with next hop as SR on EN1 (169.254.0.2) and SR on EN2 (169.254.0.3) to provide ECMP for overlay traffic coming from compute hosts.

North-South traffic from overlay workloads hosted on Compute hosts will be load balanced and sent to SR on EN1 or EN2, which will further do a routing lookup to send traffic out to the physical infrastructure.

A user does not have to configure these static default routes on Tier-0 DR. Automatic plumbing of default route happens in background depending upon the HA mode configuration.

Inter-SR Routing

To provide redundancy for physical router failure, Tier-0 SRs on both Edge nodes must establish routing adjacency or exchange routing information with different physical router or TOR. These physical routers may or may not have the same routing information. For instance, a route 192.168.100.0/24 may only be available on physical router 1 and not on physical router 2.

For such asymmetric topologies, users can enable Inter-SR routing. This feature is only available on Tier-0 gateway configured in active/active high availability mode. Figure 4-34 shows an asymmetric routing topology with Tier-0 gateway on Edge node, EN1 and EN2 peering with physical router 1 and physical router 2, both advertising different routes.

When Inter-SR routing is enabled by the user, an overlay segment is auto plumbed between SRs (similar to the transit segment auto plumbed between DR and SR) and each end gets an IP address assigned in 169.254.0.128/25 subnet by default. An IBGP session is automatically created between Tier-0 SRs and northbound routes (EBGP and static routes) are exchanged on this IBGP session.

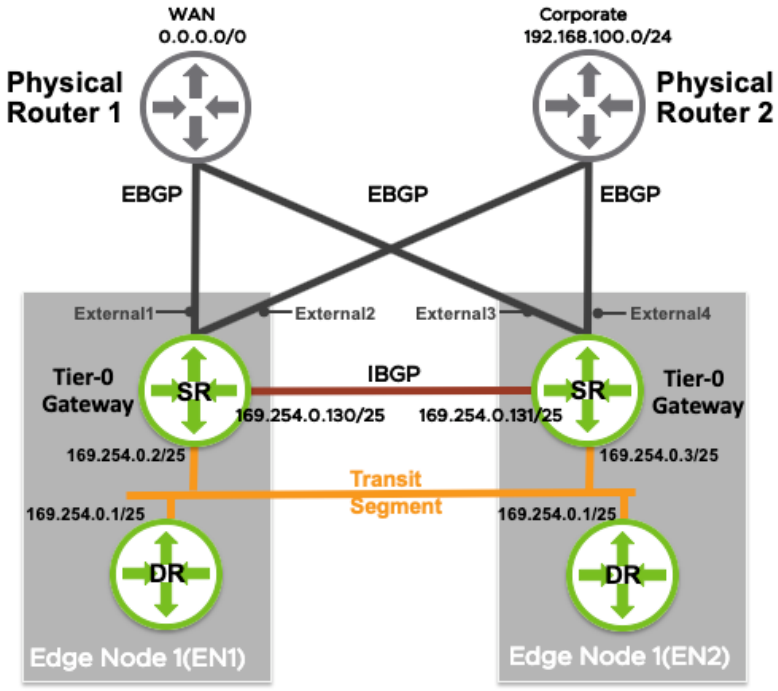


Figure 4-37: Inter-SR Routing

As explained in previous figure, Tier-0 DR has auto plumbed default routes with next hops as Tier-0 SRs and North-South traffic can go to either SR on EN1 or EN2. In case of asymmetric routing topologies, a particular Tier-0 SR may or may not have the route to a destination. In that case, traffic can follow the IBGP route to another SR that has the route to destination.

FIGURE 4-37 shows a topology where Tier-0 SR on EN1 is learning a default WAN route 0.0.0.0/0 and a corporate prefix 192.168.100.0/24 from physical router 1 and physical router 2 respectively. If "External 1" interface on Tier-0 fails and the traffic from compute workloads destined to WAN lands on Tier-0 SR on EN1, this traffic can follow the default route (0.0.0.0/0) learned via IBGP from Tier-0 SR on EN2.Traffic is being sent to EN2 through the Geneve overlay. After a route lookup on Tier-0 SR on EN2, this N-S traffic can be sent to physical router 1 using "External interface 3".

Graceful Restart and BFD Interaction with Active/Active – Tier-0 SR Only

If an Edge node is connected to a TOR switch that does not have the dual supervisor or the ability to retain forwarding traffic when the control plane is

restarting, enabling GR does not make sense. There is no value in preserving the forwarding table on either end or sending traffic to the failed or restarting device. In case of an active SR failure (i.e., the Edge node goes down), physical router failure, or path failure, forwarding will continue using another active SR or another TOR. BFD should be enabled with the physical routers for faster failure detection.

It is recommended to enable GR If the Edge node is connected to a single dual supervisor system that supports forwarding traffic when the control plane is restarting. This will ensure that forwarding table data is preserved and forwarding will continue through the restarting supervisor or control plane. Enabling BFD with such a system would depend on the device-specific BFD implementation. If the BFD session goes down during supervisor failover, then BFD should not be enabled with this system. If the BFD implementation is distributed such that the BFD session would not go down in case of supervisor or control plane failure, then enable BFD as well as GR.

Suppose the edge node connects to multiple dual supervisor systems. In that case, the network architect will have to choose what type of failover mechanism to prioritize between graceful restart and traffic rerouting over a different link.

4.5.2 Active/Standby

Active/Standby – This is a high availability mode where only one SR act as an active forwarder. This mode is required when stateful services are enabled. Services like NAT are in constant state of sync between active and standby SRs on the Edge nodes. This mode is supported on both Tier-1 and Tier-0 SRs. Preemptive and Non-Preemptive modes are available for both Tier-0 and Tier-1 SRs. Default mode for gateways configured in active/standby high availability configuration is non-preemptive.

A user can select the preferred member (Edge node) when a gateway is configured in active/standby preemptive mode. When enabled, preemptive behavior allows a SR to resume active role on preferred edge node as soon as it recovers from a failure.

For Tier-1 Gateway, active/standby SRs have the same IP addresses northbound. Only the active SR will reply to ARP requests, while the standby SR interfaces operational state is set as down so that they will automatically drop packets.

For Tier-0 Gateway, active/standby SRs have different IP addresses northbound and both have BGP sessions or OSPF adjacencies established on their uplinks. Both Tier-0 SRs (active and standby) receive routing updates from physical routers and advertise routes to the physical routers; however, the standby Tier-0 SR prepends its local AS three times in case of BGP or advertise routes with a higher metric in case of OSPF so that traffic from the physical routers prefer the active Tier-0 SR.

Southbound IP addresses on active and standby Tier-0 SRs are the same and the operational state of standby SR southbound interface is down. Since the operational state of southbound Tier-0 SR interface is down, the Tier-0 DR does not send any traffic to the standby SR. FIGURE 4-38 shows active and standby Tier-0 SRs on Edge nodes “EN1” and “EN2”.

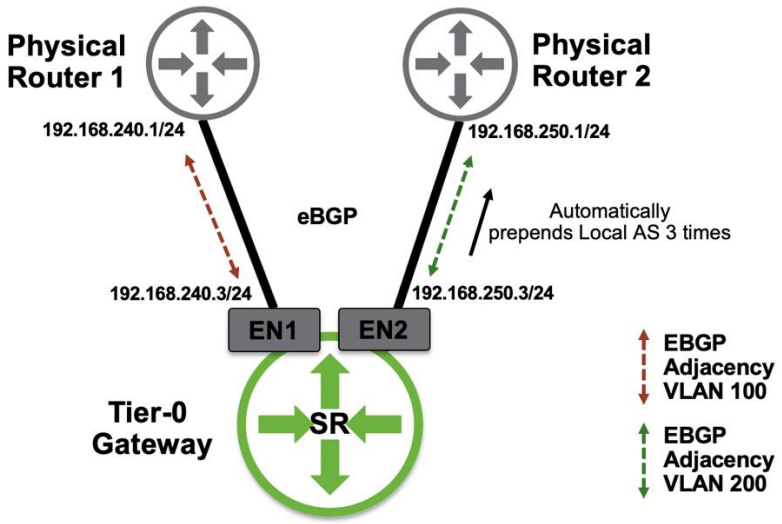


Figure 4-38: Active and Standby Routing Control with eBGP

The placement of active and standby SR in terms of connectivity to TOR or northbound infrastructure becomes an important design choice, such that any component failure should not result in a failure of both active and standby service. Diversity of connectivity to TOR for bare metal edge nodes and host-specific availability consideration for hosts where Edge node VMs are hosted, becomes an important design choice. These choices are described in the section EDGE NODE AND SERVICES DESIGN in chapter 7.

4.5.2.1 Graceful Restart and BFD Interaction with Active/Standby

Active/standby services have an active/active control plane with active/standby data forwarding. In this redundancy model, BGP or OSPF is established on active and standby Tier-0s SR with their respective TORs. If the Edge node is connected to a system that does not have the dual supervisor or the ability to keep forwarding traffic when the control plane is restarting, enabling GR does not make sense. There is no value in preserving the forwarding table on either end as well as no point sending traffic to the failed or restarting device. When the active Tier-0 SR goes down, the route advertised from standby Tier-0 becomes the best route and forwarding continues using the newly active SR. If the TOR switch supports BFD, it is recommended to run BFD on the both eBGP neighbors for faster failure detection.

It is recommended to enable GR If the Edge node is connected to a single dual supervisor system that supports forwarding traffic when the control plane is restarting. This will ensure that the forwarding table is table is preserved and forwarding will continue through the restarting supervisor or control plane. Enabling BFD with such system depends on BFD implementation of hardware vendor. If the BFD session goes down during supervisor failover, then BFD should not be enabled with this system; however, if the BFD implementation is distributed such that that the BFD session would not go down in case of supervisor or control plane failure, then enable BFD as well as GR.

Suppose the edge node connects to multiple dual supervisor systems. In that case, the network architect will have to choose what type of failover mechanism to prioritize between graceful restart and traffic rerouting over a different link.

4.5.3 High availability failover triggers

Two types of edge failover events exist. The first type works at the edge node level and causes all the SRs running on the node to fail and the SRs running on a peer edge node to take over. Peer edge nodes are defined based on each Tier-1 or Tier-1 SR. Two edge nodes are considered peers for a service if that service runs in active/standby or active/active across those edge nodes. For each active/standby gateway, we have two peer edge nodes. For an active/active Tier-0 gateway, we can have up to 8 edge peer nodes. The peer node relationship is not exclusive. Edge node one can have edge node two as a peer for a Tier-0 Gateway but have edge node three as a peer for a Tier-1 gateway (see FIGURE 4-39). The second type of failover event affects an individual service or SR. Those are recovered on the peer edge node, while others may still be running on the original node.

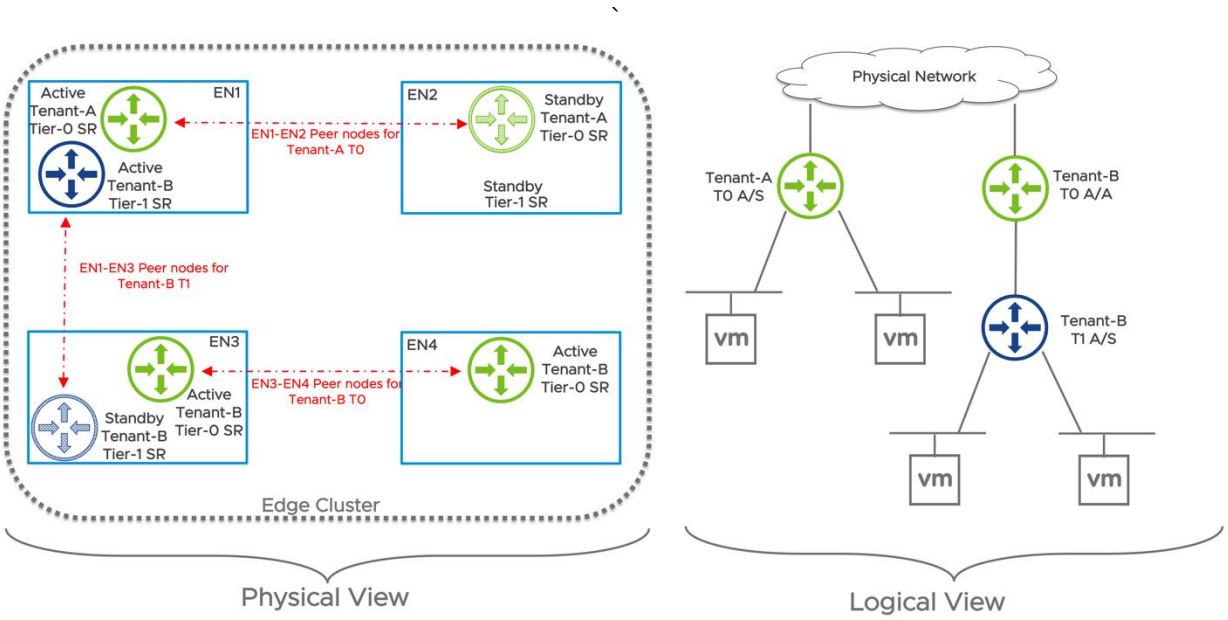


Figure 4-39: Edge HA - Peer edge node definition

Failover events that belong to the first category are (all services and SRs will failover to the corresponding peer edge node):

1. Dataplane service is not running.
2. All TEP interfaces are down. This condition only applies to bare metal edge nodes as the interface of an edge node VM should never go “physically” down. If a bare metal edge node has more than one TEP interface, all need to be down for this condition to take effect.
3. The edge node enters maintenance mode
4. Edge nodes run BFD with compute hosts and other edge nodes to monitor the health of the overlay tunnels. If host transport nodes are present, when all the overlay tunnels are down to both remote Edges and compute

hypervisors, all the SRs on the edge will be declared down. This condition does not apply if no host transport nodes are present.

5. Edge nodes in an Edge cluster exchange BFD keepalives on two interfaces, management and TEP interfaces. Each edge monitors the status of its peer edges on those two interfaces, if both are down, it will consider the peer down, and it will make the corresponding SR active.

Failover event that belongs to the second category are (Only a specific SR is declared down and failover to the peer edge):

1. Northbound routing on a Tier-0 SR is down. This is applicable to BGP, and OSPF with or without BFD enabled and to static routes with BFD enabled. When this situation occurs, the Tier-0 SR only will be declared failed. Other Tier-1 SRs on the same edge will still be active. If the Tier-0 is configured with static routes with no BFD, northbound routing will never be considered down.
2. Services are configured on SR. The health score of services on the SR is less than that on the peer SR

We will now review in more detail the failover triggers that requires a more careful consideration from a design perspective.

FIGURE 4-40 outlines condition 5. The BFD sessions between two edge nodes are lost on both the management and the overlay network. Standby Tier-0 and Tier-1 SRs will become active. This condition addresses the scenario when an edge node fails or is completely isolated. When designing edge node connectivity, it is important to consider corner-case situations when this condition may apply, but the edge node is not down or completely isolated. For example, uplink connectivity could be up while the overly and management network could be down if the uplink traffic used dedicated pNICs. Designing for fate sharing between uplink, management, and overlay traffic will mitigate the risk of a dual-active scenario. If dedicated pNICs are part of the design, providing adequate pNIC redundancy to management and overlay traffic will ensure that condition five is only triggered because of a complete failure of an edge.

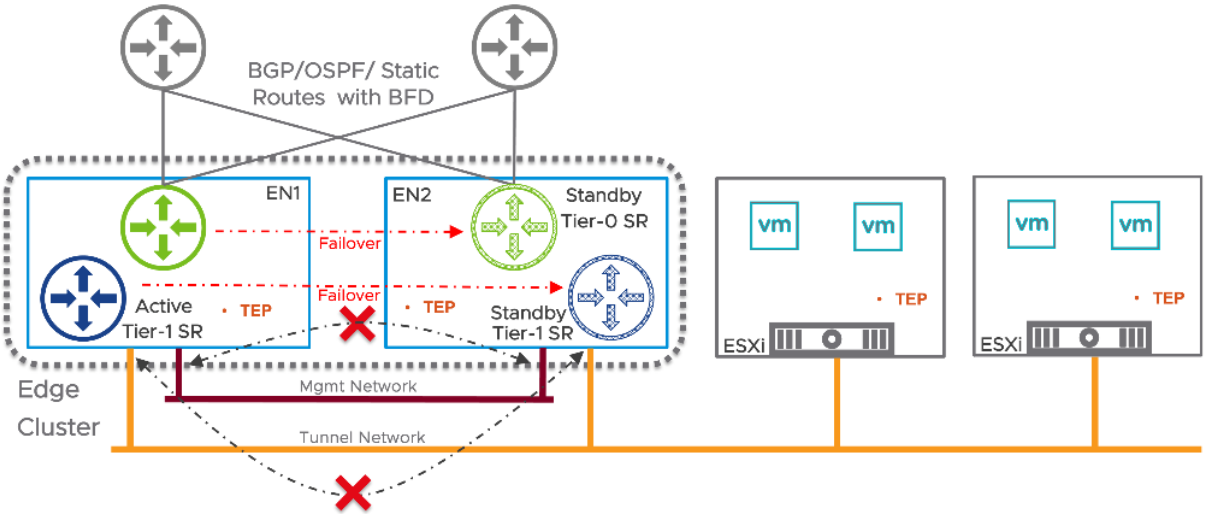


Figure 4-40: Failover triggers – BFD on management and overlay network down

FIGURE 4-41 below outlines condition 6. A northbound connectivity issue is detected by the dynamic routing protocol timers or BFD. In this case, the affected Tier-0 only will undergo a failover event. Any other Tier-1 SR on the same edge will remain active. Static routes without BFD do not allow to detect a failure in the uplink network. For this reason, configurations including static routes should always include BFD.

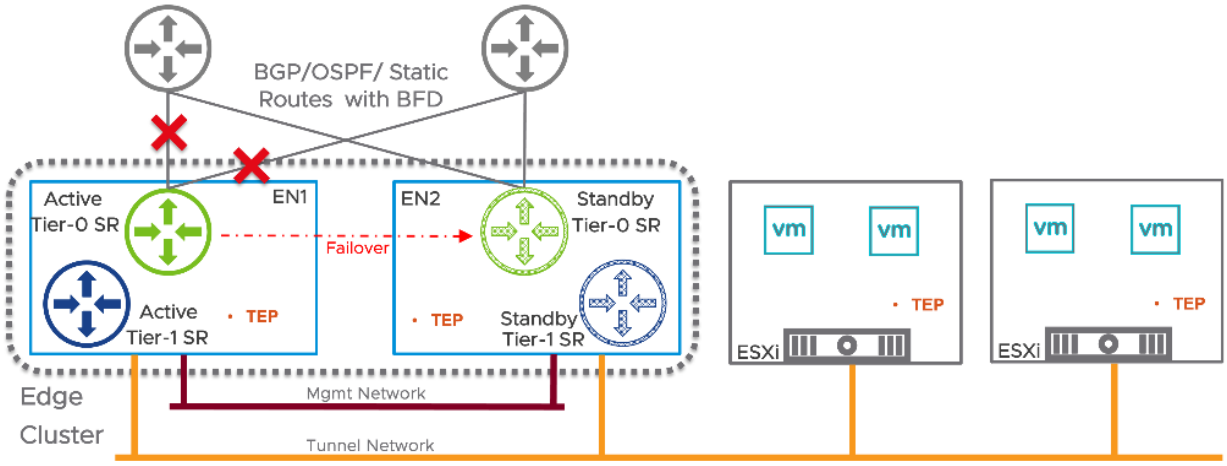


Figure 4-41: Failover triggers – northbound routing down

FIGURE 4-42 below outlines condition 4. In this case, edge node one did not fail, but it is not able to communicate to any other transport node on the overlay network. This condition mitigates situations when an edge node does not fail, but connectivity issues exist. All the SRs on the affected edge node, including all Tier-0 and Tier-1 SRs, will failover. This condition only applies to deployments including host transport nodes, which can help discriminate between the failure of the peer edge and a network connectivity problem to it.

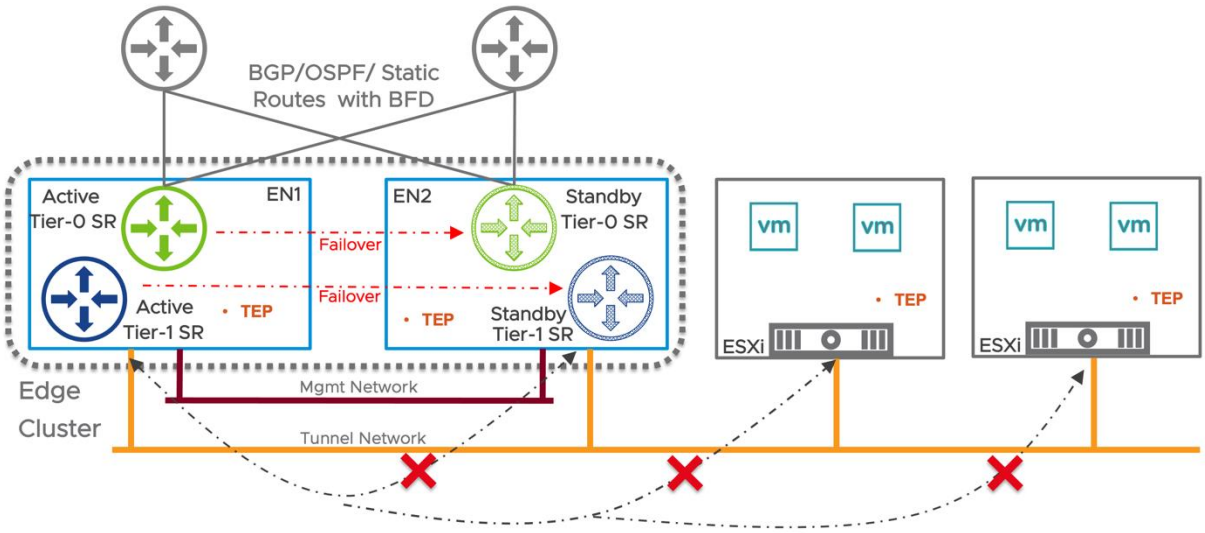


Figure 4-42: Failover triggers – All tunnels down (with host transport nodes)

FIGURE 4-43 outlines condition four again, but this time no host transport node is present in the topology. The all overlay tunnel down condition does not put the SRs running on edge node 2 in a failed state. In the scenario depicted in the diagram, edge node one failed, and the SRs on edge two took over even if all the overlay tunnels on edge 2 were down. If the all tunnel condition applied to deployment without host transport nodes, the SRs running on edge node two would go into a fail state, blackholing the traffic. This topology applies to VLAN-only deployments, where workloads are connected via service interfaces.

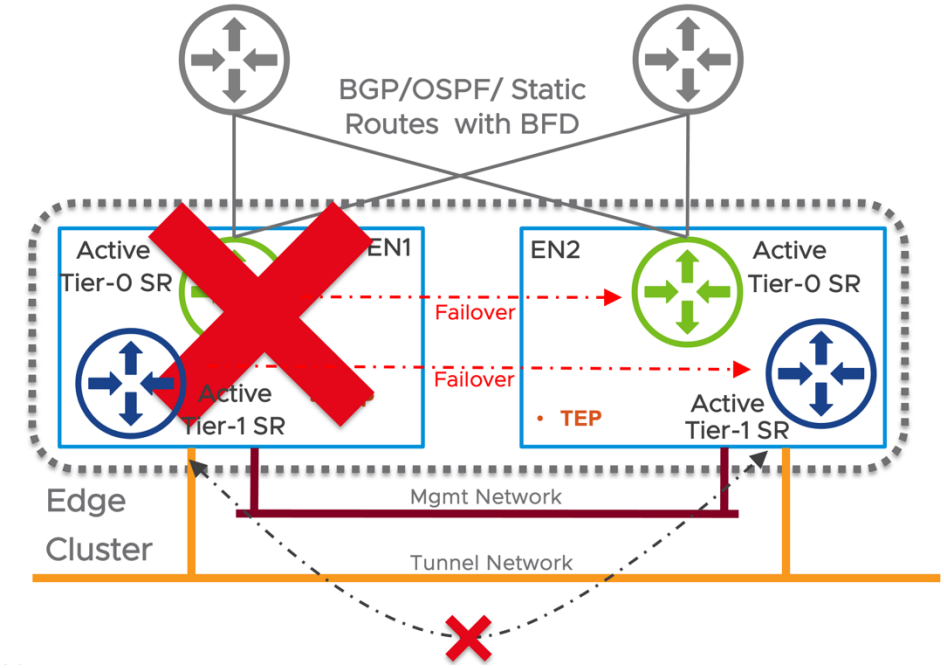


Figure 4-43: Failover triggers – All tunnels down (without host transport nodes)

We should pay special attention to the implications of deploying multiple edge node VMs on the same host and/or on hosts with TEP interfaces on the same network (possible starting NSX version 3.1) on the "all tunnel down condition" failover trigger. A failure of the upstream overlay transport network (pNIC or switch level failure) will not bring down the tunnels local to the host, those between edge VMs on the same host, or to the host TEP (See FIGURE 4-44). This situation may lead the edge node VMs to not trigger a failover if northbound routing connectivity is up. Traffic that the upstream network forward to the affected edge node VMs will be blackholed because of the lack of access to the overlay network.

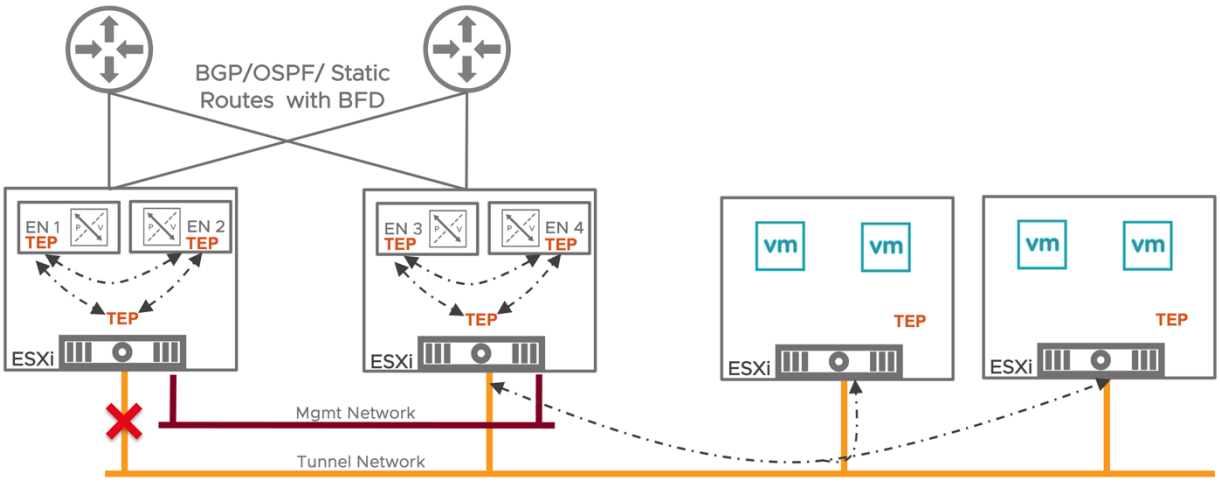


Figure 4-44: Failover Triggers - All tunnels Down - Tunnels local to ESXi host

This condition can be avoided by designing for fate sharing between the overlay and upstream network. If that is not a possibility based on the design requirements, we should eliminate the possibility of establishing overlay tunnels within a host with one of these options:

- No more than one edge node VM per host on hosts not part of the same overlay TZ
- No more than one edge node VM per host on hosts part of the same overlay TZ but edge and host overlay VLAN and subnet are different (BFD keepalives will need to be forwarded to the physical network)
- With more than one edge node VM per host, edge node VMs should have different overlay VLAN and subnet (BFD keepalives will need to be forwarded to the physical network)

4.6 VRF Lite

4.6.1 VRF Lite Generalities

Virtual Routing Forwarding (VRF) is a virtualization method that consists of creating multiple logical routing instances within a physical routing appliance. VRF instances are commonly used in enterprise and service providers networks to provide control and data plane isolation, allowing several use cases such as overlapping IP addressing between tenants, isolation of regulated workload, isolation of external and internal workload as well as hardware resources consolidation. With NSX, it is possible to extent the VRF present on the physical network onto the NSX domain. Creating a development environment that replicates the production environment is a typical use case for VRF.

Another representative use case for VRF is when multiple environments need to be isolated from each other. As stated previously, VRF instances are isolated between each other by default; allowing communications between these environments using the Route Leaking VRF feature is possible. While this feature allows inter-VRF communications, it is important to emphasize that scalability can become an issue if a design permits all VRF to communicate between each other.

In this case, VRF might not be the option. VRF should not be replaced in lieu of the DFW construct.

FIGURE 4-45 pictures a traditional VRF architecture. Logical (Switch Virtual Interface – Vlan interface) or Physical interface should be dedicated to a single VRF instance. In the following diagram, interface e1/1 and e2/2 belong to VRF-A while interface e1/2 and e2/2 belong to VRF-B. Each VRF will run their own dynamic routing protocol (or use static routes).

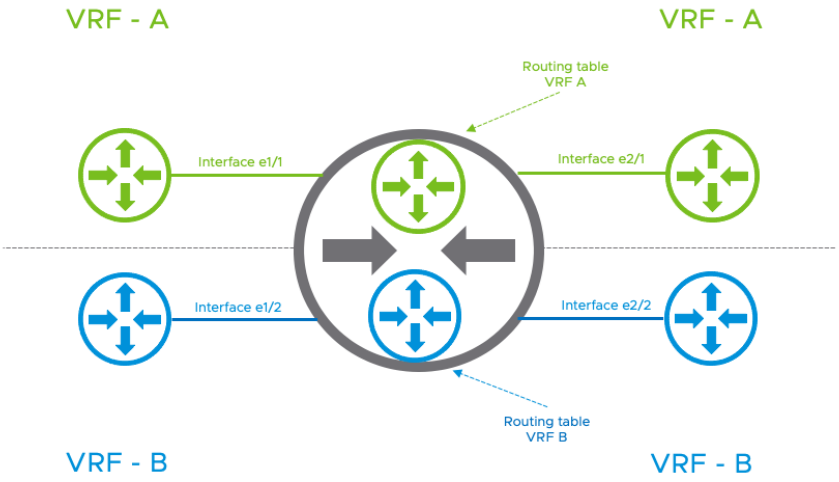


Figure 4-45: Networking VRF architecture

With NSX 2.x, several multi-tenant designs are possible.

The first option was to deploy a Tier-1 gateway for each tenant while a shared Tier-0 provides connectivity to the physical networking fabric for all tenants. In this configuration the Tier-0 gateway has a single routing table for all the tenants, requiring unique IP addresses for each tenant or NAT implemented at the tenant Tier-1 gateway level when overlapping IPs exist.

FIGURE 4-46 diagrams that option.

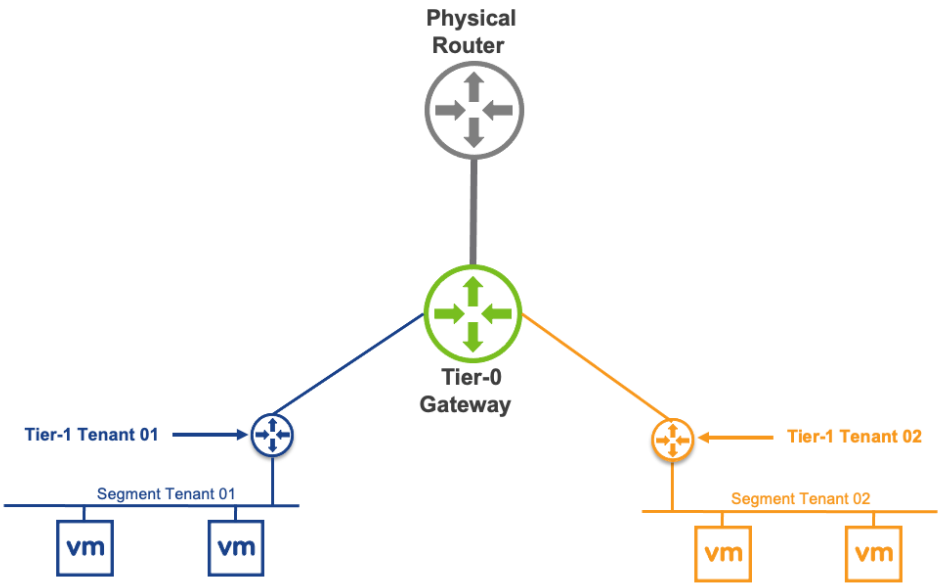


Figure 4-46: NSX 2.x multi-tenant architecture – Shared Tier-0 Gateway

Another supported design is to deploy a separate Tier-0 gateway for each tenant on a dedicated tenant edge node. This configuration allows for duplicated IP addresses between tenants, but requires dedicated edge nodes per tenant (no more than a single Tier-0 SR can run on each edge node). While providing the required separation, this solution may be limited from a scalability perspective.

FIGURE 4-47 shows a traditional multi-tenant architecture using dedicated Tier-0 per tenant in NSX 2.X .

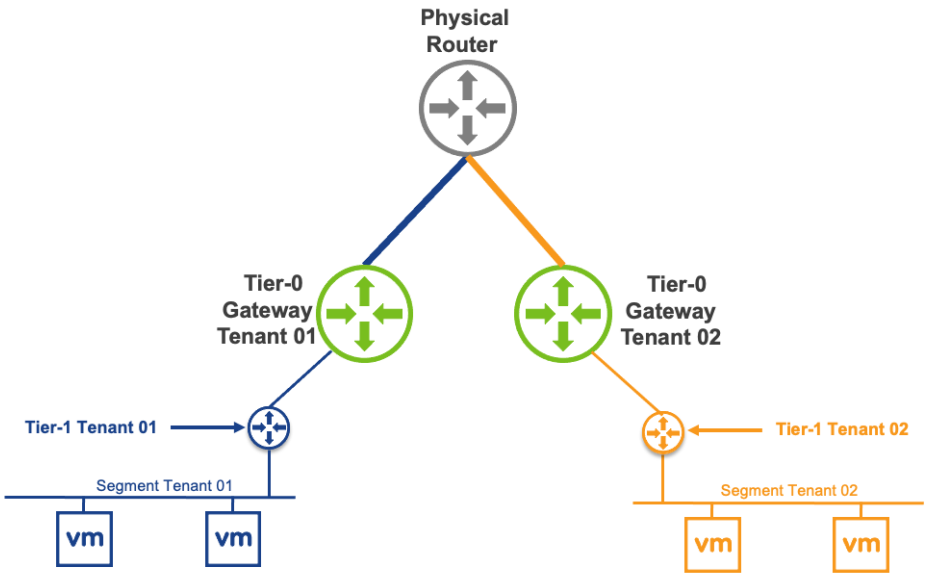


Figure 4-47: NSX 2.x multi-tenant architecture. Dedicated Tier0 for each tenant

In traditional networking, VRF instances are hosted on a physical appliance and share the resources with the global routing table. Starting with NSX 3.0, Virtual Routing and Forwarding (VRF) instances configured on the physical fabric can be extended to the NSX domain. A VRF Tier-0 gateway must be associated to a traditional Tier-0 gateway identified as the "Parent Tier-0". FIGURE 4-48 diagrams an edge node hosting a traditional Tier-0 gateway with two VRF gateways. Control plane is completely isolated between all the Tier-0 gateways instances.

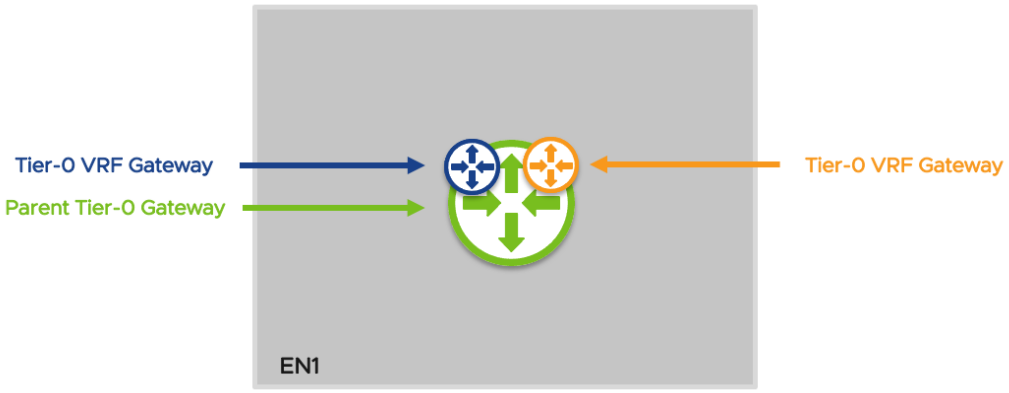


Figure 4-48: Tier-0 VRF Gateways hosted on a Parent Tier-0 Gateway

The parent Tier-0 gateway can be considered as the global routing table and must have connectivity to the physical fabric. A unique Tier-0 gateway instance (DR and SR) will be created and dedicated to a VRF. FIGURE 4-49 shows a detailed representation of the Tier-0 VRF gateway with their respective Service Router and Distributed Router components.

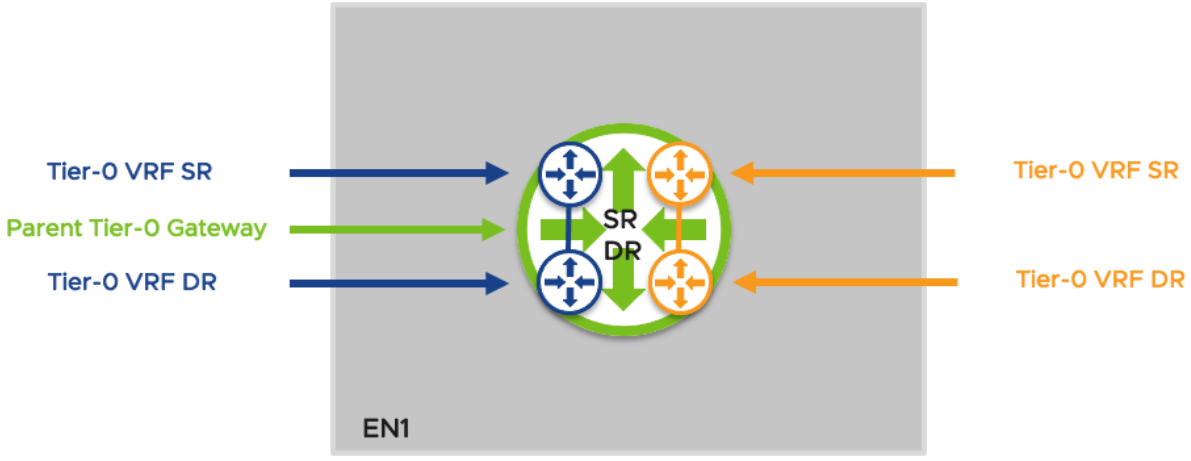


Figure 4-49: Detailed representation of the SR/DR component for Tier-0 VRF hosted on an edge node

FIGURE 4-50 shows a typical single tier routing architecture with two Tier-0 VRF gateways connected to their parent Tier-0 gateway. Traditional segments are connected to a Tier-0 VRF gateway.

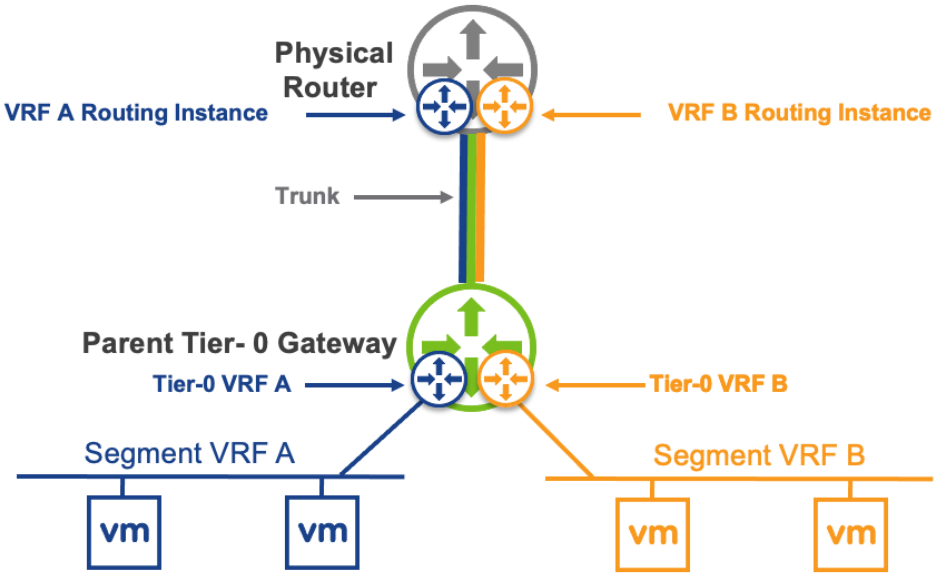


Figure 4-50: NSX 3.0 multi-tenant architecture. Dedicated Tier-0 VRF Instance for each VRF

Since control plane is isolated between Tier-0 VRF instances and the parent Tier-0 gateway, each Tier-0 VRF needs their own network and routing constructs:

- Segments
- Uplink Interfaces
- BGP configuration with dedicated peers or Static route configuration

NSX 3.0 supports BGP and static routes for the Tier-0 VRF gateway. It offers the flexibility to use static routes on a particular Tier-0 VRF while another Tier-0 VRF would use BGP. The OSPF routing protocol is not supported for VRF topologies.

FIGURE 4-51 shows a topology with two Tier-0 VRF instances and their respective BGP peers on the physical networking fabric. It is important to emphasize that the Parent Tier-0 gateway has a BGP peering adjacency with the physical routers using their respective global routing table and BGP process.

From a data plane standpoint, 802.1q VLAN tags are used to differentiate traffic between the VRFs instances as demonstrated in the following figure.

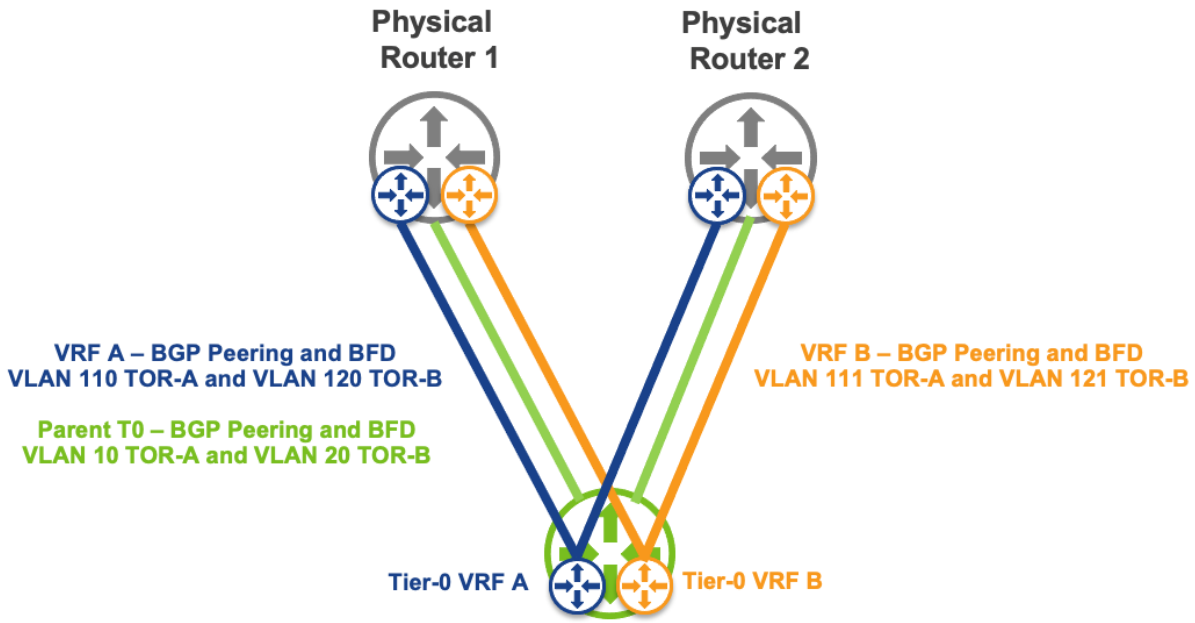


Figure 4-51: BGP peering Tier-0 VRF gateways and VRF on the networking fabric

When a Tier-0 VRF is attached to parent Tier-0, multiple parameters will be inherited by design and cannot be changed:

- Edge Cluster
- High Availability mode (Active/Active – Active/Standby)
- BGP Local AS Number
- Internal Transit Subnet
- Tier-0, Tier-1 Transit Subnet.

All other configuration parameters can be independently managed:

- External Interface IP addresses
- BGP neighbor
- Prefix list, route-map, Redistribution
- Firewall rules
- NAT rules

4.6.2 VRF Lite HA

As mentioned previously, The Tier-0 VRF is associated to a Parent Tier-0 and will follow the high availability mode and state of its Parent Tier-0.

Both Active/Active or Active/Standby high availability mode are supported on the Tier-0 VRF gateways. It is not possible to have an Active/Active Tier-0 VRF associated to an Active/Standby Parent Tier-0 and vice-versa.

In a traditional Active/Standby design, a Tier-0 gateway failover can be triggered if all northbound BGP peers are unreachable. Similar to the high availability construct between the Tier-0 VRF and the Parent Tier-0, the BGP peering design must match between the VRF Tier-0 and the Parent Tier-0.

Inter-SR routing is not supported in Active/Active Tier-0 VRF topologies. FIGURE 4-52 represents a BGP instance from both parent Tier-0 and Tier-0 VRF point of view. This topology is supported as each Tier-0 SR (on the parent and on the VRF itself) have a redundant path towards the network infrastructure. The Top of Rack switches in this case are advertising the same prefixes towards the NSX Tier-0.

Both the Parent Tier-0 gateway and the Tier-0 VRF gateway are peering with the same physical networking device. The Parent Tier-0 gateways establish BGP adjacencies with both top of rack switches in the global routing table while the Tier-0 VRF gateways establish BGP adjacencies with both top of rack switch on another BGP process dedicated to the VRF. The Tier-0 VRF leverages physical redundancy towards the networking fabric if one of its northbound link fails.

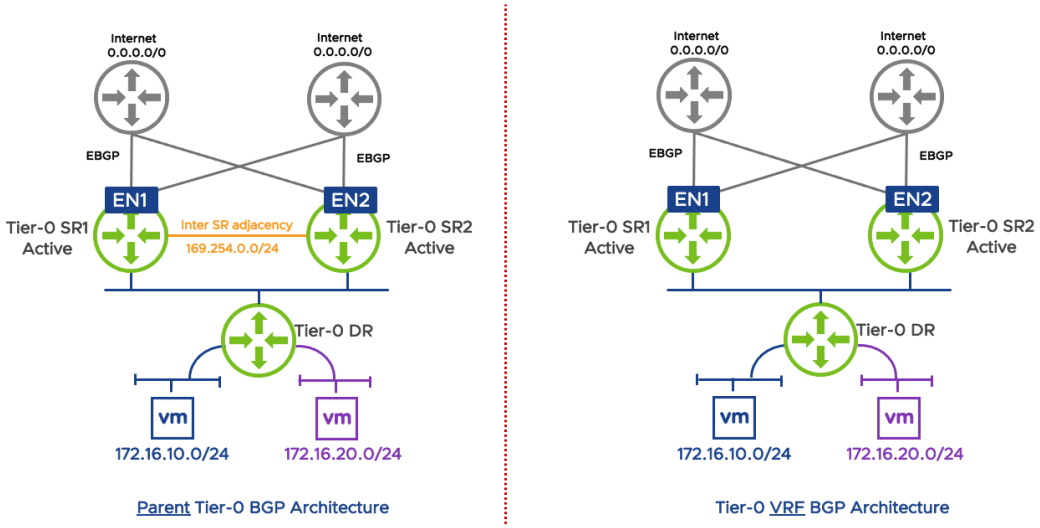


Figure 4-52: Supported BGP peering Design

FIGURE 4-53 represents an unsupported VRF Active/Active design where different routes are learned from different physical routers. Both the Parent Tier-0 and VRF Tier-0 gateways are learning their default route from a single physical router. At the same time they receive specific routes from a different single BGP peer. This kind of scenario would be supported for traditional Tier-0 architecture as Inter-SR would provide a redundant path to the networking fabric, but it is not supported for VRFs because of the lack of inter-SR peering capability. This VRF architecture is not supported in NSX 3.2.

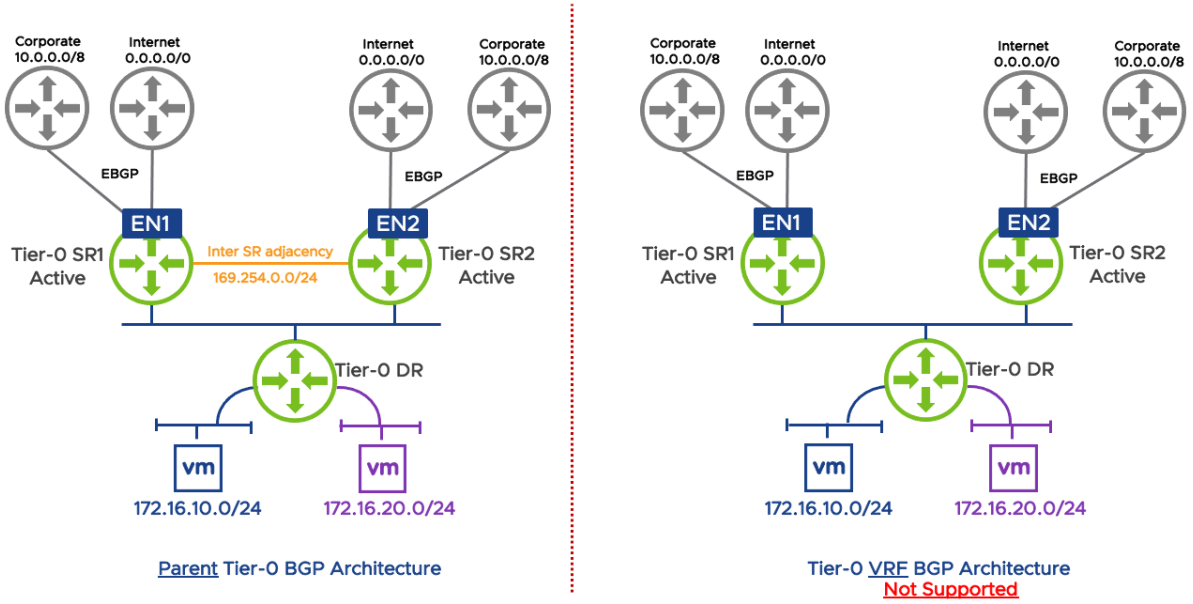


Figure 4-53: Unsupported Active/Active Topology with VRF

FIGURE 4-54 demonstrates the traffic being as one internet router fails and Tier-0 VRF gateways can't leverage another redundant path to reach the destination. Since the Parent Tier-0 gateway has an established BGP peering adjacency (VRFs HA state is inherited from the parent), failover will not be triggered, and traffic will be blackholed on the Tier-0 VRF. This situation would not arise if each VRF SR peered with every ToR.

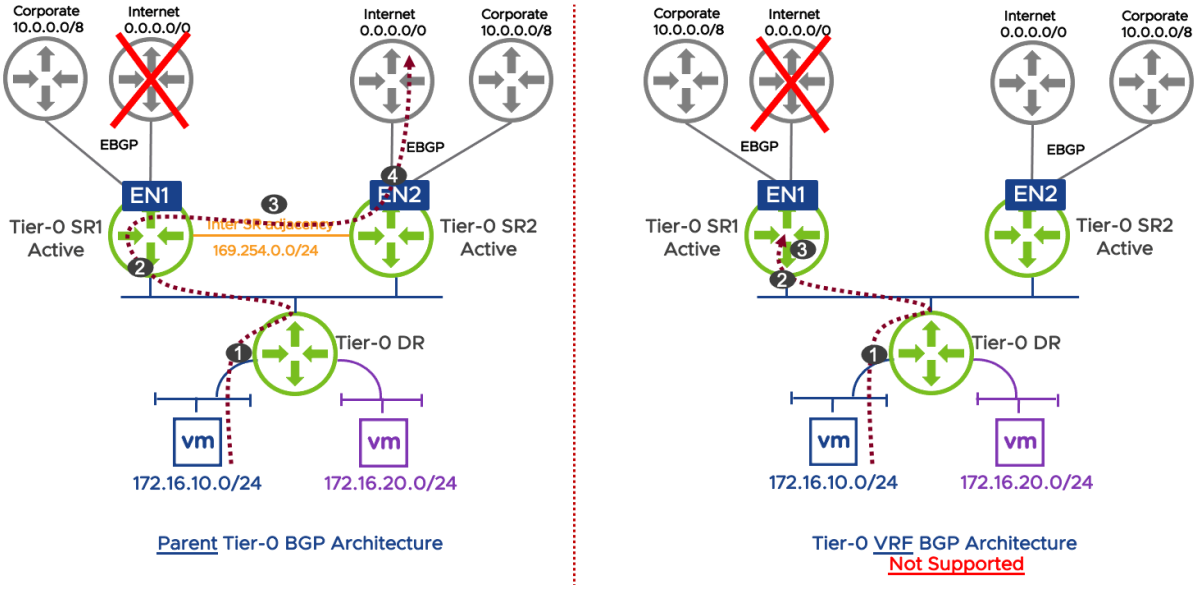


Figure 4-54: Unsupported Active-Active Topology with VRF – Failure

On the Parent Tier-0:

1. VM "172.16.10.0" sends its IP traffic towards the internet through the Tier-0 DR.
2. Since the Tier-0 topology is Active/Active, the Tier-0 DR can send the traffic to either Tier-0 SR1 and Tier-0 SR2 using a 5 tuple hashing algorithm.
3. From a Tier-0 SR1 point of view, the traffic that needs to be routed towards the internet will be sent towards Tier-0 SR2 as there is an inter-SR BGP adjacency and that Tier-0 SR2 learns the route from another internet switch.
4. Traffic is received by the Tier-0 SR2 and routed towards the physical fabric.

On the Tier-0 VRF:

1. VM "172.16.10.0" sends its IP traffic towards the internet through the Tier-0 DR.
2. Since the Tier-0 topology is Active/Active, the Tier-0 DR can send the traffic to either Tier-0 SR1 and Tier-0 SR2 using a 5 tuple hashing algorithm.
3. From a Tier-0 SR1 point of view, the traffic is blackholed as there is no inter-SR BGP adjacency between the Tier-0 SRs VRF.

Following the same BGP peering design and principle for Active/Standby topologies is also mandatory for VRF architectures as the Tier-0 VRF will inherit the behavior of the parent Tier-0 gateway.

FIGURE 4-55 represents another unsupported design with VRF architecture.

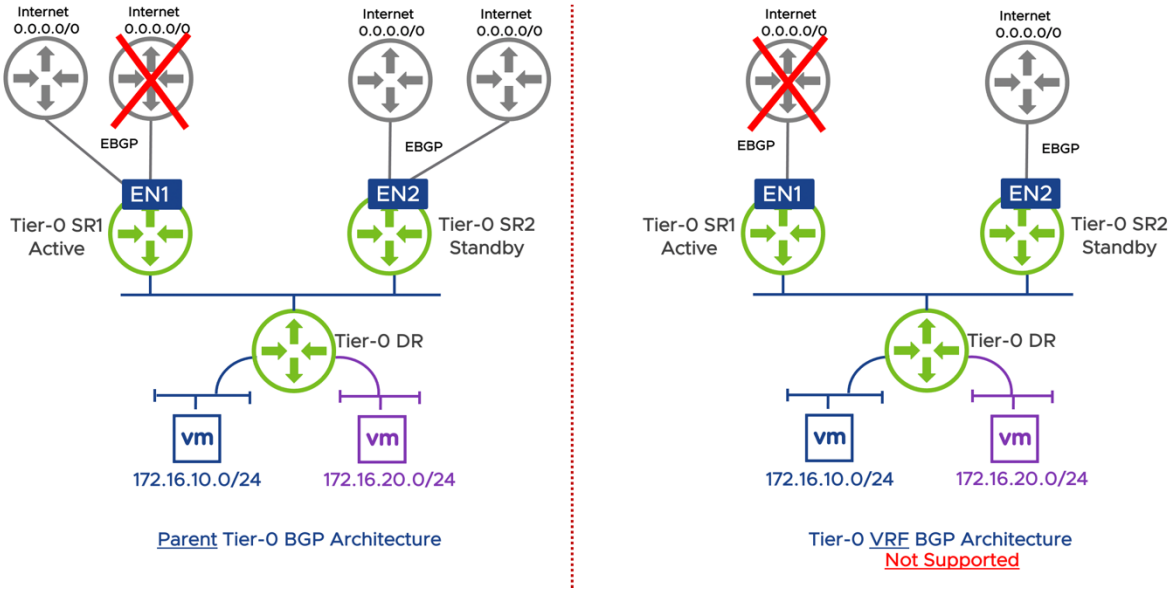


Figure 4-55: Unsupported design BGP architecture Different peering with the networking fabric

In this design, traffic will be blackholed on the Tier-0 VRF SR1 as the internet router fails. Since the Tier-0 VRF share its high availability running mode with the Parent Tier-0, it is important to note that the Tier-0 SR1 will not failover to the

Tier-0 SR2. The reason behind this behavior is because a failover is triggered only if all northbound BGP sessions change to the "down" state on the parent Tier-0 SR.

Since the parent Tier-0 gateway has an active BGP peering with a northbound router on the physical networking fabric, failover will not occur and traffic will be blackholed on the VRF that have only one BGP peer active. This situation would not arise if each VRF SR peered with every ToR.

4.6.3 Connecting Tier-1 gateways to VRFs

A Multi-Tier routing topology is supported to add granularity and flexibility to a multi-tenancy data center architecture as depicted in the following figure. Traditional Tier-1 gateway can be connected to a Tier-0 VRF gateway as demonstrated in FIGURE 4-56.

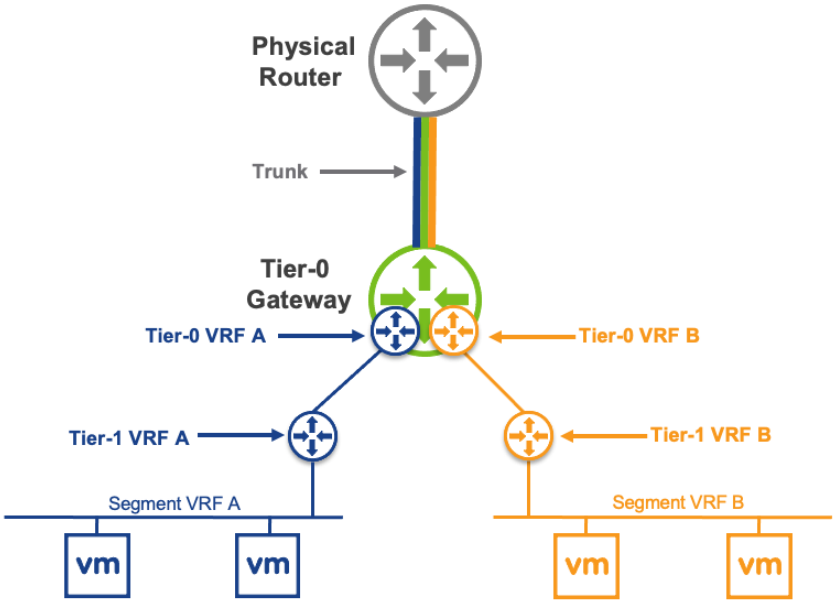


Figure 4-56: Multi-Tier routing architecture and VRF-lite

Stateful services can either run on a Tier-0 VRF gateway or a Tier-1 gateway except for VPN and Load Balancing as these features are not supported on a Tier-0 VRF. Tier-0 SR in charge of the stateful services for a particular VRF will be hosted on the same edge nodes as the Parent Tier-0 Gateway. It is recommended to run the stateful services on a Tier-1 gateways and leverage an Active/Active Tier-0 gateway to send the traffic northbound to the physical fabric.

FIGURE 4-57 represents stateful services running on traditional Tier-1 gateways SR.

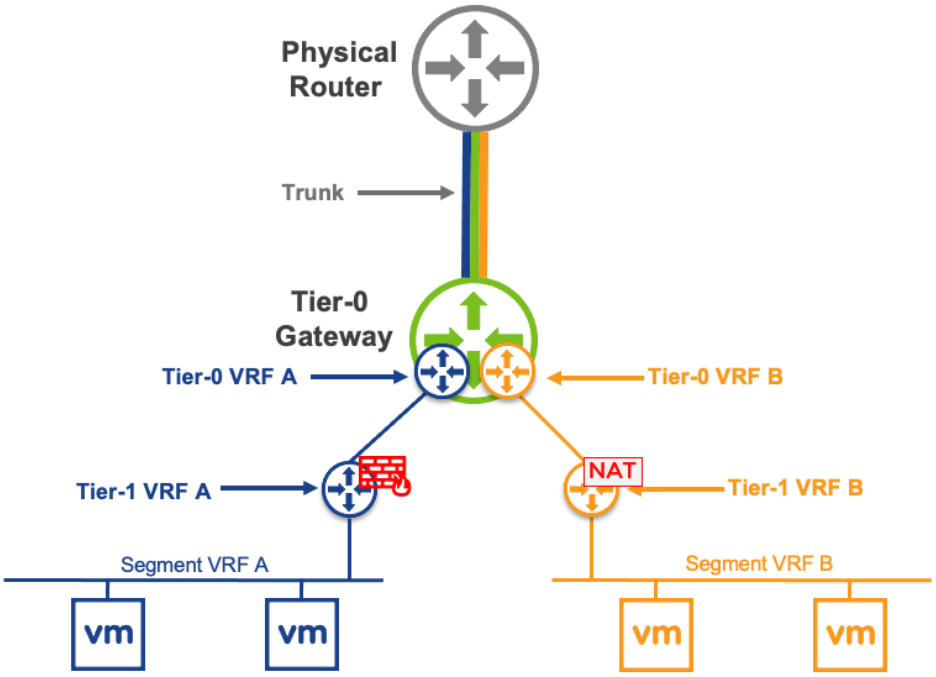


Figure 4-57: Stateful Services supported on Tier-1

4.6.4 VRF route leaking

By default, data-plane traffic between VRF instances is isolated in NSX. By configuring VRF Route Leaking, traffic can be exchanged between VRF instances. As a result, static routes must be configured on the Tier-0 VRF instances to allow traffic to be exchanged. The next hop for these static routes must not be pointing to an interface that belongs to a Tier-0 gateway (VRF or Parent). As a result, multi-tier routing architecture must be implemented to allow traffic to be exchanged between the VRF instances.

Figure 4-29 demonstrates a supported topology for VRF route leaking

Two static routes are necessary:

- Static Route on Tier-0 VRF A
  - Destination Subnet: 172.16.20.0/24
  - Next Hop
    - IP Address of Tier-1 DR in VRF B (e.g 100.64.80.3)
    - Admin Distance: 1
    - Scope: VRF-B
- Static Route on Tier-0 VRF B
  - Destination Subnet: 172.16.10.0/24
  - Next Hop
    - IP Address of Tier-1 DR in VRF A (e.g 100.64.80.1)
    - Admin Distance: 1
    - Scope: VRF-A

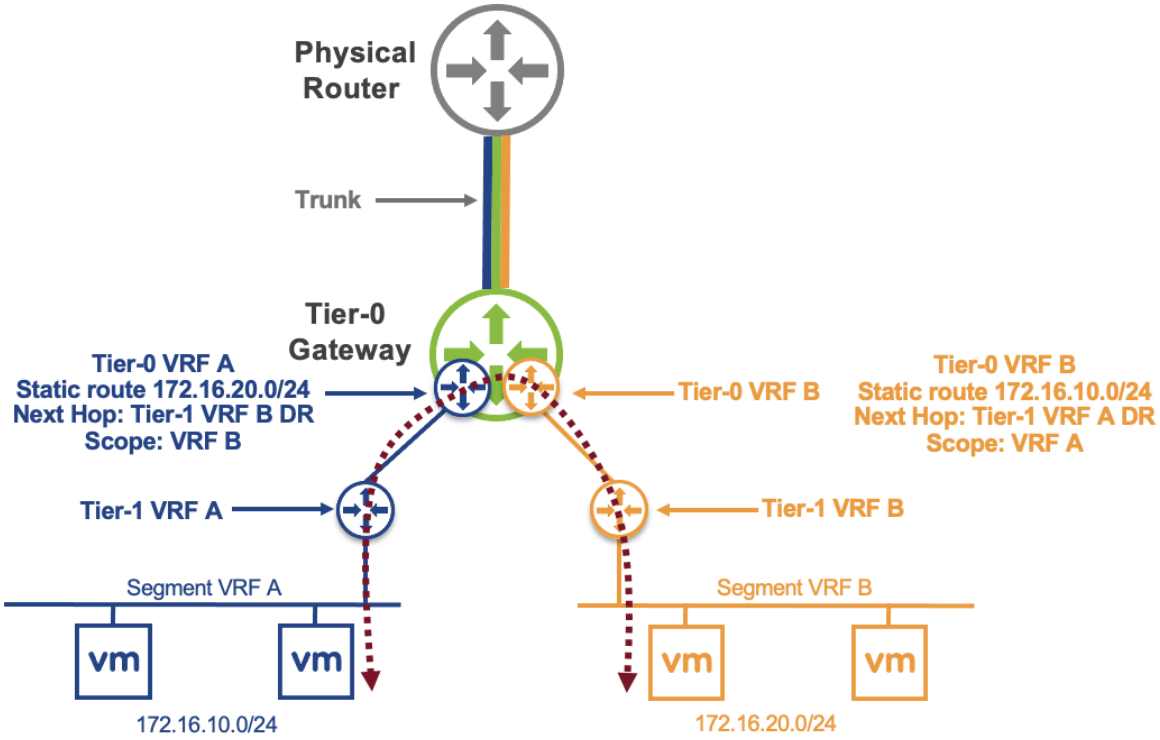


Figure 4-29 VRF Route leaking with static routes.

VRF-lite also supports northbound VRF route leaking as traffic can be exchanged between a virtual workload on an VRF overlay segment and a bare metal server hosted in a different VRF on the physical networking fabric.

NSX VRF route leaking requires that the next hop for the static route must not be pointing to an interface that belongs to a Tier-0 gateway (VRF or Parent). Static routes pointing to the directly connected IP addresses uplink on the ToR switches would not be a recommended design as the static route would fail if an outage would occur on that link or neighbor (Multiple static routes would be needed for redundancy).

A loopback or virtual IP address host route (/32) can be advertised in the network in the destination VRF. Since the host route is advertised by both top of rack switches, two ECMP routes will be installed in the Tier-0 VRF. The loopback or virtual IP address host route (/32) does not need to be advertised in the source VRF.

FIGURE 4-58 demonstrates the design and Tier-0 VRF gateways will use all available healthy paths to the networking fabric to reach the server in VRF-B.

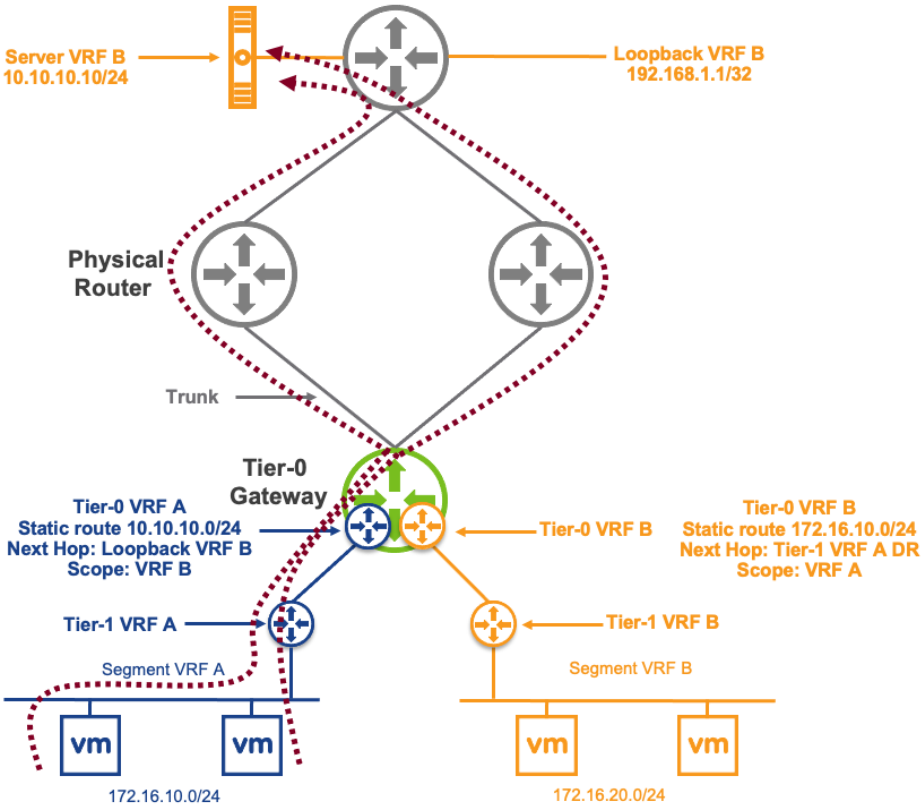


Figure 4-58: VRF leaking traffic with northbound destination

Two static routes are necessary:

- Static Route on Tier-0 VRF A
  - Destination Subnet: 10.10.10.0/24
  - Next Hop
    - 192.168.1.1 (Loopback interface)
    - Admin Distance: 1
    - Scope: VRF-B
- Static Route on Tier-0 VRF B
  - Destination Subnet: 172.16.10.0/24
  - Next Hop
    - IP Address of Tier-1 DR in VRF A (e.g 100.64.80.1)
    - Admin Distance: 1
    - Scope: VRF-A

In case of a physical router outage, the next hop for the static route on the Tier-0 VRF A can still be reached using a different healthy BGP peer advertising that host route.

4.6.5 VRFs shared Internet access over a common VLAN

In some situations, different VRFs instances require access to the Internet via a shared connection. This configuration can be accomplished on the physical network via VRF route-leaking and VRF aware NAT.

If the intention is to minimize the physical network configuration and simply provide a common VLAN where all the NSX VRFs can access the Internet, this configuration can be accomplished by deploying a shared T0 Gateway dedicated to Internet access as demonstrated in the following figure (FIGURE 4-59)

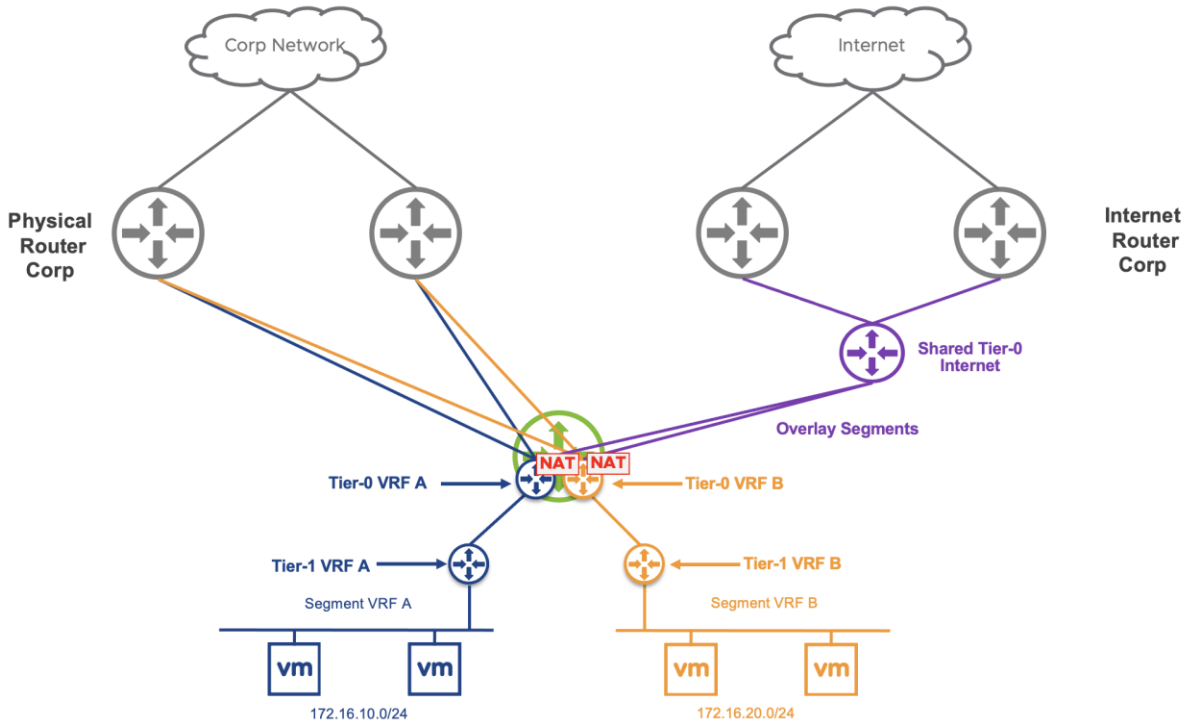


Figure 4-59: Different VRFs sharing Internet access via common Internet Tier-0

Directly connecting the different VRFs to a common VLAN is not allowed in NSX 3.2.

The VRFs can be interconnected to the shared T0 via dedicated overlay segments. The shared Tier-0 gateway is providing internet access using VLAN segments.

Internet facing NAT can implemented on the shared T0 if the VRFs are not carrying overlapping IP spaces, or at the VRF level if they are.

eBGP peering between the VRFs Tier-0 Gateways and the shared Tier-0 gateway is recommended to protect against an edge node failure.

4.7 Multi-Protocol BGP EVPN - VXLAN

In a multi tenancy network architecture based on VRF, each instances needs to establish its own BGP peering with the Top of Rack switches VRF.

Each VRF instances need their own network constructs:

- NSX Segments and VLANs
- Subnets and IP addresses
- BGP Configuration
- BFD timers

With a substantial number of tenants, the number of network objects needed both on the physical network and within NSX can introduce an operational issue.

Using NSX and EVPN VXLAN overcomes that operational issue as all the VRF prefixes are advertised over a single BGP session. BGP is one of the most popular routing protocols in Data Centers as it provides an unmatched interoperability, granularity and flexibility over the learned and advertised Network Layer Reachability Information (NLRI). MP-BGP is an extension of BGP and leverages multiple address families like IPv4/IPv6 unicast, VPNv4/VPNv6 to distribute routing information.

With the advent of the VXLAN popularity in the Data Center, an additional address and sub-address families have been implemented in the MP-BGP protocol to alleviate the native flood and learn behavior of VXLAN making scalability an issue in large data centers. The EVPN NLRI is a new addition to the MP-BGP protocol and is carried in the L2VPN Address family (AFI 25) and Subsequent Address Family EVPN (SAFI 70). The L2VPN/EVPN address family exchange capability has to be exchanged during the initial BGP OPEN message.

FIGURE 4-60 compares the different NSX multi-tenancy architectures. The left end side of the picture represents a native VRF-lite architecture requiring a BGP peering per tenant. The right end side of the picture represents a VRF architecture based on EVPN using a single MP-BGP peering for all tenants.

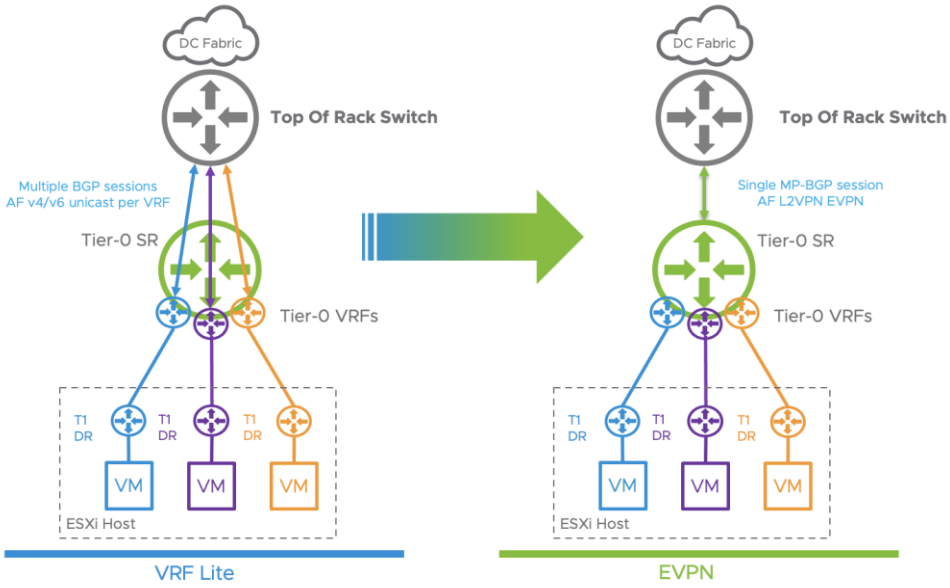


Figure 4-60: VRF Lite vs EVPN peering model

NSX supports 2 EVPN implementation in the data center:

- NSX EVPN Inline Mode
- NSX EVPN Route Server Mode

This design guide will focus on the EVPN inline mode. The EVPN Route server mode will have its own design guide as it is targeted towards telco providers.

4.7.1 MP-BGP EVPN for the Control Plane

As stated previously, MP-BGP uses the L2VPN address family and the EVPN sub-address family. In the NLRI field, different EVPN route types can be used in EVPN to advertises MAC addresses and IP prefixes as demonstrated in the next table. NSX EVPN Inline mode supports EVPN Route Type 5 to advertise VRF prefixes between the Tier-0 gateway and the physical Top of Rack Switch.

Route Type	Description	Purpose	NSX Inline mode
Type 1 (RT-1)	Ethernet Auto-Discovery (A-D) route	Used in Datacenters to support EVPN active-active of multi-homing	No
Type 2 (RT-2)	MAC/IP Advertisement route	Advertises reachability of a specific MAC address, and optionally MAC and IP address binding	No
Type 3 (RT-3)	Inclusive Multicast Ethernet Tag route	Advertises reachability of a VNI associated to a particular VTEP in a virtual network. Type 3 routes are required for BUM traffic delivery across EVPN networks.	No
Type 4 (RT-4)	Ethernet Segment route	Used in the datacenters with multihomed endpoints, and used for Designated Forwarder Election to ensure that only one of the VTEPs forwards BUM traffic	No
Type 5 (RT-5)	IP Prefix Route	Advertises IPv4 and IPv6 prefixes reachability. This advertisement of prefixes into the EVPN domain provides the ability to build L3VPN similar services.	Yes
Type 6 (RT-6)	Selective Multicast Ethernet Tag Route	Used to advertise Host's or VM's intent to receive Multicast traffic for a certain Multicast Group (*,G) or Source-Group combination (S,G)	No

Figure 4-61: EVPN Route Types

This EVPN Route type can advertise either IPv4 or IPv6 prefixes with a variable network mask length (0 to 32 bits for IPv4 and 0 to 128 for IPv6). This route type allows inter subnet forwarding, overlapping IP address scheme with the help of route distinguishers.

FIGURE 4-62 represents the field included in an EVPN Route Type 5

Field Name	Size	Purpose
Route Distinguisher	8 bytes	Route distinguisher must be defined and unique per VRF (defined in RFC7432 and RFC 8365)
Ethernet Segment Identifier	10 bytes	Must be a nonzero 10 bytes identifier.
Ethernet Tag ID	4 bytes	VLAN ID configured.
IP Prefix length	1 byte	Specified the prefix length. (0 to 32 for IPv4 or 0 to 128 for IPv6).
IP Prefix	4 or 16 bytes	Advertises the actual IPv4 and IPv6 prefixes. (4 bytes for IPv4 and 16 bytes for IPv6)
Gateway IP Address	4 or 16 bytes	Next hop used as an overlay for the advertised IP prefixes (4 bytes for IPv4 and 16 bytes for IPv6)
MPLS Label	3 bytes	Layer 3 VNI identifier (unique per VRF)

Figure 4-62: EVPN Route Type 5 Fields

When designing an EVPN architecture based on NSX, the following network constructs must be taken into account:

- Segments used for the uplink interfaces
- Uplink interfaces for the Parent Tier-0

- MP-BGP configuration – L2VPN/EVPN Address Family
- Route Distinguisher
- Route Targets
- VXLAN Virtual network Identifier

4.7.1.1 Uplink interfaces, Loopback and Segment

Since NSX supports a single VXLAN TEP in the parent Tier-0 gateway, it is recommended to use a loopback interface.

The BGP architecture has no different requirements than a traditional non EVPN architecture, redundancy must be present in the BGP peering architecture towards the physical fabric.

It is recommended to have an edge cluster with at least 2 nodes. The Tier-0 deployed on this edge cluster should have interfaces (2 per edge nodes) connected to different top of rack switches. These interfaces should be /28 to allow the design to scale up to 8 ECMP edge nodes as demonstrated in FIGURE 4-63.

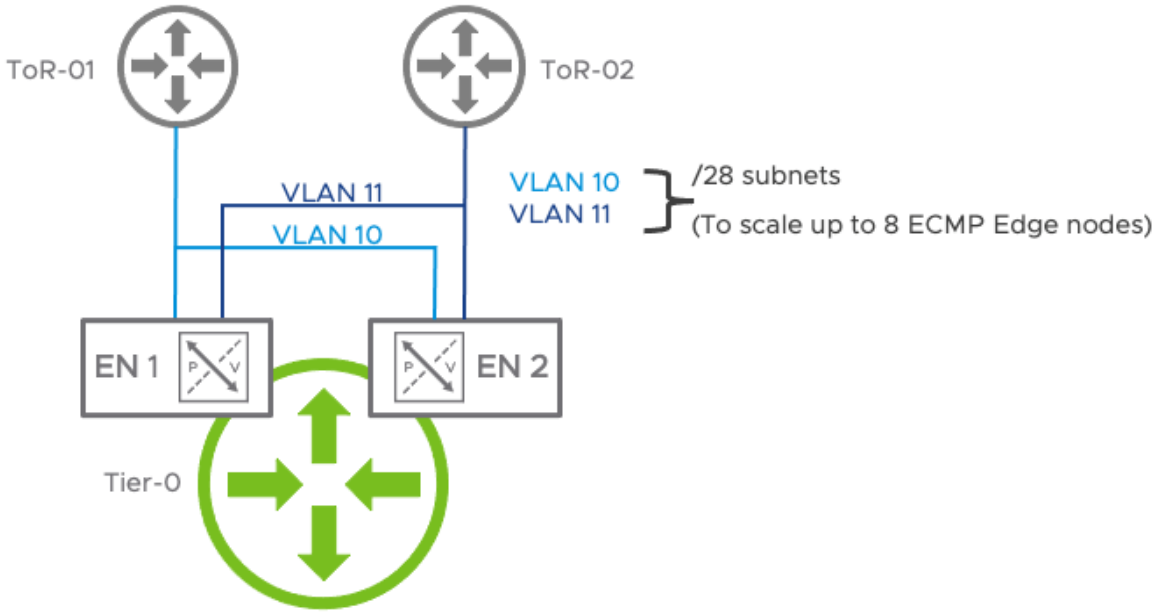


Figure 4-63: BGP Peering for EVPN

From a BGP standpoint, there are 2 mains requirements needed to setup the basic MP-BGP Peering:

- TEP interfaces (loopbacks) need to have reachability (Top of racks loopback interfaces to NSX Tier-0 Loopbacks hosted on each edge nodes)
- L2VPN/EVPN address family negotiated between the Top of Rack Switch and the Tier-0 Service Router.

To achieve connectivity between the loopbacks hosted on the Tier-0 SR and the Top of Rack switches, 2 designs are available as demonstrated in FIGURE 4-64. The first design will have a BGP peering between the NSX Tier-0 SR and the Top of Rack switches using their uplink interfaces. The loopbacks must be redistributed into BGP so that the entire network fabric is aware of these TEP IP addresses to reach the different VRF prefixes hosted on the NSX domain. The second design uses BGP Multi-hop between the loopback interfaces. To achieve reachability between the loopbacks in this case, the use of static routing or OSPF is needed, otherwise the BGP Multi-hop peering will never reach the "established" state.

The first design with loopback redistribution is recommended as it simplify the overall architecture and reduce the operational load.

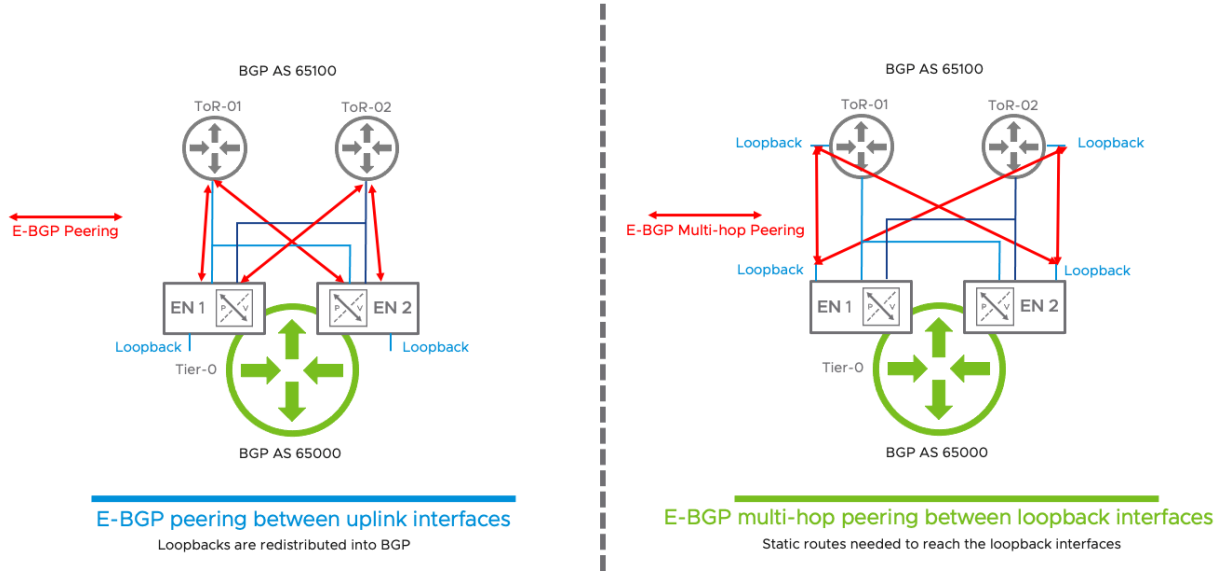
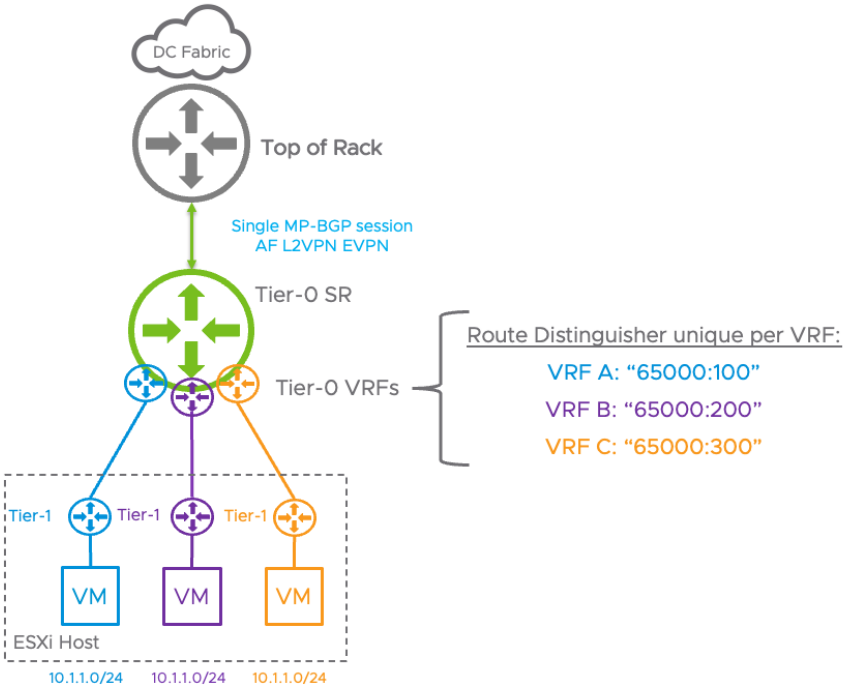


Figure 4-64: Peering over uplink interfaces with loopback redistribution into BGP (recommended) vs multi-hop E-BGP and peering over the loopback interfaces

The **Route Distinguisher** field in a Type 5 EVPN route has the following functionalities and properties:

- Allows VRF prefixes to be considered unique. If two tenants advertise the same prefix, the Tier-0 (and the network fabric) must have a way to differentiate these 2 overlapping prefixes so that they are globally unique among the different isolated routed instances.
- As a result, the Route Distinguisher must be unique per VRF. It is an 8 bytes identifier
- NSX can auto assign Route Distinguishers, or an administrator can configure them
- Network engineer traditional use the format "BGP_AS:VRF" to configure the route distinguishers

The following pictures show a multi-tenancy architecture based on a single MP-BGP EVPN session that advertises the same prefix owned by 3 different tenants using route distinguishers to make the routes unique.



The role of the **Route Target** configuration per VRF is:

- A route target is a BGP extended community and is configured to import/export the prefixes in specifics VRF
- When a particular prefix is exported in NSX with a route-target that matches the VRF import route target configuration on the top of rack, the prefix will be imported in that particular VRF routing table.
- When a particular prefix is exported in NSX with a route-target that does not matches the VRF import route target configuration on the top of rack, the prefix will not be imported in that particular VRF routing table.
- The route targets import/export should match between the NSX Tier-0 and the Top of Rack switches and should be unique per VRF instance/tenants.
- It is possible to configure multiple route-target import/export statement per VRF.

The role of the **Virtual Extensible LAN (VXLAN) ID** is:

- Each VRF should be mapped to a unique VNI to differentiate the network traffic between tenants from a data plane standpoint.

- Geneve is the overlay encapsulation protocol used internally within NSX. VXLAN is used as an overlay between the Top of Rack Switches and the NSX
- Since NSX already uses its own Geneve Overlay to allow communication between NSX TEP, the VXLAN VNI range should be unique
- VXLAN VNI Range must be between 75001 and 16777215

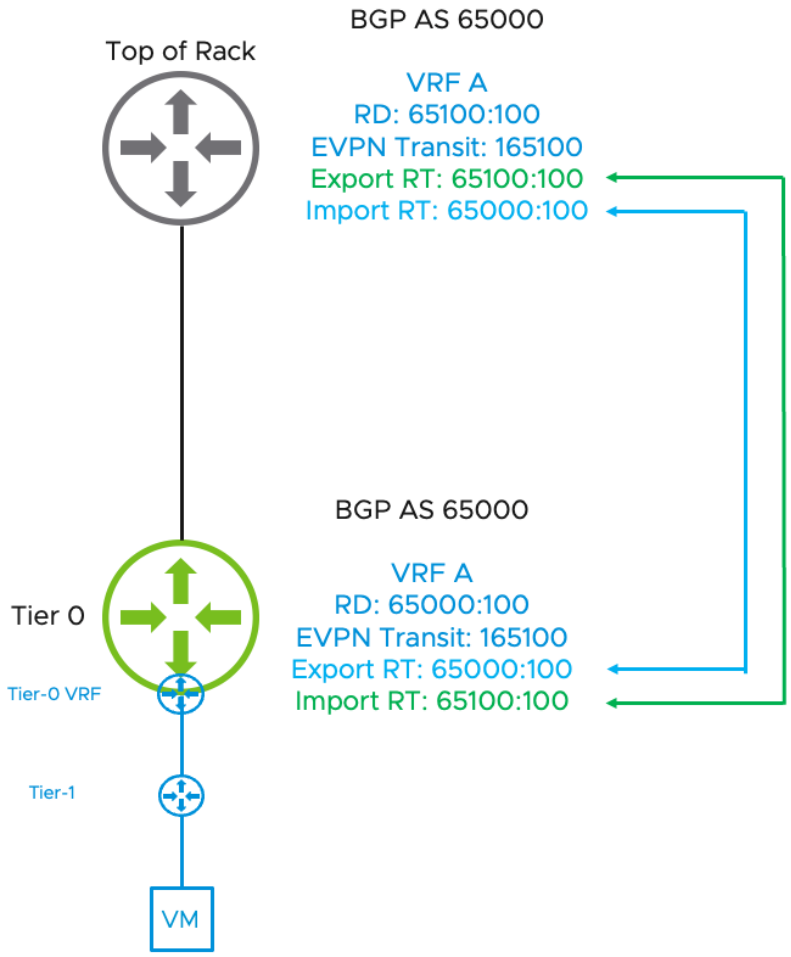


Figure 4-65: Relationships between Route Distinguisher, Route Target and VXLAN ID

FIGURE 4-66 summarizes the networking constructs needed per VRF to design and implement MP-BGP EVPN with VXLAN between the Tier-0 gateways and the Top of Rack switches.

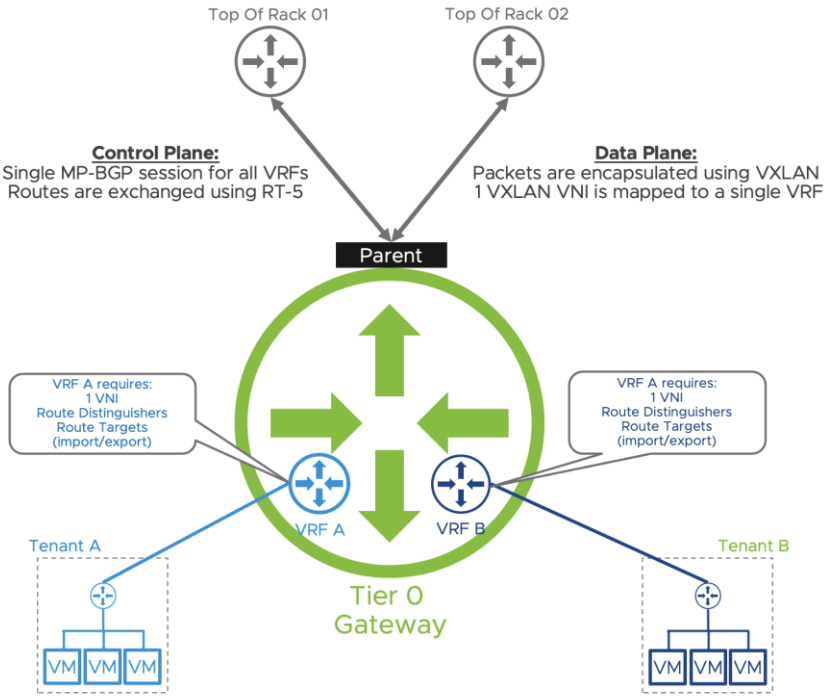


Figure 4-66: Inline EVPN Topology

The NSX edge node uses Geneve TEP interfaces for all NSX overlay traffic. To exchange network traffic with the Top of Rack switches, the Tier-0 must have a different TEP interfaces enabled for VXLAN. One VXLAN TEP being supported on each edge node, loopback interfaces acting as VXLAN TEP interfaces are recommended in an NSX EVPN architecture.

4.8 Multicast routing within NSX

This section details the capabilities of NSX for routing multicast traffic and provides a high-level technical overview of the implementation of its data plane operations. This knowledge is helpful in understanding the recommendations listed in the last part of this chapter.

4.8.1 Multicast in a virtual environment

When considering a virtual network environment, it's essential to notice that it includes a significant part handled by the physical infrastructure. NSX is implementing most networking functions, but in the end, physical switches and routers are responsible for forwarding traffic between the transport nodes. FIGURE 4-67 illustrates this point in the context of multicast routing.

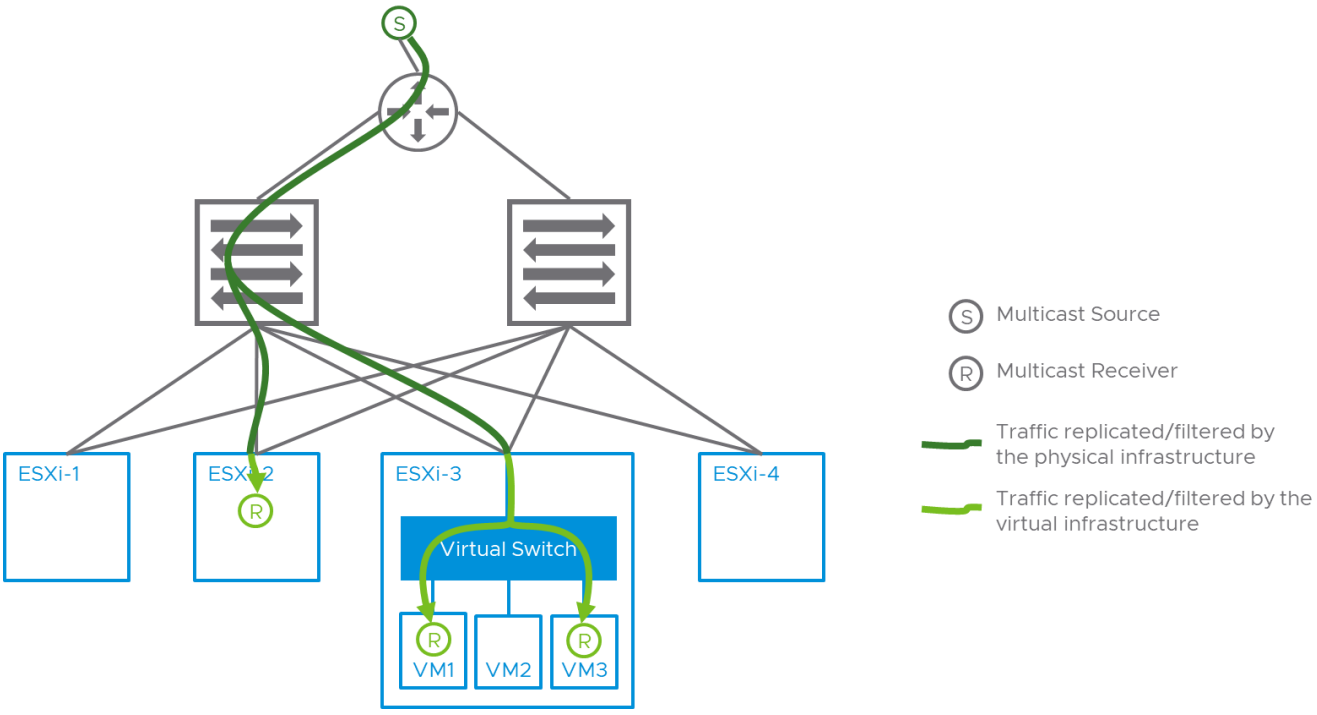


Figure 4-67: multicast between hosts and within a host

The diagram represents a multicast source located in the physical infrastructure sending to multiple virtual machines on different ESXi hosts. Two kinds of multicast are involved here:

- multicast between physical devices, represented in dark green,
- multicast handling on the virtual switch inside the ESXi hosts, represented in light green. More specifically, you can see that within ESXi-3, traffic has been replicated to VM1 and VM3, but filtered to VM2.

When running multicast in a VLAN-backed environment (NSX VLAN segments, or simple dvportgroups, if NSX is not present in the picture), the physical infrastructure can directly snoop the IGMP traffic sent by the virtual machines and take care of the multicast replication and filtering between the hosts. In that scenario, there is no difference between NSX and regular vSphere networking: the virtual switch is identifying the receivers using IGMP snooping and delivers the multicast traffic to the appropriate vnics. We will not elaborate on this capability in this guide.

For overlay segments however, the physical infrastructure is unaware of the traffic being tunneled and it's up to NSX to handle efficiently the distribution of multicast traffic to the appropriate transport nodes. This design guide section is going to focus on that scenario: the replication/filtering of multicast traffic between transport nodes with an NSX overlay.

4.8.2 NSX multicast capabilities

As of NSX 3.2, Tier0 and Tier1 Gateways are capable of routing IPv4 multicast traffic between overlay segments (no IPv6 multicast.) This capability exists within the NSX domain, but also between the Tier0 Gateway and the physical

infrastructure, thus allowing multicast sources and receivers inside and outside NSX.

Between NSX and the physical infrastructure:

- The Tier0 gateway must be explicitly enabled for multicast routing and have at least one uplink configured for multicast routing.
- The Tier0 gateway runs PIM sparse mode on its multiple uplinks (other PIM flavors are not supported.)
- The rendez-vous point (RP) must be in the physical infrastructure. NSX does not provide the RP functionality.
- The RP can be identified within NSX using static RP (different RPs can be assigned statically on different IP multicast address ranges) or via multicast PIM bootstrap (BSR), a standard protocol included in PIMv2 that is used to automatically find the RP in a multicast network.

Inside NSX:

- Segments must be enabled for multicast routing (this is the default) and attached to a gateway with multicast routing enabled.
- Just like Tier0 gateways, Tier1 gateways must be enabled explicitly for multicast routing and also connected to a Tier0 with multicast routing enabled.
- NSX snoops IGMPv2 messages from the virtual machines to identify multicast receivers. NSX 3.2 does not support IGMPv3 yet and source specific multicast (SSM) is not possible at that stage.

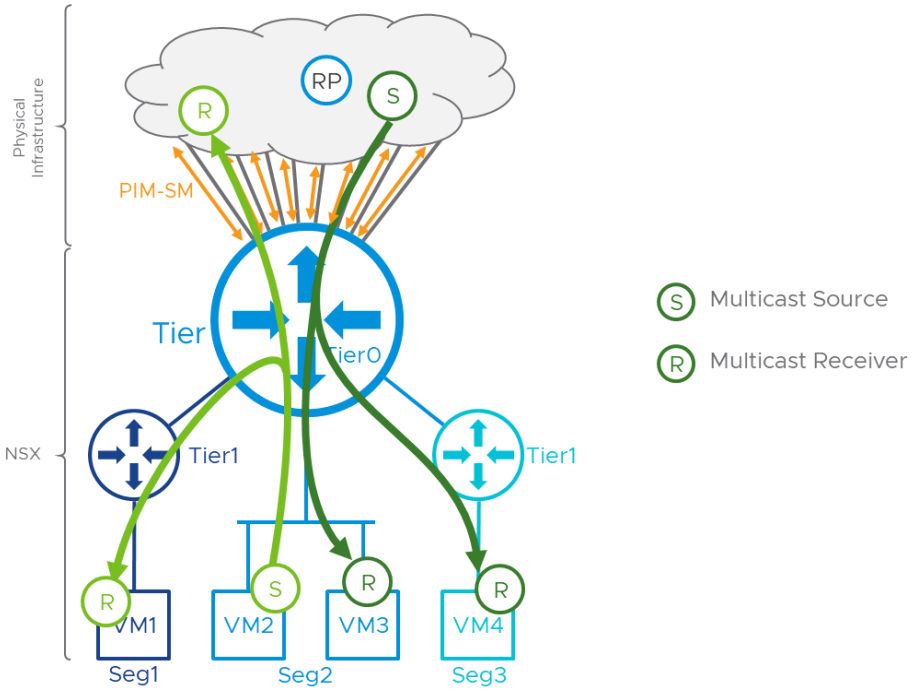


Figure 4-68: NSX multicast routing capabilities

FIGURE 4-68 summarizes the kind of multicast connectivity provided by NSX with multicast routing. Sources and receivers can be inside or outside the NSX domain, the Tier0 gateway is running PIM SM to the physical infrastructure where the RP is located.

4.8.3 Technical overview

NSX routes multicast traffic between overlay segment thanks to its Tier0 and Tier1 gateways. This section focuses on the data plane operations within NSX: how NSX achieves efficient multicast replication between transport nodes. This multicast traffic is carried on an overlay network and the physical infrastructure cannot look inside the tunnels NSX creates.

4.8.3.1 Control plane

As mentioned above, the Tier0 gateway runs PIM sparse mode and relies on an external rendez-vous point. Other than that, there is absolutely no multicast routing protocol running inside NSX between transport nodes. NSX simply distributes the IGMP reports sent by the virtual machines to all the transport nodes within a gateway routing domain (the routing domain of a gateway is the collection of transport nodes where this gateway is instantiated) so that they get a complete picture of where the receivers are disseminated.

4.8.3.2 Data plane operations between Tier0 gateway and physical infrastructure

The Tier0 gateway connecting NSX to the physical infrastructure is implemented as multiple Tier0-SRs on multiple edge transport nodes. FIGURE 4-69 illustrates a Tier0 made of four active/active Tier0-SRs, each configured with two physical uplinks running PIM sparse mode. Thanks to ECMP, unicast traffic can be sent and received on all the uplinks of those Tier0-SRs. NSX is also spreading multicast traffic across the different Tier0-SRs on a per-multicast group basis.

For a given destination multicast IP address, NSX uses a hash to determine a unique “active” Tier0-SR that will be responsible for handing traffic in and out of the NSX domain for this group. Different groups are hashed to different Tier0-SRs, thus providing some form of load balancing for multicast traffic. There is no configuration option to manually select the multicast active Tier0-SR for a given multicast group.

All Tier0-SRs must have a configuration and connectivity to the physical infrastructure capable of handling all the groups. If a Tier0-SR fails, the multicast groups it was handling will be assigned to an arbitrary Tier0-SR that is still active. No multicast routing state is synchronized between the Tier0-SRs, and all the affected multicast groups will see their traffic disrupted until PIM has re-populated its tables. Note that only the groups that were associated to the failed Tier0-SR are impacted, other groups remain associated to their existing active Tier0-SR.

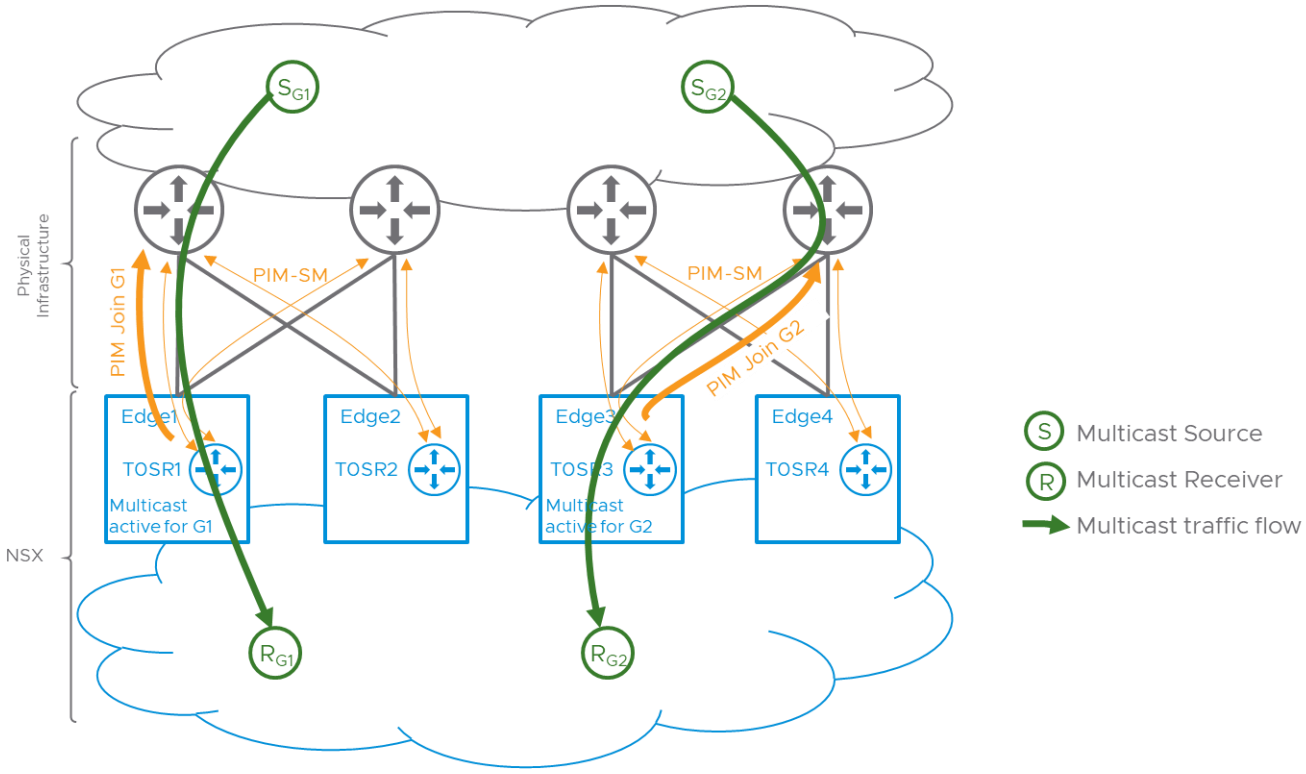


Figure 4-69: multicast between edges and physical infrastructure, external sources, internal receivers

NSX will only join the source tree of a multicast source in the physical infrastructure on one of the uplinks of the multicast active Tier0-SR for that group. That means that external multicast traffic will always enter NSX through the multicast active Tier0-SR. In the above FIGURE 4-69, the multicast traffic for G1 is entering NSX through Tier0-SR1, the multicast active Tier0-SR for group G1. Multicast group G2 could be associated to another Tier0-SR, like Tier0-SR3 in the diagram, thus spreading inbound traffic across multiple Tier0-SRs.

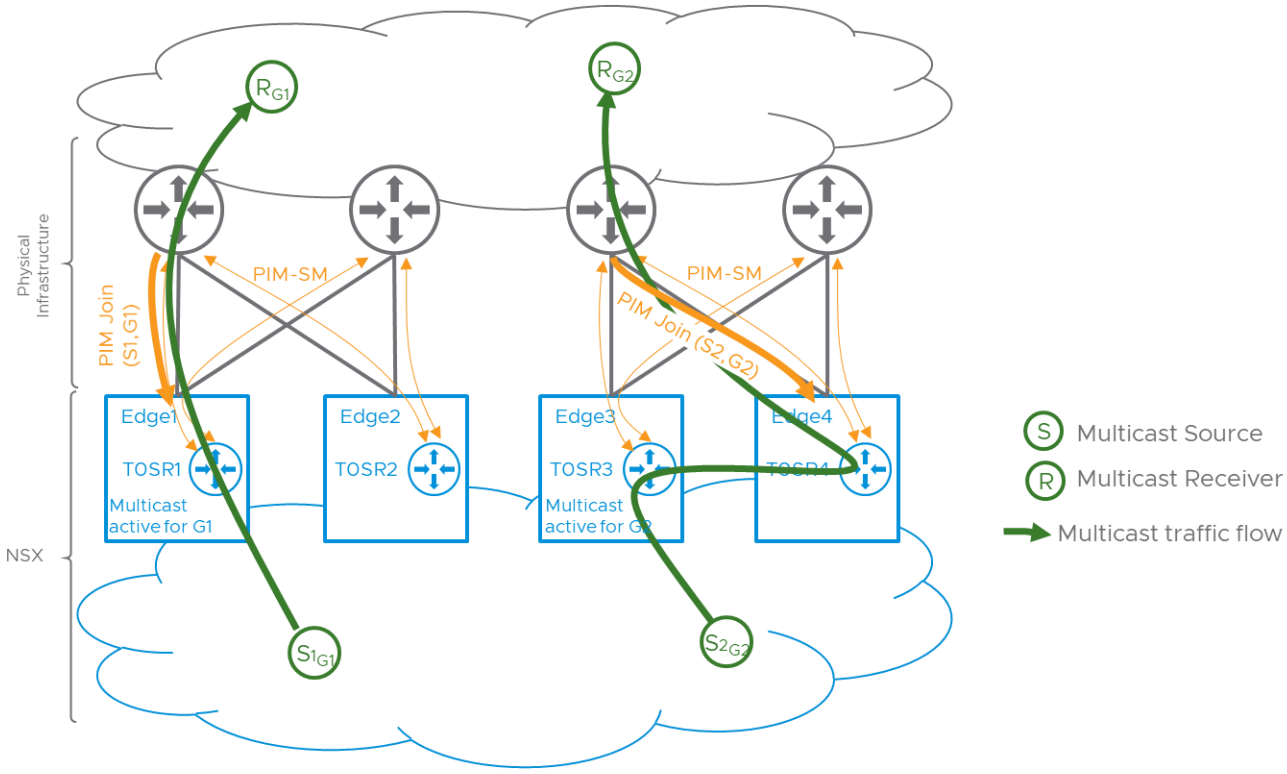


Figure 4-70: multicast between edges and physical infrastructure, internal sources, external receivers.

When the source for the group is located inside the NSX domain, the traffic pattern might be slightly different. As soon as the IP address of the source is known to the physical infrastructure, a PIM join will be sent toward the IP address of the source, in order to join the receiver to the source tree.

This PIM join can be forwarded by the physical infrastructure on any path toward the source. Thanks to unicast ECMP, the IP address of the source, is reachable via any Tier0-SR. In FIGURE 4-70 above, the external receiver for group G1 ended up joining the source tree rooted at $S1_{G1}$ through Tier0-SR1, the Tier0-SR active for G1. This is purely by chance. By contrast, the receiver for group G2 ended up trying to join the source tree rooted at $S2_{G2}$ via Tier0-SR4.

This is not the Tier0-SR active for G2. Tier0-SR4 will itself need to join the source tree rooted inside NSX in order to provide connectivity to the receiver. Because G2 is associated with Tier0-SR3, the multicast traffic to G2 is always sent to Tier0-SR3. From there, it will be forwarded to Tier0-SR4 so that it can reach the receiver that joined the source tree through an uplink of Tier0-SR4. In the "south-north" direction, multicast traffic can thus hairpin to the multicast active Tier0-SR before reaching its destination.

4.8.3.3 Data plane operations within NSX for segments attached to a Tier0 gateway

We have seen that the multicast traffic going in and out the NSX domain for group G is always routed through the multicast active Tier0-SR for G. The rest of this section is dedicated to the multicast traffic flow within NSX. We're only going to represent traffic using active Tier0-SR1 in the diagrams, but of course, traffic could be going through any other Tier0-SR, depending on the group.

The following diagram show the path for multicast traffic initiated in the physical infrastructure and distributed inside NSX via the multicast active Tier0-SR1. In this scenario, we're going to assume that all the receivers are attached to segments directly connected to the Tier0 gateway (in other words, there is no Tier1 gateway in the picture).

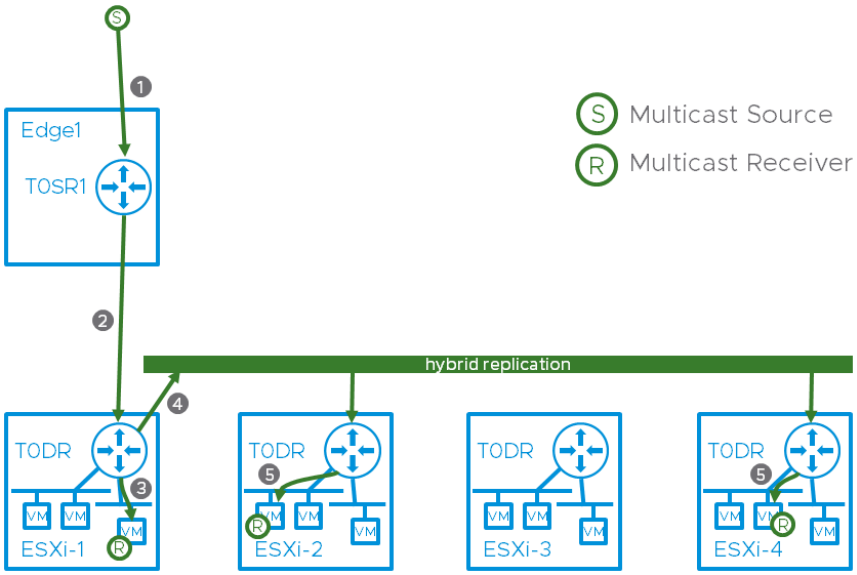


Figure 4-71: external source, receivers on Tier0 segments

1. Multicast traffic from the external source S reaches the Tier0-SR multicast active uplink. There are receivers in the NSX domain, so this packet needs to be replicated to one or multiple ESXi transport nodes.
2. For performance reasons, Tier0-SR1 is not going to replicate the traffic to the interested transport nodes in the NSX domain. Instead, it's going to "offload" the replication to an ESXi host, picked arbitrarily in the Tier0 routing domain (again, the routing domain is the group of transport nodes where the Tier0 gateway spans.) The host selected for replication does not need to have a local receiver for the traffic, but the destination multicast IP address is part of the hash used to select the host. This way, this multicast offload is distributed across all the hosts in the routing domain, based on multicast group. A note on the format of the packet that is forwarded to the ESXi-1 transport node: this is a multicast packet tunneled unicast to ESXi-1. The destination IP address of the overlay packet is the unicast IP address of a TEP in ESXi-1.
3. In this example, ESXi-1 has been selected to replicate the multicast packet. First, the local Tier0-DR routes the traffic to any receiver on local VMs.
4. Then the Tier0-DR initiate a "hybrid replication" to all the remote hosts in the Tier0 routing domain that have at least one receiver (and only those hosts.) The next section will detail what this hybrid replication means, at that stage, just assume that it's a way of leveraging the physical infrastructure to assist packet replication between ESXi hosts.
5. At that final step, the remote Tier0-DRs receive the multicast packet and route it to their local receivers.

FIGURE 4-72 details the multicast packets flow when a source is in the NSX domain (on a segment attached to a Tier0) and there are receivers inside and outside NSX.

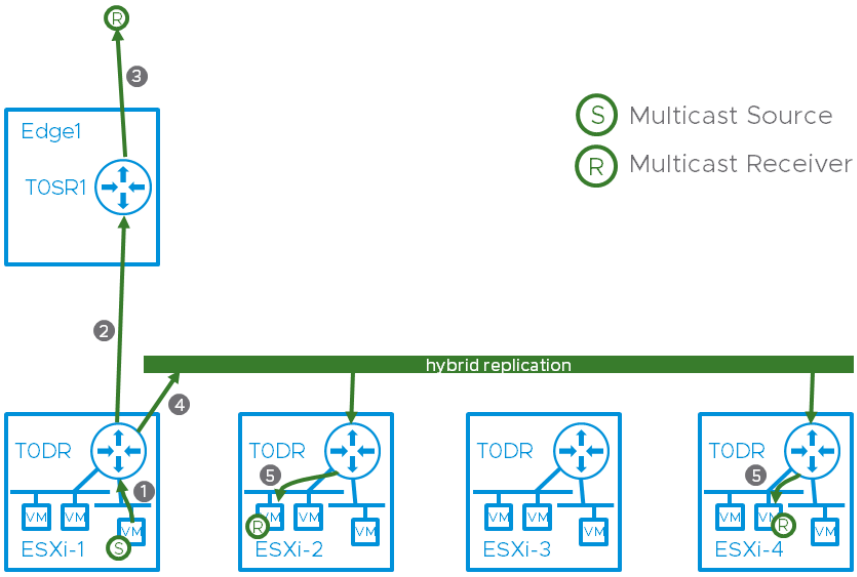


Figure 4-72: internal source on Tier0 segment

1. The VM sends multicast traffic on the segment to which it is connected. It reaches the local Tier0-DR. Note: there is no receiver on ESXi-1. If there were some in the source segment, they would directly receive the multicast traffic, forwarded at Layer 2 (with IGMP snooping.) If there were some receivers on other segments in ESXi-1, the T0-DR would route the traffic directly to them too.
2. Again, we assumed that Tier0-SR1 is multicast active for the group. It is thus considered as being the multicast router and needs to receive a copy of every multicast packet generated in the NSX domain for this group. The Tier0-DR on ESXi-1 sends a unicast copy of the multicast packet to Tier0-SR1 (multicast traffic encapsulated in unicast overlay.)
3. The Tier0-SR receives the traffic and routes it to the outside world on one of its uplinks (or potentially through another Tier0-SR, as we have seen earlier.)
4. The Tier0-DR of ESXi-1 also starts hybrid replication to forward the multicast traffic to all other ESXi transport nodes with receivers in the routing domain (the part dedicated to hybrid replication will detail how this is achieved.)
5. The remote Tier0-DRs forward the multicast traffic to their local receivers.

4.8.3.4 Data plane operations within NSX: Tier0 and Tier1 segments combined

Until NSX 3.0, multicast routing was only possible on Tier0 gateways. NSX 3.1 introduces the capability of routing multicast on Tier1 gateways. The handling of the Tier1 gateways is an extension of the model described in the previous part for the Tier0 gateways.

When multicast routing is enabled on a Tier1 gateway, a Tier1-SR **must** be instantiated on an edge. This Tier1-SR will be considered as the multicast router for the Tier1 routing domain and will receive a copy of all multicast traffic initiated on a segment attached to this Tier1 gateway. Then this Tier1-SR will be attached as a leaf to the existing multicast tree of its Tier0 gateway (remember that in order to run multicast, a Tier1 gateway must itself be attached to a Tier0 gateway running multicast.)

The following diagram represents the traffic path of some external multicast traffic that needs to be replicated within NSX to receivers off a Tier0 gateway as well as two Tier1 gateways (green and purple.) Step 1-5 are about replicating the traffic to segments attached to the Tier0 gateway, thus exactly similar to the previous example.

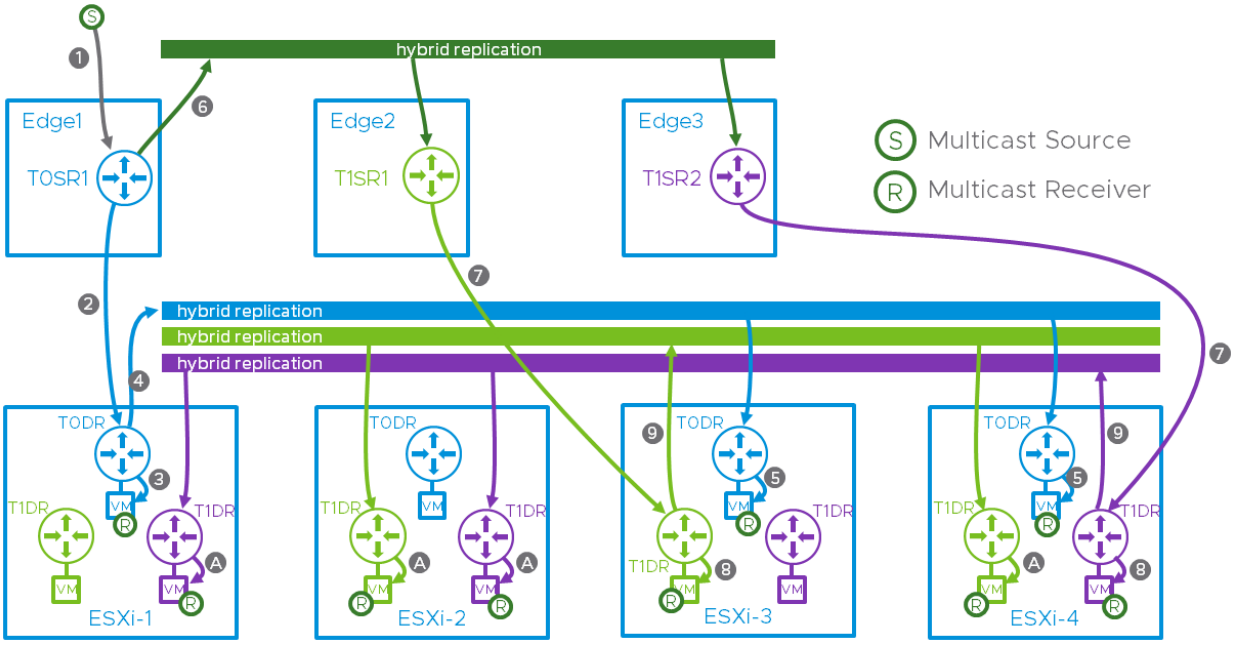


Figure 4-73: external source, receivers on Tier0 and Tier1 segments

1. The multicast traffic is received by the multicast active Tier0-SR for the group (Tier0-SR1.)
2. The Tier0-SR1 offloads replication to an arbitrary ESXi transport node, here ESXi-1.
3. The Tier0-DR of the offload transport node (ESXi-1) routes the multicast traffic to local receivers.
4. The Tier0-DR of ESXi-1 starts hybrid replication to reach all the ESXi hosts that have receivers on segments attached to the Tier0 gateway.
5. The Tier0-DRs of those remote ESXi transport nodes route the traffic to their local receivers.
   So far, those steps have been exactly like those described in the previous part.
6. This is the first step specific to multicast on Tier1 gateways: the multicast active Tier0-SR1 initiate a hybrid replication targeting all the Tier1-SRs with receivers for this group.

7. The Tier1-SRs (green and purple) offload the multicast replication to an arbitrary ESXi transport node in their routing domain. In this respect, they act exactly the same way as the multicast active Tier0-SR for its routing domain.
8. The ESXi transport nodes selected for offloading their Tier1-SR (ESXi-3 for the green Tier1 and ESXi-4 for the purple Tier1) route the traffic to their local receivers, if any.
9. The ESXi hosts selected for offloading their Tier1-SR initiate a hybrid replication for reaching all the interested Tier1-DRs in their Tier1 routing domain.
10. Finally, the Tier1-DRs on remote transport nodes route the multicast traffic to their local receivers.

For the sake of being complete, FIGURE 4-74 is showing the traffic flow when a multicast source is in the routing domain of a Tier1 gateway and there are receivers off another Tier1 gateway, Tier0 gateway and external.

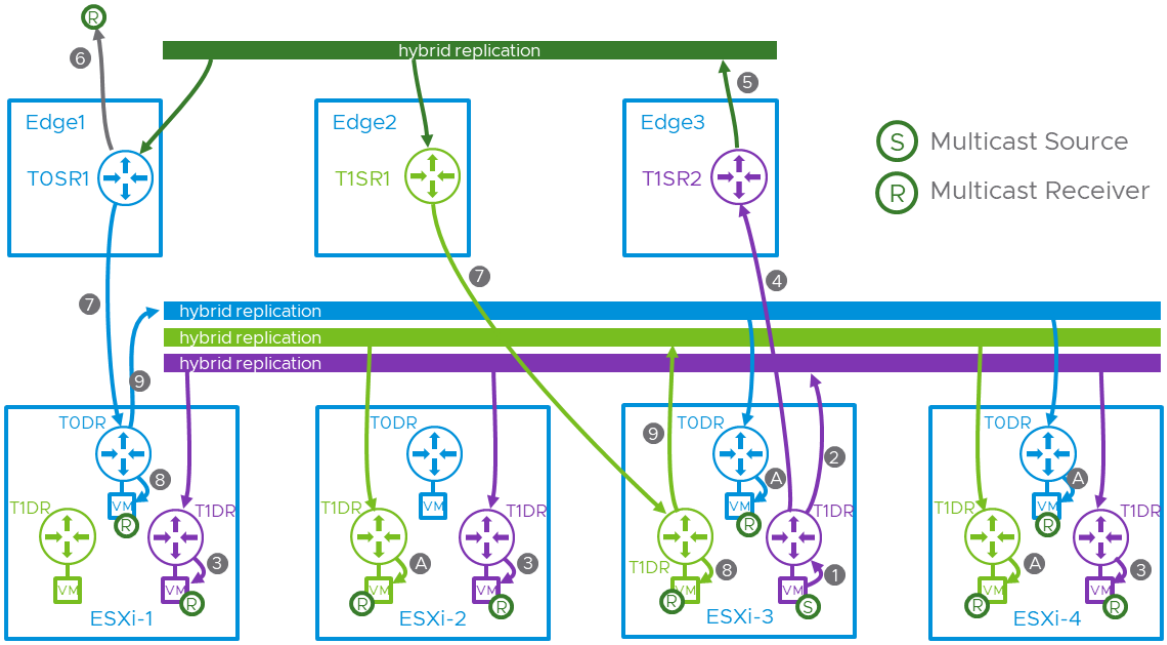


Figure 4-74: internal source, external receiver and receivers on Tier0 and Tier1 segments

1. The source is located on ESXi-3, off a segment attached to the purple Tier1. The multicast traffic is first received by the local purple Tier1-DR.
2. The purple Tier1-DR initiate a hybrid replication to reach every remote purple Tier1-DR.
3. The remote purple Tier1-DRs route the traffic to their local destination.
4. The purple Tier1-DR of ESXi-3 also sends a copy of the multicast traffic to the multicast router of the routing domain, i.e. the purple Tier1-SR on edge3.
5. The purple Tier1-SR forwards the traffic to all SRs with receivers for this group (Tier0SR and green T1SR). Note that the diagram is simplifying the real process. You don't need to understand the details, but for accuracy, let's just mention that the purple Tier1-SR is in fact sending a unicast copy

(not represented) to Tier0SR1, its multicast router. Then it initiates a hybrid replication that will reach all SRs with receivers. So, the T0SR receives two copies: one is replicated on the uplink toward external receivers (step 6 below), while the other will go down the tree to internal receivers on segments attached to the Tier0 (part of step 7 below.)

6. The multicast active Tier0-SR routes the traffic on one of its uplinks toward the external receiver(s).
7. The Tier0SR and green Tier1-SR now offload replication to an arbitrary ESXi transport node in their routing domain.
8. The Tier0-DR on ESXi-1 and green Tier1-DR on ESXi-3 (the host selected to offload the green Tier1-SR) route the multicast traffic to their local receivers, if any.
9. The Tier0-DR on ESXi-1 and green Tier1-DR on ESXi-3 initiate a hybrid replication to their peers in their respective routing domain.
10. The remote Tier0-DR and green Tier1-DR route the traffic to their local receivers.

Some few notes on the Tier1 multicast model:

- The use of a Tier1-SR for multicast has an impact on unicast traffic. Indeed, unicast traffic that needs to be routed to other Tier1s or to the outside world will now transit through this Tier1-SR.
- Multicast traffic for Tier1 attached segments follows a tree that is rooted on the corresponding Tier1-SR. The consequence is that a host with receivers on segments that are attached to different Tier0/Tier1s will get multiple copies of the same multicast traffic, one for each Tier0-DR or Tier1-DR involved. Check ESXi-4 in the above FIGURE 4-74: it received three separate copies of the same multicast packet, one for each gateway.

4.8.3.5 Hybrid replication

NSX implements its overlay segments by maintaining point to point tunnels between its transport nodes. If we kept the exact same model for implementing the multicast data path, this would imply that NSX would be responsible for performing in software all the multicast replication work. Indeed, the physical infrastructure cannot help directly, as it's not aware in any way of the kind of traffic the NSX tunnels carry.

4.8.3.5.1 Hybrid replication concept

Hybrid replication is all about leveraging the replication capability of the physical infrastructure. To achieve that, NSX will now map a multicast group in the overlay segment to an underlay multicast group. This will de facto create "multi-point" tunnels, where the overlay segment multicast traffic is encapsulated in a tunnel packet with a multicast destination address and replicated by the physical infrastructure to multiple hosts.

NSX does not assume that the physical network (the "underlay") is capable of routing multicast traffic though. NSX will just assume that if it sends IP multicast traffic on the physical infrastructure, this traffic will be replicated to all the interested device in the same Layer 2 domain as the source. Ethernet switches

can replicate in hardware multicast traffic, and most of them run IGMP snooping by default, ensuring an adequate filtering of multicast traffic within a Layer 2 domain. NSX will take care of replicating the multicast traffic across the underlay IP subnets.

4.8.3.5.2 Hybrid replication example

When configuring NSX for multicast routing, the administrator is prompted for a "Replication Multicast Range". This is a range of multicast addresses reserved in the physical infrastructure for hybrid replication. This range is split evenly in two ranges, let's call them range1 and range2. Range1 will be used for hybrid replication between ESXi transport nodes, while range2 will be used for hybrid replication between service routers (Tier0-SR and Tier1-SRs.)

Suppose that a transport node is interested in receiving multicast traffic sent to multicast group G in the overlay. This group G will be hashed by NSX into two multicast IP addresses M1 and M2 in the replication multicast range1 and range2 respectively. Then if the transport node is an ESXi host, it will join group M1 in the underlay. This means that a TEP on this ESXi host will send an IGMPv2 report with the intend of joining group M1. The physical infrastructure (running IGMP snooping) can act on this IGMP message and record that this ESXi host is a receiver for M1. If the transport node was a service router on an edge node, it would join the multicast group M2 in the underlay. That's the reason for the even split of the replication multicast range: one is for hybrid replication between ESXi transport nodes, and the other is for hybrid replication between edge transport nodes (you'll notice in the previous multicast flow examples that there is no hybrid replication between edges and ESXi hosts, only unicast communication.) Now that all the transport nodes interested in G have joined a common multicast group in the underlay, NSX can send tunnel traffic to multicast address M1 or M2 to replicate efficiently multicast traffic for group G between ESXi hosts or edges: within the same Layer 2 domain, the physical infrastructure will be able to take care of the replication. When traffic needs to be replicated across underlay subnets, NSX will send a single unicast copy of the multicast traffic to a remote transport node in each remote subnet. Don't worry if this sounds confusing at that stage, the following diagrams will hopefully make that clear!

In this following scenario, let's assume that a virtual machine on transport node TN-1 generates multicast traffic to multicast group G, and that there are receivers on multiple ESXi transport nodes. This example is not about repeating the multicast packet flow presented in the previous part, we're not going to consider the multicast traffic sent to the multicast router for example, we're just going to show how hybrid replication can deliver this multicast IP packet to all the other transport nodes that have a receiver for G. In this diagram those transport nodes with receivers are TN-2, TN4, TN-5, TN-7, TN-8, TN-9, TN-10, and TN-11. The transport nodes have their TEPs in three different subnets. This is a common design: typically, hosts under the same top of rack switch (TOR) have their IP address in the same subnet. In this example, transport nodes TN-1 to TN-4 have their TEP IP addresses in subnet 10.0.0.0/24, TN-5 to TN-8 have TEPs in subnet 20.0.0.0/24 and TN-9 to TN-12 have TEP addresses in 30.0.0.0.24.

Let's assume that NSX has mapped group G to group M in the replication multicast range for ESXi transport nodes (called earlier as "range1".) All the ESXi hosts with a receiver for G in the overlay are joining group M in the physical infrastructure (that's what the "M" on the TEPs is representing.)

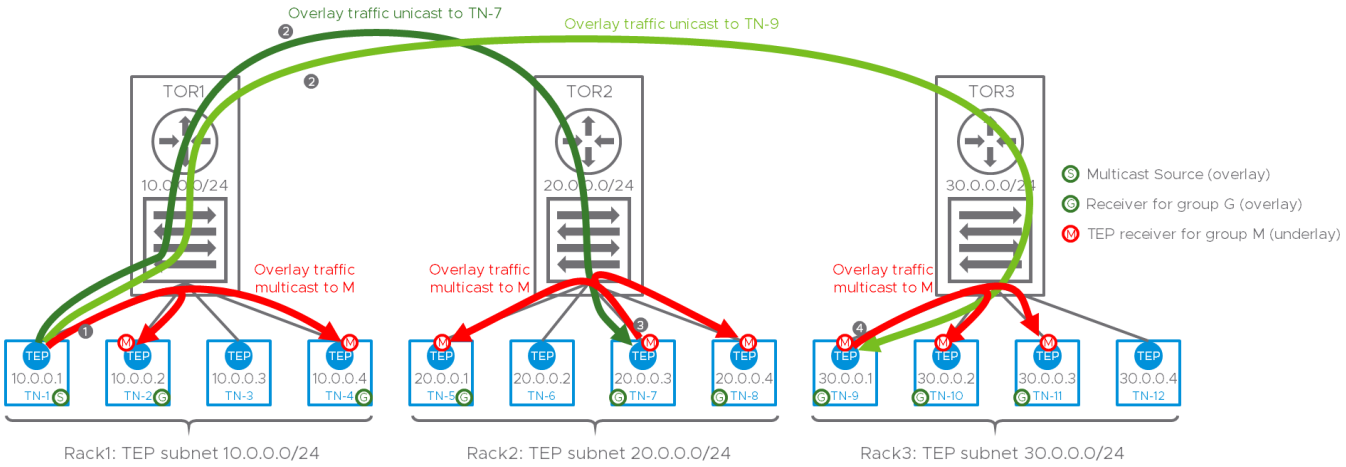


Figure 4-75: hybrid replication

Transport nodes TN-1 knows the IP address of all the transport nodes interested in G (as mentioned earlier, this has been advertised by relaying IGMP packets.) More specifically, TN-1 knows that it has receivers in its local subnet and in two remote subnets.

1. TN-1 encapsulate the multicast traffic to G in an overlay packet with M as a destination and send it on its uplink. The physical infrastructure sees a packet to M and knows that there are two hosts that have joined group M. The top of rack switch takes care of replicating this packet to TN-2 and TN-4. Note again that we don't expect the physical infrastructure to route multicast traffic across subnet. The packet sent to M is thus constrained to its source subnet 10.0.0.0/24 and not routed to the remote subnets 20.0.0.0/24 and 30.0.0.0/24 by the physical infrastructure.
2. TN-1 knows that receivers TN-5, TN-7 and TN-8 are all in the remote underlay subnet 20.0.0.0/24. It picks one of those three transport nodes (let's say TN-7) and forward the multicast traffic to G as an overlay unicast to it. TN-1 repeats the operation for the remote underlay subnet 30.0.0.0/24 and sends the multicast traffic to G as an overlay unicast to TN-9.
3. TN-7 receives an encapsulated multicast packet from TN-1. In the metadata of this packet, TN-1 has set a bit indicating that this packet needs to be replicated to the other transport nodes in the underlay subnet (reference section OVERLAY ENCAPSULATION). TN-7 delivers the multicast packet to its local receiver VMs and copies the multicast traffic to G on its uplink, encapsulated with an overlay destination multicast IP address M. The physical infrastructure is then again taking care of replicating this packet to TN-5 and TN-8. The metadata of this overlay packet created by TN-7 is not indicating this time that this packet needs to be replicated to the other hosts in the same subnet, so TN-5 and TN-8 only forward the multicast traffic to their local receivers.
4. TN-9 receives the overlay unicast copy of the multicast traffic to G and perform the same operation as TN-7 in the previous step.

4.8.4 NSX multicast design recommendations

4.8.4.1 Beware of the TTL

The NSX gateways behave like regular routers and do decrement the TTL of the packet they forward. Multicast packets going through a Tier0 gateway and a Tier1 gateway will have their TTL decremented twice. Make sure that the multicast traffic generated has an appropriate TTL, we've seen multiple examples where customers migrating from NSX for vSphere losing multicast traffic because of a wrong TTL (in NSX for vSphere, the edge and distributed logical router represent a single logical router and only decrement the TTL once.)

4.8.4.2 Connecting the Tier0-SRs to the physical infrastructure

The administrator has no influence over which Tier0-SR is selected by NSX as the multicast active for a specific group. When the design includes multiple active/active Tier0-SRs, make sure they have consistent configuration.

Also, make sure that PIM is enabled on all the uplinks of the Tier0-SRs. This is not really an NSX specific recommendation. Multicast routing must be enabled on all the unicast paths toward the multicast source. As seen earlier, a PIM join from a receiver in the physical infrastructure can reach the Tier0 gateway on any uplink leading to this source in the NSX domain.

4.8.4.3 Hybrid replication

This multicast section of the design guide included a relatively detailed description of the data plane so that the reader could understand the following recommendations, tightly linked to the way NSX uses hybrid replication.

The replication multicast range should be a dedicated non-routed range of multicast addresses in the physical infrastructure

When configuring multicast, the administrator needs to provide a range of multicast IP addresses that will be used by NSX in the physical infrastructure for hybrid replication. Theoretically, the overlay and underlay IP spaces are entirely independent, and you could potentially use the same multicast addresses in overlay and the underlay. However, it is recommended to reserve a separate range of multicast IP address for the NSX replication range. First, it will make it easier to identify the traffic generated by NSX for its hybrid replication. Second, it's important to make sure that the physical infrastructure does not enable multicast routing for the replication range. The reason is that NSX is responsible for propagating overlay multicast between subnets. For example, suppose that in Figure 4-75: hybrid replication, multicast routing was enabled for group M. In that scenario, the encapsulated multicast traffic sent to multicast address M by TN-1 (in step 1) would be routed to the remote subnets 20.0.0.0/24 and 30.0.0.0/24 by the physical infrastructure. Note that the underlay multicast traffic generated by NSX carries a TTL of 1. So, the remote hosts would most likely not receive multiple copy of the multicast traffic already propagated by NSX between subnets (and even if they received it, they would drop it.) The result would however be suboptimal, as some bandwidth would be wasted for transmitting traffic that would be eventually discarded.

Reserve the largest replication multicast range possible

There is a mapping between each multicast IP address used in the overlay and a unique pair of multicast addresses in this replication multicast range.

Suppose that the virtual network uses 1000 different multicast groups, but that the replication multicast range only reserves 100 multicast addresses (leading to 50 addresses in range1 and range2.) Obviously, there is a 1:20 over-subscription and multiple multicast groups from the overlay address space will map to the same multicast address in the replication range. In the following diagram, due to a hashing collision, two overlay multicast groups G1 and G2 are mapped by NSX to the same underlay multicast group M.

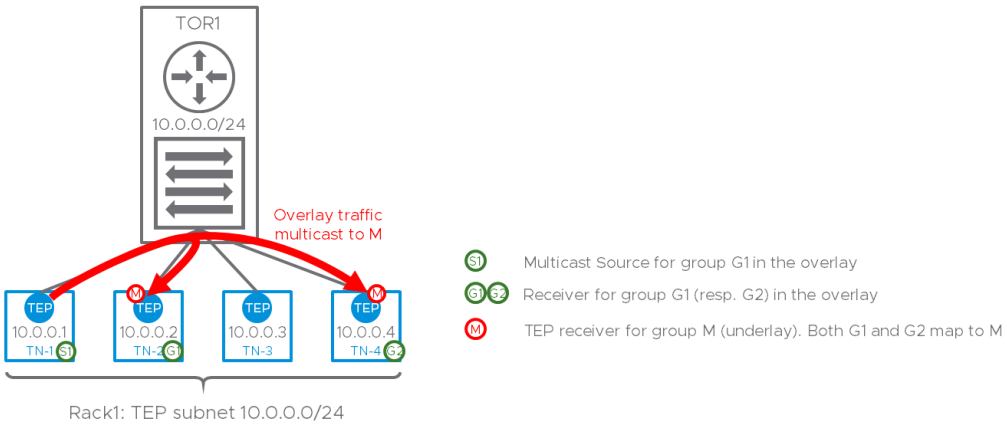


Figure 4-76: hashing collision

When TN-1 sends traffic to group G1, it's encapsulated in a multicast tunnel packet and sent to multicast address M. As a result, both TN-2 and TN-4 receive this packet. This is sub-optimal because TN-4 has no receiver for group G1. TN-4 just discards this packet. Note that, from a performance standpoint, this issue is not catastrophic. The additional replication work is done by the physical infrastructure, the cost for discarding superfluous packets it not that great on TN-4, but it's still better to minimize this scenario.

Enable IGMP snooping in the physical infrastructure

As mentioned already in this document, NSX only uses the physical infrastructure for multicast within the same Layer 2 domain. One of the reasons for this implementation choice is that we did not want NSX to be dependent on configuring multicast routing in the physical infrastructure (something that many of our customers are not able to do.) It is however beneficial to have IGMP snooping enabled on the Layer 2 switches of the physical infrastructure, in order to optimize the multicast replication within the rack.

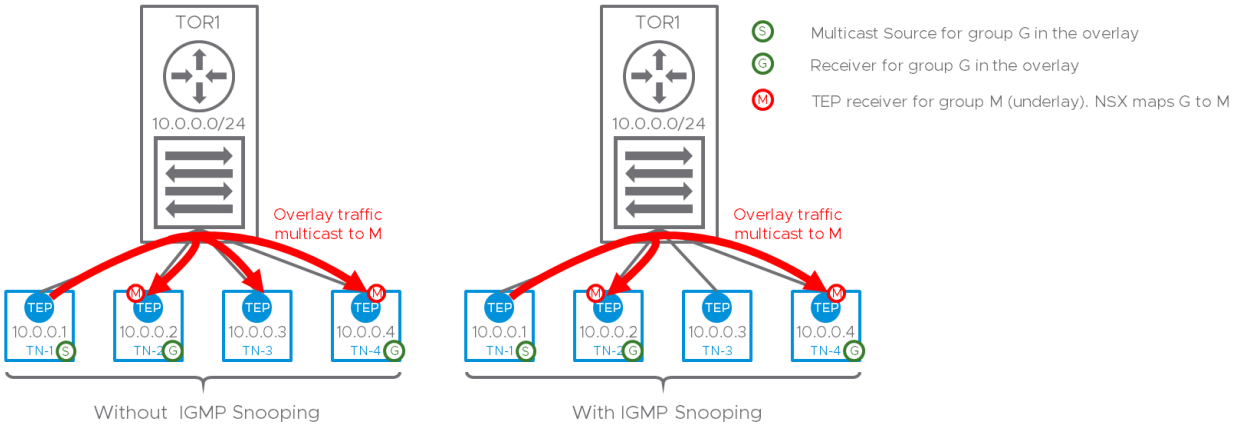


Figure 4-77: benefits of IGMP snooping

Most switches default to IGMP snooping enabled, just check that your infrastructure is configured that way. Without IGMP snooping, IP multicast is treated like broadcast by Layer 2 switches, which results in sub-optimal replication.

Limit the number of TEP subnets

This one is just a recommendation related to performance. With the hybrid replication model, NSX performs a unicast copy for each remote TEP subnets (see FIGURE 4-75: HYBRID REPLICATION.) This copy consumes CPU on the source transport node and wastes bandwidth on its uplink. The following diagram proposes a relatively extreme example comparing the replication of a single multicast frame to ten remote ESXi transport nodes. In the case represented on the top of the diagram, all the receiving transport nodes are in different subnets, in the bottom part of the diagram, the receiving transport nodes are in the same subnet as the source.

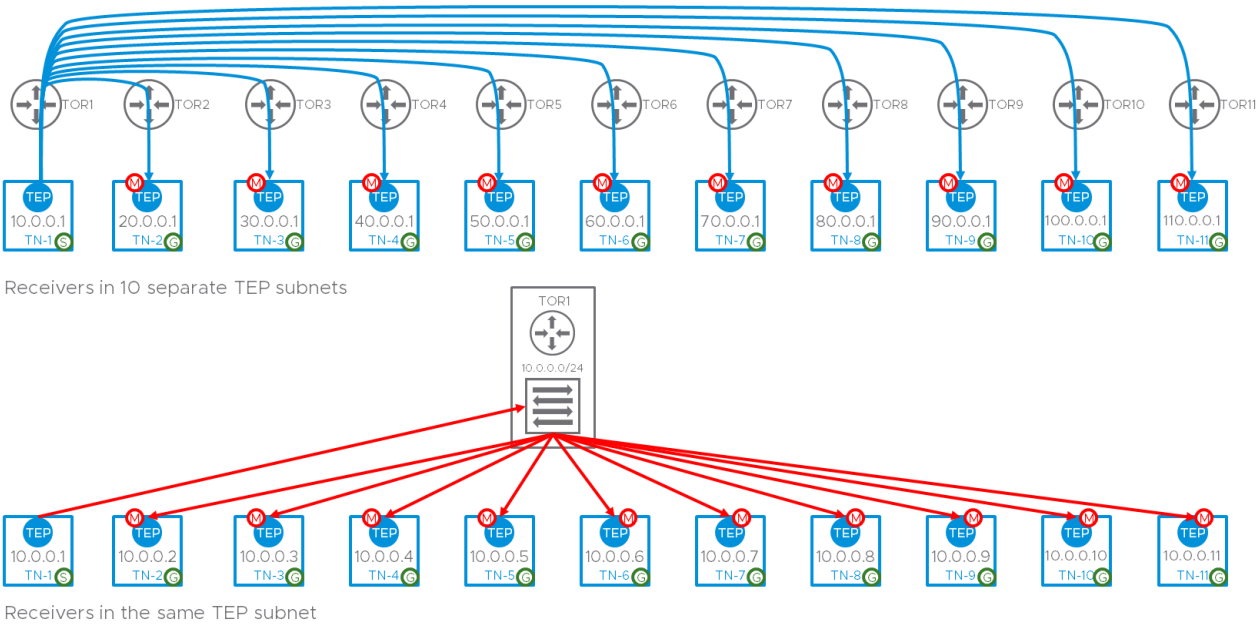


Figure 4-78: replication across multiple subnets vs. within a single subnet

In the top example, the source transport node (TN-1) needs to send 10 unicast copies of the multicast packet, one for each and every receiving transport node. Practically, it means that the bandwidth of the uplink of TN-1 has been divided by 10. There is a CPU cost to this 10-fold replication too.

In the example represented at the bottom, the source transport node just needs to send a single multicast copy of the multicast packet. It is received by the physical switch local to the rack and directly replicated to the receivers. The source transport node thus only had to send a single packet, and the burden of the replication has been entirely put on the hardware switch.

Of course, we're not recommending putting all your hosts in the same Layer 2 domain. In fact, one of the benefits of the NSX overlay is precisely that you can design your physical infrastructure without the need of extending Layer 2. The cost associated to replicating across subnets is difficult to avoid for ESXi hosts, that are bound to be spread across subnets. It might however be possible to group VMs joining the same multicast groups on hosts that are spread across a minimum number of racks.

Hybrid replication is also used between edges hosting the Tier0-SRs and Tier1-SRs. The edges of your edge cluster should not be dependent on the TORs of a single rack, but it's perfectly possible to deploy all the edges of your edge cluster across two racks (and two TEP subnets.) This will adequately reduce the amount of unicast replication that the hybrid model needs to perform.

Using Tier1-SRs in the model has consequences, minimize the number of Tier1 routers with multicast

This has already been mentioned in the chapter DATA PLANE OPERATIONS WITHIN NSX: TIER0 AND TIER1 SEGMENTS COMBINED. An ESXi transport node will receive a copy of the same multicast packet for each Tier0 and Tier1 with local receivers. The following diagram makes a simpler representation of this property:

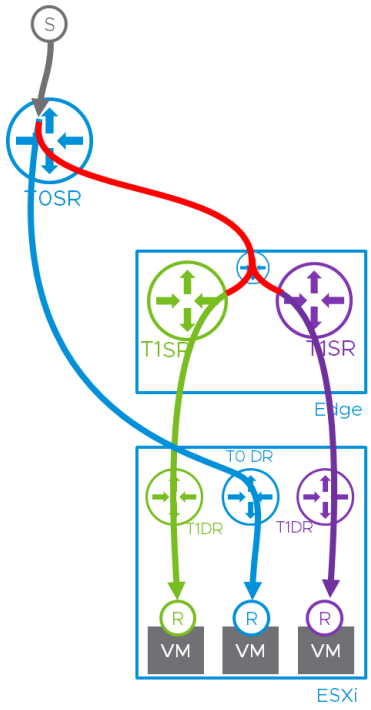


Figure 4-79: host transport node receiving multiple copies of the same multicast packet

Here, the ESXi host at the bottom receives three copies of a multicast packet generated outside NSX, behind a Tier0 gateway. The multicast trees for the Tier1 gateways are rooted in their Tier1-SRs. You can see in the diagram that the ESXi host receives the multicast packet for the local receiver under the green Tier1-DR straight from the green Tier1-SR, not from the local Tier0-DR (as it would, for unicast traffic.) This looks sub-optimal (multicast is all about avoiding the same links to carry the same multicast traffic multiple times), but it's mandatory for inserting stateful services on the Tier1 gateways, in a future release. Just be aware that multiplying Tier1 gateways with receivers for the same groups will cause NSX to replicate the same traffic multiple times.

Using Tier1-SRs in the model has consequences, be aware of a Tier1-SR impact on unicast traffic

The choice of implementing a Tier1-SR for multicast traffic also has an impact on unicast traffic: unicast traffic can only enter/exit the Tier1 routing domain through this centralized Tier1-SR, as represented in the figure below.

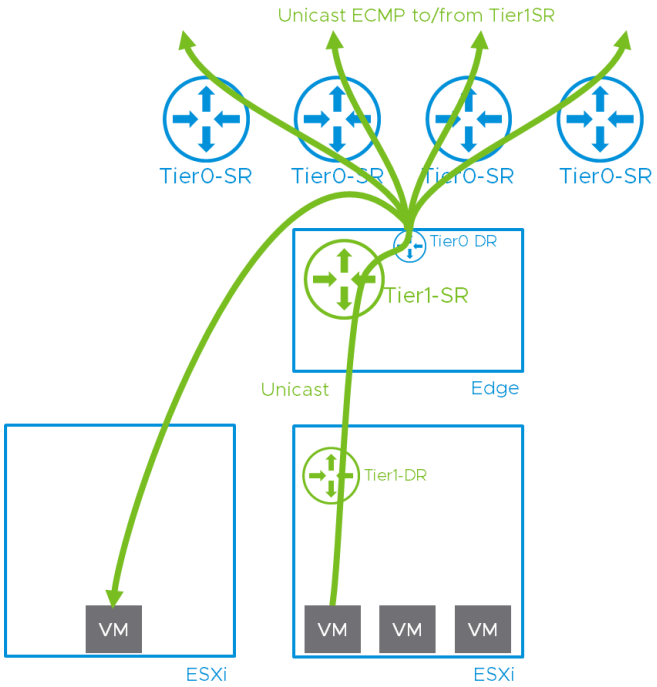


Figure 4-80: Tier1-SR impacts unicast traffic

The impact of inserting a Tier1-SR is well known, and not specific to multicast. It is still possible to have unicast ECMP straight from the host transport node to the Tier0-SRs (and the physical infrastructure) by attaching workloads directly to a segment off the Tier0 gateway.

4.8.5 Static RP vs. BSR

The definition of the rendez-vous point in NSX can be achieved by static RP configuration or through BSR. For faster convergence, we recommend using static RP. NSX allows entering multiple static RPs, each with an associated range of IP

multicast addresses. Those ranges can even be overlapping, as long as they don't use the same subnet mask.

Static RP also allows for greater control on which groups can interact with the physical infrastructure. By using static RP and disabling BSR, it is indeed possible to define exactly which groups will have an associated RP. Groups with no RP will not be able to receive/transmit multicast with the outside world.

Note also that static RP has precedence over BSR. If no RP is assigned to a group via static RP, it can still be associated to an RP via BSR (if BSR is enabled, of course.)

4.9 Edge Node

Edge nodes are service appliances with pools of capacity, dedicated to running network and security services that cannot be distributed to the hypervisors. Edge node also provides connectivity to the physical infrastructure. Previous sections mentioned that centralized services will run on the SR component of Tier-0 or Tier-1 gateways. These features include:

- Connectivity to physical infrastructure (static routing / BGP / MP-BGP)
- VRF-lite
- EVPN
- NAT
- DHCP server
- Metadata proxy
- Gateway Firewall
- NSX Native Load Balancer (No NSX Advanced Load Balancer)
- L2 Bridging
- Service Interface
- VPN
- TLS Decryption
- URL Filtering
- IDS/IPS (Also available as a Distributed Service)
- Malware Prevention (Also available as a Distributed Service)

As soon as one of these services is configured or an external interface is defined on the Tier-0 gateway, a SR is instantiated on the Edge node. The Edge node is also a transport node just like compute nodes in NSX, and like a compute node it can connect to more than one transport zones. The NSX Edge runs one or more N-VDSs managing pNICs of a bare metal edge, or the vNics of a virtual machine edge. A specific Edge node can be connected to only one overlay transport zone and depending upon the topology, is connected to one or more VLAN transport zones for N-S connectivity.

There are two transport zones on the Edge:

- Overlay Transport Zone: Any traffic that originates from a VM participating in NSX domain may require reachability to external devices or networks. This is typically described as external North-South traffic. Traffic from VMs may also require some centralized service like NAT, load balancer etc. To provide reachability for N-S traffic and to consume centralized services, overlay traffic is sent from compute transport nodes to Edge nodes. Edge node needs to be configured with a single overlay transport zone so that it can decapsulate the overlay traffic received from compute nodes as well as encapsulate the traffic sent to compute nodes. If the NSX deployment includes multiple overlay transport zones, multiple sets of edge nodes must be deployed, one for each overlay transport zone.
- VLAN Transport Zone: Edge nodes connect to the physical infrastructure using VLANs. Edge node needs to be configured for VLAN transport zone to provide external or N-S connectivity to the physical infrastructure. Depending upon the N-S topology, an edge node can be configured with one or more VLAN transport zones.

Edge node can have one or more N-VDS to provide the desired connectivity. Each N-VDS on the Edge node uses an uplink profile which can be the same or unique per N-VDS. The teaming policies defined in the uplink profile defines how the N-VDS balances traffic across its uplinks. The uplinks can in turn be individual pNICs or LAGs. While supported, implementing LAGs on edge nodes is discouraged and not included in any of the deployment examples in this guide.

4.9.1 Types of Edge Nodes

Edge nodes are available in two form factors – VM and bare metal. Both leverage the data plane development kit (DPDK) for faster packet processing and high performance. There are different VM form factors available. Each of them has a different resource footprint and can be used to achieve different guidelines. These are detailed in below table.

Size	Memory	vCPU	Disk	Specific Usage Guidelines
Small	4GB	2	200 GB	PoC only. L7 rules for firewall, load balancing are not realized on a Tier-1 gateway if you deploy a small sized NSX Edge VM.
Medium	8GB	4	200 GB	Suitable when only L2 through L4 features such as NAT, routing, L4 firewall, L4 load balancer are required and the total throughput requirement is less than 2 Gbps.

Large	32GB	8	200 GB	Suitable when only L2 through L4 features such as NAT, routing, L4 firewall, L4 load balancer and the total throughput requirement is more than 2 Gbps .
Extra Large	64GB	16	200GB	Suitable when the total throughput required is multiple Gbps for L7 load balancer and VPN .
Bare metal Edge	32GB	8	200 GB	Suitable for production with centralized services like NAT, Gateway Firewall, load balancer etc. Typically deployed, where higher performance at low packet size and sub-second N-S convergence is desired.

Table 4-2: Edge VM Form Factors and Usage Guideline

The Bare Metal Edge resources specified above specify the minimum resources needed. It is recommended to deploy an edge node on a bare metal server with the following specifications for maximum performance:

- Memory: 256GB
- CPU Cores: 24
- Disk Space: 200GB

When NSX Edge is installed as a VM, vCPUs are allocated to the Linux IP stack and DPDK. The number of vCPU assigned to a Linux IP stack or DPDK depends on the size of the Edge VM. A medium Edge VM has two vCPUs for Linux IP stack and two vCPUs dedicated for DPDK. This changes to four vCPUs for Linux IP stack and four vCPUs for DPDK in a large size Edge VM. Starting with NSX 3.0, several AMD CPUs are supported both for the virtualized and Bare Metal Edge node form factor. Specifications can be found HERE.

4.9.2 Multi-TEP support on Edge Node

Staring with NSX 2.4 release, Edge nodes support multiple overlay tunnels (multi-TEP) configuration to load balance overlay traffic for overlay segments/logical switches. Multi-TEP is supported in both Edge VM and bare metal. FIGURE 4-81 shows two TEPs configured on the bare metal Edge. Each overlay segment/logical switch is pinned to a specific tunnel end point IP, TEP IP1 or TEP IP2. Each TEP uses a different uplink, for instance, TEP IP1 uses Uplink1 that's mapped to pNIC P1 and TEP IP2 uses Uplink2 that's mapped to pNIC P2. This feature offers a better design choice by load balancing overlay traffic across both physical pNICs and also simplifies N-VDS design on the Edge.

Notice that a single N-VDS is used in this topology that carries both overlay and external traffic.

In-band management feature is leveraged for management traffic. Overlay traffic gets load balanced by using multi-TEP feature on Edge and external traffic gets load balanced using "Named Teaming policy" as described in section TEAMING POLICY in chapter 3.

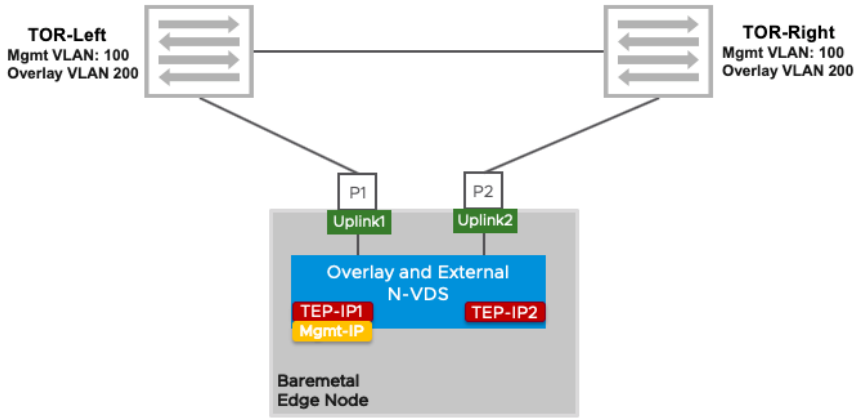


Figure 4-81: Bare metal Edge -Same N-VDS for overlay and external traffic with Multi-TEP

Following pre-requisites must be met for multi-TEP support:

- TEP configuration must be done on one N-VDS only.
- All TEPs must use same transport VLAN for overlay traffic.
- All TEP IPs must be in same subnet and use same default gateway.

During a pNIC failure, Edge performs a TEP failover by migrating TEP IP and its MAC address to another uplink. For instance, if pNIC P1 fails, TEP IP1 along with its MAC address will be migrated to use Uplink2 that's mapped to pNIC P2. In case of pNIC P1 failure, pNIC P2 will carry the traffic for both TEP IP1 and TEP IP2.

4.9.3 Bare Metal Edge Node

NSX bare metal Edge runs on a physical server and is installed using an ISO file or PXE boot.

Legacy BIOS mode is the only supported booting mode on NSXT-T 3.0. A bare metal Edge differs from the VM form factor Edge in terms of performance. It provides sub-second convergence, faster failover, and higher throughput at low packet size (discussed in chapter 8 NSX PERFORMANCE & OPTIMIZATION). There are certain hardware requirements including CPU specifics and supported NICs can be found in the NSX EDGE BARE METAL REQUIREMENTS section of the NSX installation guide.

When a bare metal Edge node is installed, a dedicated interface is retained for management. If redundancy is desired, two NICs can be used for management plane high availability. These management interfaces can also be 1G. Bare metal Edge also supports in-band management where management traffic can leverage an interface being used for overlay or external (N-S) traffic.

Bare metal Edge node supports a maximum of 16 datapath physical NICs. For each of these 16 physical NICs on the server, an internal interface is created following the naming scheme "fp-ethX". These internal interfaces are assigned to the DPDK Fast Path. There is a flexibility in assigning these Fast Path interfaces (fp-eth) for overlay or external connectivity.

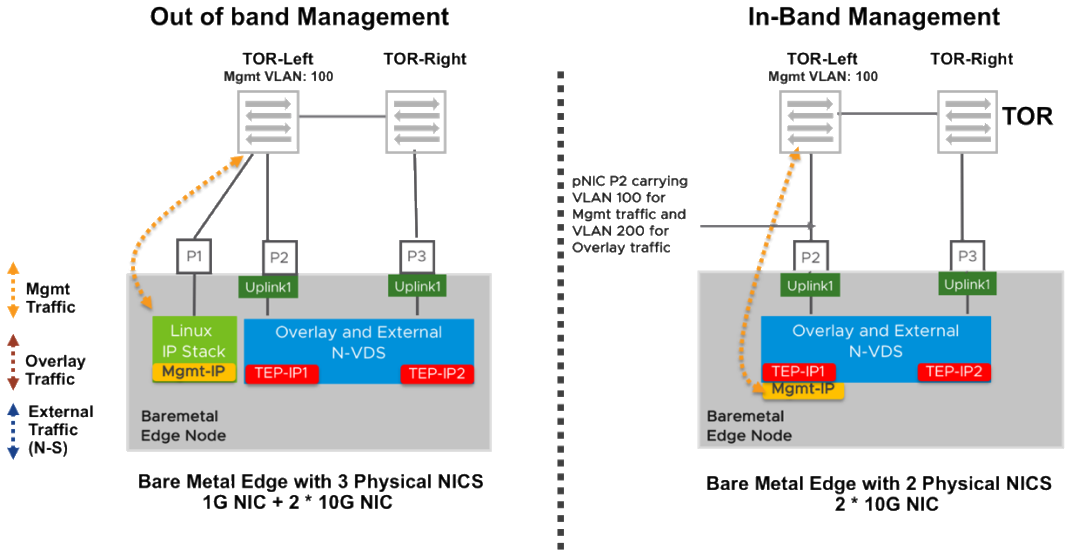


Figure 4-37: Bare metal Edge Management Configuration Choices

4.9.4 VM Edge Node

NSX VM Edge in VM form factor can be installed using an OVA, OVF, or ISO file. NSX Edge VM is only supported on ESXi host.

Up to NSX 3.2.0, an NSX Edge VM has four internal interfaces: eth0, fp-eth0, fp-eth1, and fp-eth2. Eth0 is reserved for management, while the rest of the interfaces are assigned to DPDK Fast Path. Starting with NSX 3.2.1, four datapath interfaces are available: fp-eth0, fp-eth1, fp-eth2, and fp-eth3.

These interfaces are allocated for external connectivity to TOR switches and for NSX overlay tunneling. There is complete flexibility in assigning Fast Path interfaces (fp-eth) for overlay or external connectivity. As an example, fp-eth0 could be assigned for overlay traffic with fp-eth1, fp-eth2, or both for external traffic.

FIGURE 4-82 shows an edge VM where only two of the fast path interfaces are in use. They are managed by the same NVDS and carry both overlay and VLAN traffic. This design is in line with the edge reference architecture and will be explained in detailed in chapter 7 section EDGE CONNECTIVITY GUIDELINES FOR LAYER 3 PEERING USE CASE.

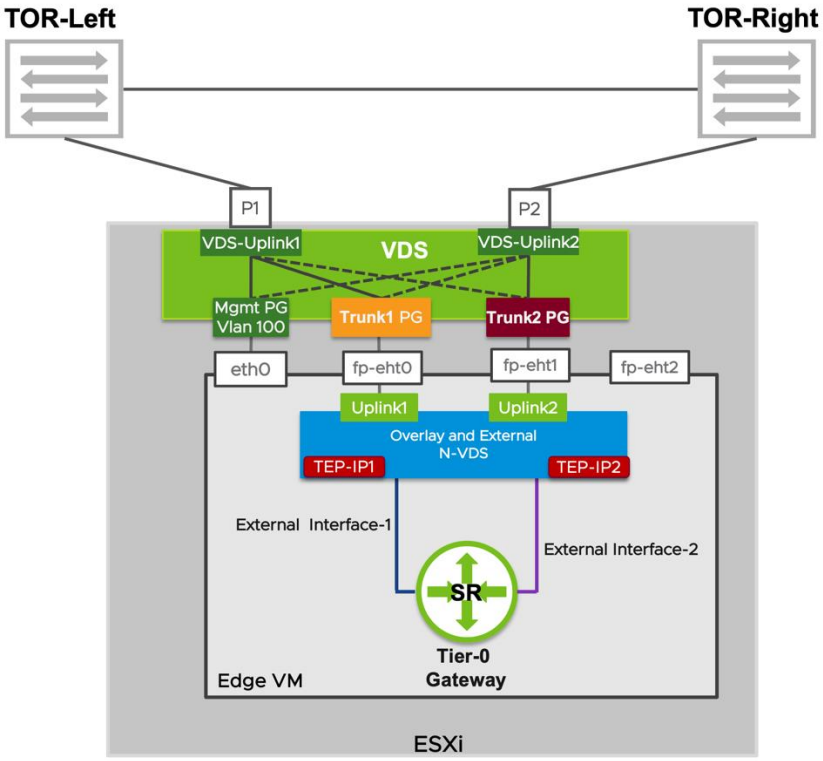


Figure 4-82: Edge VM

4.9.5 Edge Cluster

An Edge cluster is a group of Edge transport nodes. It provides scale out, redundant, and high-throughput gateway functionality for logical networks. Scale out from the logical networks to the Edge nodes is achieved using ECMP. There is a flexibility in assigning Tier-0 or Tier-1 gateways to Edge nodes and clusters. Tier-0 and Tier-1 gateways can be hosted on either same or different Edge clusters.

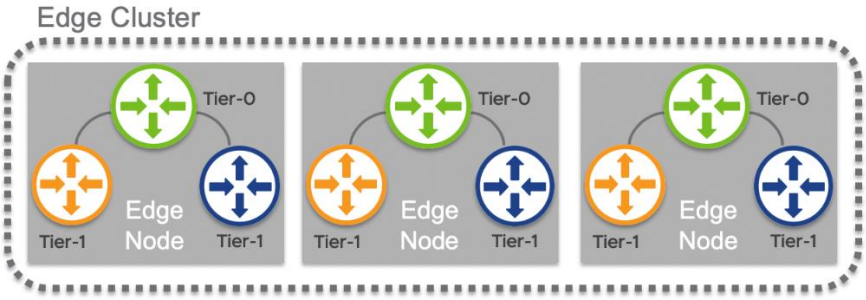


Figure 4-83: Edge Cluster with Tier-0 and Tier 1 Services

Depending upon the services hosted on the Edge node and their usage, an Edge cluster could be dedicated simply for running centralized services (e.g., NAT). FIGURE 4-84 shows two clusters of Edge nodes. Edge Cluster 1 is dedicated for Tier-0 gateways only and provides external connectivity to the physical infrastructure. Edge Cluster 2 is responsible for NAT functionality on Tier-1 gateways.

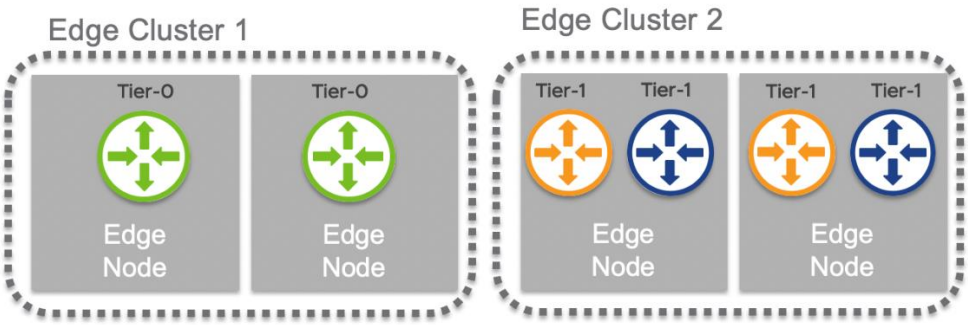


Figure 4-84: Multiple Edge Clusters with Dedicated Tier-0 and Tier-1 Services

There can be only one Tier-0 gateway per Edge node; however, multiple Tier-1 gateways can be hosted on one Edge node.

A Tier-0 gateway supports a maximum of eight equal cost paths per SR or DR component, thus a maximum of eight Edge nodes are supported for ECMP. Edge nodes in an Edge cluster run Bidirectional Forwarding Detection (BFD) on both tunnel and management networks to detect Edge node failure. Edge VMs support BFD with minimum BFD timer of 500ms with three retries, providing a 1.5 second failure detection time. Bare metal Edges support BFD with minimum BFD TX/RX timer of 50ms with three retries which implies 150ms failure detection time.

4.9.6 Failure Domain

Failure domain is a logical grouping of Edge nodes within an Edge Cluster. This feature can be enabled on the Edge cluster level via API.

As discussed in high availability section, a Tier-1 gateway with centralized services runs on Edge nodes in active/standby HA configuration mode. When a user assigns a Tier-1 gateway to an Edge cluster, NSX manager automatically chooses the Edge nodes in the cluster to run the active and standby Tier-1 SR. The auto placement of Tier-1 SRs on different Edge nodes considers several parameters like Edge capacity, active/standby HA state etc.

Failure domains compliment auto placement algorithm and guarantee service availability in case of a failure affecting multiple edge nodes. Active and standby instance of a Tier-1 SR always run in different failure domains.

FIGURE 4-85 shows an edge cluster comprised of four Edge nodes, EN1, EN2, EN3 and EN4. EN1 and EN2 connected to two TOR switches in rack 1 and EN3 and EN4 connected to two TOR switches in rack 2. Without failure domain, a Tier-1 SR could be auto placed in EN1 and EN2. If rack1 fails, both active and standby instance of this Tier-1 SR fail as well.

EN1 and EN2 are configured to be a part of failure domain 1, while EN3 and EN4 are in failure domain 2. When a new Tier-1 SR is created and if the active instance of that Tier-1 is hosted on EN1, then the standby Tier-1 SR will be instantiated in failure domain 2 (EN3 or EN4).

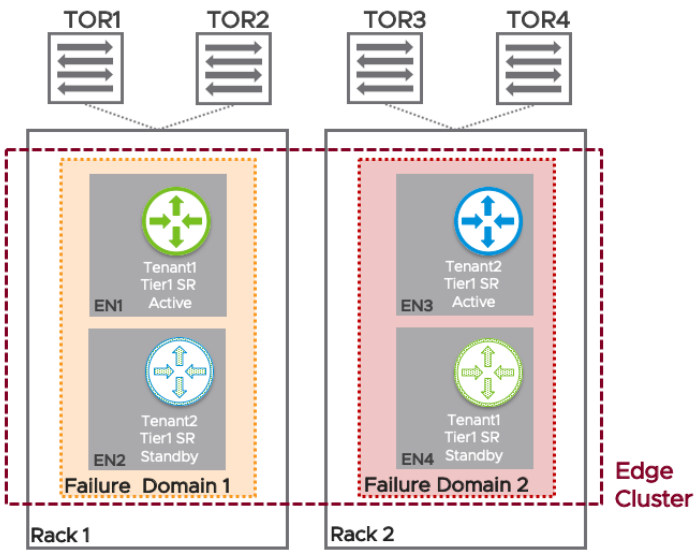


Figure 4-85: Failure Domains

To ensure that all Tier-1 services are active on a set of edge nodes, a user can also enforce that all active Tier-1 SRs are placed in one failure domain. This configuration is supported for Tier-1 gateway in preemptive mode only.

4.10 Other Network Services

4.10.1Unicast Reverse Path Forwarding (uRPF)

A router forwards packets based on the value of the destination IP address field that is present in the IP header. The source IP address field is generally not used when forwarding a packet on a network (except when networks implement source-based routing).

Unicast Reverse Path Forwarding is defined in RFC 2827 and 3704. Prevent packets with spoofed source IP address to be forwarded in the network. uURPF is generally enabled on a router per interface and not globally. A uRPF enabled router A router will inspect the source IP address of theevery packet of each packet received on an interface. It will validates that packets are coming from a legitimate source by inspecting the routing table. The This protection prevents spoofed source IP address attacks that are commonly used by sending packets with random source IP addresses. When a packet arrives on an interface, the router will verify if the receiving interface would be used to reach the source of the packet. It will discard the packets if the interfaces are different.

This protection prevents spoofed source IP address attacks that are commonly used by sending packets with random source IP addresses.

FIGURE 4-86 diagrams a physical network with URPF enabled on the core router.

1. The core router receives a packet with a source IP address of 10.1.1.1 on interface ethernet0/2.
2. The core router has the URPF feature enabled on all its interfaces and will check in its routing table if the source IP address of the packet would be routed through interface ethernet 0/2. In this case, 10.1.1.1 is the source IP

address present in the IP header. The core router has a longest prefix match for 10.1.1.0/24 via interface ethernet 0/0.

3. Since the packet does not come from the interface ethernet 0/0, the packet will be discarded.

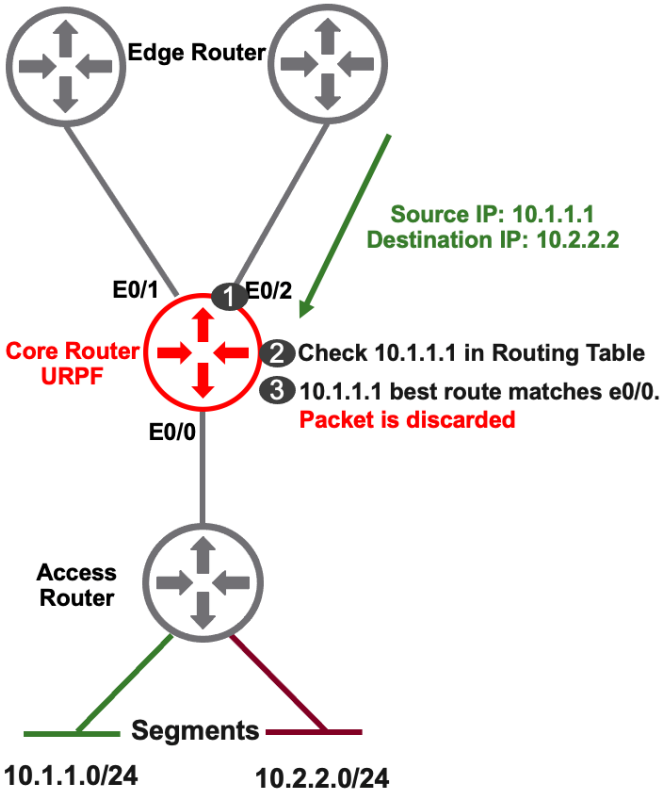


Figure 4-86: URPF

In NSX, URPF is enabled by default on external, internal and service interfaces. From a security standpoint, it is a best practice to keep uRPF enabled on these interfaces. uRPF is also recommended in architectures that leverage ECMP. On intra-tier and router link interfaces, a simplified anti-spoofing mechanism is implemented. It is checking that a packet is never sent back to the interface the packet was received on.

It is possible to disable uRPF in complex routing architecture where the upstream BGP or OSPF peers do not advertise the same networks.

4.10.2 Network Address Translation

Users can enable NAT as a network service on NSX. This is a centralized service which can be enabled on both Tier-0 and Tier-1 gateways.

Supported NAT rule types include:

- Source NAT (SNAT): Source NAT translates the source IP of the outbound packets to a known public IP address so that the application can

communicate with the outside world without using its private IP address. It also keeps track of the reply.

- **Destination NAT (DNAT)**: DNAT allows for access to internal private IP addresses from the outside world by translating the destination IP address when inbound communication is initiated. It also takes care of the reply. For both SNAT and DNAT, users can apply NAT rules based on 5 tuple match criteria.
- **Reflexive NAT:** Reflexive NAT rules are stateless ACLs which must be defined in both directions. These do not keep track of the connection. Reflexive NAT rules can be used in cases where stateful NAT cannot be used due to asymmetric paths (e.g., user needs to enable NAT on active/active ECMP routers).

Table 4-3 summarizes NAT rules and usage restrictions.

NAT Rules Type	Type	Specific Usage Guidelines
Stateful	Source NAT (SNAT) Destination NAT (DNAT)	Can be enabled on both Tier-0 and Tier-1 gateways
Stateless	Reflexive NAT	Can be enabled on Tier-0 gateway; generally used when the Tier-0 is in active/active mode.

Table 4-3: NAT Usage Guideline

Table 4-4 summarizes the use cases and advantages of running NAT on Tier-0 and Tier-1 gateways.

Gateway Type	NAT Rule Type	Specific Usage Guidelines
Tier-0	Stateful	E-W routing between different tenants remains completely distributed.
Tier-1	Stateful	Recommended for high throughput ECMP topologies. Recommended for topologies with overlapping IP address space.

Table 4-4: Tier-0 and Tier-1 NAT use cases

NAT Service Router Placement

As a centralized service, whenever NAT is enabled, a service component or SR must be instantiated on an Edge cluster. In order to configure NAT, specify the Edge cluster where the service should run; it is also possible the NAT service on a specific Edge node pair. If no specific Edge node is identified, the platform will perform auto placement of the services component on an Edge node in the cluster using a weighted round robin algorithm.

4.10.3 DHCP Services

NSX provides both DHCP relay and DHCP server functionality. DHCP relay can be enabled at the gateway level and can act as relay between non-NSX managed environment and DHCP servers. DHCP server functionality can be enabled to service DHCP requests from VMs connected to NSX-managed segments. DHCP server functionality is a stateful service and must be bound to an Edge cluster or a specific pair of Edge nodes as with NAT functionality.

4.10.4 Metadata Proxy Service

With a metadata proxy server, VM instances can retrieve instance-specific metadata from an OpenStack Nova API server. This functionality is specific to OpenStack use-cases only. Metadata proxy service runs as a service on an NSX Edge node. For high availability, configure metadata proxy to run on two or more NSX Edge nodes in an NSX Edge cluster.

4.10.5 Gateway Firewall Service

Gateway Firewall service can be enabled on the Tier-0 and Tier-1 gateway for North-South firewalling. Table 4-5 summarizes Gateway Firewalling usage criteria.

Gateway Firewall	Specific Usage Guidelines
Stateful	Can be enabled on both Tier-0 and Tier-1 gateways.
Stateless	Can be enabled on Tier-0 gateway; generally used when the Tier-0 is in active/active mode.

Table 4-5: Gateway Firewall Usage Guideline

Since Gateway Firewalling is a centralized service, it needs to run on an Edge cluster or a set of Edge nodes. This service is described in more detail in the security chapter 5.

4.10.6 Proxy ARP

Proxy ARP is a method that consist of answering an ARP request on behalf of another host. This method is performed by a layer 3 networking device (usually a router). The purpose is to provide connectivity between 2 hosts when routing wouldn't be possible for various reasons.

Proxy ARP in an NSX infrastructure can be considered in environments where IP subnets are limited. Proof of concepts and VMware Enterprise PKS environments are usually using Proxy-ARP to simplify the network topology.

For production environment, it is recommended to implement proper routing between a physical fabric and the NSX Tier-0 by using either static routes or Border Gateway Protocol with BFD. If proper routing is used between the Tier-0 gateway and the physical fabric, BFD with its sub-second timers will converge faster. In case of failover with proxy ARP, the convergence relies on gratuitous ARP (broadcast) to update all hosts on the VLAN segment with the new MAC Address to use. If the Tier-0 gateway has proxy ARP enabled for 100 IP addresses, the newly active Tier-0 SR needs to send 100 Gratuitous ARP packets.

Edge Node HA	Specific Usage Guidelines
Active / Standby	Supported
Active / Active	Not supported

Table 4-6: Proxy ARP Support

By enabling proxy-ARP, hosts on the overlay segments and hosts on a VLAN segment can exchange network traffic together without implementing any change in the physical networking fabric. Proxy ARP is automatically enabled when a NAT rule or a load balancer VIP uses an IP address from the subnet of the Tier-0 gateway uplink.

FIGURE 4-87 presents the logical packet flow between a virtual machine connected to an NSX overlay segment and a virtual machine or physical appliance connected to a VLAN segment shared with the NSX Tier-0 uplinks.

In this example, the virtual machine connected to the overlay segment initiates networking traffic toward 20.20.20.100.

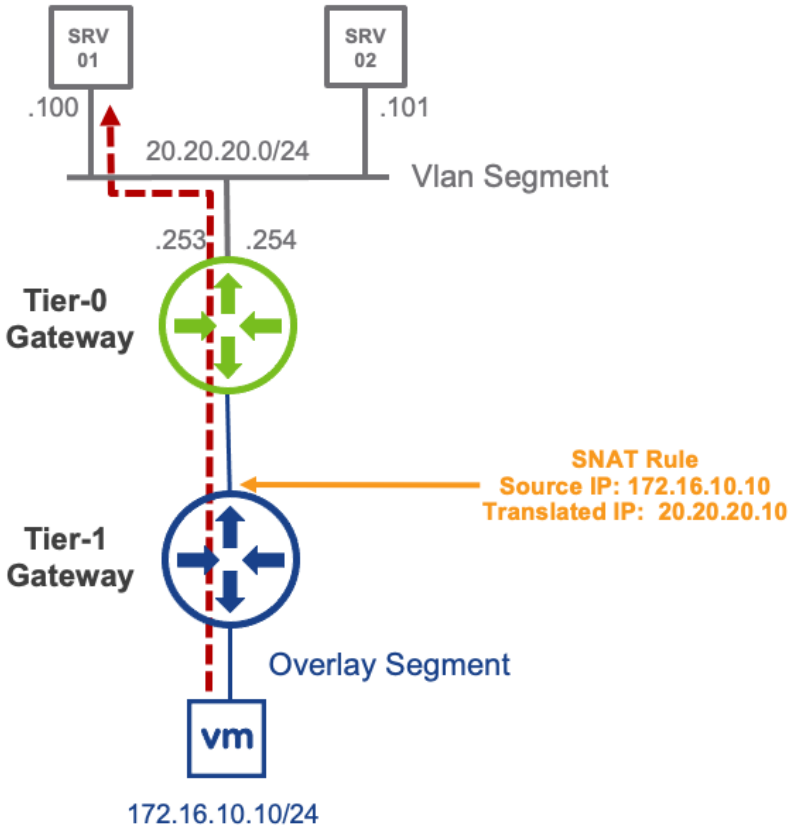


Figure 4-87: Proxy ARP topology

1. The virtual machine connected to the overlay segment with an IP address of 172.16.10.10 sends a packet to the physical appliance "SRV01" with an IP address of 20.20.20.100. the local DR hosted on the local hypervisor performs a routing lookup and sends the traffic to the SR.
2. The SR hosted on an edge node translates the source IP of 172.16.10.10 with a value of 20.20.20.10 and sends the traffic to the Tier-0.
3. Tier-0 SR has proxy ARP enabled on its uplink interface and will send an ARP request (broadcast) on the vlan segment to map the IP address of 20.20.20.100 with the correct MAC address.
4. The physical appliance "SRV01" answers to that ARP request with an ARP reply.
5. Tier-0 SR sends the packet to the physical appliance with a source IP of 20.20.20.10 and a destination IP of 20.20.20.100.
6. The physical appliance "SRV01" receives the packet and sends an ARP broadcast on the VLAN segment to map the IP address of the virtual machine (20.20.20.10) to the corresponding MAC address.
7. Tier-0 receives the ARP request for 20.20.20.10 (broadcast) and has the proxy ARP feature enabled on its uplink interfaces. It replies to the ARP request with an ARP reply that contains the Tier-0 SR MAC address for the interface uplink.

8. The physical appliance "SRV01" receives the ARP request and sends a packet on the vlan segment with a source IP of 20.20.20.100 and a destination IP of 20.20.20.10.
9. The packet is being received by the Tier-0 SR and is being routed to the Tier-1 who does translate the Destination IP of 20.20.20.10 with a value of 172.16.10.10. Packet is sent to the overlay segment and the virtual machine receives it.

It is crucial to note that in this case, the traffic is initiated by the virtual machine which is connected to the overlay segment on the Tier-1. If the initial traffic was initiated by a server on the VLAN segment, a Destination NAT rule would have been required on the Tier-1/Tier-0 since the initial traffic would not match the SNAT rule that has been configured previously.

FIGURE 4-88 represents an outage on an active Tier-0 gateway with Proxy ARP enabled. The newly active Tier-0 gateway will send a gratuitous ARP to announce the new MAC address to be used by the hosts on the VLAN segment in order to reach the virtual machine connected to the overlay. It is critical to fathom that the newly active Tier-0 will send a Gratuitous ARP for each IP address that are configured for Proxy ARP.

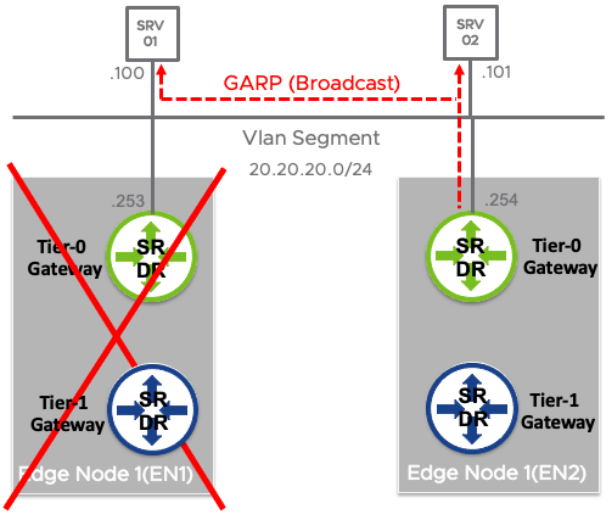


Figure 4-88: Edge node failover and Proxy ARP

4.11 Topology Consideration

This section covers a few of the many topologies that customers can build with NSX. NSX routing components - Tier-1 and Tier-0 gateways - enable flexible deployment of multi-tiered routing topologies. Topology design also depends on what services are enabled and where those services are provided at the provider or tenant level.

4.11.1 Supported Topologies

FIGURE 4-89 shows three topologies with Tier-0 gateway providing N-S traffic connectivity via multiple Edge nodes. The first topology is single-tiered where

Tier-0 gateway connects directly to the segments and provides E-W routing between subnets. Tier-0 gateway provides multiple active paths for N-S L3 forwarding using ECMP. The second topology shows the multi-tiered approach where Tier-0 gateway provides multiple active paths for L3 forwarding using ECMP and Tier-1 gateways as first hops for the segments connected to them. Routing is fully distributed in this multi-tier topology. The third topology shows a multi-tiered topology with Tier-0 gateway configured in Active/Standby HA mode to provide some centralized or stateful services like NAT, VPN etc.

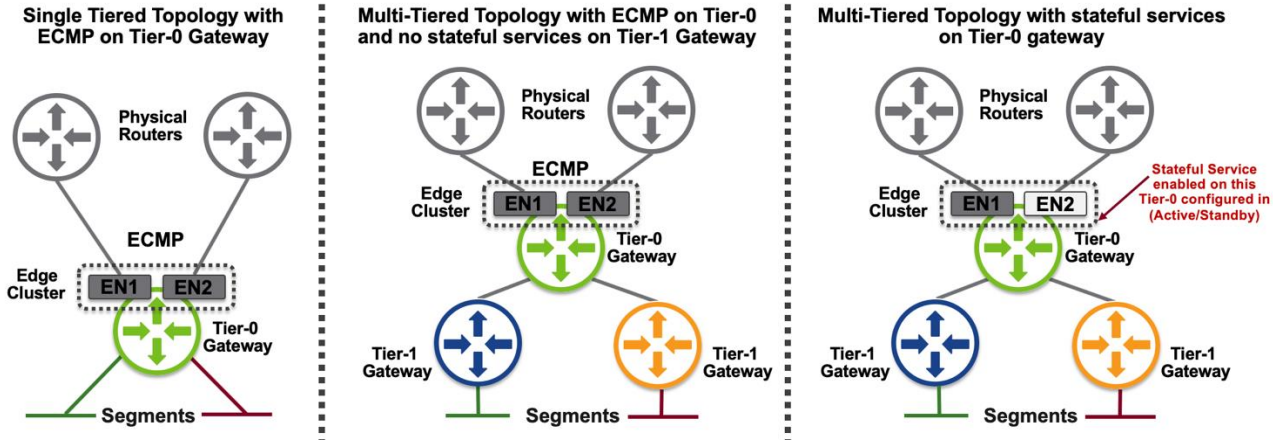


Figure 4-89: Single tier and multi-tier routing topologies

As discussed in the two-tier routing section, centralized services can be enabled on Tier-1 or Tier-0 gateway level. FIGURE 4-90 shows two multi-tiered topologies.

The first topology shows centralized services like NAT, load balancer on Tier-1 gateways while Tier-0 gateway provides multiple active paths for L3 forwarding using ECMP.

The second topology shows centralized services configured on a Tier-1 and Tier-0 gateway. Some centralized services are only available on Tier-1.

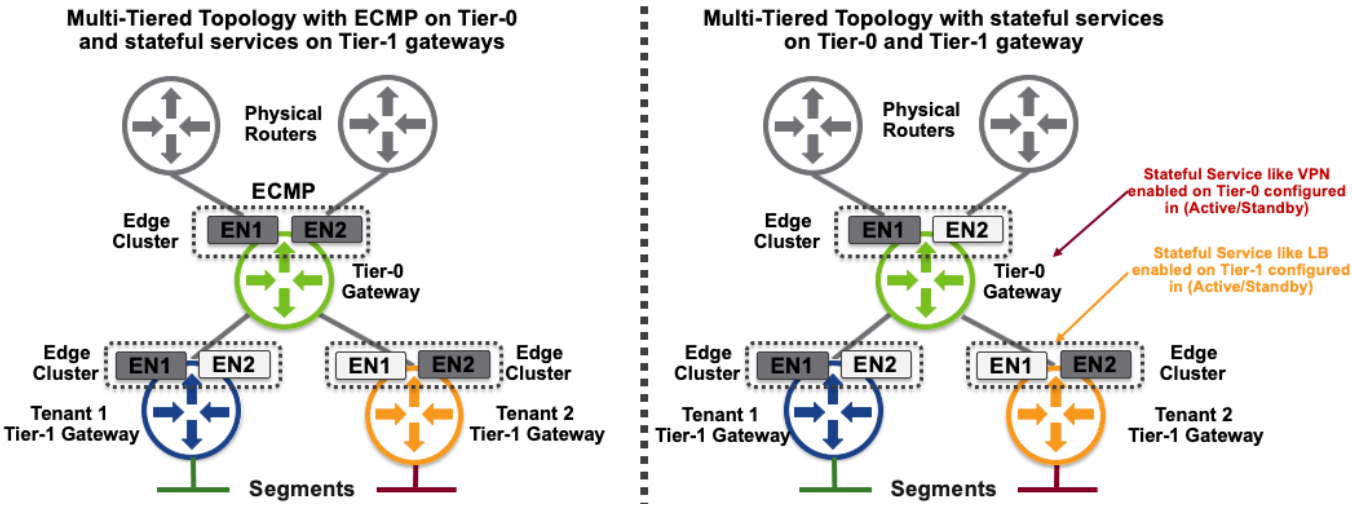


Figure 4-90: Stateful and Stateless (ECMP) Services Topologies Choices at Each Tier

FIGURE 4-91 shows a topology with Tier-0 gateways connected back to back. "Tenant-1 Tier-0 Gateway" is configured for a stateful firewall while "Tenant-2 Tier-0 Gateway" has stateful NAT configured. Since stateful services are configured on both "Tenant-1 Tier-0 Gateway" and "Tenant-2 Tier-0 Gateway", they are configured in Active/Standby high availability mode. The top layer of Tier-0 gateway, "Aggregate Tier-0 Gateway" provides ECMP for North-South traffic. Note that only external interfaces should be used to connect a Tier-0 gateway to another Tier-0 gateway. Static routing and BGP are supported to exchange routes between two Tier-0 gateways and full mesh connectivity is recommended for optimal traffic forwarding. This topology provides high N-S throughput with centralized stateful services running on different Tier-0 gateways. This topology also provides complete separation of routing tables on the tenant Tier-0 gateway level and allows services that are only available on Tier-0 gateways (like VPN with redundant remote peers) to leverage ECMP northbound. Note that VPN is available on Tier-1 gateways starting NSX 2.5 release. NSX 3.0 introduces new multi tenancy features such as EVPN and VRF-lite. These features are recommended and suitable for true multi-tenant architecture where stateful services need to be run on multiple layers or Tier-0

Full mesh connectivity is recommended for optimal traffic forwarding.

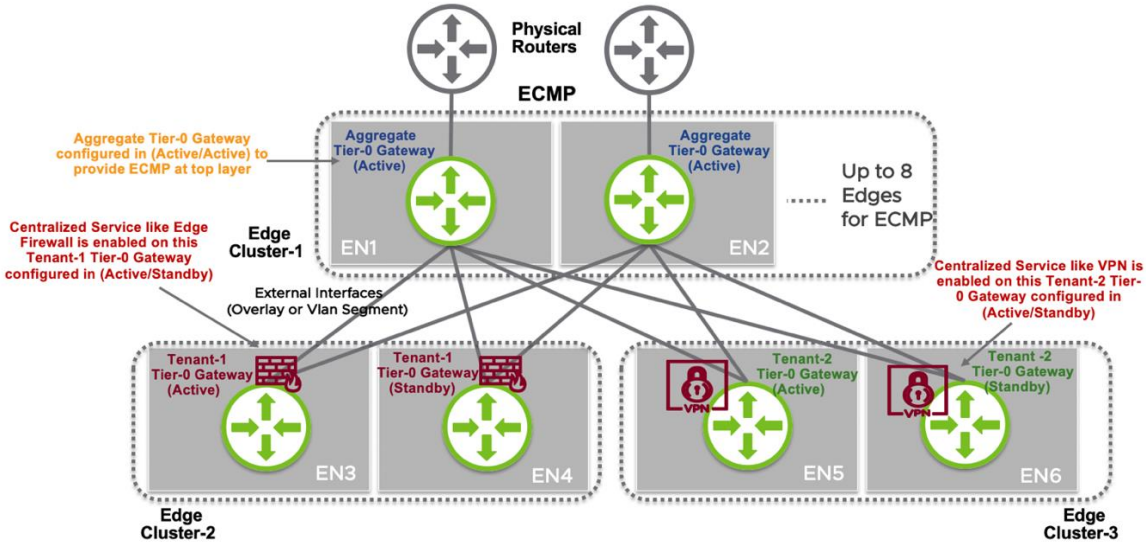


Figure 4-91: Multiple Tier-0 Topologies with Stateful and Stateless (ECMP) Services

FIGURE 4-92 shows another topology with Tier-0 gateways connected back-to-back. "Corporate Tier-0 Gateway" on Edge cluster-1 provides connectivity to the corporate resources (172.16.0.0/16 subnet) learned via a pair of physical routers on the left. This Tier-0 has stateful Gateway Firewall enabled to allow access to restricted users only.

"WAN Tier-0 Gateway" on Edge-Cluster-2 provides WAN connectivity via WAN routers and is also configured for stateful NAT.

"Aggregate Tier-0 gateway" on the Edge cluster-3 learns specific routes for corporate subnet (172.16.0.0/16) from "Corporate Tier-0 Gateway" and a default route from "WAN Tier-0 Gateway". "Aggregate Tier-0 Gateway" provides ECMP for both corporate and WAN traffic originating from any segments connected to

it or connected to a Tier-1 southbound. Full mesh connectivity is recommended for optimal traffic forwarding.

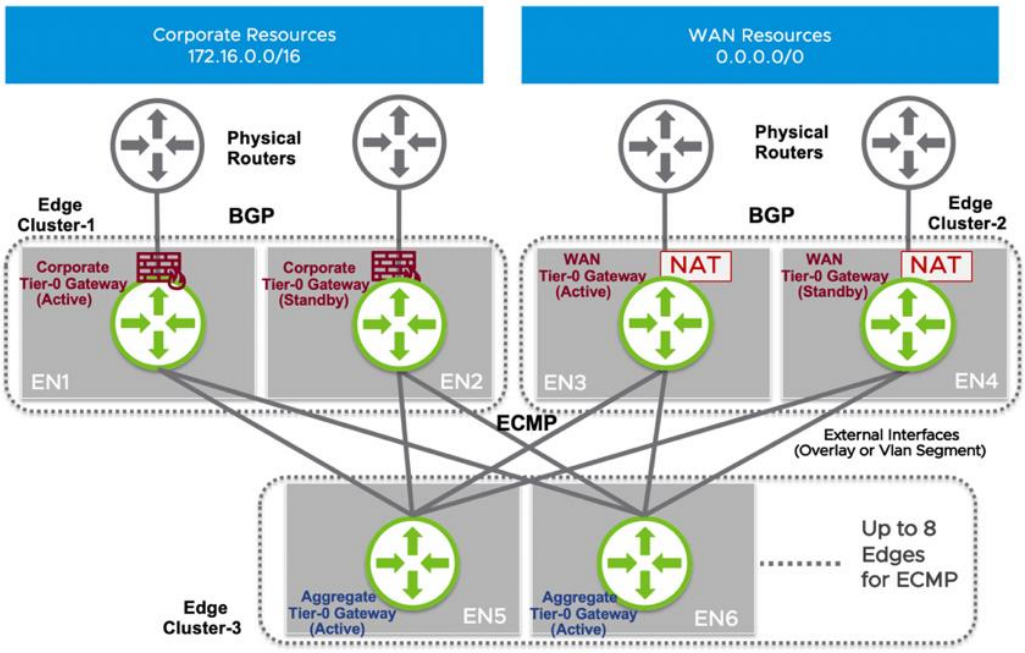


Figure 4-92: Multiple Tier-0 Topologies with Stateful and Stateless (ECMP) Services

A Tier-1 gateway usually connects to a Tier-0 gateway but it is possible for some use cases to interconnect it to another Tier-1 gateway using service interfaces (SI) and downlink as depicted in FIGURE 4-93. Static routing must be configured on both Tier-1 gateways in this case as dynamic routing is not supported. The Tier-0 gateway must be aware of the all the overlay segments prefixes to provide connectivity.

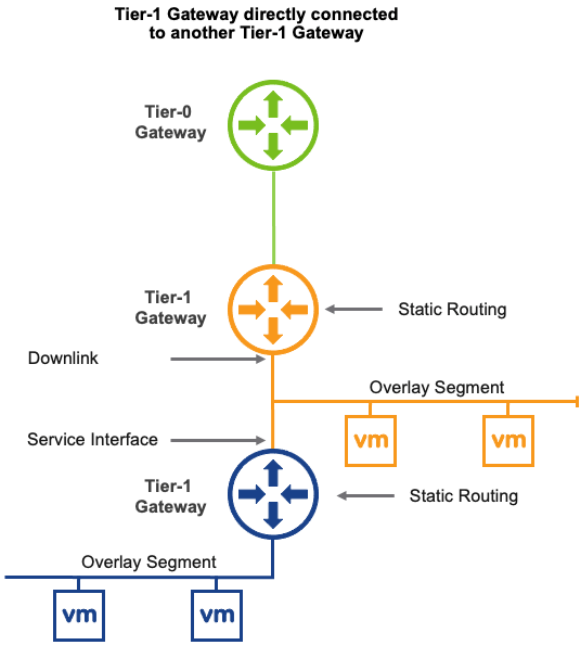


Figure 4-93: Supported Topology – T1 gateways interconnected to each other using Service Interface and Downlink

4.11.2 Unsupported Topologies

While the deployment of logical routing components enables customers to deploy flexible multi-tiered routing topologies, FIGURE 4-94 presents topologies that are not supported. The topology on the left shows that a tenant Tier-1 gateway cannot be connected directly to another tenant Tier-1 gateway using downlinks exclusively.

The rightmost topology highlights that a Tier-1 gateway cannot be connected to two different upstream Tier-0 gateways.

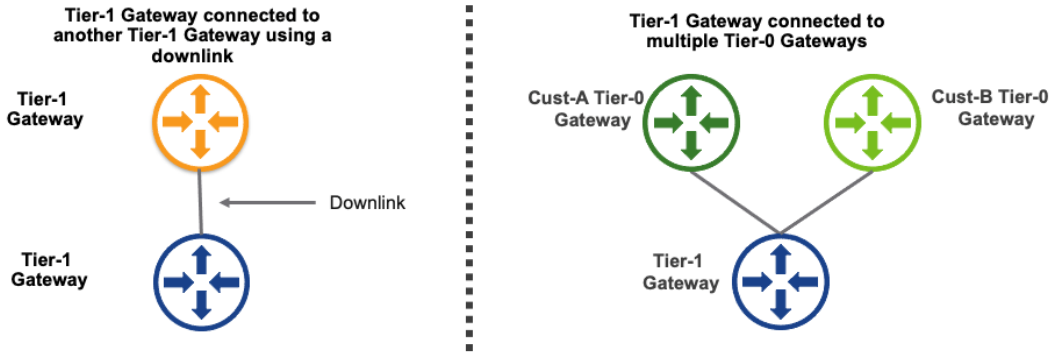


Figure 4-94: Unsupported Topologies

5 NSX Security

In addition to providing network virtualization, NSX also serves as an advanced security platform, providing a rich set of features to streamline the deployment of security solutions. This chapter focuses on core NSX security capabilities, architecture, components, and implementation. Key concepts for examination include:

- NSX distributed firewall (DFW) provides stateful protection of the workload at the vNIC level. For ESXi, the DFW enforcement occurs in the hypervisor kernel, helping deliver micro-segmentation. However, the DFW extends to physical servers, KVM hypervisors, containers, and public clouds providing distributed policy enforcement.
- Uniform security policy model for on-premises and cloud deployment, supporting multi-hypervisor (i.e., ESXi and KVM) and multi-workload, with a level of granularity down to VM/containers/bare metal attributes.
- Agnostic to compute domain - supporting hypervisors managed by different compute-managers while allowing any defined micro-segmentation policy to be applied across hypervisors spanning multiple vCenter environments.
- Support for Layer 3, Layer 4, Layer-7 APP-ID, & Identity based firewall policies provide security via protocol, port, and or deeper packet/session intelligence to suit diverse needs.
- NSX Gateway firewall serves as a centralized stateful firewall service for N-S traffic. Gateway firewall is implemented per gateway and supported at both Tier-0 and Tier-1. Gateway firewall is independent of NSX DFW from policy configuration and enforcement perspective, providing a means for defining perimeter security control in addition to distributed security control.
- Gateway & Distributed Firewall Service insertion capability to integrate existing security investments using integration with partner ecosystem products on a granular basis without the need for interrupting natural traffic flows.
- Distributed IDS extends IDS capabilities to every host in the environment.
- Dynamic grouping of objects into logical constructs called Groups based on various criteria including tag, virtual machine name or operating system, subnet, and segments which automates policy application.
- The scope of policy enforcement can be selective, with application or workload-level granularity.
- Firewall Flood Protection capability to protect the workload & hypervisor resources.
- IP discovery mechanism dynamically identifies workload addressing.
- SpoofGuard blocks IP spoofing at vNIC level.
- Switch Security provides storm control and security against unauthorized traffic.

NSX 3.2 introduces advanced security features such as security on vCenter dvpgs, distributed and centralized malware protection, centralized IDS/IPS, URL Filtering, extensive next generation firewall App Identification support, Network Traffic Anomaly Detection, and Network Detection and Response (NDR). These new features will be covered in the new version if the SECURITY DESIGN GUIDE. This document will cover the NSX core security feature.

5.1 NSX Security Use Cases

The NSX security platform is designed to address the security challenges faced by IT admins. Although it started with firewalling, the NSX security feature set has since grown to encompass Identity Firewalling, IPS, and many more. The NSX firewall is delivered as part of a distributed platform that offers ubiquitous enforcement, scalability, line rate performance, multi-hypervisor support, and API-driven orchestration. These fundamental pillars of the NSX firewall allow it to address many different use cases for production deployment.

One of the leading use cases NSX supports is micro-segmentation. Micro-segmentation enables an organization to logically divide its data center into distinct security segments down to the individual workload level, then define distinct security controls for and deliver services to each unique segment. This is all possible without the need to change underlying network architecture or addressing. A central benefit of micro-segmentation is its ability to deny attackers the opportunity to pivot laterally within the internal network, even after the perimeter has been breached.

VMware NSX supports micro-segmentation as it allows for a centrally controlled, operationally distributed firewall to be attached directly to workloads within an organization's network. The distribution of the firewall for the application of security policy to protect individual workloads is highly efficient; rules can be applied that are specific to the requirements of each workload. Of additional value is that NSX's capabilities are not limited to homogeneous vSphere environments. NSX supports the heterogeneity of platforms and infrastructure that is common in organizations today.

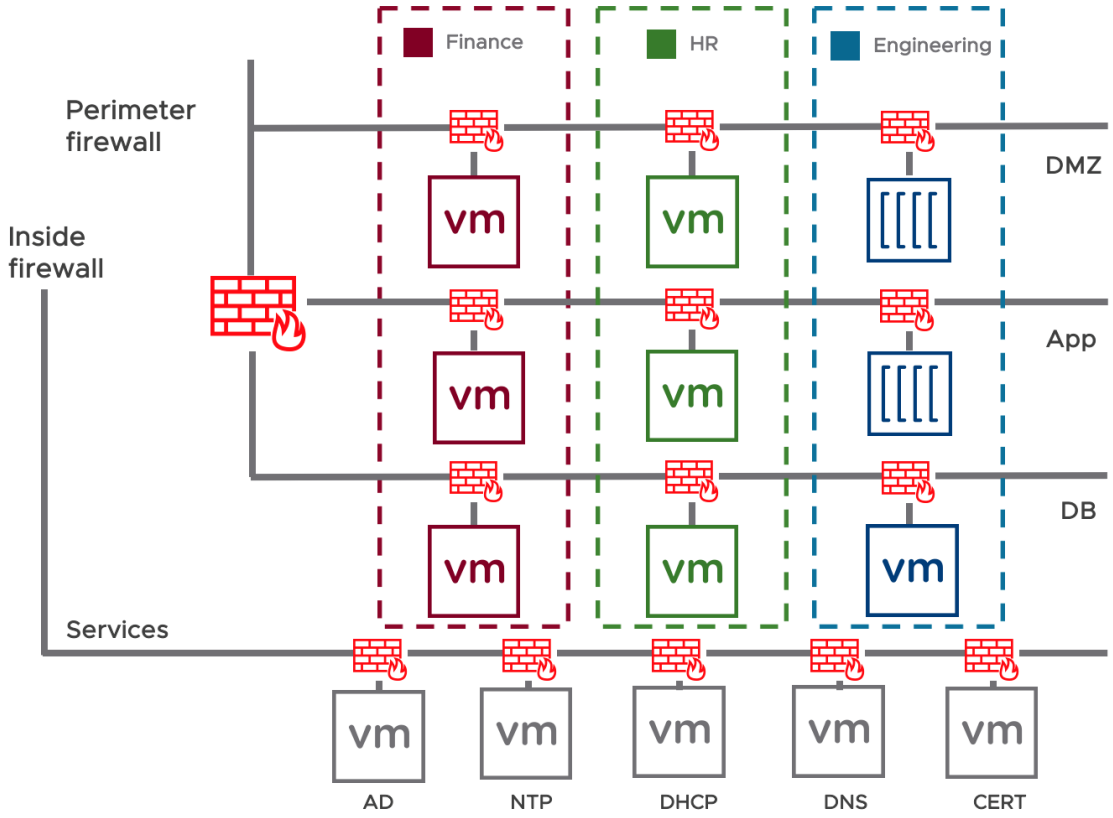


Figure 5-1: Example of Micro-segmentation with NSX

Micro-segmentation provided by NSX is an essential element of zero trust, specifically it embodies "making the *access control enforcement as granular as possible*." (NIST ZTA publication). It establishes a security perimeter around each VM or container workload with a dynamically defined policy which can be down to the user level of granularity. Legacy security models assume that everything on the inside of an organization's network can be trusted; zero-trust assumes the opposite - trust nothing and verify everything. This addresses the increased sophistication of networks attacks and insider threats that frequently exploit the conventional perimeter-controlled approach. For each system in an organization's network, trust of the underlying network is removed. A perimeter is defined per system within the network to limit the possibility of lateral (i.e., East-West) movement of an attacker.

Implementation of a zero-trust architecture with traditional network security solutions can be costly, complex, and come with a high management burden. Moreover, the lack of visibility for organization's internal networks can slow down implementation of a zero-trust architecture and leave gaps that may only be discovered after they have been exploited. Additionally, conventional internal perimeters may have granularity only down to a VLAN or subnet – as is common with many traditional DMZs – rather than down to the individual system.

5.2 NSX DFW Architecture and Components

The NSX DFW architecture management plane, control plane, and data plane work together to enable a centralized policy configuration model with distributed firewalling. This section will examine the role of each plane and its associated components, detailing how they interact with each other to provide a scalable, topology agnostic distributed firewall solution.

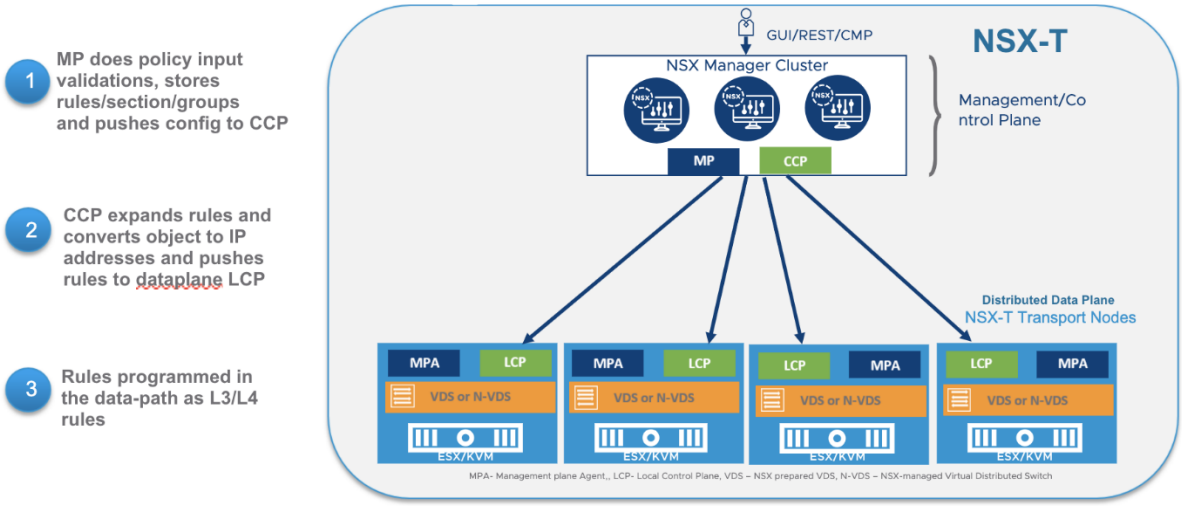


Figure 5-2: NSX DFW Architecture and Components

5.2.1 Management Plane

The NSX management plane is implemented through NSX Managers. NSX Managers are deployed as a cluster of 3 manager nodes. Access to the NSX Manager is available through a GUI or REST API framework. When a firewall policy rule is configured, the NSX management plane service validates the configuration and locally stores a persistent copy. Then the NSX Manager pushes user-published policies to the control plane service within Manager Cluster which in turn pushes to the data plane. A typical DFW policy configuration consists of one or more sections with a set of rules using objects like Groups, Segments, and application level gateway (ALGs). For monitoring and troubleshooting, the NSX Manager interacts with a host-based management plane agent (MPA) to retrieve DFW status along with rule and flow statistics. The NSX Manager also collects an inventory of all hosted virtualized workloads on NSX transport nodes. This is dynamically collected and updated from all NSX transport nodes.

5.2.2 Control Plane

The NSX control plane consists of two components - the central control plane (CCP) and the Local Control Plane (LCP). The CCP is implemented on NSX Manager Cluster, while the LCP includes the user space module on all of the NSX transport nodes. This module interacts with the CCP to exchange configuration and state information.

From a DFW policy configuration perspective, NSX Control plane will receive policy rules pushed by the NSX Management plane. If the policy contains objects including segments or Groups, it converts them into IP addresses using an object-to-IP mapping table. This table is maintained by the control plane and updated using an IP discovery mechanism. Once the policy is converted into a set of rules based on IP addresses, the CCP pushes the rules to the LCP on all the NSX transport nodes.

The CCP utilizes a hierarchy system to distribute the load of CCP-to-LCP communication. The responsibility for transport node notification is distributed across the managers in the manager clusters based on an internal hashing mechanism. For example, for 30 transport nodes with three managers, each manager will be responsible for roughly ten transport nodes.

5.2.3 Data Plane

The NSX transport nodes comprise the distributed data plane with DFW enforcement done at the hypervisor kernel level. Each of the transport nodes, at any given time, connects to only one of the CCP managers, based on mastership for that node. On each of the transport nodes, once the local control plane (LCP) has received policy configuration from CCP, it pushes the firewall policy and rules to the data plane filters (in kernel) for each of the virtual NICs. With the "Applied To" field in the rule or section which defines scope of enforcement, the LCP makes sure only relevant DFW rules are programmed on relevant virtual NICs instead of every rule everywhere, which would be a suboptimal use of hypervisor resources. Additional details on data plane components for both ESXi and KVM hosts explained in following sections.

5.3 NSX Data Plane Implementation - ESXi vs. KVM Hosts

NSX provides network virtualization and security services in a heterogeneous hypervisor environment, managing ESXi and KVM hosts as part of the same NSX cluster. The DFW is functionally identical in both environments; however, there are architectural and implementation differences depending on the hypervisor specifics.

Management and control plane components are identical for both ESXi and KVM hosts. For the data plane, they use a different implementation for packet handling. NSX uses either the VDS in ESXi 7.0 and later or the N-VDS (which is derived from the VDS) on earlier ESXi hosts, along with the VMware Internetworking Service Insertion Platform (vSIP) kernel module for firewalling. (For details on the differences between the N-VDS and the VDS, see NSX Logical Switching. For KVM, the N-VDS leverages Open vSwitch (OVS) and its utilities. The following sections highlight data plane implementation details and differences between these two options.

5.3.1 ESXi Hosts- Data Plane Components

NSX uses VDS or N-VDS on ESXi hosts for connecting virtual workloads, managing it with the NSX Manager application. The NSX DFW kernel space implementation for ESXi is same as the implementation of NSX for vSphere – it uses the VMware Internetworking Service Insertion Platform (vSIP) kernel module and kernel IO chains filters. NSX does not require vCenter to be present. Figure 5-3:

NSX MANAGEMENT PLANE COMPONENTS ON KVM provides details on the data plane components for the ESX host.

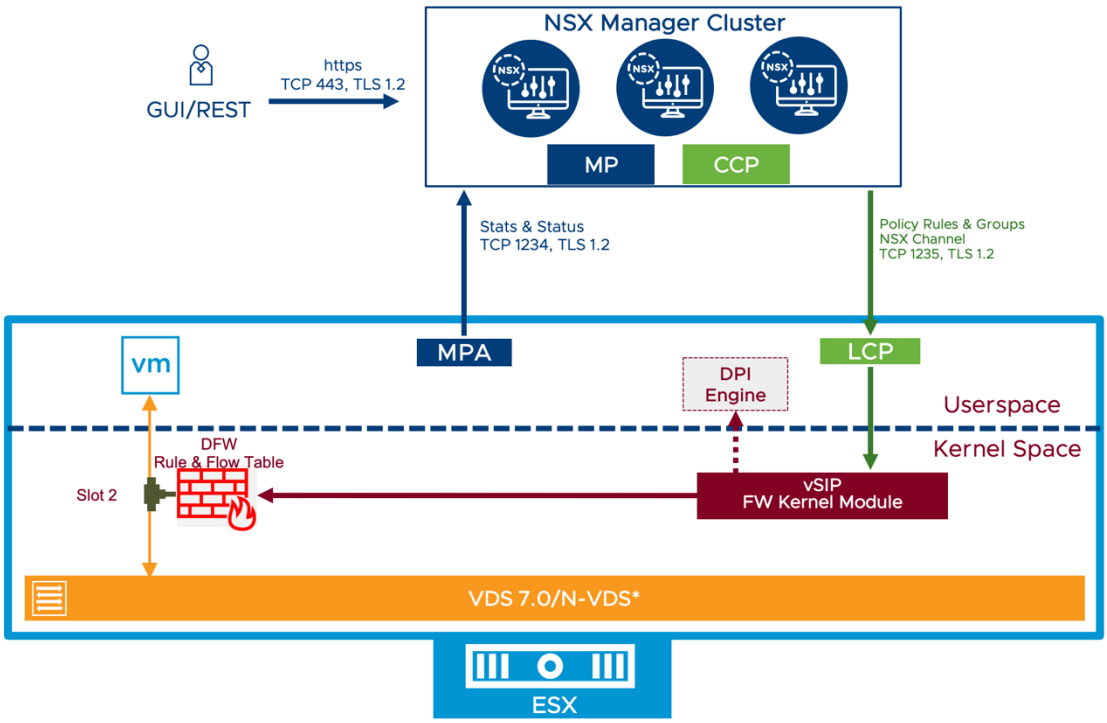


Figure 5-3: NSX Management Plane Components on KVM

NSX uses OVS and its utilities on KVM to provide DFW functionality, thus the LCP agent implementation differs from an ESXi host. For KVM, there is an additional component called the NSX agent in addition to LCP, with both running as user space agents. When LCP receives DFW policy from the CCP, it sends it to the NSX-agent. The NSX-agent will process and convert policy messages received to a format appropriate for the OVS data path. Then, the NSX agent programs the policy rules onto the OVS data path using OpenFlow messages. For stateful DFW rules, NSX uses the Linux conntrack utilities to keep track of the state of permitted flow connections allowed by a stateful firewall rule. For DFW policy rule logging, NSX uses the ovs-fwd module.

The MPA interacts with NSX Manager to export status, rules, and flow statistics. The MPA module gets the rules and flows statistics from data path tables using the stats exporter module.

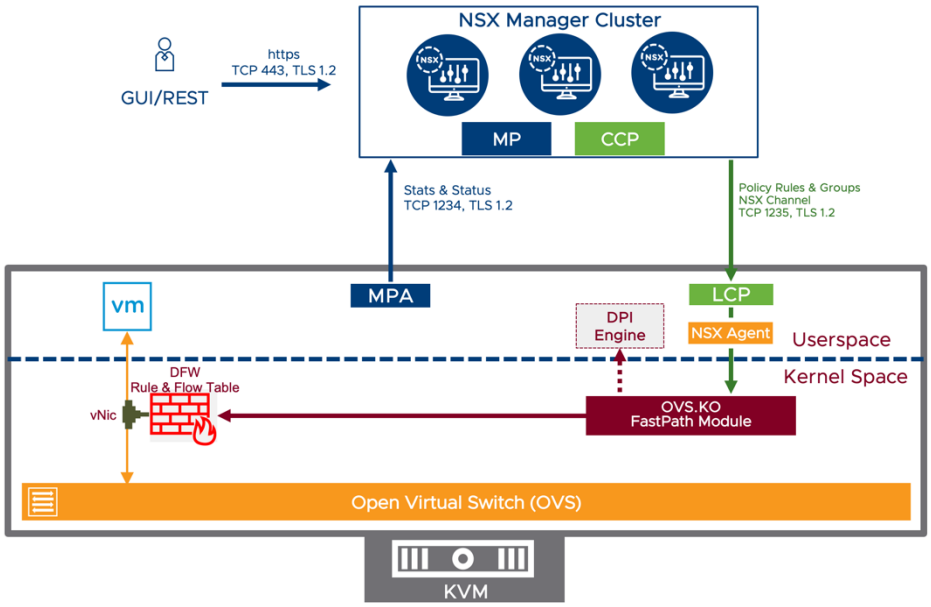


Figure 5-4: NSX DFW Data Plane Components on KVM

5.3.2 NSX DFW Policy Lookup and Packet Flow

In the data path, the DFW maintains two tables: a rule table and a connection tracker table. The LCP populates the rule table with the configured policy rules, while the connection tracker table is updated dynamically to cache flows permitted by rule table. NSX DFW can allow for a policy to be stateful or stateless with section-level granularity in the DFW rule table. The connection tracker table is populated only for stateful policy rules; it contains no information on stateless policies. This applies to both ESXi and KVM environments.

NSX DFW rules are enforced as follows:

- Rules are processed in top-to-bottom order.
- Each packet is checked against the top rule in the rule table before moving down the subsequent rules in the table.
- The first rule in the table that matches the traffic parameters is enforced. The search is then terminated, so no subsequent rules will be examined or enforced.

Because of this behavior, it is always recommended to put the most granular policies at the top of the rule table. This will ensure more specific policies are enforced first. The DFW default policy rule, located at the bottom of the rule table, is a catchall rule; packets not matching any other rule will be enforced by the default rule - which is set to “allow” by default. This ensures that VM-to-VM communication is not broken during staging or migration phases. It is a best practice to then change this default rule to a “drop” action and enforce access control through an explicit allow model (i.e., only traffic defined in the firewall policy is allowed onto the network). FIGURE 5-5: NSX DFW POLICY LOOKUP diagrams the policy rule lookup and packet flow.

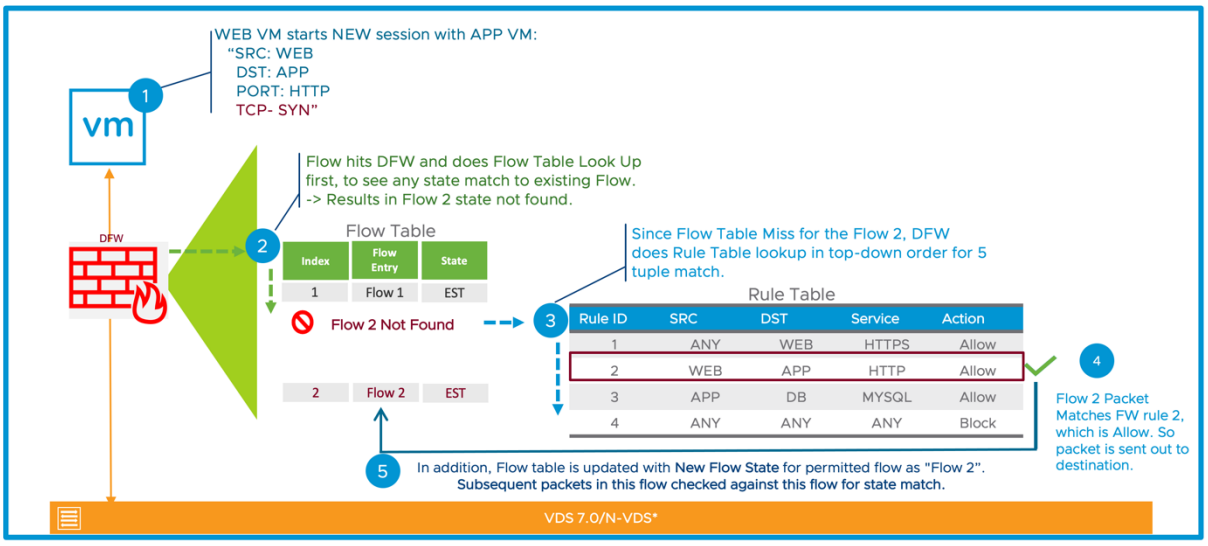


Figure 5-5: NSX DFW Policy Lookup

In the example shown above,

1. WEB VM initiates a session to APP VM by sending TCP SYN packet.
2. The TCP SYN packets hit the DFW on vNIC and does a Flow Table lookup first, to see if any state matches the existing Flow. Given it's the first packet of the new session, lookup results in "Flow state not found".
3. Since the Flow Table missed, the DFW does a Rule Table lookup in top-down order for 5-Tuple match.
4. Flow Matches FW rule 2, which is Allow so the packet is sent out to the destination.
5. In addition, the Flow table is updated with New Flow State for permitted flow as "Flow 2".

Subsequent packets in this TCP session checked against this flow in the flow table for the state match. Once the session terminates, the flow information is removed from the flow table.

5.4 NSX Security Policy - Plan, Design and Implement

Planning, designing, and implementing NSX security policy is a three-step process:

1. Policy Methodology – Decide on the policy approach - application-centric, infrastructure-centric, or network-centric
2. Policy Rule Model – Select grouping and management strategy for policy rules by the NSX DFW policy categories and sections.
3. Policy Consumption – Implement the policy rules using the abstraction through grouping constructs and options provided by NSX.

5.4.1 Security Policy Methodology

This section details the considerations behind policy creation strategies to help determine which capabilities of the NSX platform should be utilized as well as how various grouping methodologies and policy strategies can be adopted for a specific design.

The three general methodologies reviewed in FIGURE 5-6: MICRO-SEGMENTATION METHODOLOGIES can be utilized for grouping application workloads and building security rule sets within the NSX DFW. This section will look at each methodology and highlight appropriate usage.

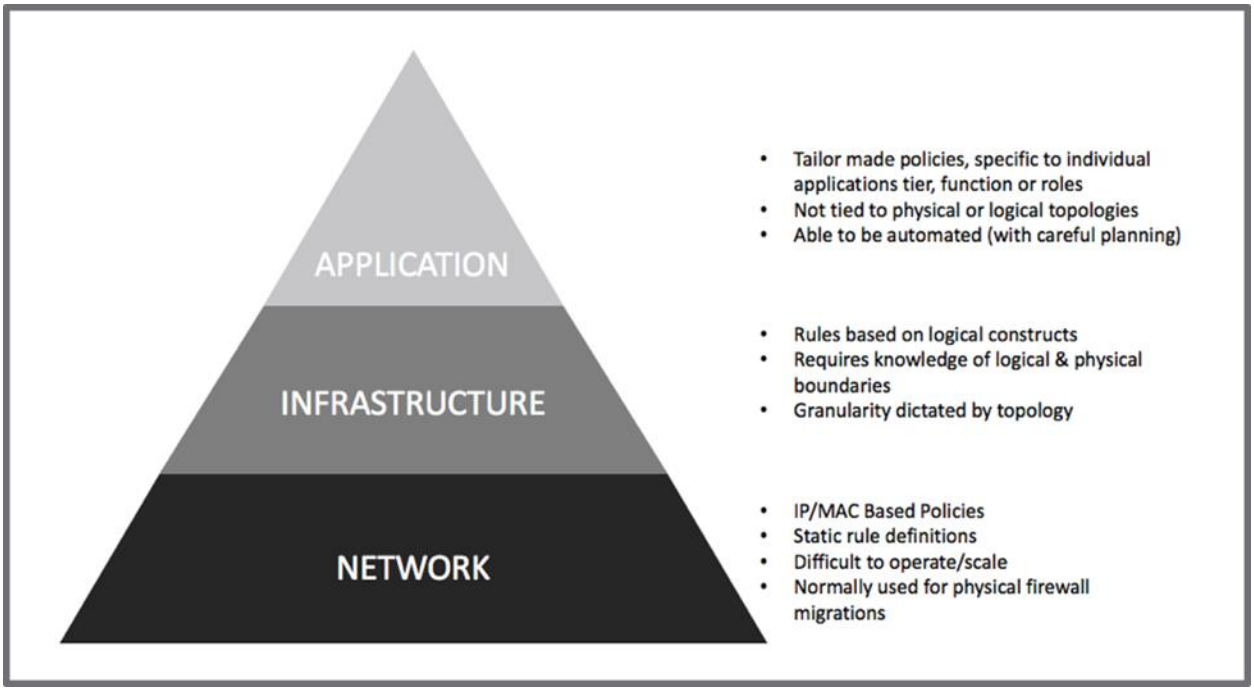


Figure 5-6: Micro-segmentation Methodologies

5.4.1.1 Ethernet

The Ethernet Section of the policy is a Layer 2 firewalling section. All rules in this section must use MAC Addresses for their source or destination objects. Any rule defined with any other object type will be ignored.

5.4.1.2 Application

In an application-centric approach, grouping is based on the application type (e.g., VMs tagged as "Web-Servers"), application environment (e.g., all resources tagged as "Production-Zone") and application security posture. An advantage of this approach is the security posture of the application is not tied to network constructs or infrastructure. Security policies can move with the application irrespective of network or infrastructure boundaries, allowing security teams to focus on the policy rather than the architecture. Policies can be templated and reused across instances of the same types of applications and workloads while following the application lifecycle; they will be applied when the application is

deployed and is destroyed when the application is decommissioned. An application-based policy approach will significantly aid in moving towards a self-service IT model. In an environment where there is strong adherence to a strict naming convention, the VM substring grouping option allows for simple policy definition.

An application-centric model does not provide significant benefits in an environment that is static, lacks mobility, and has infrastructure functions that are properly demarcated.

5.4.1.3 Infrastructure

Infrastructure-centric grouping is based on infrastructure components such as segments or segment ports, identifying where application VMs are connected. Security teams must work closely with the network administrators to understand logical and physical boundaries.

If there are no physical or logical boundaries in the environment, then an infrastructure-centric approach is not suitable.

5.4.1.4 Network

Network-centric is the traditional approach of grouping based on L2 or L3 elements. Grouping can be done based on MAC addresses, IP addresses, or a combination of both. NSX supports this approach of grouping objects. A security team needs to aware of networking infrastructure to deploy network-based policies. There is a high probability of security rule sprawl as grouping based on dynamic attributes is not used. This method of grouping works well for migrating existing rules from an existing firewall.

A network-centric approach is not recommended in dynamic environments where there is a rapid rate of infrastructure change or VM addition/deletion.

5.4.2 Security Rule Model

Policy rule models in a data center are essential to achieve optimal micro-segmentation strategies. The first criteria in developing a policy model is to align with the natural boundaries in the data center, such as tiers of application, SLAs, isolation requirements, and zonal access restrictions. Associating a top-level zone or boundary to a policy helps apply consistent, yet flexible control.

Global changes for a zone can be applied via single policy; however, within the zone there could be a secondary policy with sub-grouping mapping to a specific sub-zone. An example production zone might itself be carved into sub-zones like PCI or HIPAA. There are also zones for each department as well as shared services. Zoning creates relationships between various groups, providing basic segmentation and policy strategies.

A second criterion in developing policy models is identifying reactions to security events and workflows. If a vulnerability is discovered, what are the mitigation strategies? Where is the source of the exposure – internal or external? Is the exposure limited to a specific application or operating system version?

When east-west security is first implemented in a brownfield environment, there are two common approaches, depending on corporate culture: either an incremental zonal approach where one application is secured before moving to the next, or a top-down iterative approach where first prod and non-prod are divided then each of those areas are further subdivided. Regardless of the chosen approach, there will likely be a variety of security postures taken within each zone. A lab zone, for example may merely be ring-fenced with a policy that allows any traffic type from lab device to lab device and only allows basic common services such as LDAP, NTP, and DNS to penetrate the perimeter in. On the other end of the spectrum, any zone containing regulated or sensitive data (such as customer info) will often be tightly defined traffic between entities, many types being further inspected by partner L7 firewall offerings using Service Insertion.

The answers to these questions help shape a policy rule model. Policy models should be flexible enough to address ever-changing deployment scenarios, rather than simply be part of the initial setup. Concepts such as intelligent grouping, tags and hierarchy provide flexible yet agile response capability for steady state protection as well as during instantaneous threat response. The model shown in FIGURE 5-7: SECURITY RULE MODEL represents an overview of the different classifications of security rules that can be placed into the NSX DFW rule table. Each of the classification shown represents a category on NSX firewall table layout. The Firewall table category aligns with the best practice around organizing rules to help admin with grouping Policy based on the category. Each firewall category can have one or more policy within it to organize firewall rules under that category.

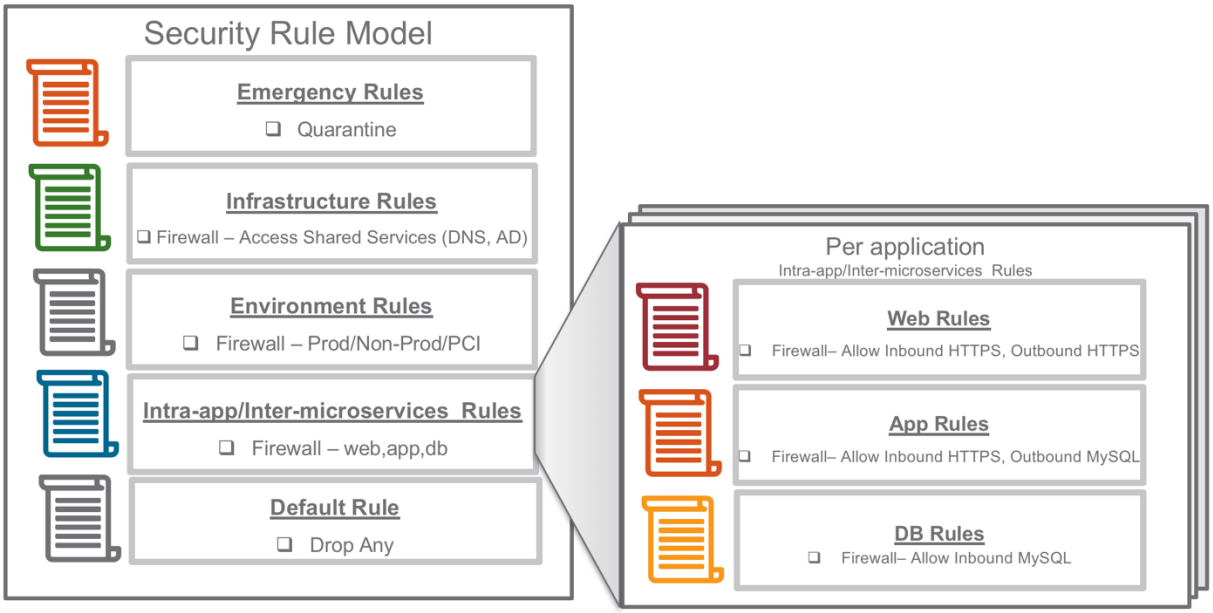


Figure 5-7: Security Rule Model

5.4.3 Security Policy - Consumption Model

NSX Security policy is consumed by the firewall rule table, which is using NSX Manager GUI or REST API framework. When defining security policy rules for the firewall table, it is recommended to follow these high-level steps:

- VM Inventory Collection – Identify and organize a list of all hosted virtualized workloads on NSX transport nodes. This is dynamically collected and saved by NSX Manager as the nodes – ESXi or KVM – are added as NSX transport nodes.
- Tag Workload – Use VM inventory collection to organize VMs with one or more tags. Each designation consists of scope and tag association of the workload to an application, environment, or tenant. For example, a VM tag could be "Scope = Prod, Tag = web" or "Scope=tenant-1, Tag = app-1". Often, these categories will dive several layers deep including BU, project, environment, and regulatory flags. When following the iterative approach of segmentation, categories and tags can be added to entities with existing tags. In the application centric approach, new categories can be added with each application.
- Group Workloads – Use the NSX logical grouping construct with dynamic or static membership criteria based on VM name, tags, segment, segment port, IP's, or other attributes. NSX allows for thousands of groups based on tags, although rarely are more than a dozen or so needed.
- Define Security Policy – Using the firewall rule table, define the security policy. Have categories and policies to separate and identify emergency, infrastructure, environment, and application-specific policy rules based on the rule model.

The methodology and rule model mentioned earlier would influence how to tag and group the workloads as well as affect policy definition. The following sections offer more details on grouping and firewall rule table construction with an example of grouping objects and defining NSX DFW policy.

5.4.3.1 Group Creation Strategies

The most basic grouping strategy is creation of a group around every application which is hosted in the NSX environment. Each 3-tier, 2-tier, or single-tier applications should have its own security group to enable faster operationalization of micro-segmentation. When combined with a basic rule restricting inter-application communication to only essential shared services (e.g., DNS, AD, DHCP server) this enforces granular security inside the perimeter. Once this basic micro-segmentation is in place, the writing of per-application rules can begin.

Groups

NSX provides collection of referenceable objects represented in a construct called Groups. The selection of a specific policy methodology approach – application, infrastructure, or network – will help dictate how grouping construct is used. Groups allow abstraction of workload grouping from the underlying infrastructure topology. This allows a security policy to be written for either a workload or zone (e.g., PCI zone, DMZ, or production environment).

A Group is a logical construct that allows grouping into a common container of static (e.g., IPSet/NSX objects) and dynamic (e.g., VM names/VM tags) elements. This is a generic construct which can be leveraged across a variety of NSX features where applicable.

Static criteria provide capability to manually include particular objects into the Group. For dynamic inclusion criteria, Boolean logic can be used to create groups between various criteria. A Group creates a logical grouping of VMs based on static and dynamic criteria. TABLE 5-1: NSX OBJECTS USED FOR GROUPS shows one type of grouping criteria based on NSX Objects.

NSX Object	Description
IP Address	Grouping of IP addresses and subnets.
Segment	All VMs/vNICs connected to this segment/logical switch segment will be selected.
Group	Nested (Sub-group) of collection of referenceable objects - all VMs/vNICs defined within the Group will be selected
Segment Port	This particular vNIC instance will be selected.
MAC Address	Selected MAC sets container will be used. MAC sets contain a list of individual MAC addresses.
AD Groups	Grouping based on Active Directory groups for Identity Firewall (VDI/RDSH) use case.

Table 5-1: NSX Objects used for Groups

TABLE 5-2: VM PROPERTIES USED FOR GROUPS list the selection criteria based on VM properties.

VM Property	Description
VM Name	All VMs that contain/equal/starts/not-equals with the string as part of their name.
Tags	All VMs that are applied with specified NSX security tags
OS Name	All VM with specific operating System type and version
Computer name	All VMs that contain/equal/starts/not-equals with the string as part of their hostname.

Table 5-2: VM Properties used for Groups

The use of Groups gives more flexibility as an environment changes over time. This approach has three major advantages:

- Rules stay more constant for a given policy model, even as the data center environment changes. The addition or deletion of workloads will affect group membership alone, not the rules.
- Publishing a change of group membership to the underlying hosts is more efficient than publishing a rule change. It is faster to send down to all the affected hosts and cheaper in terms of memory and CPU utilization.
- As NSX adds more grouping object criteria, the group criteria can be edited to better reflect the data center environment.

Using Nested Groups

Groups can be nested. A Group may contain multiple groups or a combination of groups and other grouping objects. A security rule applied to the parent Group is automatically applied to the child Groups. Nesting should be limited to 3 levels, although more are supported. This is to ease troubleshooting, minimize unintentional policy results, and to optimize the computational burden of publishing policy. Nothing prolongs downtime like trying to follow the logic of a grouping nested 5 levels deep.

In the example shown in FIGURE 5-8: GROUP AND NESTED GROUP EXAMPLE, three Groups have been defined with different inclusion criteria to demonstrate the flexibility and the power of grouping construct.

- Using dynamic inclusion criteria, all VMs with name starting by “WEB” are included in Group named “SG-WEB”.
- Using dynamic inclusion criteria, all VMs containing the name “APP” and having a tag “Scope=PCI” are included in Group named “SG-PCI-APP”.
- Using static inclusion criteria, all VMs that are connected to a segment “SEG-DB” are included in Group named “SG-DB”.

Nesting of Group is also possible; all three of the Groups in the list above could be children of a parent Group named “SG-APP-1-AllTier”. This organization is also shown in FIGURE 5-8: GROUP AND NESTED GROUP EXAMPLE.

Figure 5-8: Group and Nested Group Example

Efficient Grouping Considerations

Calculation of groups adds a processing load to the NSX management and control planes. Different grouping mechanisms add different types of loads. Static groupings are more efficient than dynamic groupings in terms of calculation. At scale, grouping considerations should consider the frequency of group changes for associated VMs. A large number of group changes applied frequently means the grouping criteria is suboptimal.

5.4.3.2 Define Policy using DFW Rule Table

The NSX DFW rule table starts with a default rule to allow any traffic. An administrator can add multiple policies on top of default rule under different categories based on the specific policy model. NSX distributed firewall table layout consists of Categories like Ethernet, Emergency, Infrastructure, Environment, and Application to help users to organize security policies. Each category can have one or more policy/section with one or more firewall rules. Please refer to Security Rule Model section above to understand the best practices around organizing the policies.

In the data path, the packet lookup will be performed from top to bottom order, starting with policies from category Ethernet, Emergency, Infrastructure, Environment and Application. Any packet not matching an explicit rule will be enforced by the last rule in the table (i.e., default rule). This final rule is set to the "allow" action by default, but it can be changed to "block" if desired.

The NSX DFW enables policy to be stateful or stateless with policy-level granularity. By default, NSX DFW is a stateful firewall; this is a requirement for most deployments. In some scenarios where an application has less network activity, the stateless section may be appropriate to avoid connection reset due to inactive timeout of the DFW stateful connection table.

NSX Firewall policy can also be locked by a user to avoid losing any update to policy with multiple people editing same policy at the same time.

Name	ID	Source	Destination	Service	Profiles	Applied To	Action	Advanced Setting	Stats

Table 5-3: Policy Rule Fields

A rule within a policy is composed of field shown in *Table 5-3: Policy Rule Fields* and its meaning is described below

Rule Name: User field; supports up to 30 characters.

ID: Unique rule ID auto generated by System. The rule id helps in monitoring and troubleshooting. Firewall Log carries this Rule ID when rule logging is enabled.

Source and Destination: Source and destination fields of the packet. This will be a GROUP which could be static or dynamic groups as mentioned under Group section.

Service: Predefined services, predefined services groups, or raw protocols can be selected. When selecting raw protocols like TCP or UDP, it is possible to define individual port numbers or a range. There are four options for the services field:

- **Pre-defined Service** – A pre-defined Service from the list of available objects.
- **Add Custom Services** – Define custom services by clicking on the "Create New Service" option. Custom services can be created based on L4 Port Set, application level gateways (ALGs), IP protocols, and other criteria. This is done using the "service type" option in the configuration menu. When selecting an L4 port set with TCP or UDP, it is possible to define individual destination ports or a range of destination ports. When selecting ALG, select supported protocols for ALG from the list. ALGs are only supported in stateful mode; if the section is marked as stateless, the ALGs will not be implemented. Additionally, some ALGs may be supported only on ESXi hosts, not KVM. Please review release-specific documentation for supported ALGs and hosts.
- **Custom Services Group** – Define a custom Services group, selecting from single or multiple services. Workflow is similar to adding Custom services, except you would be adding multiple service entries.

Profiles: This is used to select & define Layer 7 Application ID & FQDN profile. This is used for Layer 7 based security rules.

Applied To: Define the scope of rule publishing. The policy rule could be published all workloads (default value) or restricted to a specific GROUP. When GROUP is used in Applied-To it needs to be based on NON-IP members like VM object, Segments etc. Not using the Applied To field can result in very large firewall tables being loaded on vNICs, which will negatively affect performance.

Action: Define enforcement method for this policy rule; available options are listed in TABLE 5-4: FIREWALL RULE TABLE – "ACTION" VALUES

Action	Description

Drop	Block silently the traffic.
Allow	Allow the traffic.
Reject	Reject action will send back to initiator: • RST packets for TCP connections. • ICMP unreachable with network administratively prohibited code for UDP, ICMP and other IP connections.

Table 5-4: Firewall Rule Table – "Action" Values

Advanced Settings: Following settings are under advanced settings options:

- Logging: Enable or disable packet logging. When enabled, each DFW enabled host will send DFW packet logs in a syslog file called "dfwpktlog.log" to the configured syslog server. This information from the default rule will provide insight to traffic not currently being caught by existing policy. Best practice for deploying east-west traffic in a brownfield environment is to define policy with a default allow rule with logging. This allows for the verification of traffic not currently caught by policy.
- Direction: This field matches the direction of the packet, default both In-Out. It can be set to match packet exiting the VM, entering the VM, or both directions.
- IP Protocol: By default, both IPv4 & IPv6 protocols. Option to choose IPv4 or IPv6.
- Log Label: You can Label the rule; this will be sent as part of DFW packet log when traffic hits this rule.
- Notes: This field can be used for any free-flowing string and is useful to store comments. Best practice is to use this field for change control by pointing to a ticket ID.
- Stats: Provides packets/bytes/sessions statistics along with popularity index associated with that rule entry. The Also provides Popularity Index for the given rule. Stats per rule are Polled Aggregated every 15 minutes from all the transport nodes.

Examples of Policy Rules for 3-Tier Application

FIGURE 5-9: 3-TIER APPLICATION NETWORK TOPOLOGY shows a standard 3-Tier application topology used to define NSX DFW policy. Three web servers are connected to "SEG Web", two applications servers are connected to "SEG App", and 2 DB servers connected to "SEG DB". A distributed Gateway is used to interconnect the three tiers by providing inter-tier routing. NSX DFW has been enabled, so each VM has a dedicated instance of DFW attached to its vNIC/segment port.

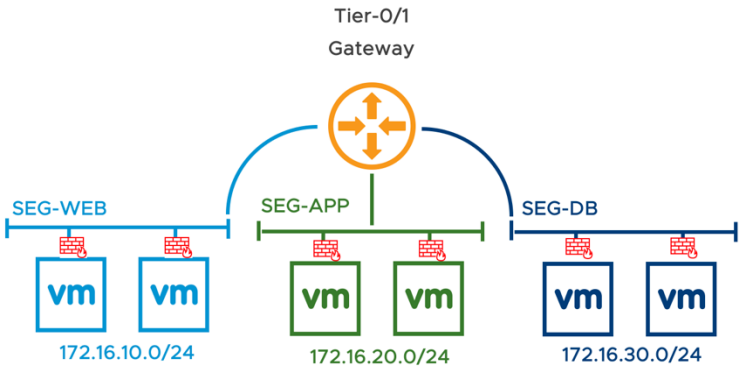


Figure 5-9: 3-Tier Application Network Topology

In order to define micro-segmentation policy for this application use the category Application on DFW rule table and add a new policy session and rules within it for each application.

The following use cases employ present policy rules based on the different methodologies introduced earlier.

Example 1: Static IP addresses/subnets Group in security policy rule.

This example shows use of the network methodology to define policy rule. Groups in this example are identified in TABLE 5-5: FIREWALL RULE TABLE - EXAMPLE 1 – GROUP DEFINITION while the firewall policy configuration is shown in TABLE 5-6: FIREWALL RULE TABLE - EXAMPLE 1- POLICY.

Group name	Group definition
Group-WEB-IP	IP Members: 172.16.10.0/24
Group-APP-IP	IP Members: 172.16.20.0/24
Group-DB-IP	IP Members: 172.16.30.0/24

Table 5-5: Firewall Rule Table - Example 1 – Group Definition

Name	Source	Destination	Service	Action	Applied To
Any to Web	Any	Group-WEB-IP	https	Allow	All

Web to App	Group-WEB-IP	Group-APP-IP	<Enterprise Service Bus>	Allow	All
App to DB	Group-APP-IP	Group-DB-IP	SQL	Allow	All
Block-Other	Any	Any	Any	Drop	All

Table 5-6: Firewall Rule Table - Example 1- Policy

The DFW engine is able to enforce network traffic access control based on the provided information. To use this type of construct, exact IP information is required for the policy rule. This construct is quite static and does not fully leverage dynamic capabilities with modern cloud systems.

Example 2: Using Segment object Group in Security Policy rule.

This example uses the infrastructure methodology to define policy rule. Groups in this example are identified in TABLE 5-7: FIREWALL RULE TABLE - EXAMPLE 2 – GROUP DEFINITION while the firewall policy configuration is shown in TABLE 5-8: FIREWALL RULE TABLE - EXAMPLE 2 – POLICY.

Group name	Group definition
Group-SEG-WEB	Static inclusion: SEG-WEB
Group-SEG-APP	Static inclusion: SEG-APP
Group-SEG-DB	Static inclusion: SEG-DB

Table 5-7: Firewall Rule Table - Example 2 – Group Definition

Name	Source	Destination	Service	Action	Applied To
Any to Web	Any	Group-SEG-WEB	https	Allow	Group-SEG-WEB
Web to App	Group-SEG-WEB	Group-SEG-APP	<Enterprise Service Bus>	Allow	Group-SEG-WEB Group-SEG-APP
App to DB	Group-SEG-APP	Group-SEG-DB	SQL	Allow	Group-SEG-APP Group-SEG-DB

Block-Other	Any	Any	Any	Drop	Group-SEG-WEB Group-SEG-APP Group-SEG-DB

Table 5-8: Firewall Rule Table - Example 2 – Policy

Reading this policy rule table would be easier for all teams in the organization, ranging from security auditors to architects to operations. Any new VM connected on any segment will be automatically secured with the corresponding security posture. For instance, a newly installed web server will be seamlessly protected by the first policy rule with no human intervention, while VM disconnected from a segment will no longer have a security policy applied to it. This type of construct fully leverages the dynamic nature of NSX. It will monitor VM connectivity at any given point in time, and if a VM is no longer connected to a particular segment, any associated security policies are removed.

This policy rule also uses the "Applied To" option to apply the policy to only relevant objects rather than populating the rule everywhere. In this example, the first rule is applied to the vNIC associated with "SEG-Web". Use of "Applied To" is recommended to define the enforcement point for the given rule for better resource usage.

Security policy and IP Discovery

Both NSX DFW and Gateway Firewall (GFW) has a dependency on VM-to-IP discovery which is used to translate objects to IP before rules are pushed to data path. This is mainly required when the policy is defined using grouped objects. This VM-to-IP table is maintained by NSX Control plane and populated by the IP discovery mechanism. IP discovery used as a central mechanism to ascertain the IP address of a VM. By default, this is done using DHCP and ARP snooping, with VMware Tools available as another mechanism with ESXi hosts. These discovered VM-to-IP mappings can be overridden by manual input if needed, and multiple IP addresses are possible on a single vNIC. The IP and MAC addresses learned are added to the VM-to-IP table. This table is used internally by NSX for SpoofGuard, ARP suppression, and firewall object-to-IP translation.

5.5 Intrusion Detection

Much like distributed firewalling changed the game on firewalling by providing a distributed, ubiquitous enforcement plane, NSX distributed IPS/IPS changes the game on IPS by providing a distributed, ubiquitous enforcement plane. However, there are additional benefits that the NSX distributed IPS model brings beyond ubiquity (which in itself is a game changer). NSX IPS is IPS distributed across all the hosts. Much like with DFW, the distributed nature allows the IPS capacity to grow linearly with compute capacity. Beyond that, however, there is an added benefit to distributing IPS. This is the added context. Legacy network Intrusion Detection and Prevention systems are deployed centrally in the network and rely either on traffic to be hairpinned through them or a copy of the traffic to be sent to them via techniques like SPAN or TAPs. These sensors typically match all traffic against all or a broad set of signatures and have very little context about the assets they are protecting. Applying all signatures to all traffic is very inefficient,

as IDS/IPS unlike firewalling needs to look at the packet payload, not just the network headers. Each signature that needs to be matched against the traffic adds inspection overhead and potential latency introduced. Also, because legacy network IDS/IPS appliances just see packets without having context about the protected workloads, it's very difficult for security teams to determine the appropriate priority for each incident. Obviously, a successful intrusion against a vulnerable database server in production which holds mission-critical data needs more attention than someone in the IT staff triggering an IDS event by running a vulnerability scan. Because the NSX distributed IDS/IPS is applied to the vNIC of every workload, traffic does not need to hairpinned to a centralized appliance, and we can be very selective as to what signatures are applied. Signatures related to a windows vulnerability don't need to be applied to Linux workloads, or servers running Apache don't need signatures that detect an exploit of a database service. Through the Guest Introspection Framework, and in-guest drivers, NSX has access to context about each guest, including the operating system version, users logged in or any running process. This context can be leveraged to selectively apply only the relevant signatures, not only reducing the processing impact, but more importantly reducing the noise and quantity of false positives compared to what would be seen if all signatures are applied to all traffic with a traditional appliance. For a detailed description of IDS configuration, see the NSX Product Documentation.

5.6 Service Insertion

The value of NSX security extends beyond NSX to your pre-existing security infrastructure; NSX is the mortar that ties your security bricks to build a stronger wall. Legacy security strategies were intolerant of pre-existing security infrastructure. Anyone who had a Checkpoint firewall and wanted to move to a Palo Alto Networks firewall would run the 2 managers, side by side until the transition was complete. Troubleshooting during this transition period required a lot of chair swiveling. NSX brings a new model, complementing pre-existing infrastructure. Service Insertion is the feature which allows NSX firewalls (both gateway and DFW) to send traffic to legacy firewall infrastructure for processing. This can be done as granularly as a port level, without any modification to existing network architecture. Service Insertion not only sends the traffic to other services for processing, Service Insertion offers and a deep integration which allows the exchange of NSX Manager objects to SI service managers. So, a group in NSX which is comprised on VMs which a substring of "web" (for example) would get shared to the SI service manager. Thus, when a new VM is spun up which becomes a member of the new group, the NSX Manager will send that update to the SI Service Manager so that policy can be consistently applied across platforms.

5.7 Additional Security Features

NSX extends the security solution beyond DFW with additional features to enhance data center security posture on top of micro-segmentation. These features include:

- SpoofGuard - Provides protection against spoofing with MAC+IP+VLAN bindings. This can be enforced at a per logical port level. The SpoofGuard

feature requires static or dynamic bindings (e.g., DHCP/ARP snooping) of IP+MAC for enforcement.

- Segment Security - Provides stateless L2 and L3 security to protect segment integrity by filtering out malicious attacks (e.g., denial of service using broadcast/multicast storms) and unauthorized traffic entering segment from VMs. This is accomplished by attaching the segment security profile to a segment for enforcement. The segment security profile has options to allow/block bridge protocol data unit (BPDU), DHCP server/client traffic, non-IP traffic. It allows for rate limiting of broadcast and multicast traffic, both transmitted and received.

5.8 NSX Security Enforcement – Agnostic to Network Isolation

The NSX security solution is agnostic to network isolation and topology requirements. Below are the different possible deployment options for adapting NSX micro-segmentation policies based on different network isolation requirements.

The consumption of security policies requires no changes from policy planning, design, and implementation perspective. This applies to all of the deployment options mentioned below. However, the following initial provisioning steps required to enforce NSX security policies:

a) Preparation of compute hosts for NSX.

b) Create VLAN or overlay segments on NSX based on network isolation and

c) Move relevant workloads to relevant VLAN or overlay segments/networks on compute hosts for policy enforcement.

5.8.1 NSX Distributed Firewall for VLAN Backed workloads

This is a very common use case for our customer who is looking at NSX as a platform only for micro-segmentation security use case without changing existing network isolation, which is VLAN backed. This is the ideal use case for a brownfield deployment where customer wants to enhance the security posture for existing applications without changing network design.

The following diagram depicts this use case with logical and physical topology.

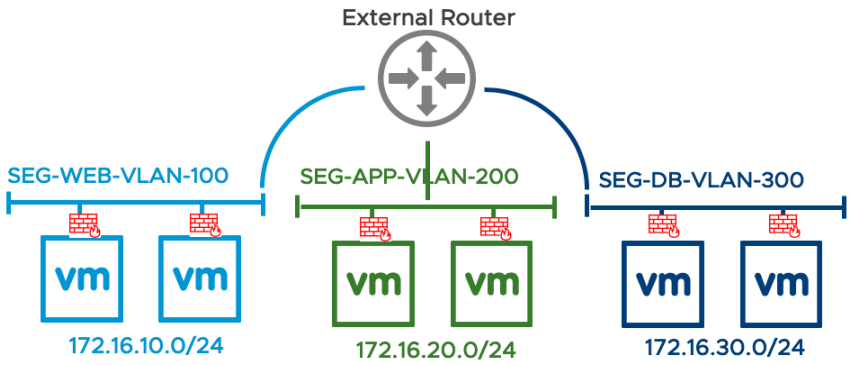


Figure 5-10: NSX DFW Logical topology – VLAN Backed Workloads

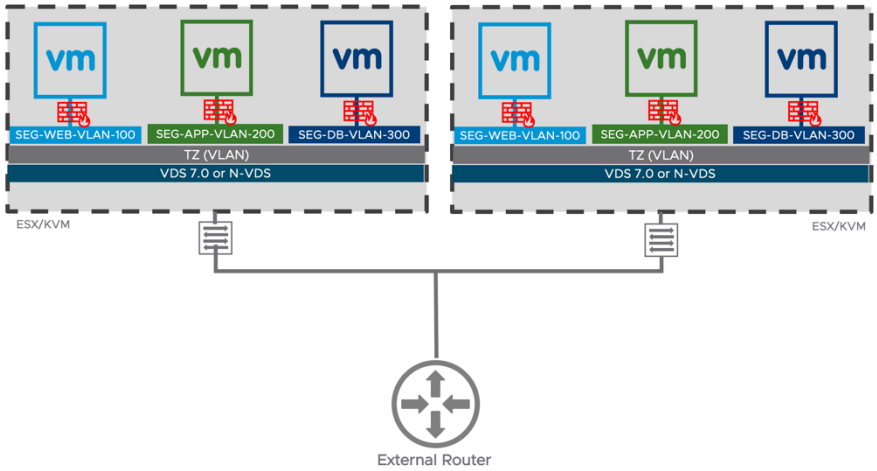


Figure 5-11: NSX DFW Physical Topology – VLAN Backed Workloads

5.8.2 NSX Distributed Firewall for Mix of VLAN and Overlay backed workloads

This use case mainly applies to customer who wants to adapt NSX micro-segmentation policies to all of their workloads and looking at adapting NSX network virtualization (overlay) for their application networking needs in phases. This scenario may arise when customer starts to either deploy new application with network virtualization or migrating existing applications in phases from VLAN to overlay backed networking to avail the advantages of NSX network virtualization. This scenario is also common where there are applications which prevent overlay backed networking from being adopted fully (as described in section <BRIDGING> above). The order of operations in this environment is as follows: on egress, DFW processing happens first, then overlay network processing happens second. On traffic arrival at a remote host, overlay network processing happens first, then DFW processing happens before traffic arrives at the VM.

The following diagram depicts this use case with logical and physical topology.

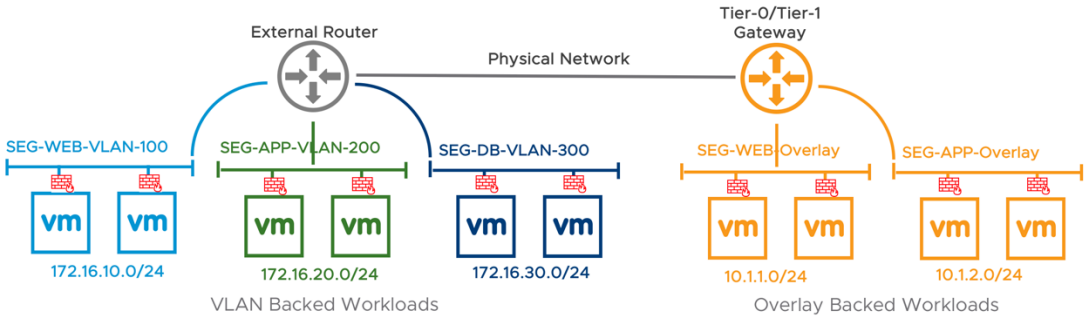


Figure 5-12: NSX DFW Logical Topology – Mix of VLAN & Overlay Backed Workloads

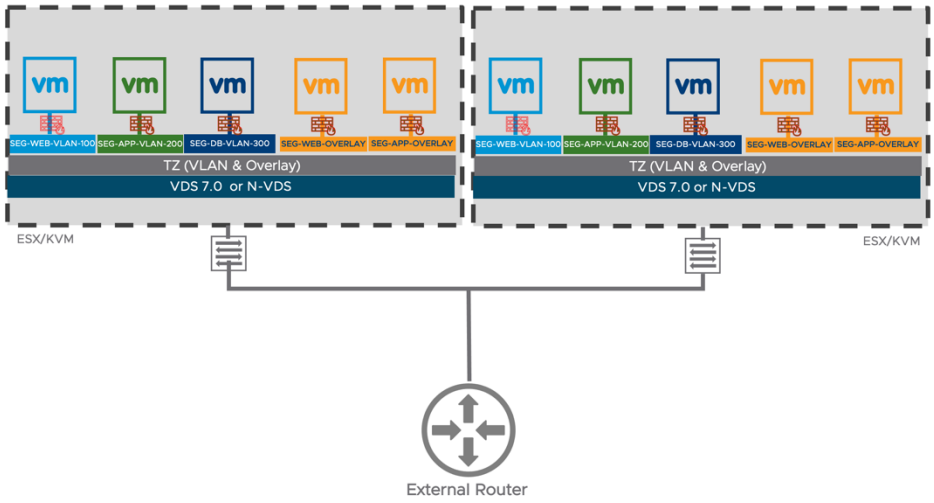


Figure 5-13: NSX DFW Physical Topology – Mix of VLAN & Overlay Backed Workloads

5.8.3 NSX Distributed Firewall for Overlay Backed workloads

In this use case where all the virtualized applications are hosted or moved from VLAN to NSX overlay backed networking from the network isolation perspective. This could apply to green field deployment or final phase of brownfield deployments where all virtualized applications have been moved from VLAN to NSX overlay backed networking.

In summary, NSX Platform enforces micro-segmentation policies irrespective of network isolation, VLAN or overlay or Mix, without having to change policy planning, design, and implementation. A user can define NSX micro-segmentation policy once for the application, and it will continue to work as you migrate application from VLAN based networking to NSX overlay backed networking.

5.9 Gateway Firewall

The NSX Gateway firewall provides essential perimeter firewall protection which can be used in addition to a physical perimeter firewall. Gateway firewall service is part of the NSX Edge node for both bare metal and VM form factors. The Gateway firewall is useful in developing PCI zones, multi-tenant environments, or DevOps style connectivity without forcing the inter-tenant or inter-zone traffic

onto the physical network. The Gateway firewall data path uses DPDK framework supported on Edge to provide better throughput.

Optionally, Gateway Firewall service insertion capability can be leveraged with the partner ecosystem to provide integrated security which leverages existing security investments. This enhances the security posture by providing next-generation firewall (NGFW) services on top of native firewall capability NSX provides. This is applicable for the design where security compliance requirements mandate zone or group of workloads need to be secured using NGFW, for example, DMZ or PCI zones or Multi-Tenant environments. Service insertion leverages existing security infrastructure investments and extends NSX dynamic security groups to them.

5.9.1 Consumption

NSX Gateway firewall is instantiated per gateway and supported at both Tier-0 and Tier-1. Gateway firewall works independently of NSX DFW from a policy configuration and enforcement perspective, although objects can be shared from the DFW. A user can consume the Gateway firewall using either the GUI or REST API framework provided by NSX Manager. The Gateway firewall configuration is similar to DFW firewall policy; it is defined as a set of individual rules within a section. Like the DFW, the Gateway firewall rules can use logical objects, tagging and grouping constructs (e.g., Groups) to build policies. Similarly, regarding L4 services in a rule, it is valid to use predefined Services, custom Services, predefined service groups, custom service groups, or TCP/UDP protocols with the ports. NSX Gateway firewall also supports multiple Application Level Gateways (ALGs). The user can select an ALG and supported protocols by using the other setting for type of service. Gateway FW supports only FTP and TFTP as part of ALG. ALGs are only supported in stateful mode; if the section is marked as stateless, the ALGs will not be implemented.

When partner services are leveraged through service insertion, the implementation requires registering the NSX Manager on the partner management console and the registration of the partner management console in the NSX manager. Once the two managers are integrated, they will share relevant objects, which will improve security policy consistency across the board.

5.9.2 Implementation

Gateway firewall is an optional centralized firewall implemented on NSX Tier-0 gateway uplinks and Tier-1 gateway links. This is implemented on a Tier-0/1 SR component which is hosted on NSX Edge. Tier-0 Gateway firewall supports stateful firewalling only with active/standby HA mode. It can also be enabled in an active/active mode, though it will be only working in stateless mode. Gateway firewall uses a similar model as DFW for defining policy, and NSX grouping construct can be used as well. Gateway firewall policy rules are organized using one or more policy sections in the firewall table for each Tier-0 and Tier-1 Gateway. Firewalling at the perimeter allows for a coarse grain policy definition which can greatly reduce the security policy size inside.

5.9.3 Deployment Scenarios

This section provides two examples for possible deployment and data path implementation.

Gateway FW as Perimeter FW at Virtual and Physical Boundary

The Tier-0 Gateway firewall is used as perimeter firewall between physical and virtual domains. This is mainly used for N-S traffic from the virtualized environment to physical world. In this case, the Tier-0 SR component which resides on the Edge node enforces the firewall policy before traffic enters or leaves the NSX virtual environment. The E-W traffic continues to leverage the distributed routing and firewalling capability which NSX natively provides in the hypervisor. In addition to firewalling, the T1 Gateway can perform per tenant NAT. This is highly desirable in containerized environments to reduce the consumption of IP addresses.

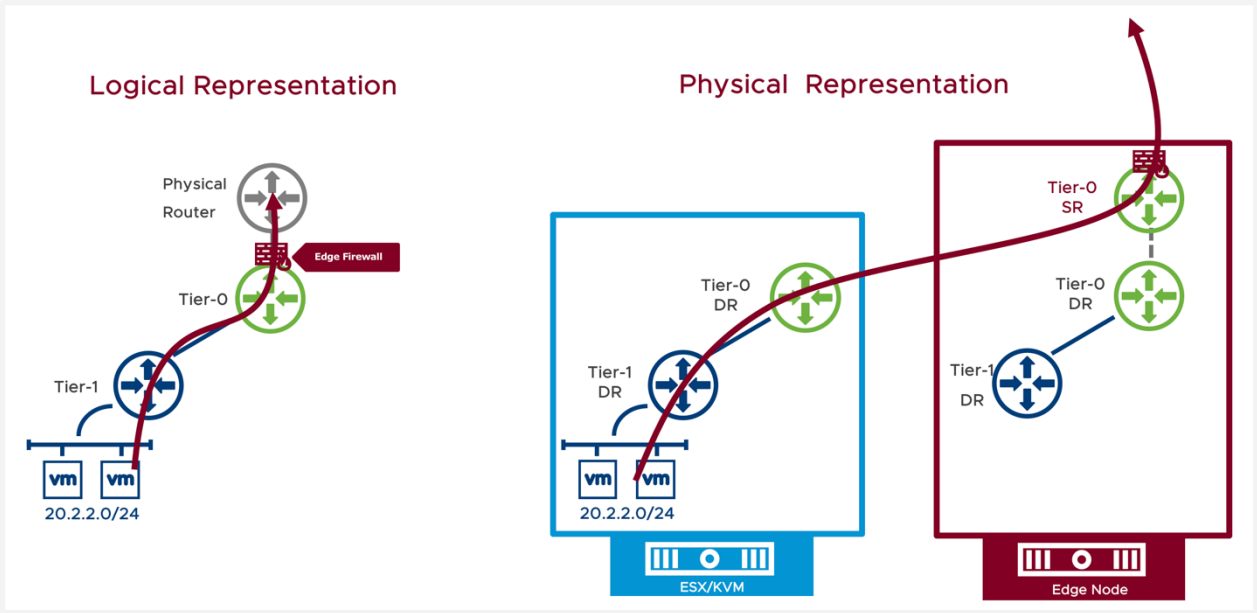


Figure 5-14: Tier-0 Gateway Firewall – Virtual-to-Physical Boundary

Gateway FW as Inter-tenant FW

The Tier-1 Gateway firewall is used as inter-tenant firewall within an NSX virtual domain. This is used to define policies between different tenants who resides within an NSX environment. This firewall is enforced for the traffic leaving the Tier-1 router and uses the Tier-0 SR component which resides on the Edge node to enforce the firewall policy before sending to the Tier-0 Gateway for further processing of the traffic. The intra-tenant traffic continues to leverage distributed routing and firewalling capabilities native to the NSX.

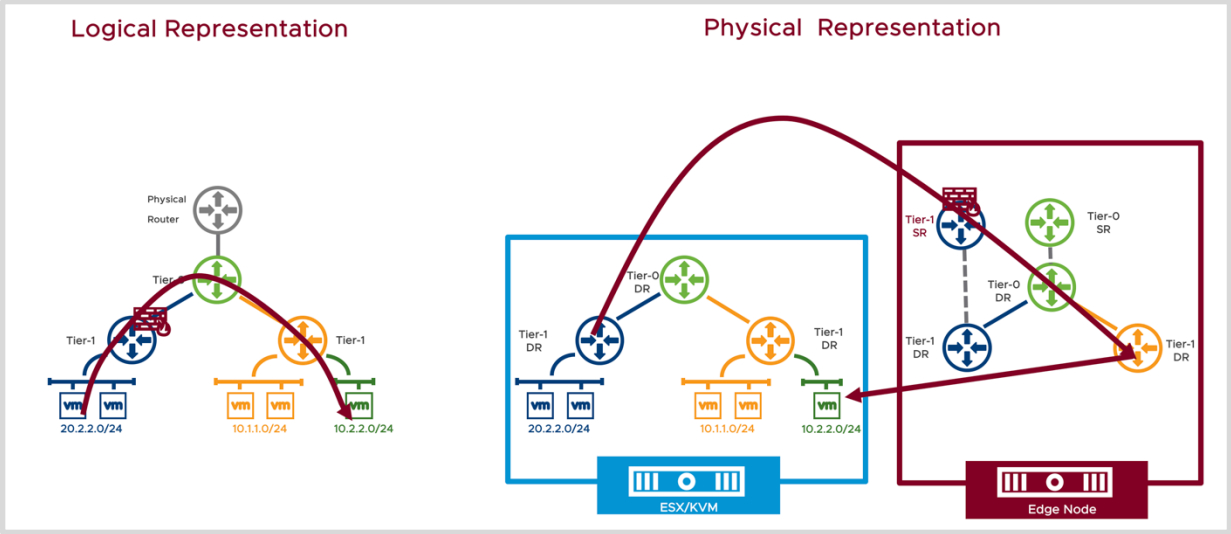


Figure 5-15: Tier-1 Gateway Firewall - Inter-tenant

Gateway FW with NGFW Service Insertion – As perimeter or Inter Tenant Service

This deployment scenario extends the Gateway Firewall scenarios depicted above with additional capability to insert the NGFW on top of native firewall capability NSX Gateway Firewall provides. This is applicable for the design where security compliance requirements mandate zones or groups of workloads be secured using NGFW, for example, DMZ or PCI zones or Multi-Tenant environments. The service insertion can be enabled per Gateway for both Tier-0 and Tier-1 Gateways depending on the scenario. Because traffic is redirected to partner services in a policy model, traffic can be redirected by protocol only for relevant traffic. This allows the insertion of partner firewalls in a surgical manner, without disruption to underlying network topology. As a best practice, Gateway firewall policy can be leveraged as the first level of defense to allow traffic based on L3/L4 policy and partner services as the second level defense. To do this, define policy on Gateway firewall to redirect the traffic which needs to be inspected by NGFW. This will optimize the NGFW performance and throughput, as well as reduce the capacity required of the partner service (which often impacts license cost).

The following diagram provides the logical representation of overall deployment scenario. Please refer to NSX interoperability matrix to check certified partners for the given use case.

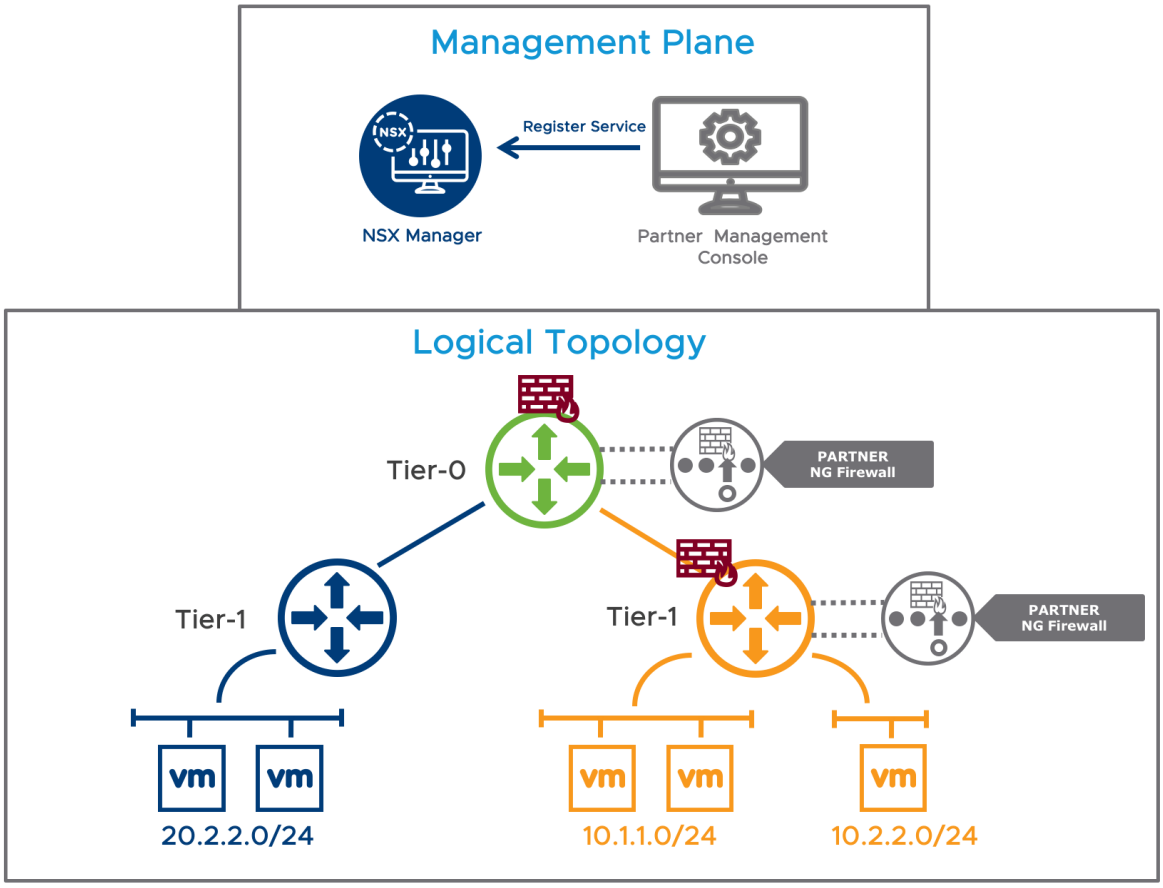


Figure 5-16: Gateway Firewall – Service Insertion

5.10 Endpoint Protection with NSX

NSX provides the Endpoint Protection platform to allow 3rd party partners to run agentless Anti-Virus/Anti-Malware (AV/AM) capabilities for virtualized workloads on ESXi. Traditional AV/AM services require agents be run inside the guest operating system of a virtual workload. These agents can consume small amounts of resources for each workload on an ESXi host. In the case of Horizon, VDI desktop hosts typically attempt to achieve high consolidation ratios on the ESXi host, providing 10s to 100s of desktops per ESXi host. With each AV/AM agent inside the virtualized workload consuming a small amount of virtual CPU and memory, the resource costs can be noticeable and possibly reduce the overall number of virtual desktops an ESXi host can accommodate, thus increasing the size and cost of the overall VDI deployment. The Guest Introspection platform allows the AV/AM partner to remove their agent from the virtual workload and provide the same services using a Service Virtual Machine (SVM) that is installed on each host. These SVMs consume much less virtual CPU and memory overall than running agents on every workload on the ESXi host. Removing the agent also removes that processing tax from the VDI, resulting in greater individual VDI performance. Many AV/AM partner solutions have the ability to add tags to the workloads based on the result of the AV/AM scan. This allows for an automated immediate quarantine policy based on the result of the AV/AM scan with the definition of DFW security rules based on the partner tags.

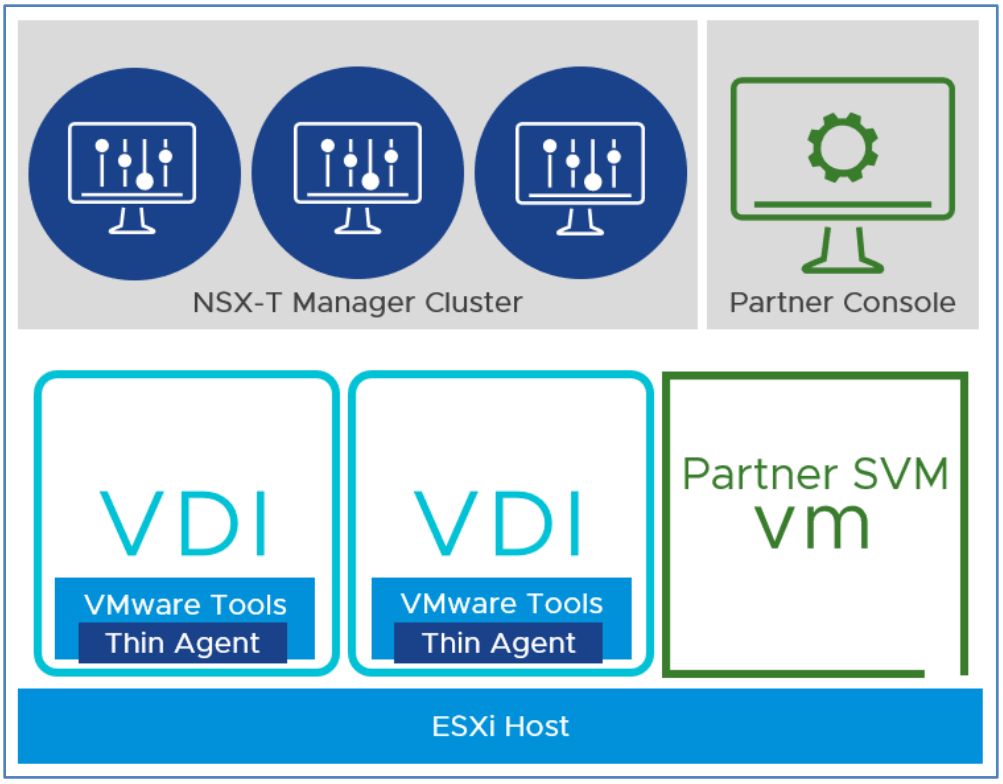


Figure 5-17: Endpoint Protection Components

The Endpoint Protection platform for NSX following a simple 3 step process to use.

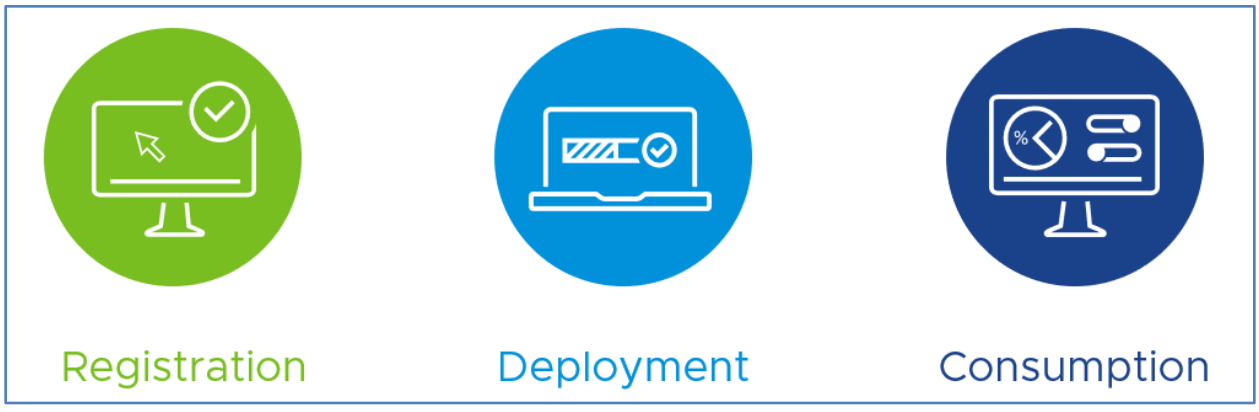


Figure 5-18: Endpoint Protection Steps

Registration

Registration of the VMware Partner console with NSX and vCenter.

Deployment

Creating a Service Deployment of the VMware Partner SVM and deployment to the ESXi Clusters. The SVMs require a Management network with which to talk to the Partner Management Console. This can be handled by IP Pool in NSX or by DHCP from the network. Management networks must be on a VSS or VDS switch.

Consumption

Consumption of the Endpoint Protection platform consists of creating a Service Profile of which references the Service Deployment and then creating Service Endpoint Protection Policy with Endpoint Rule that specifies which Service Profile should be applied to what NSX Group of Virtual Machines.

5.11 Recommendation for Security Deployments

This list provides best practices and recommendation for the NSX DFW. These can be used as guidelines while deploying an NSX security solution.

- For individual NSX software releases, always refer to release notes, compatibility guides, HARDENING GUIDE and recommended CONFIGURATION MAXIMUMS.
- Exclude management components like vCenter Server, and security tools from the DFW policy to avoid lockout, at least in the early days of DFW use. Once there is a level of comfort and proficiency, the management components can be added back in with the appropriate policy. This can be done by adding those VMs to the exclusion list.
- Use the Applied To field in the DFW to limit the rule growth on individual vNICs.
- Choose the policy methodology and rule model to enable optimum groupings and policies for micro-segmentation.
- Use NSX tagging and grouping constructs to group an application or environment to its natural boundaries. This will enable simpler policy management.
- Consider the flexibility and simplicity of a policy model for Day-2 operations. It should address ever-changing deployment scenarios rather than simply be part of the initial setup.
- Leverage DFW category and policies to group and manage policies based on the chosen rule model. (e.g., emergency, infrastructure, environment, application…)
- Use an explicit allow model; create explicit rules for allowed traffic and change DFW the default rule from “allow” to “drop”.

5.11.1 A Practical Approach to Building Micro-Segmentation Policy

The ideal way to have least privilege security model is to define security policies explicitly allowing all the applications within your data center and denying all other traffic by default. However, it is a big challenge to profile tens/hundreds of

applications and build security policies for each. Often times, application owners don't know very much about their application's details, which further complicates the process. This may take some time to profile your application and come up with a port defined security policy. Instead of waiting for profiling of an application to be complete, one can start with basic outside-in fencing approach to start defining broader security policies to enhancing the security posture and then move gradually to the desired explicit allow model over time as you complete the application profiling. The tag model of NSX allows this to be done quite easily as one can start with just Prod and Non-prod tags, then add more detailed tags as appropriate (PCI, HR, Finance, etc) to applicable VMs. (Note: Regardless of approach, experience has shown that the best starting point is common services such as DNS, NTP, AD, SNMP, etc. A great second step is the backup infrastructure.)

In this section we will walk you through a practical approach to start securing the data center workloads in a phased outside-in fencing approach as you are working on profiling your application to provide zero trust model.

First, we layout the data center topology and requirements. Then we will walk you through an approach to micro-segmentation policies in phases. This approach can be applied to both brownfield and green field deployment.

5.11.1.1 Data Center Topology and Requirements:

The following data center topology used which matches with most of our customer data center. This approach can be applied to both brownfield and greenfield deployment.

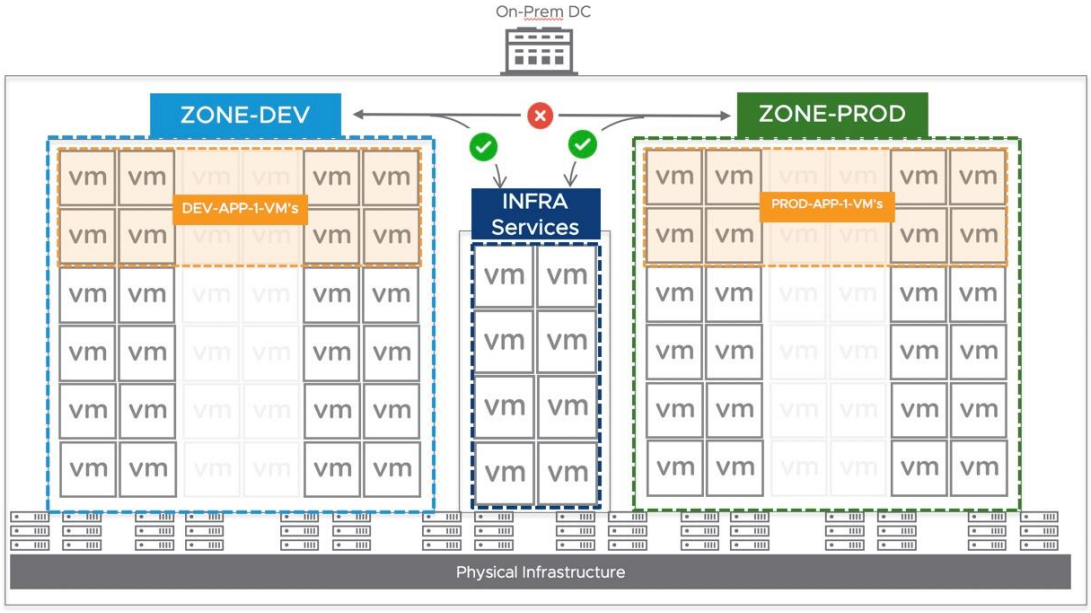


Figure 5-19: Data Center Topology Example

The data center has following characteristics:

1. Application deployment is split into two zones - production & development
2. Multiple applications hosted in both DEV and PROD ZONE

3. All application access same set of common services such as AD, DNS and NTP

The data center network has following characteristics:

1. The Zones have been assigned with dedicated IP CIDR block.
    a) Development zone has 10.1.16.0/20 and 10.1.32.0/20 IP CIDR block assigned
    b) Production zone has 10.2.16.0/20 and 10.2.32.0/20 IP CIDR block assigned.
2. Infrastructure Services have 10.3.16.0/24 subnet assigned.
3. The application within the ZONE would be given IP subnets within that ZONE specific CIDR block.
4. In most cases application VM's belonging to same application share same L2 segments. In some cases, they have separate L2 segments, especially for Database's. (In brownfield environments where L2 segments may be mixed populations of workloads, one can easily create static groups of workloads.)

The data center security has following Requirements:

1. All applications need to be allowed communicate with common Infrastructure services.
2. Between the ZONE - Workloads should not be allowed to communicate with each other.
3. Within the ZONE - Applications VM's belonging to a certain application should not be talking to other application VM's.
4. Some application within a ZONE have common Database services which runs within that ZONE.
5. Log all unauthorized communication between workloads for monitoring and for compliance.

5.11.1.2 Phased approach for NSX micro-segmentation policies:

Phase-1: Define common-services policy.

Here are the suggested steps:

1. Define NSX Groups for each of the Infrastructure Services. Following example shows the group for DNS and NTP servers with IP addresses of the respective servers as group members.

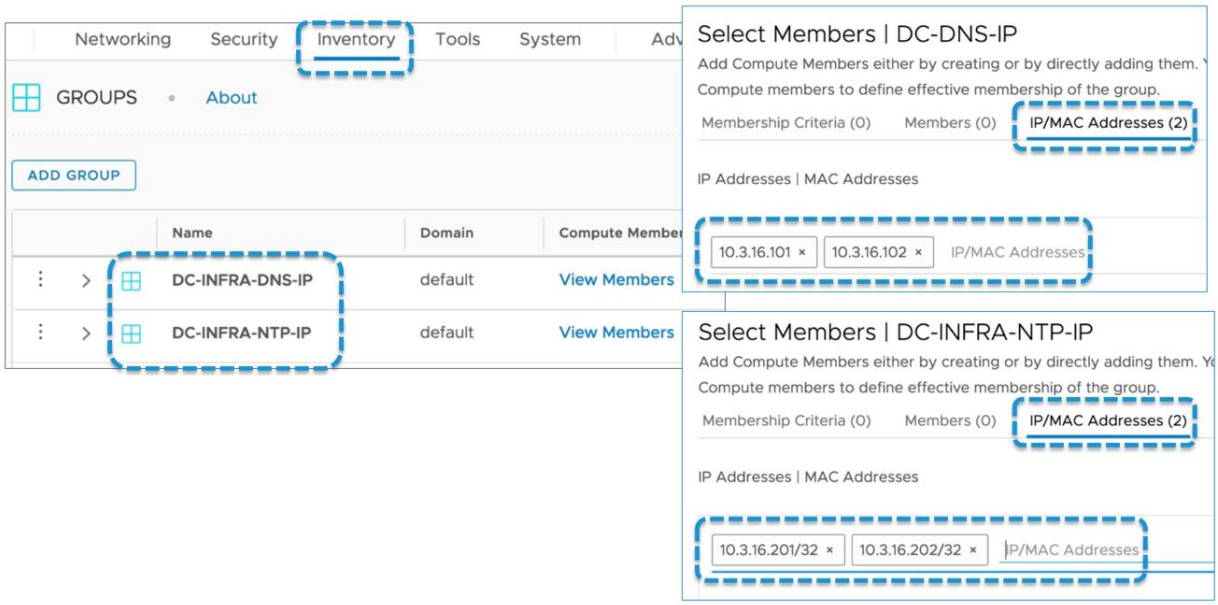


Figure 5-20: NSX Groups Example

2. Define policy for common services; like DNS, NTP as in the figure below.
   a) Define this policy under Infrastructure tab as shown below.
   b) Have two rules allows all workloads to access the common services using GROUPS created in step 1 above.
   c) Use Layer 7 context profile, DNS and NTP, in the rule to further enhance the security posture.
   d) Have catch-all deny rule to deny any other destination for the common services with logging enabled, for compliance and monitoring any unauthorized communication.

Note: If the management entities are not in an exclusion list, this section would need to have rules to allow the required protocols between the appropriate entities. See HTTPS://PORTS.VMWARE.COM/HOME/VSPHERE for the ports for all VMware products.

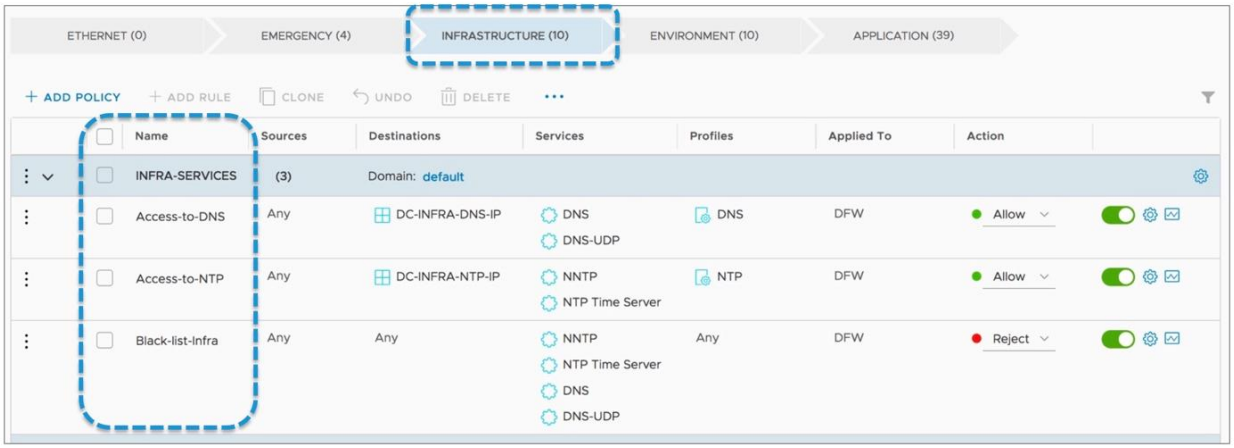


Figure 5-21: Common Services Policy Example

Phase-2: Define Segmentation around ZONES - by having an explicit allow policy between ZONES

As per the requirement, define policy between zones to deny any traffic between zones. This can be done using IP CIDR block as data center zones have pre-assigned IP CIDR block. Alternatively, this can be done using workload tags and other approach. However, IP-GROUP based approach is simpler (as admin has pre-assigned IP CIDR Block per zone), no additional workflow to tag workload and also less toll, compare to tagged approach, on NSX Manager and control plane. Tagged approach may add additional burden on NSX Manager to compute polices and update, in an environment with scale and churn. As a rule of thumb, the larger the IP block that can be defined in a rule, the more the policy can be optimized using CIDR blocks. In cases where there is no convenient CIDR block to group workloads, static groupings may be used to create entities without churn on the NSX Manager.

Here are the suggested steps:

1. Define 2 NSX Groups for each of the ZONE, Development and Production, say DC-ZONE-DEV-IP & DC-ZONE-PROD-IP with respective IP CIDR BLOCKs associated with the respective zones as members.

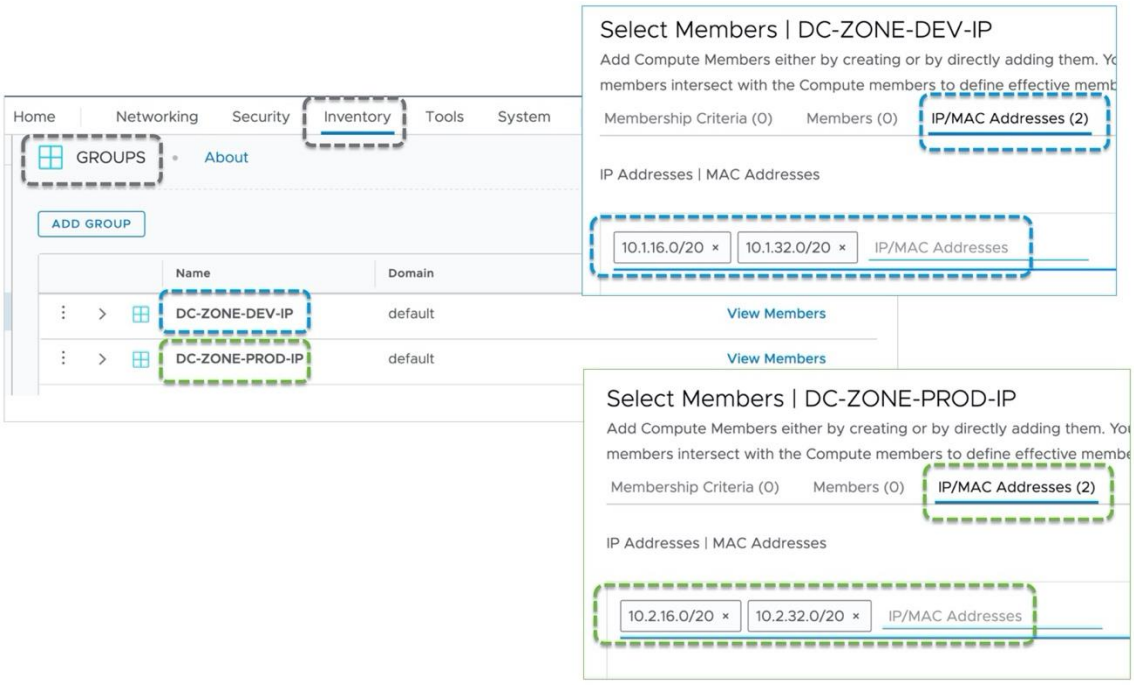


Figure 5-22: Policies Between Zones Example

2. Define policy in environment category using the IP GROUPS created in step-1 to restrict all communication between Development and Production ZONE's.
3. Have logging enabled for this policy to track all unauthorized communication attempts. (Note: In many industries, it is sufficient to log only the default action for troubleshooting purposes. In others, there may be a compliance mandate to log every action. Logging requirements are driven by the balance between storage costs and compliance requirements.)

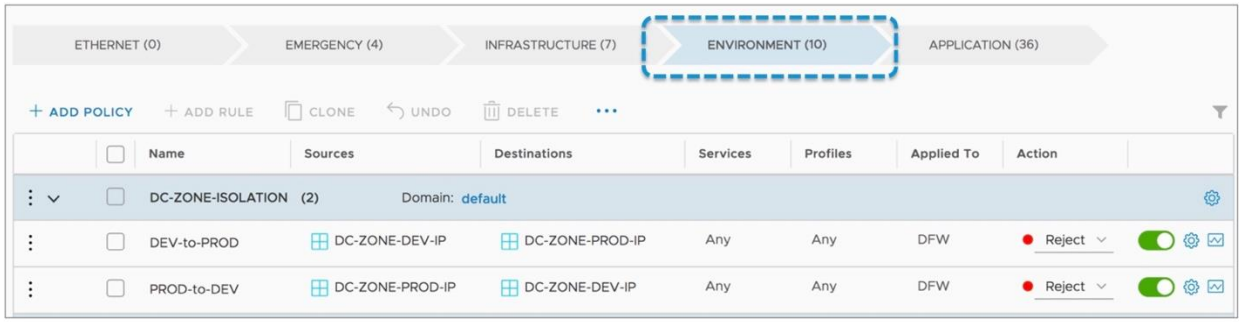


Figure 5-23: Policy Example

Phase-3: Define Segmentation around every Application, one at a time

This is two step approach to build a policy for the application. First step is to start with fence around application to build security boundary. Then as a second step profile the application further to plan and build more granular port-defined security policies between tiers.

- Start with DEV zone first and identify an application to be micro-segmented, say DEV-ZONE-APP-1.
- Identify all VM's associated with the Application within the zone.
- Check application has its own dedicated network Segments or IP Subnets.
    - If yes, you can leverage Segment or IP-based Group.
    - If no, tag application VM's with uniquely zone and application specific tags, say ZONE-DEV & APP-1.
- Check this application requires any other communication other than infra services and communication within group. For example, APP is accessed from outside on HTTPS.

Once you have above information about DEV-ZONE-APP-1, create segmentation around application by following steps:

1- Apply two tags to all the VM's belonging to APP-1 in the ZONE DEV, ZONE-DEV & APP-1.

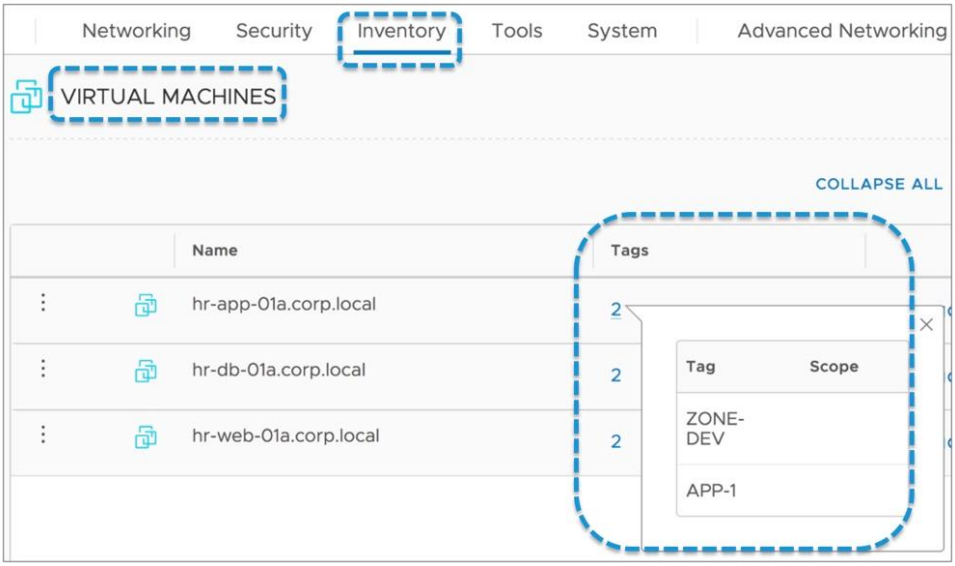


Figure 5-24: Segmentation Example

2- Create a GROUP, say "ZONE-DEV-APP-1" with criteria to match on tag equal to "ZONE-DEV & APP-1".

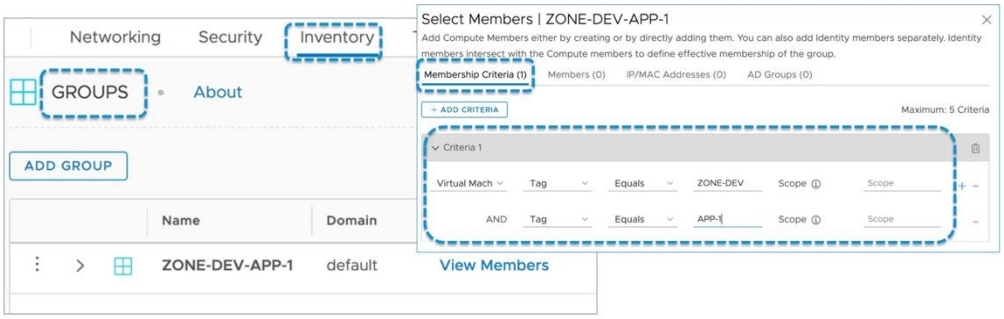


Figure 5-25: Group Example

3- Define a policy under Application category with 3 rules as in the FIGURE 5-26: APPLICATION POLICY EXAMPLE.

   a. Have "Applied To" set to "ZONE-DEV-APP-1" to limit the scope of policy only to the application VM's.

   b. The first rule allows all internal communications between the application VM's. Enable logging for this rule to profile the application tiers and protocols. (Each log entry will contain 5 tuple details about every connection.)

   c. The second rule allows access to front end of the application from outside. Use the L7 context profile to allow only SSL traffic. The below example uses Exclude Source from within ZONE, so that application is only accessible from outside, not from within except APP's other VM's, as per rule one.

   d. Default deny all other communication to these "ZONE-DEV-APP-1" VM's. Enable log for compliance and monitoring any unauthorized communication.

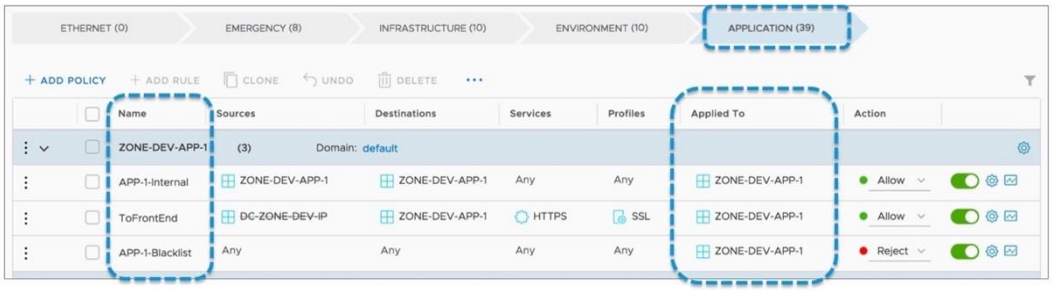


Figure 5-26: Application Policy Example

Phase-4: Review Logs for Application Profile.

Log entries will identify the direction (In/Out) as well as the protocol and source IP address/port and destination IP addresses/port for each flow. If using the log file for policy definition, it is often advisable to process the log files using excel to sort traffic. Typically, 2 sheets are created, one for IN traffic and one for OUT traffic. Then, each sheet is sorted by port first then IP address. (In the case of IN traffic

by destination IP address and in the case of OUT traffic by source address. This sorting methodology allows for the grouping of multiple servers serving/accessing the same traffic.) For each of these groupings, a rule can be inserted above rule 1 for the application. This will prevent the known traffic from appearing in the log. Once sufficient confidence is gained that the application is completely understood (this is typically when the logs are empty), the original rule ZONE-DEV-APP-1 can be removed. At this point, the security model has transitioned from zone-based to micro segmentation. (Note: Certain environments - such as labs - may be best served by ring fencing, whereas other environments may wish to add service insertion for certain traffic types on top of micro segmentation – such as sensitive financial information. The value of NSX is that a customer provides the means to implement appropriate security in one environment without impacting the other.)

Phase-5: Repeat Phase-3 for other applications and ZONES.

Repeat the same approach as in Phase-3 for other applications, to have security boundary for every application within the ZONE-DEV and ZONE-PROD. Note that the securing of each of these applications can happen asynchronously, without impact to the others. This accommodates application-specific maintenance windows, where required.

Phase-6: Define Emergency policy, Kill Switch, in case of Security Event

An emergency policy mainly leveraged for following use case and enforced on top of the firewall table:

1- To quarantine vulnerable or compromised workloads in order to protect other workloads.

2- May want to explicitly deny known bad actors by their IP Subnet based on GEO location or reputation.

This policy is defined in Emergency Category as shown:

1- First two rules quarantine all traffic from workloads belonging to group GRP-QUARANTINE.

   a. “GRP-QUARANTINE” is a group which matches all VM with tag equal to “QUARANTINE”. (If guest introspection is implemented, the AV/AM tags can be used to define different quarantine levels.)

   b. In order to enforce this policy to vulnerable VM’s, add tag “QUARANTINE” to isolate the VM’s and allow only admin to access the hosts to fix the vulnerability.

2- Other two rule uses Group with known bad IP’s to stop any communication with those IP’s.

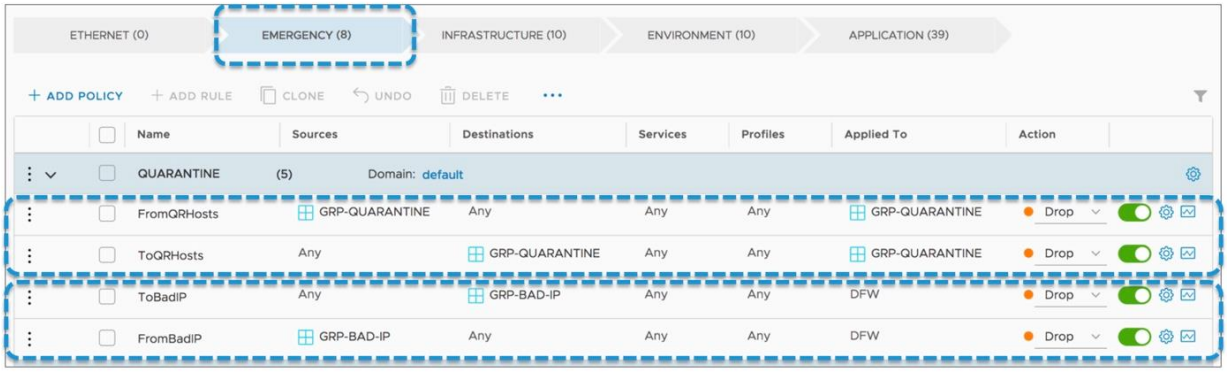


Figure 5-27: Emergency Category Example

In creating these policies, the iterative addition of rules to the policy is something that can be done at any time. It is only when the action of the default rule changes from allow to deny/drop that a maintenance window is advised. As logging has been on throughout the process, it is highly unusual to see an application break during the window. What is most frequently the case is that something within the next week or month may emerge as an unforeseen rule that was missed. For this reason, it is advised that even in environments where compliance does not dictate the collection of logs, the Deny All rule be set to logging. Aside from the security value of understanding the traffic that is being blocked, the Deny All rule logs are very useful when troubleshooting applications.

At this point you have basic level of micro-segmentation policy applied to all the workloads to shrink the attack surface. As a next step you further break the application into application tiers and its communication by profiling application flows using firewall logs or exporting IPFIX flows to Network Insight platform. This will help to group the application workload based on the function within the application and define policy based on associated port & protocols used. Once you have these groupings and protocols identified for a given application, update the policy for that application by creating additional groups and rules with right protocols to have granularly defined rules one at a time.

With this approach you start with outside-in fencing to start with micro-segmentation policies and finally come up with a granular port-based micro-segmentation policy for all the application.

5.12 NSX Firewall- For All Deployment Scenario

NSX firewall provides different security controls: Distribute Firewall, Distributed IDS, Gateway Firewall & Bridge Firewall, as an option to provide firewalling to different deployment scenarios.

A typical data center would have different workloads: VM's, Containers, Physical Server, and a mix of NSX managed and non-managed workloads. These workloads may also have a combination of a VLAN-based network or an NSX based overlay network.

The following FIGURE 5-28: NSX FIREWALL FOR ALL DEPLOYMENT SCENARIO summarizes different datacenter deployment scenarios and associated NSX firewall security controls, which best fits the design. You can use same NSX

manager as a single pane of glass to define Security policies to all of these different scenarios using different security controls.

1- NSX Managed Workloads with standard VLAN based networking.

- NSX Distributed Firewall can be used to protect NSX managed VM's, Containers & Physical Server workloads.

2- NSX Managed Workloads with NSX Overlay for networking:

- NSX Distributed Firewall can be used to protect NSX managed VM's, Containers & Physical Server workloads.

3- Non-NSX Managed workloads on traditional VLAN based network.

- NSX Gateway Firewall can provide the Inter VLAN routing and Firewalling. The Service Interface on NSX Gateway is used as a gateway & firewall for all non-NSX managed VLAN workloads.

4- NSX managed Overlay workload bridged to Non-NSX managed VLAN.

- This is the bridge scenario where an Overlay network is extended at Laye-2 into a VLAN network using NSX Bridge. In this case, NSX managed Overlay workloads can use DFW/D-IDS, and Bridge Firewall can secure traffic at the boundary between VLAN and overlay network.

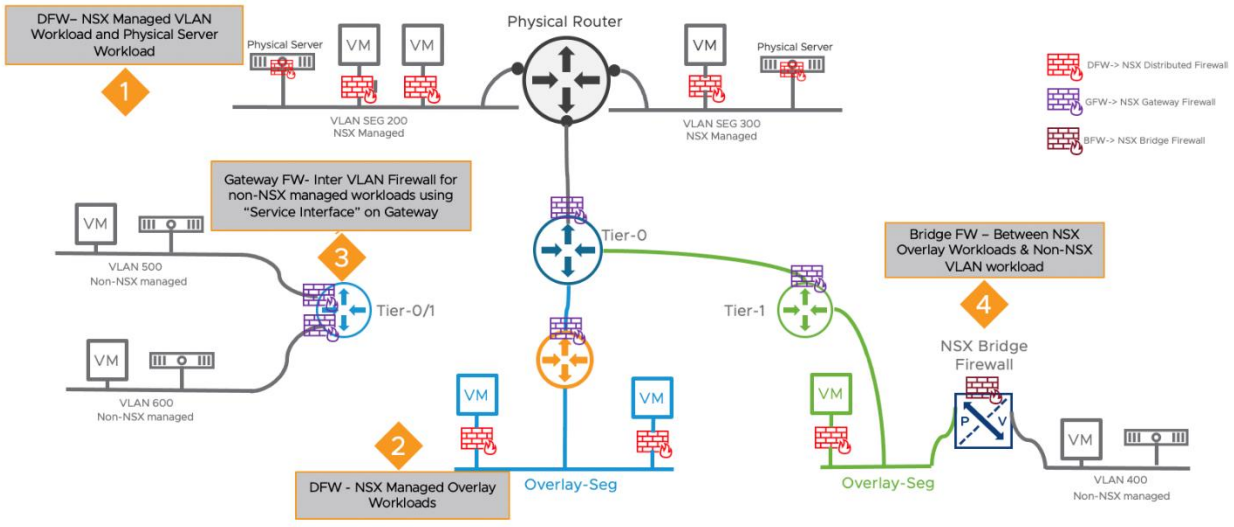


Figure 5-28: NSX Firewall For all Deployment Scenario

6 NSX Load Balancer

6.1 NSX Load Balancer Roadmap

This chapter presents an overview of the NSX Native Load Balancer. Starting with NSX version 3.2, NSX native load balancing features will not be added or enhanced, and NSX platform enhancements will not be extended to the NSX native load balancer. The official announcement is part of the NSX 3.2 RELEASE NOTES. The release notes include the following recommendations:

- It is recommended that new deployments with NSX take advantage of VMware NSX Advanced Load Balancer (Avi) using release v20.1.6 or later and not use the native NSX Load Balancer.
- If you are using Load Balancing in NSX, you are advised to migrate to VMware NSX Advanced Load Balancer (Avi), which provides a superset of the NSX load balancing functionality.
- If you have purchased NSX Advanced, NSX Enterprise Plus, NSX Advanced, or NSX Enterprise, you are entitled to the Basic edition of VMware NSX Advanced Load Balancer (Avi), which has feature parity with NSX LB.
- It is recommended that you purchase VMware NSX Advanced Load Balancer (Avi) Enterprise to unlock enterprise grade load balancing, GSLB, advanced analytics, container ingress, application security, and WAF.

The NSX DESIGN GUIDE FOR AVI VANTAGE describes the integration between NSX Advanced load Balancer (AVI) and NSX Datacenter. The process outlined in that document is the recommended integration path, and any new deployments should follow its guidelines.

NSX 3.2 introduced additional integrations between NSX Datacenter and NSX Advanced Load Balancer at the UI/API level. Such functionalities include:

1. An installation workflow that allows deploying NSX Advanced Load Balancer through the NSX Datacenter UI
2. Launch re NVMwaSX ALB (Avi) UI from the NSX Manager
3. Configure VMware NSX Advanced Load Balancer (Avi) from within NSX Manager.
4. The NSX Policy API can be used to manage the NSX Advanced Load Balancer configurations of virtual services and their dependent objects (Since NSX 3.1.1)

Integrations 3 and 4 have been deprecated and will be removed in a future release. For this reason, we recommend avoiding making them part of any deployment. The official deprecation announcement is available in this KNOWLEDGE BASE ARTICLE.

6.2 Load Balancing Overview

A load-balancer defines a virtual service, or virtual server, identified by a virtual IP address (VIP) and a UDP/TCP port. This virtual server offers an external

representation of an application while decoupling it from its physical implementation: traffic received by the load balancer can be distributed to other network-attached devices that will perform the service as if it was handled by the virtual server itself. This model is popular as it provides benefits for application scale-out and high-availability:

- Application scale-out:
  The following diagram represents traffic sent by users to the VIP of a virtual server, running on a load balancer. This traffic is distributed across the members of a pre-defined pool of capacity.

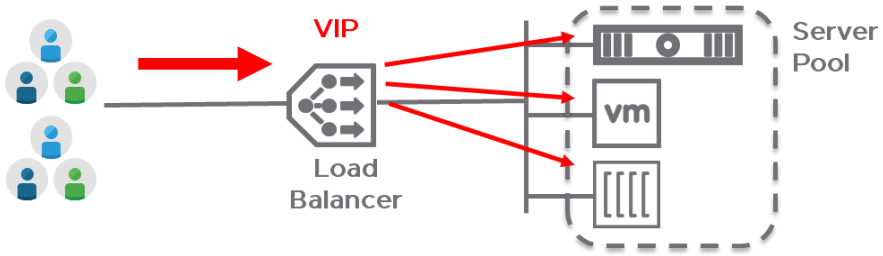


Figure 6-1: Load Balancing Offers Application Scale-out

  The server pool can include an arbitrary mix of physical servers, VMs or containers that together, allow scaling out the application.

- Application high-availability:
  The load balancer is also tracking the health of the servers and can transparently remove a failing server from the pool, redistributing the traffic it was handling to the other members:

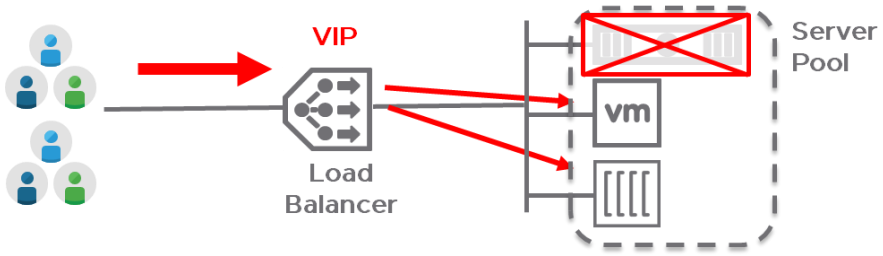


Figure 6-2: Load Balancing Offers Application High-availability

Modern applications are often built around **advanced load balancing** capabilities, which go far beyond the initial benefits of scale and availability. In the example below, the load balancer selects different target servers based on the URL of the requests received at the VIP:

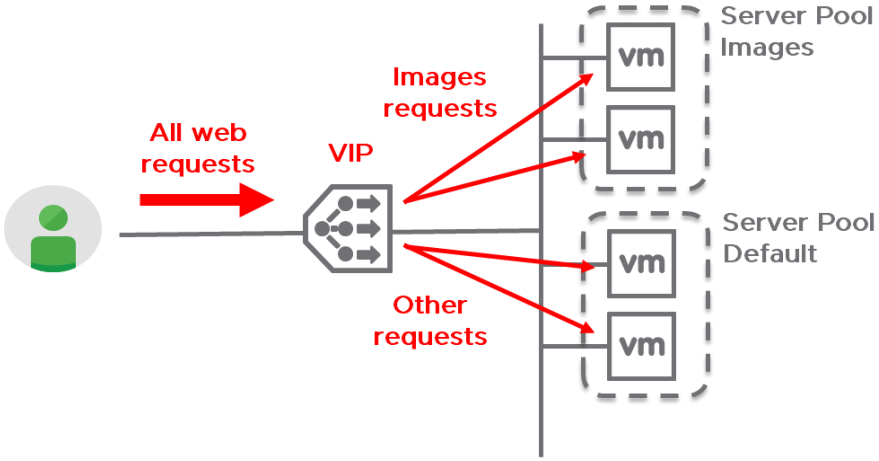


Figure 6-3: Load Balancing Offers Advanced Application Load Balancing

Thanks to its native capabilities, modern applications can be deployed in NSX without requiring any third party physical or virtual load balancer. The next sections in this part describe the architecture of the NSX load balancer and its deployment modes.

6.3 NSX Load Balancing Architecture

In order to make its adoption straightforward, the different constructs associated to the NSX load balancer have been kept similar to those of a physical load balancer. The following diagram show a logical view of those components.

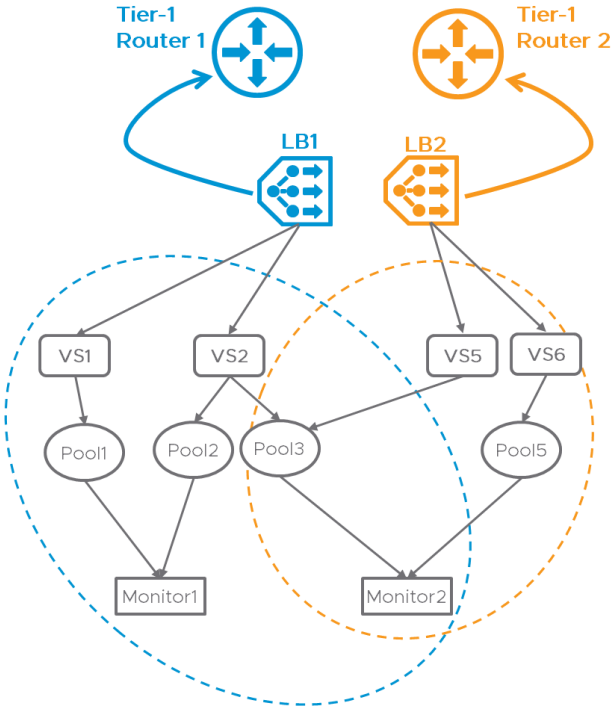


Figure 6-4: NSX Load Balancing Main Components

Load Balancer

The NSX load balancer is running on a Tier-1 gateway. The arrows in the above diagram represent a dependency: the two load balancers LB1 and LB2 are respectively attached to the Tier-1 gateways 1 and 2. Load balancers can only be attached to Tier-1 gateways (not Tier-0 gateways), and one Tier-1 gateway can only have one load balancer attached to it.

Virtual Server

On a load balancer, the user can define one or more virtual server (the maximum number depends on the load balancer form factor – See NSX Administrator Guide for load balancer scale information). As mentioned earlier, a virtual server is defined by a VIP and a TCP/UDP port number, for example IP: 20.20.20.20 TCP port 80. The diagram represents four virtual servers VS1, VS2, VS5 and VS6. A virtual server can have basic or advanced load balancing options such as forward specific client requests to specific pools (see below), or redirect them to external sites, or even block them.

Pool

A pool is a construct grouping servers hosting the same application. Grouping can be configured using server IP addresses or for more flexibility using Groups. NSX provides advanced load balancing rules that allow a virtual server to forward traffic to multiple pools. In the above diagram for example, virtual server VS2 could load balance image requests to Pool2, while directing other requests to Pool3.

Monitor

A monitor defines how the load balancer tests application availability. Those tests can range from basic ICMP requests to matching patterns in complex HTTPS queries. The health of the individual pool members is then validated according to a simple check (server replied), or more advanced ones, like checking whether a web page response contains a specific string.

Monitors are specified by pools: a single pool can use only 1 monitor, but the same monitor can be used by different Pools.

6.4 NSX Load Balancing deployment modes

NSX load balancer is flexible and can be installed in either traditional in-line or one-arm topologies. This section goes over each of those options and examine their traffic patterns.

6.4.1 In-line load balancing

In in-line load balancing mode, the clients and the pool servers are on different side of the load balancer. In the design below, the clients are on the Tier-1 uplink side, and servers are on the Tier-1 downlink side:

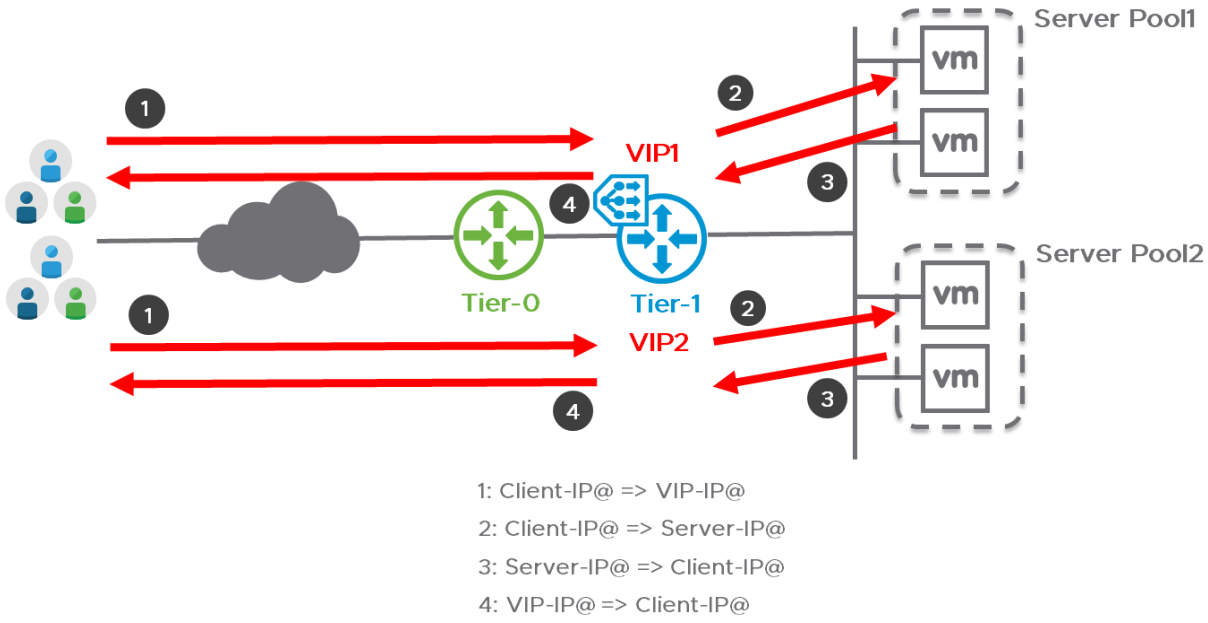


Figure 6-5: In-Line Load Balancing

Because the traffic between client and servers necessarily go through the load-balancer, there is no need to perform any LB Source-NAT (Load Balancer Network Address Translation at virtual server VIP).

The in-line mode is the simplest load-balancer deployment model. Its main benefit is that the pool members can directly identify the clients from the source IP address, which is passed unchanged (step2). The load-balancer being a centralized service, it is instantiated on a Tier-1 gateway SR (Service Router). The drawback from this model is that, because the Tier-1 gateway now has a centralized component, East-West traffic for Segments behind different Tier-1 will be pinned to an Edge node in order to get to the SR. This is the case even for traffic that does not need to go through the load-balancer.

6.4.2 One-arm load balancing

In one-arm load balancing mode, both client traffic (client traffic to the load-balancer VIP) and server traffic (load-balancer to server) use the same load balancer interface. In that case, LB-SNAT will be used to make sure that the traffic from the servers back to the client indeed go through the load-balancer. There are two variations over this one-arm load-balancing scenario: a case where both clients are servers are on the same subnet and a case where they are on different subnets. For both cases the solution leverages load-balancer source NAT in order to make sure that traffic from a server to its clients is directed to the load-balancer. As a result, the server will not see the real IP address of the clients. Note that the load-balancer can inject an “X-Forwarded-For” header for HTTP/HTTPS traffic in order to work around this issue.

6.4.2.1 Clients and servers on the same subnet

In the design below, the clients and servers are on the same Tier-1 gateway downlink.

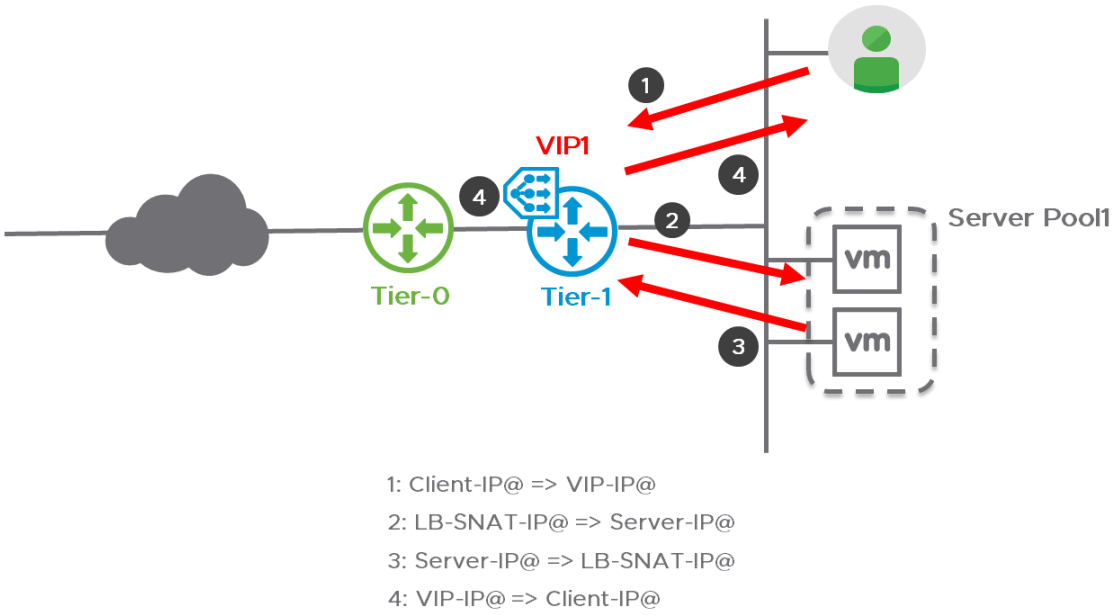


Figure 6-6: One-Arm Load Balancing with Clients and Servers on the same segment

The need for a Tier-1 SR in for the centralized load-balancer service result in East-West traffic for Segments behind different Tier-1 being pinned to an Edge node. This is the same drawback as for the inline model described in the previous part.

6.4.2.2 Load Balancer One-Arm attached to Segment

In the design below, the blue Tier-1 gateway does not run any load-balancer service. Instead, the load-balancer has been deployed as a standalone Tier-1 gateway (represented in orange in the diagram), with a single Service Interface. This gateway is acting as an appliance instantiating a load-balancer. This way, several segments below the blue Tier-1 gateway can have their own dedicated load-balancer.

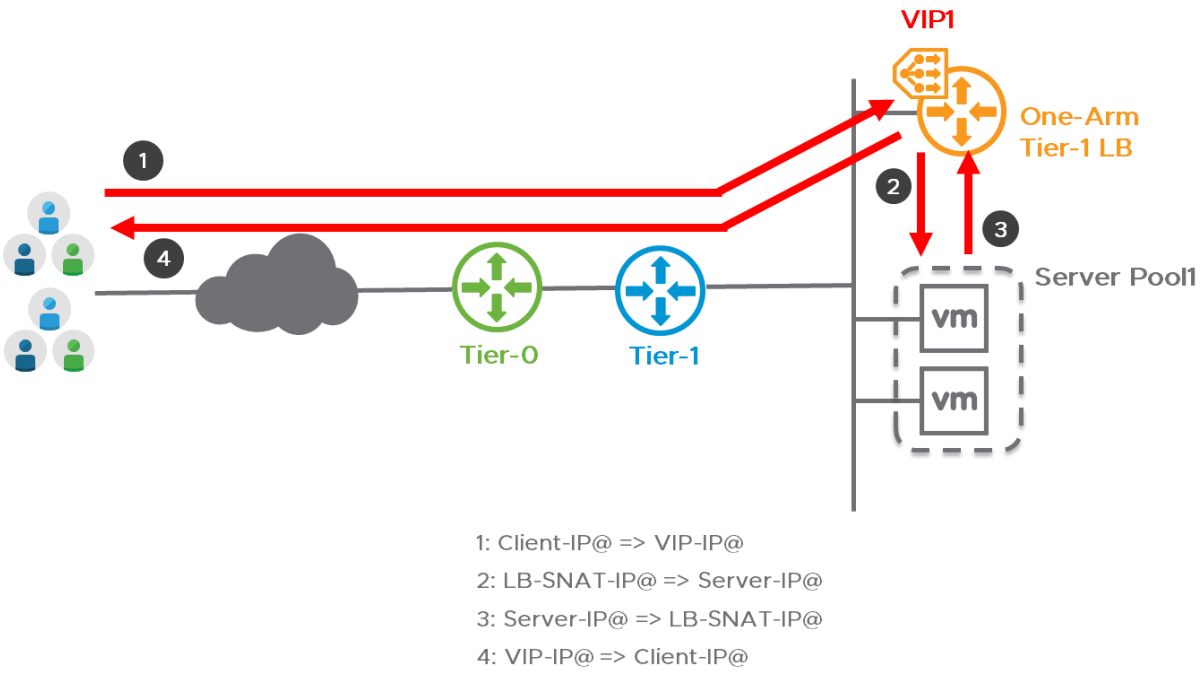


Figure 6-7: Load Balancer One-Arm attached to segment overlay

This design allows for better horizontal scale, as an individual segment can have its own dedicated load-balancer service appliance(s). This flexibility in the assignment of load-balancing resources comes at the expense of potentially instantiating several additional Tier-1 SRs on several Edge nodes. Because the load-balancer service has its dedicated appliance, in East-West traffic for Segments behind different Tier-1 gateway (the blue Tier-1 gateway in the above diagram) can still be distributed. The diagram above represented a Tier-1 One-Arm attached to overlay segment.

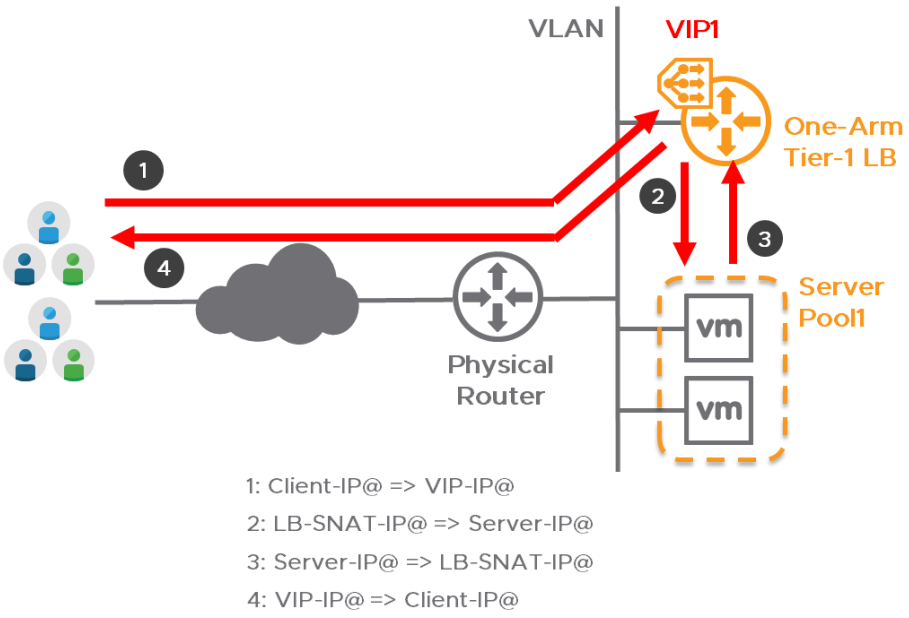


Figure 6-8: Load Balancer One-Arm attached to segment VLAN

Tier-1 One-Arm LB can also be attached to physical VLAN segments as shown in above figure, and thus offering load balancing service even for applications on VLAN. In this use case, the Tier-1 interface is also using a Service Interface, but this time connected to a segment-VLAN instead of a segment-overlay.

6.5 NSX load-balancing technical details

This section provides additional details on how the load-balancer components are physically implemented in an NSX environment. Even if it's not necessary for implementing the designs described in the previous part, understanding the traffic flow between the components, the high-availability model or the way the monitor service is implemented will help the reader optimize resource usage in their network.

6.5.1 Load-balancer high-availability

The load-balancer is a centralized service running on a Tier-1 gateway, meaning that it runs on a Tier-1 gateway Service Router (SR). The load-balancer will thus run on the Edge node of its associated Tier-1 SR, and its redundancy model will follow the Edge high-availability design.

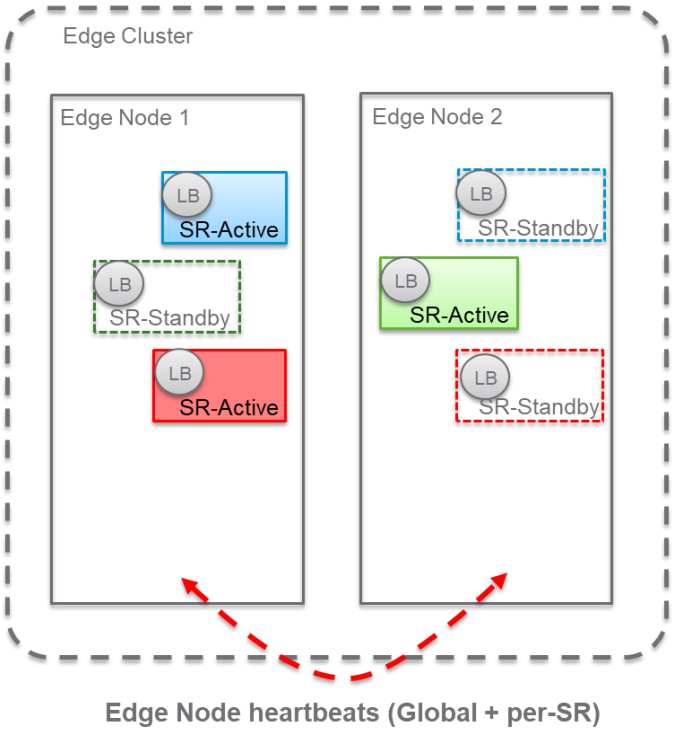


Figure 6-9: Load Balancing High-Availability

The above diagram represents two Edge nodes hosting three redundant Tier-1 SRs with a load-balancer each. The Edge High Availability (HA) model is based on periodic keep alive messages exchanged between each pair of Edges in an Edge Cluster. This keepalive protects against the loss of an Edge as a whole. In the above diagram, should Edge node 2 go down, the standby green SR on Edge node 1, along with its associated load-balancer, would become active immediately.

There is a second messaging protocol between the Edges. This one is event driven (not periodic), and per-application. This means that if a failure of the load-balancer of the red Tier-1 SR on Edge node 1 is detected, this mechanism can trigger a failover of just this red Tier-1 SR from Edge node 1 to Edge node 2, without impacting the other services.

The active load balancer service will always synchronize the following information to the standby load balancer:

- State Synchronization
- L4 Flow State
- Source-IP Persistence State
- Monitor State

This way, in case of failover, the standby load balancer (and its associated Tier-1 SR) can immediately take over with minimal traffic interruption.

6.5.2 Load-balancer monitor

The pools targeted by the virtual servers configured on a load-balancer have their monitor services running on the same load-balancer. This ensure that the monitor service cannot fail without the load-balancer failing itself (fate sharing.) The left part of the following diagram is representing the same example of relation between the different load-balancer components as the one used in part 6.3. The right part of the diagram is providing an example of where those components would be physically located in a real-life scenario.

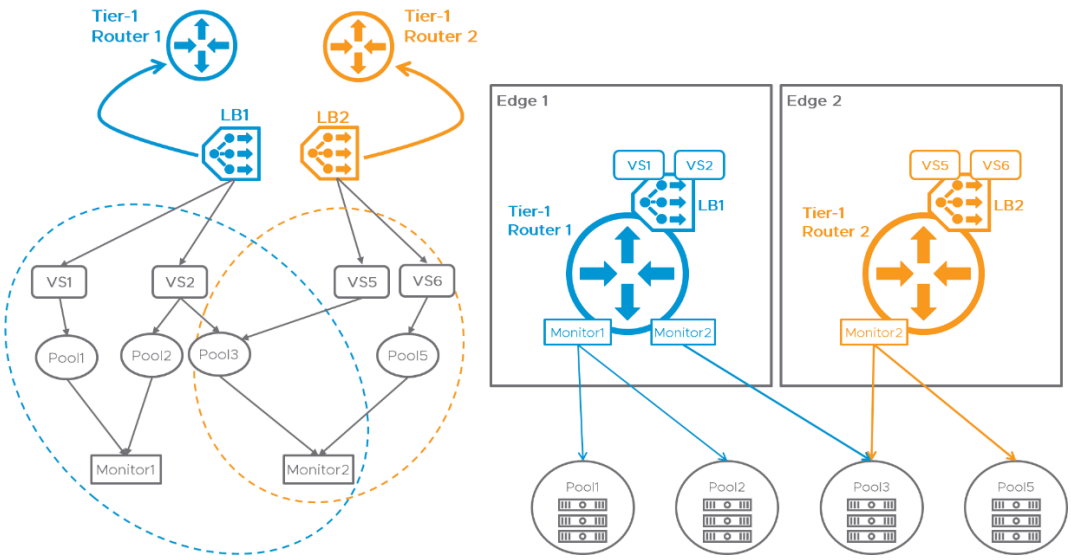


Figure 6-10: NSX Load Balancer Monitor

Here, LB1 is a load-balancer attached to Tier-1 Gateway 1 and running two virtual servers VS1 and VS2. The SR for Tier-1 Gateway 1 is instantiated on Edge 1. Similarly, load-balancer LB2 is on gateway Tier-1 Gateway 2, running VS5 and VS6.

Monitor1 and Monitor2 protecting server pools Pool1, Pool2 and Pool3 used by LB1. As a result, both Monitor1 and Monitor2 are implemented on the SR where LB1 reside. Monitor2 is also polling servers used by LB2, thus it is also implemented on the SR where LB2 is running. The Monitor2 example highlights the fact that a monitor service can be instantiated in several physical locations and that a given pool can be monitored from different SRs.

6.5.3 Load-balancer Layer4 or Layer7 load balancing

NSX LB offers Layer4 (L4) and Layer7 (L7) load balancing. L4 VIP load balances UDP and TCP connections.The client connection is load balanced by the VIP and its connection is terminated by one of the pool members.

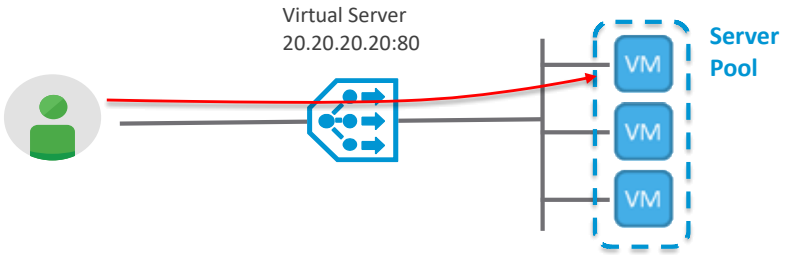


Figure 6-11: NSX L4 VIP

L7 VIP load balances HTTP or HTTPS connections. The client connection is terminated by the VIP, and once the client's HTTP or HTTPS request is received then the load balancer establishes another connection to one of the pool members. If needed, some specific load balancing configuration can also be done by L7 VIP, like a selection of specific pool members based on the request.

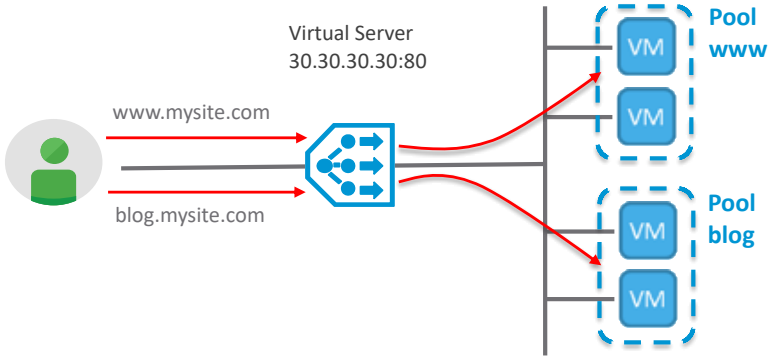


Figure 6-12: NSX L7 VIP

For L7 VIP HTTPS, NSX LB offers 3 modes: HTTPS Off-Load, HTTPS End-to-End SSL, and SSL Passthrough.

HTTPS Off-Load decrypts the HTTPS traffic at the VIP and forward the traffic in clear HTTP to the pool members. It is the best balance between security, performance, and LB flexibility:

- Security: traffic is encrypted on the external side.
- Performance: web servers don't have to run encryption.
- LB flexibility: all advanced configuration on HTTP traffic available like URL load balancing.

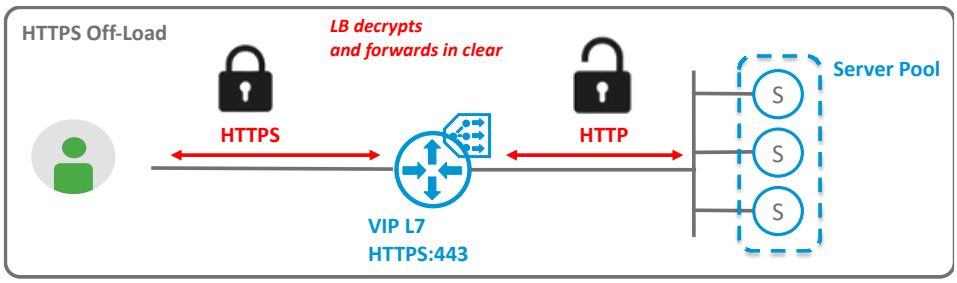


Figure 6-13: NSX L7 HTTPS Off-Load VIP

HTTPS End-to-End SSL decrypts the HTTPS traffic at the VIP and re-encrypts the traffic in another HTTPS session to the pool members. It is the best security and LB flexibility:

- Security: traffic is encrypted end to end.
- Performance: this mode has lower performance with traffic decrypted/encrypted twice.
- LB flexibility: all advanced configuration on HTTP traffic available like URL load balancing.

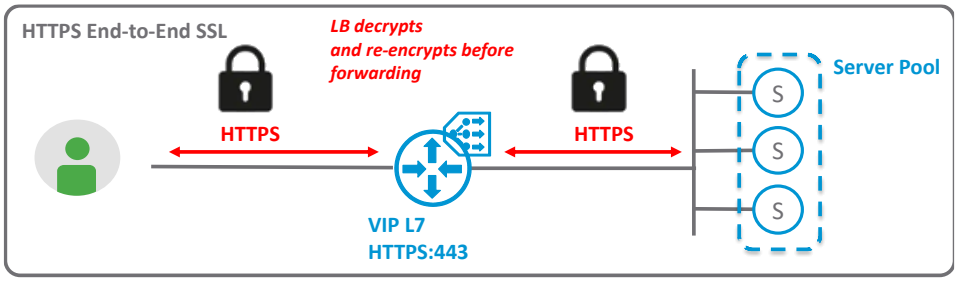


Figure 6-14: NSX L7 HTTPS End-to-End VIP

HTTPS SSL Passthrough does not decrypt the HTTPS traffic at the VIP and SSL connection is terminated on the pool members. It is the best security and performance, but with limited LB flexibility:

- Security: traffic is encrypted end to end.
- Performance: highest performance since LB does not terminate SSL traffic
- LB flexibility: advanced configuration based on HTTP traffic is not available. Only advanced configuration based on SSL traffic is available like SSL SNI load balancing.

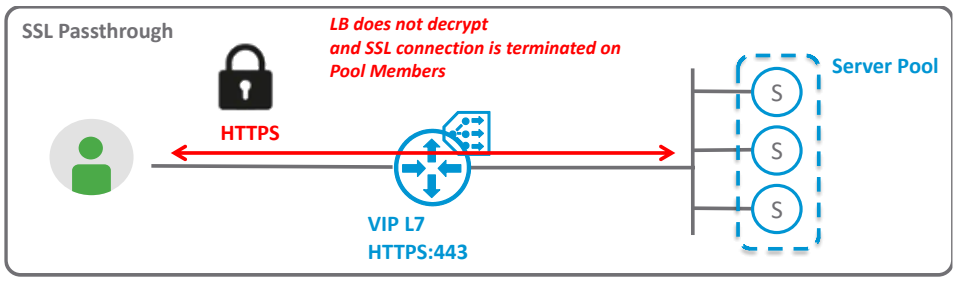


Figure 6-15: NSX L7 SSL Passthrough VIP

6.5.4 Load-balancer IPv6

NSX LB has many NSX network and security services offers its service for IPv4 and IPv6 clients.

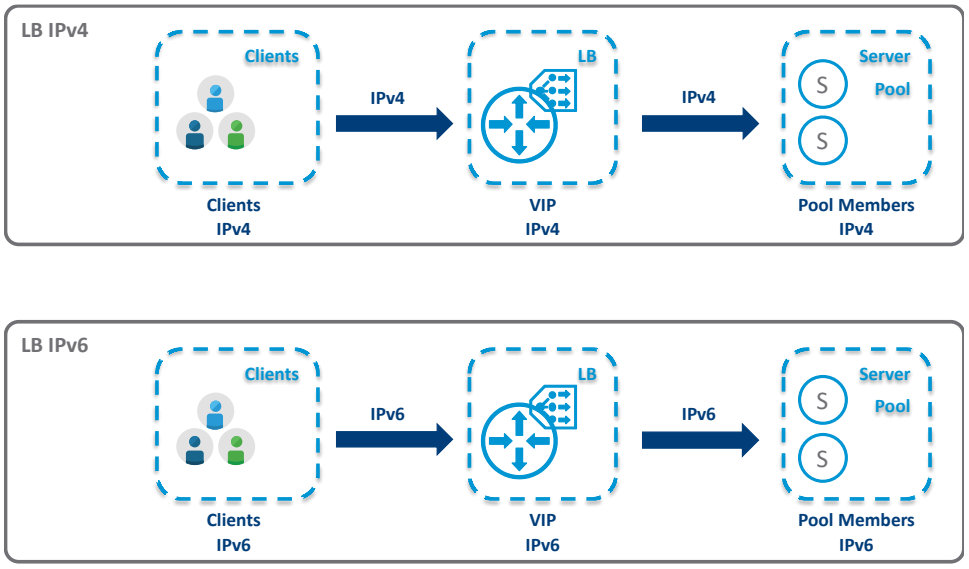


Figure 6-16: NSX LB IPv4 and IPv6

6.5.5 Load-balancer traffic flows

Tier-1 gateway looks like a single entity from a logical point of view. However, and as mentioned several times already, when a load-balancer is configured on a Tier-1 gateway, it is physically instantiated on a Tier-1 Gateway Service Router. This part is exploring the scenarios where the logical representation of a Tier-1 gateway, hiding the distinction between SR and DR, can lead to confusion.

6.5.5.1 The in-line model

With the in-line model, traffic between the clients and the servers necessarily go through the load balancer. Thanks to this property, there is no need for source LB-SNAT in order to make sure that traffic goes through the load balancer both ways. The following diagram shows both logical and physical representation of a Tier-1 gateway used to host a load-balancer operating in-line. Clearly, traffic from clients must go through the Tier-1 SR where the load-balancer is instantiated in order to reach the server and vice versa:

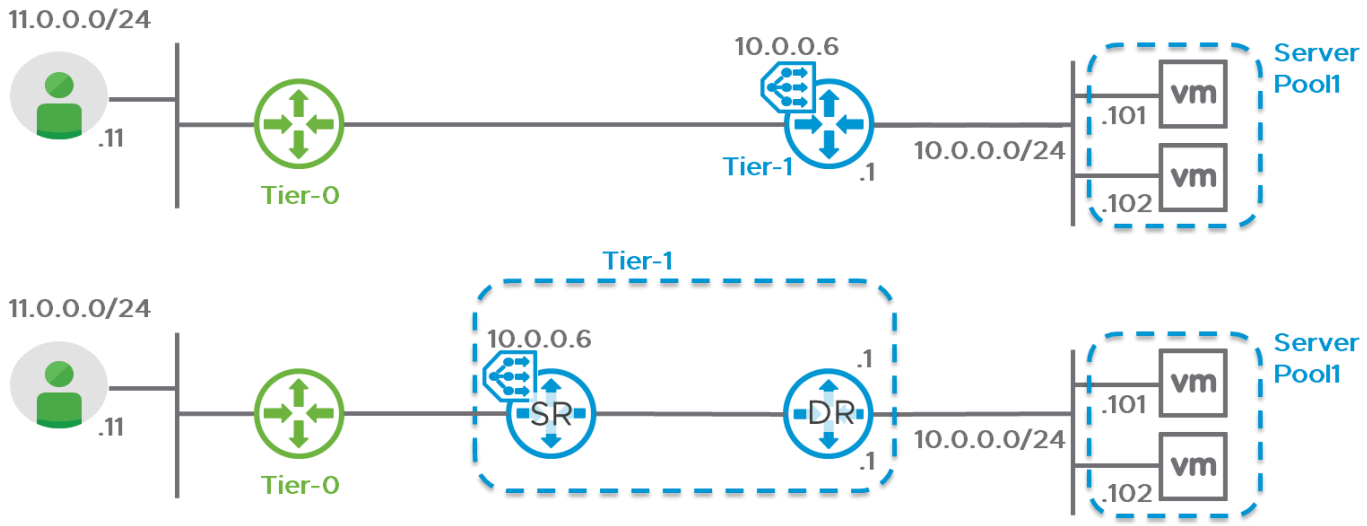


Figure 6-17: In-Line Model: Logical and Expanded View

The following diagram represents another scenario that, from a logical standpoint at least, looks like an in-line load-balancer design. However, source LB-SNAT is required in this design, even if the traffic between the clients and the servers cannot apparently avoid the Tier-1 gateway where the load-balancer is instantiated.

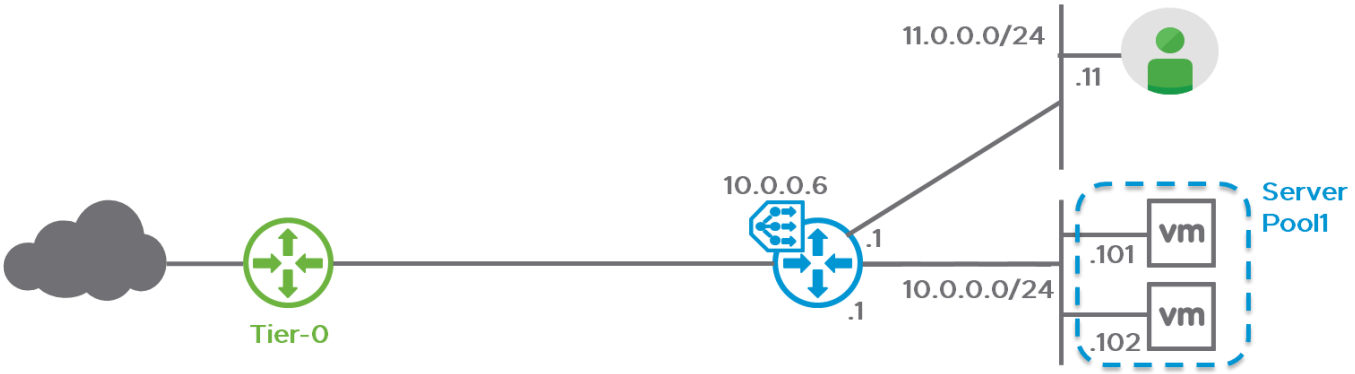


Figure 6-18: Load Balancing VIP IP@ in Tier-1 Downlink Subnet – Tier-1 Expanded View

The following expanded view, where the Tier-1 SR and DR are represented as distinct entities and hosted physically in different location in the network, clarifies the reason why source LB-SNAT is mandatory:

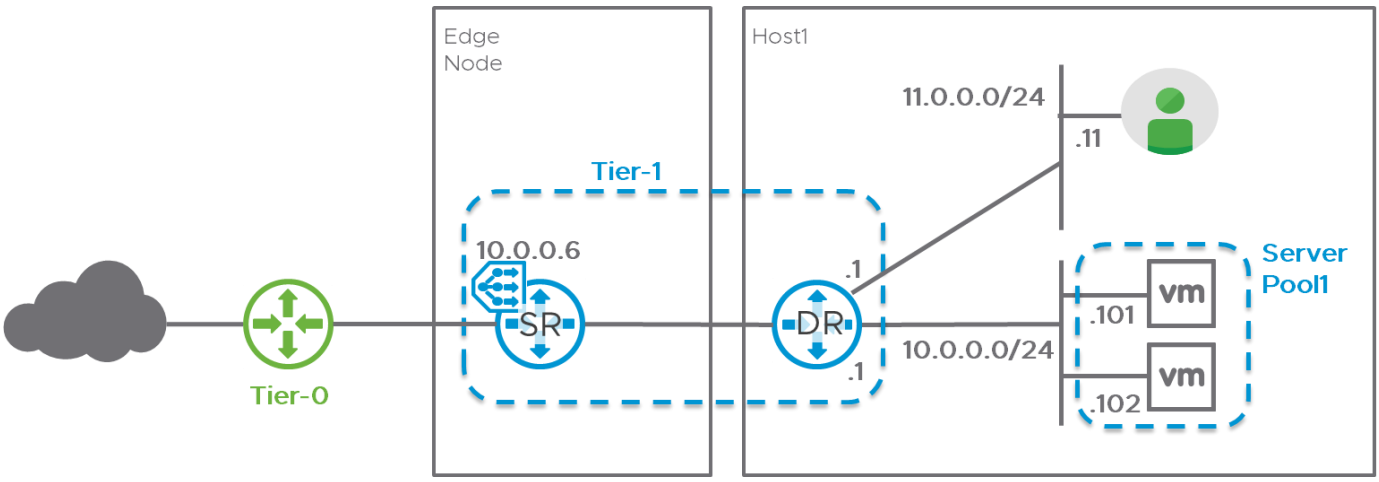


Figure 6-19: Load Balancing VIP IP@ in Tier-1 Downlink Subnet – Tier-1 Expanded View

Traffic from server to client would be switched directly by the Tier-1 DR without going through the load-balancer on the SR if source LB-SNAT was not configured. This design is not in fact a true in-line deployment of the load-balancer and does require LB-SNAT.

6.5.5.2 One-arm model

From a logical standpoint, the VIP of a virtual server belongs to the subnet of the downlink of the Tier-1 gateway associated to the load-balancer. The following diagram represents a load-balancer on a Tier-1 gateway with a downlink to subnet 10.0.0/24. The Tier-1 gateway interface has the IP address 10.0.0.1, and a virtual server with VIP 10.0.0.6 has been configured on the load balancer.

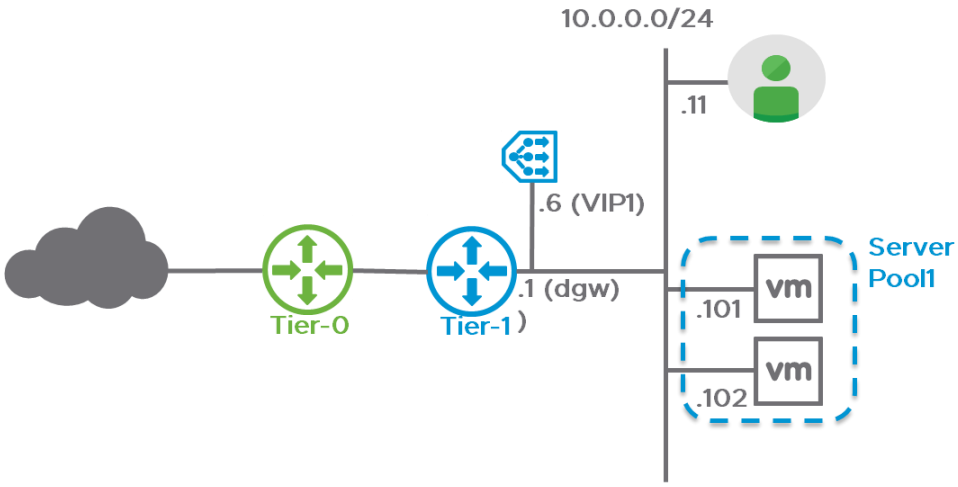


Figure 6-20: Load Balancing VIP IP@ in Tier-1 Downlink Subnet – Logical View

The diagram below offers a possible physical representation of the same network, where the Tier-1 gateway is broken down between an SR on an Edge Node, and a

DR on the host where both client and servers are instantiated (note that, in order to simplify the representation, the DR on the Edge was omitted.)

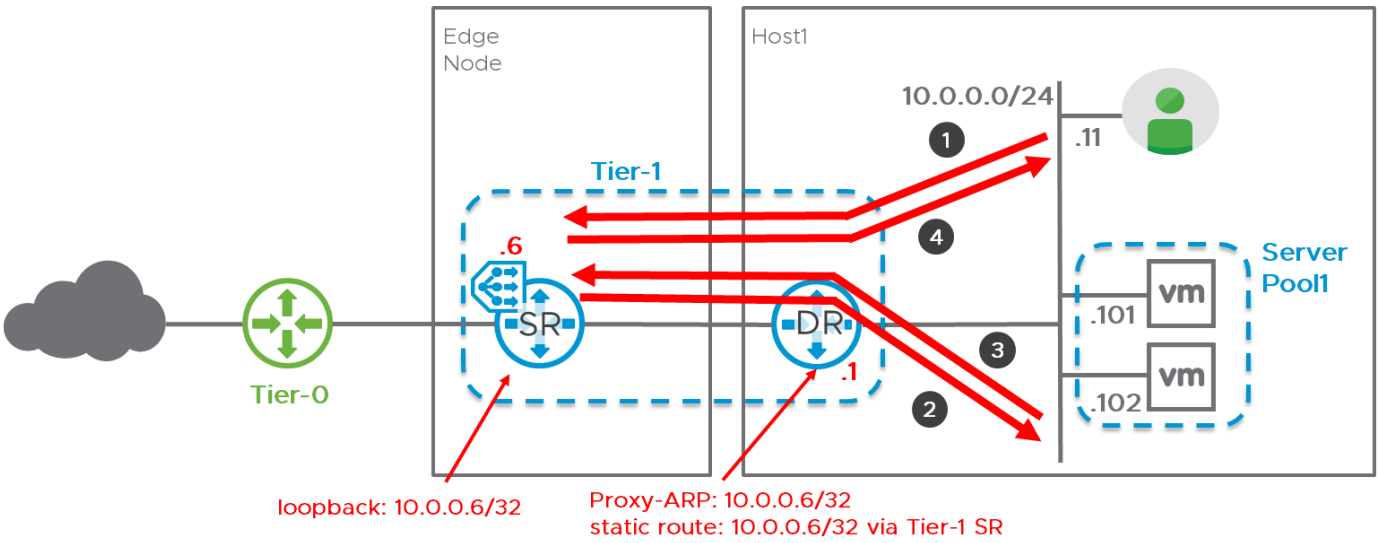


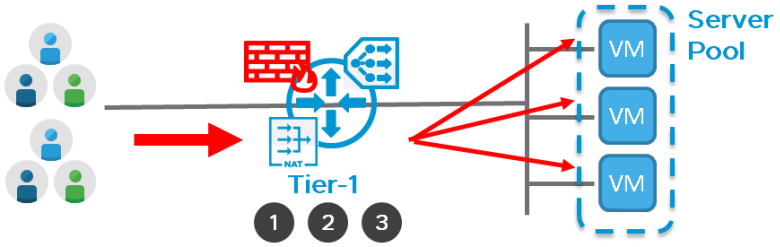
Figure 6-21: Load Balancing VIP IP@ in Tier-1 Downlink Subnet – Tier-1 Expanded View

This representation makes it clear that because the VIP is not physically instantiated on the DR, even if it belongs to the subnet of the downlink of the Tier-1 gateway, some additional "plumbing" is needed in order to make sure that traffic destined to the load-balancer reach its destination. Thus, NSX configures proxy-ARP on the DR to answer local request for the VIP and adds a static route for the VIP pointing to the SR (represented in red in the diagram.)

6.5.6 Load-balancing combined with SR services (NAT and Firewall)

Since NSX 2.4, the load-balancing service can be inserted in a service chain along with NAT and centralized firewall.

In case of service chaining, the order is NAT, then central firewall, at last load balancing.



1: DNAT: Client-IP@ => L3-DNAT-IP@ (translated to L3-Internal-VIP-IP@)

2: FW: Client-IP@ => L3-Internal-VIP-IP@ (allowed)

3: LB: Client-IP@ => Pool-Member-IP@

Service Chaining NAT / L3-FW / LB

Figure 6-22: LB + NAT + FW Services Chaining

7 NSX Design Considerations

This section examines the technical details of a typical NSX-based enterprise data center design. It looks at the physical infrastructure and requirements and discusses the design considerations for specific components of NSX. Central concepts include:

- NSX Deployment models
- Physical network fabric considerations
- Connectivity of management and control plane components (NSX Manager.)
- Design for connecting the compute hosts with both ESXi and KVM hypervisors.
- Design for the NSX Edge and Edge clusters.
- Organization of compute domains and NSX resources.
- Review of sample deployment scenarios.

7.1 NSX Deployment Models

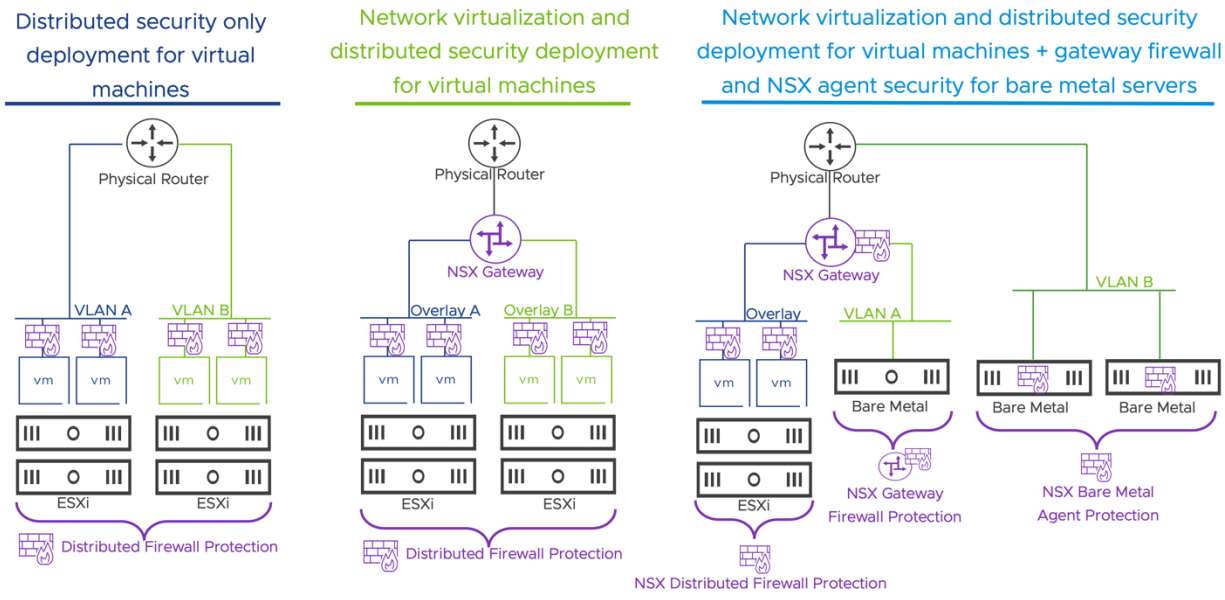


Figure 7-1: NSX Deployment Models

The first step in planning an NSX installation is understanding the deployment model we will adopt. NSX can be adopted in three modes. They are not mutually exclusive, as they can coexist in the same NSX installation and even vSphere cluster in some case. The three deployment models map to different NSX use cases: virtual workload security, network virtualization, bare metal server security. NSX can help in a variety of more advanced use cases, such as multi location

connectivity, disaster recovery, and cloud native applications, but here we focus on the core use cases to create a design framework.

The three deployment models are:

1. Distributed security model for virtualized workloads
2. Network virtualization with distributed security for virtualized workloads
3. Bare metal workloads security via gateway firewall (centralized security) or NSX bare metal agent

In the **distributed security model**, NSX provides distributed security services such as distributed firewall and distributed IDS/IPS. Still, the physical network retains a significant part of the switching and routing responsibilities. The only NSX component participating in the networking functionalities is the VMware VDS, responsible for switching the traffic between the VMs running in the hypervisor and the physical uplinks. Once the packets are delivered to the upstream physical switch, it is the responsibility of the physical fabric to deliver it to the appropriate destination. This networking model is the most common in vSphere deployments that are not running NSX, making the adoption of NSX distributed security entirely transparent for the physical network in brownfield environments.

The **networking and security model** provides the full benefit of NSX network and security virtualization. Distributed security services operate as in the security-only model, being orthogonal to the adoption of the NSX network virtualization services. NSX network virtualization decouples the physical network from the topology requirements of the applications and services that run on top of it. The physical fabric can be designed and operated based on a simpler and more stable model that does not require following the application's lifecycle. The role of adapting to the ever-changing application requirements is offloaded to the NSX network virtualization layer. At the same time, the physical fabric design can be centered around maximizing throughput and high availability, minimizing latency, and optimizing operational efficiency.

Overlay Networking Benefits		
Area	VLAN Only Networking	NSX Overlay Networking
VM Mobility	VM mobility is limited to the span of a VLAN (usually one rack)	VM mobility across layer 3 boundaries – any topology
Rack Availability	Achieving rack availability via vSphere HA is not possible without extending VLANs between racks	Easy to achieve rack availability via vSphere HA for compute workloads
Resource Pooling	Resource pooling is limited by the VLAN span	Heterogenous islands of compute are easily integrated

Infrastructure Agility	Slow application deployment dependent on physical network reconfigurations	The physical network is abstracted. No changes are required to support the applications lifecycle
Change Management	Network changes require reconfiguration of multiple devices	Single point of management for the entire logical network. The network can embrace an Infrastructure as Code (IoC) approach, where network constructs follow the application lifecycle
Disaster Recovery	Disaster recovery requires re-IP	Easy disaster recovery without re-IP
L2 Scale	Fundamentally tied to MAC learning and L2 discovery over different types of underlay technology (e.g., traditional L2, EVPN)	Programmed from vSphere, and thus no MAC learning and abstracted from physical network, reducing the leaf switch scale requirements
L2 Security	Physical fabric is directly exposed to L2 frames generated by the VMs (e.g., STP BPDU).	NSX Overlay isolates the physical fabric from the layer 2 frames generated by the endpoints
Support of Advanced Networking – NAT, Multicast, IPV6, etc.	Tied to physical ASIC and limited by switch vendor	Software defined allows scale the networking and security stack
Elastic BW	BW is tied to VLAN/Subnet to a rack switch uplink Tie to uplink oversubscription per rack Tied to Spine scale	BW is scalable via the rapid deployment of the software gateway 160 Gateways per NSX Domains

Figure 7-2: Network Virtualization Benefits

The following sections list the physical network requirements for both deployment models and provide considerations around specific network fabric designs.

The **centralized security** (or gateway firewall) model is a third less standard but emerging deployment model. In this option, NSX Edges act as the perimeter firewall for the data center or a branch location and protect not virtualized workloads operating as a next-generation firewall appliance. This model has no

requirements on the physical network. Physical server security via the NSX agent is covered in this DOCUMENT.

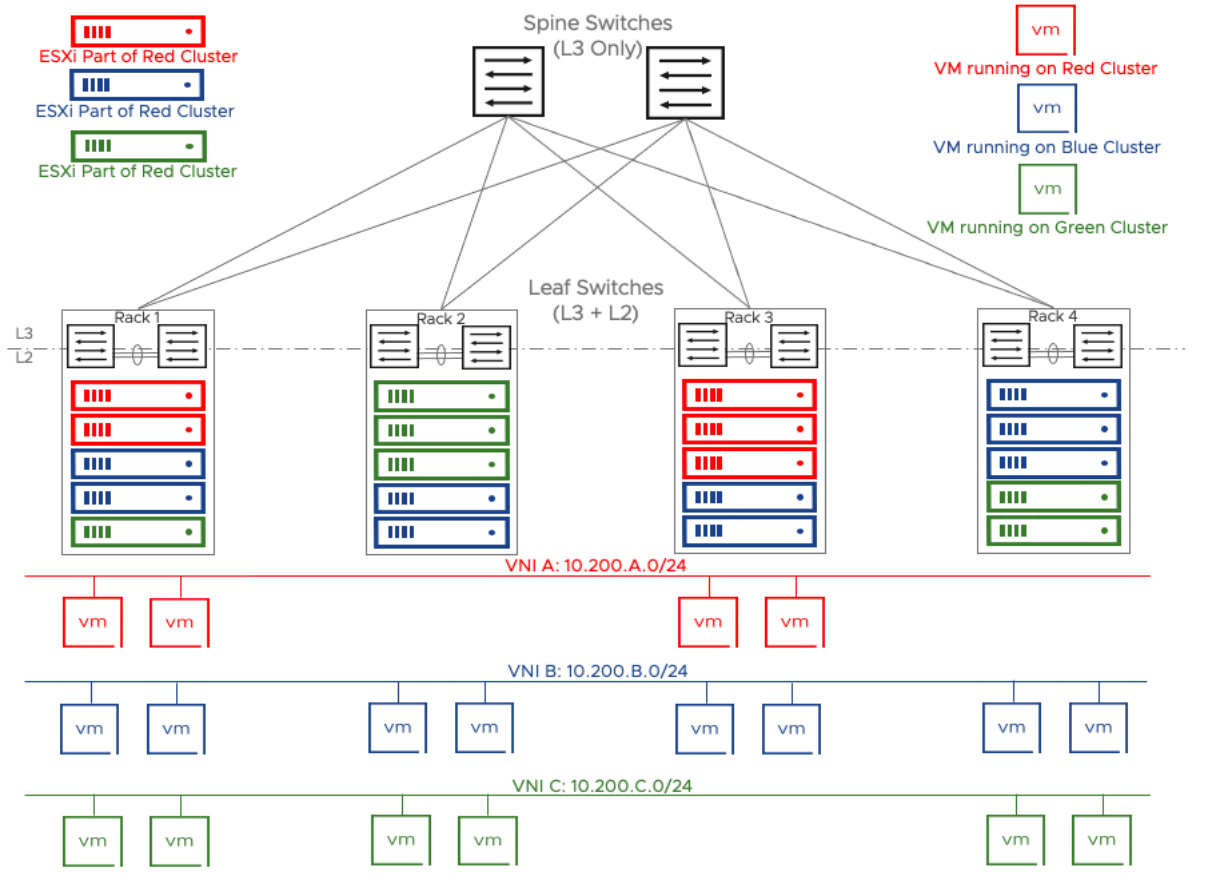
7.1.1 Role of overlay networking in the modern datacenter

Overlay networking allows the decoupling of network topology from the physical infrastructure. The physical network only requires a basic set of VLANs and subnets common to all the vSphere deployments: management, storage, and vMotion, plus an additional VLAN/subnet for the NSX overlay traffic. FIGURE 7-3 below shows an example where four racks, representing four independent pods of computing resources, are separated by layer-3 boundaries. Each rack has an independent set of those subnets: management, vMotion, storage, and overlay (Geneve). VLAN IDs are kept consistent across the racks for a more standardized configuration, but it is not required. FIGURE 7-3 also shows an addressing schema, where the third octet of the IP ranges represents the rack or compute pod. Again, this is an approach we recommend to encourage standardization, but technically not required.

Once this basic network connectivity is in place, we can place VMs on overlay networks (VNIs) spanning all the racks. No VLAN in the physical network is required to span between racks. vSphere clusters can span multiple racks without impacting vSphere HA, vMotion, and DRS.

Modern data centers do not grow organically. Physical network and compute resources have different lifecycles and growth patterns. Physical network atomic units of scale are usually racks and the top of the rack switches in them. When the switches' ports or the rack's space are depleted, a new rack with new top-of-rack switches is deployed. In a vSphere deployment, the atomic units of scale are the hosts pooled in clusters often dedicated for different purposes. In an ideal world, the growth patterns for computing and physical networks coincide. In reality, it rarely happens.

Looking at FIGURE 7-3, we can see the erratic growth pattern for the different vSphere clusters that required the three clusters to be striped non-uniformly across all four racks. If we performed capacity planning correctly and the budget required to purchase those compute resources was available on day 1, the distribution could have been more orderly. Each rack could have been dedicated to a cluster, or maybe two for rack redundancy, to limit the span of VM mobility. In the real world, a different line of business can fund their initiatives in different phases, and IT operators must expand their clusters on demand leveraging the available network resources.



VLANs & IP Subnet Defined at the Leaf SW POD A		
SVI	VLAN ID	IP Subnet
Management	100	10.100.A.x/24
vMotion	101	10.101.A.x/24
Storage	102	10.102.A.x/24
Geneve	**103**	**10.103.A.x/24**

VLANs & IP Subnet Defined at the Leaf SW POD B		
SVI	VLAN ID	IP Subnet
Management	100	10.100.B.x/24
vMotion	101	10.101.B.x/24
Storage	102	10.102.B.x/24
Geneve	**103**	**10.103.B.x/24**

VLANs & IP Subnet Defined at the Leaf SW POD C		
SVI	VLAN ID	IP Subnet
Management	100	10.100.C.x/24
vMotion	101	10.101.C.x/24
Storage	102	10.102.C.x/24
Geneve	**103**	**10.103.C.x/24**

VLANs & IP Subnet Defined at the Leaf SW POD D		
SVI	VLAN ID	IP Subnet
Management	100	10.100.D.x/24
vMotion	101	10.101.D.x/24
Storage	102	10.102.D.x/24
Geneve	**103**	**10.103.D.x/24**

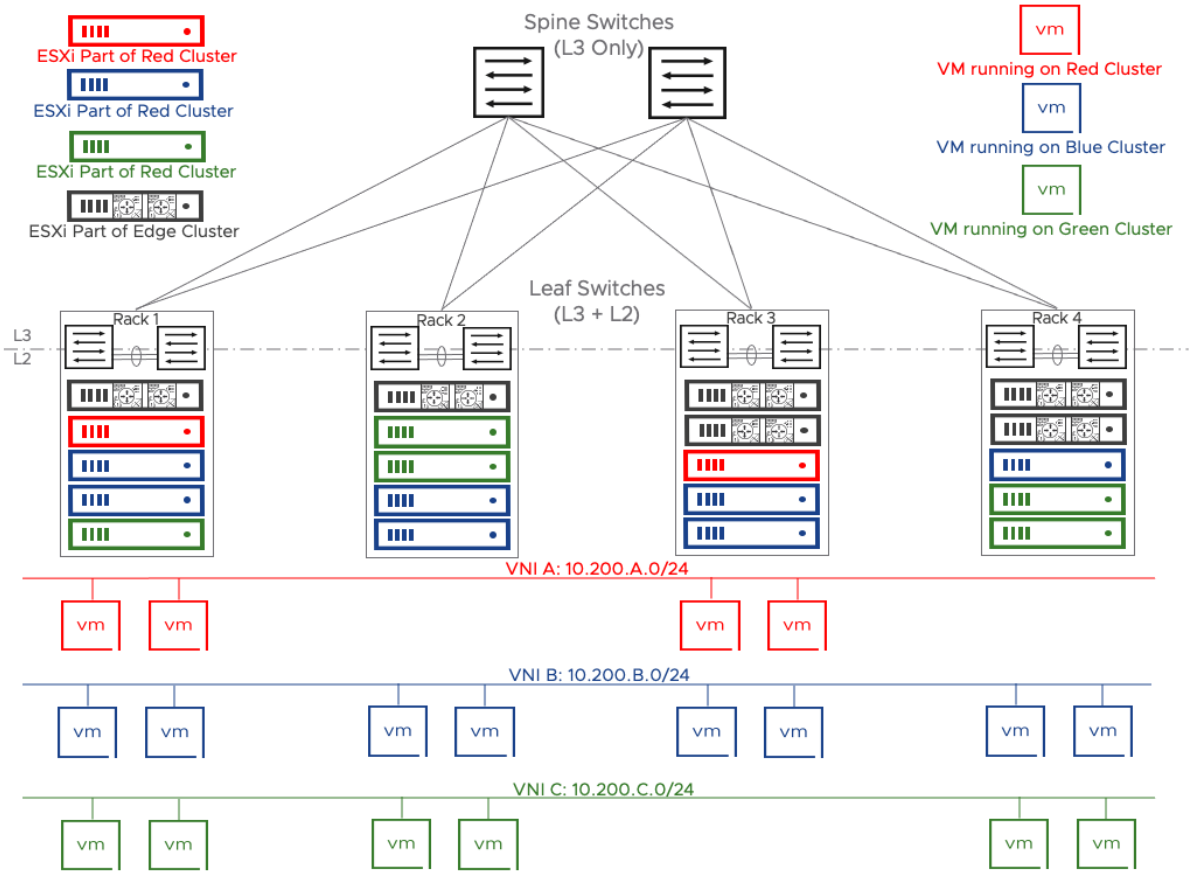
Figure 7-3: Network Overlays for elastic and non-uniform compute requirements

The role of the physical network is to provide reliable high throughput connectivity between the hosts regardless of their nature. Dedicating a set of physical switches or racks to specific workloads (vSphere clusters) is not a valid approach in data centers required to reach cloud scale. It represents a waste of resources when the rack and switches are underutilized, and they are a scale upper boundary for the workloads. What happens when you need to add a host to a cluster with a dedicated rack and the ports, or the rack space is unavailable? NSX overlays allows for the different vSphere cluster to grown inorganically while preserving connectivity and VM mobility.

So far, we have explored the benefits and flexibility that overlays bring to private clouds requiring elastic compute and network connectivity properties. We can extend the model to north-south bandwidth requirements and network services. NSX edge nodes are responsible for scaling the software-defined network solution over those dimensions. OF LIFE cycle that comes with vSphere Figure 7-4 shows how edge hosts (ESXi hosts running NSX edge VMs) can be added to the design over time based on changing requirements. The only additional

requirement on the physical network is two additional subnets/VLANs to establish BGP peering in each rack. As for the other infrastructure VLANs, those are local and do not stretch across racks.

The NSX Edge nodes can be thought has the “spine” layer of the NSX SDN solution. A single NSX deployment can horizontally scale up to 160 edge nodes. T hey can be distributed over multiple racks to reduce the physical fabric oversubscription ratio between the leaf and spine layers.



VLANs & IP Subnet Defined at the Leaf SW POD A		
SVI	VLAN ID	IP Subnet
Management	100	10.100.A.x/24
vMotion	101	10.101.A.x/24
Storage	102	10.102.A.x/24
Geneve	103	10.103.A.x/24
BGP p2p-1	104	10.104.A.x/24
BGP p2p-2	105	10.105.A.x/24

VLANs & IP Subnet Defined at the Leaf SW POD B		
SVI	VLAN ID	IP Subnet
Management	100	10.100.B.x/24
vMotion	101	10.101.B.x/24
Storage	102	10.102.B.x/24
Geneve	103	10.103.B.x/24
BGP p2p-1	104	10.104.B.x/24
BGP p2p-2	105	10.105.B.x/24

VLANs & IP Subnet Defined at the Leaf SW POD C		
SVI	VLAN ID	IP Subnet
Management	100	10.100.C.x/24
vMotion	101	10.101.C.x/24
Storage	102	10.102.C.x/24
Geneve	103	10.103.C.x/24
BGP p2p-1	104	10.104.C.x/24
BGP p2p-2	105	10.105.C.x/24

VLANs & IP Subnet Defined at the Leaf SW POD D		
SVI	VLAN ID	IP Subnet
Management	100	10.100.D.x/24
vMotion	101	10.101.D.x/24
Storage	102	10.102.D.x/24
Geneve	103	10.103.D.x/24
BGP p2p-1	104	10.104.D.x/24
BGP p2p-2	105	10.105.D.x/24

of life cycle that comes with vSphere Figure 7-4: Network Overlays for elastic North-South Bandwidth and network services

7.1.2 Distributed security model, dvpgs vs NSX VLAN segments

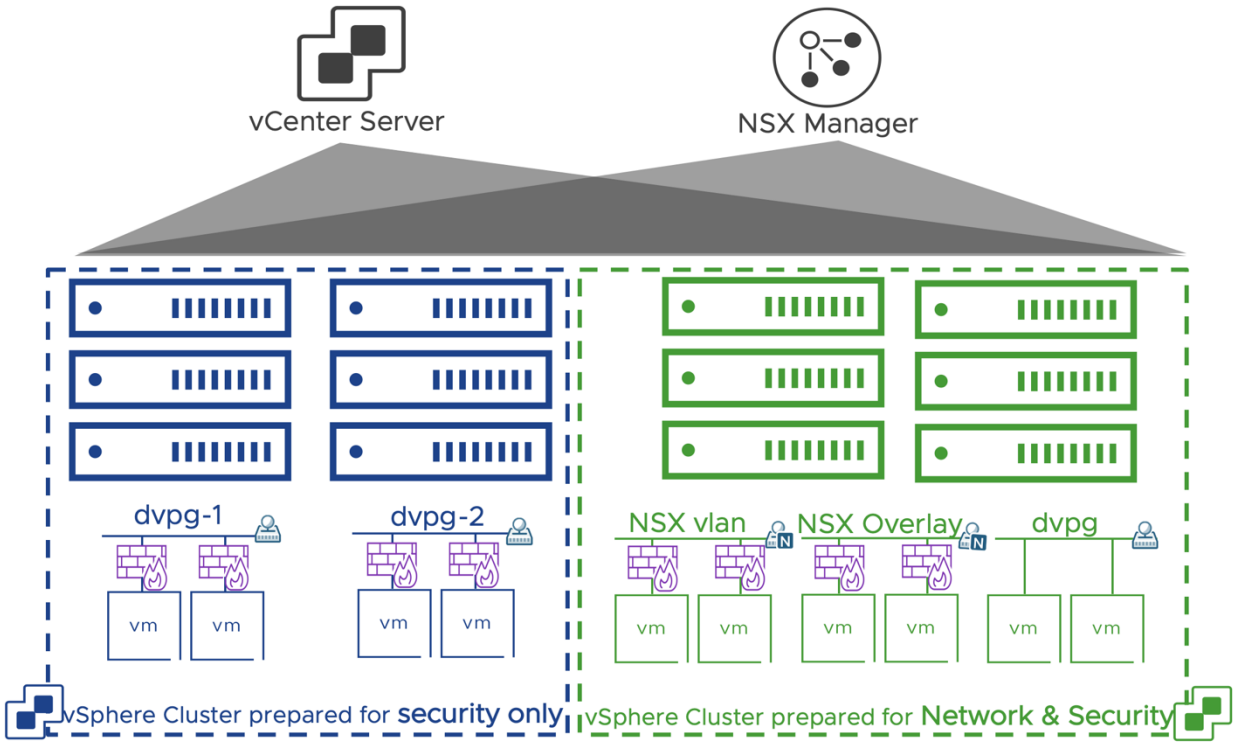


Figure 7-5: Security only vs Network & Security cluster preparation

When planning a distributed security deployment without overlay, two options exist:

1. Prepare the vSphere hosts for NSX Security only, and leave the virtual machines connected to the virtual distributed port-groups managed by vCenter
2. Prepare the vSphere hosts for NSX Networking & Security, and migrate the virtual machines to NSX VLAN port-groups

Option 1 does not require migrating the VM, and the entire networking configuration is retained by vCenter. All virtual machines connected to dvpgs are automatically protected by the NSX DFW. This is the most straightforward and recommended approach, however as of now (NSX version 3.2) it does not allow the implementation of NSX overlays. If it is expected to implement NSX Overlay soon, option 2 should be considered. Migrating from a security-only deployment to a network & security deployment requires uninstalling NSX from the vSphere cluster. Option 1 is available with NSX 3.2, vSphere 6.5 or later, and VDS 6.6 or later.

Option 2 requires placing the workloads on NSX-managed distributed virtual port-groups. In a brownfield, we can create NSX segments with the same VLAN as the dvpgs where the VMs reside and then migrate them, reconfiguring the vNic with no disruption. VMs on dvpg are excluded by the NSX distributed firewall. With this option, we can adopt overlays at a later stage. Option 2 requires NSX 3.0 or later, vSphere 7, and VDS 7.

The two deployment models can coexist within the same installation on a per cluster basis. A vSphere cluster can be prepared for security only, another for network & security. This is valuable for existing deployments where one can deploy NSX for security only while in parallel overlay based clusters providing a path for a future migration.

Type of connectivity	Transport Zone	DFW	Service Interface
DVPG on VDS (connects Guest VMs to VLANs)	N/A	Yes, cannot have NSX overlay in the same cluster	VMs connected to a dvpg can have a SI as the default gateway. SI is connected to an Edge VLAN Segment.
ESXi NSX VLAN Segment (connects Guest VMs to VLANs)	ESXi VLAN Transport Zone (Different than the one for NSX Edge TNs)	Yes, it can have NSX overlay segments in the same cluster	VMs connected to an NSX VLAN Segment can have a SI as the default gateway
NSX Edge VM VLAN Segment (routing peering or service interfaces)	Edge VLAN Transport Zone (Different than the one for ESXi TNs)	N/A. NSX Edge VMs are automatically excluded from DFW	SI is connected to NSX Edge VLAN when it serves as the default gateway for VMs on dvpgs or VLAN segments. VLAN ID must match. Dynamic routing over SI is only supported for EVPN route servers mode
NSX Overlay Segment (connects Guest VMs to NSX Overlays)	Single NSX Overlay Transport Zone common with NSX edge nodes	Yes	VMs connected to an NSX overlay can have an SI as the default gateway. SI is connected to the same overlay (Not recommended, use a regular downlink interface)

7.2 Physical Infrastructure of the Data Center

An essential characteristic of NSX is that it is agnostic to the physical network configuration, allowing for great flexibility in adapting to a variety of underlay fabrics and physical network topologies.

NSX deployments can be generally categorized into two models: distributed security and network and security.

7.2.1 Physical Network Requirements

The security-only and the network and security deployment models share the following basic requirements for the physical network fabric.

Distributed Security and Network and Security deployment models:

- IP Connectivity – IP connectivity between all components of NSX and the compute hosts. This includes ESXi hosts management interfaces and NSX managers only for the security-only deployment model as NSX edge nodes are not part of it. IP connectivity should extend to the edge nodes VMs or bare metal servers for a network and security deployment model.
- IP and MAC mobility across switches in the same rack – This is not NSX specific. It is based on vSphere networking. When ESXi are connected to different switches, vSphere networking assumes that moving the IP and MAC of VMs and VMK interface from one uplink to another is transparent to the physical network. This is how vSphere teaming policies protect against the failure of a pNIC or an upstream switch. If the two switches where the ESXi is connected are not layer two adjacent, this mechanism cannot work. Most network fabrics provide this capability; the exceptions consist in pure layer three spine and leaf architectures. In such a scenario, the only option (not recommended) is to single connect the servers and provide high availability at a different level.

Once the above requirements are met, an NSX security-only deployment is agnostic to any physical fabric topology and configuration. NSX network virtualization has an additional MTU requirement.

Network and Security deployment model:

- Jumbo Frame Support – A minimum required MTU is 1600. However, MTU of 1700 bytes is recommended to address the whole possibility of a variety of functions and future proof the environment for an expanding Geneve header. As the recommended MTU for the NSX is 9000, the underlay network should support this value.
- The VM MTU – VM MTU should be set 100 bytes or more (200 preferred) lower than the MTU of the physical fabric. Typical deployment carries 1500 byte MTU for the guest VMs so no change is required for the VM MTU if the physical fabric has an MTU of 1700 bytes or higher. To improve the throughput, one can increase the MTU up to 8800 (a ballpark number to accommodate bridging and future header expansion). Modern Operating Systems support a TCP/IP stack with a feature called Path MTU (PMTU) discovery. PMTU allows the discovery of underlying networks' capability to carry message size measured by MSS (maximum segment size) for a given

TCP session. Thus, every TCP session will continue to make this discovery periodically and thus allows maximum efficiency payload for a given session.

Once the above requirements are met, an NSX network virtualization deployment is agnostic to any underlay topology and configurations, specifically:

- Any physical topology – core/aggregation/access, leaf-spine, etc.
- On any switch from any physical switch vendor, including legacy switches.
- With any underlying technology. IP connectivity can be achieved over an end-to-end layer 2 network as well as across a fully routed environment.

For an optimal design and operation of NSX, well-known baseline standards are applicable. These standards include:

- Device availability (e.g., host, TOR, rack-level)
- TOR bandwidth - both host-to-TOR and TOR uplinks
- Fault and operational domain consistency (e.g., localized peering of Edge node to the northbound network, separation of host compute domains, etc.)

7.2.2 Physical fabric considerations

This design guide uses a routed leaf-spine architecture for most examples. This model is a superset of other network topologies and fabric configurations, so its concepts also apply to layer two and non-leaf-spine topologies. This section will provide some high-level examples of deployments and considerations for different types of network fabrics to clarify the nuances of deploying NSX over different fabric topologies.

7.2.2.1 Layer 3 spine and leaf with layer2 adjacent ToRs.

In this topology, the spine switches are pure layer three devices and connect to the leaf switches via routed point-to-point links. The leaf switches are deployed in pairs in each rack. They share a common layer two domain over a direct link (most likely a redundant port-channel) to satisfy the vSphere networking requirements for MAC and IP mobility within a single rack. Leaf switches may or may not implement an MLAG proprietary solution. The L2/L3 demarcation point is at the leaf level.

The following considerations apply to this topology in an **NSX security only deployment** (FIGURE 7-6):

- Different racks require different VLANs and network configurations. Rack 1 and 2 (yellow) host the compute clusters, rack 3 (green) hosts management workloads, rack 4 (blue) is dedicated to WAN connectivity, all with different VLAN requirements.
- While WAN and Management blocks are usually static, the compute rack switches may require frequent configuration updates based on the applications lifecycle.

- Virtual machine mobility is not available across racks. This may limit the agility and elasticity properties of the design and reduce the consolidation ratio in the data center.
- vSphere clusters cannot be striped across racks, which may limit high availability as the infrastructure cannot protect against a rack failure.

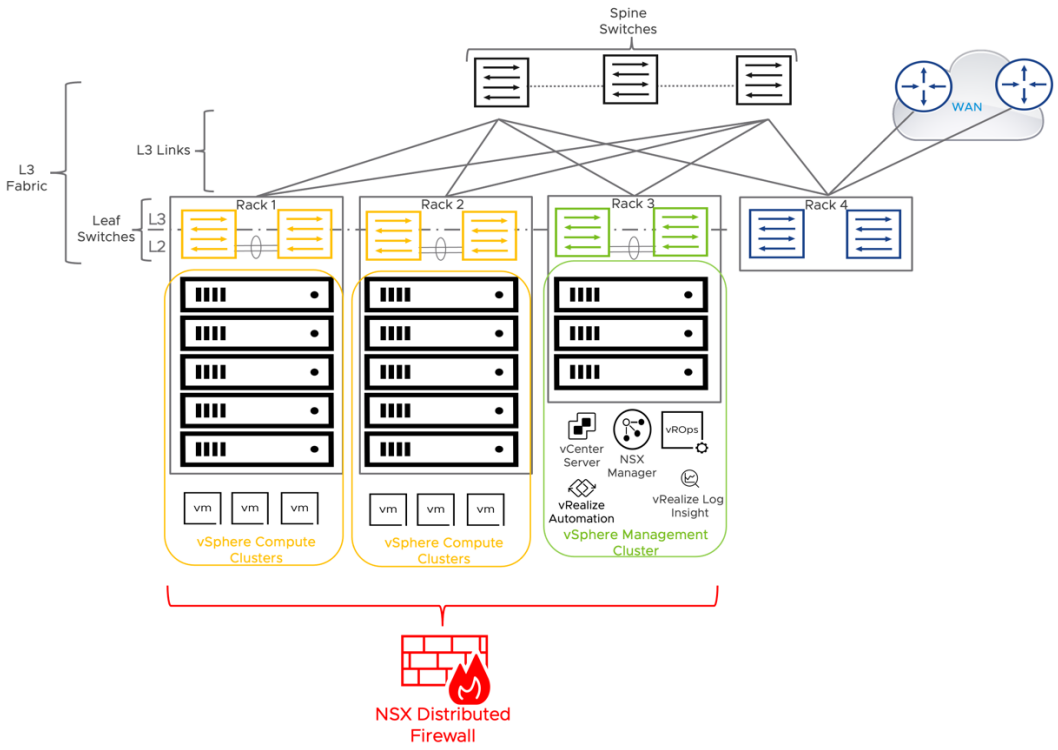


Figure 7-6: NSX Security Only Deployment – L3 fabric

The following considerations apply to this topology in an **NSX network and security deployment** (FIGURE 7-7):

- The ToR VLAN configuration is more consistent than in a security-only deployment, and it is agnostic to the application lifecycle. Generally, three configuration templates must be provided for the ToR switches, one for the compute block (yellow), one for the management block (green), and one for the edge block (blue). Those configurations templates tend to be static as they do not require modification when new NSX virtual networks are deployed.
- Virtual machine mobility is available across racks. NSX overlay segments extend layer two networks across the layer three boundaries of each rack.
- Compute clusters can be stretched across racks for rack level high availability.

- Resource pooling is streamlined as hosts in different racks (or datacenter rooms) can be added to the same vSphere cluster.
- While some infrastructure management VMs such as the vRealize suite components can be deployed on NSX overlay segments, NSX managers and vCenter should be deployed on a VLAN network. This limits the span of the management vSphere cluster to a single rack. The management VLAN must be extended across racks, to provide rack high availability. The NSX manager cluster appliances can be deployed in different IP subnets to avoid extending layer two between racks, but no solution is currently available for the vCenter server.

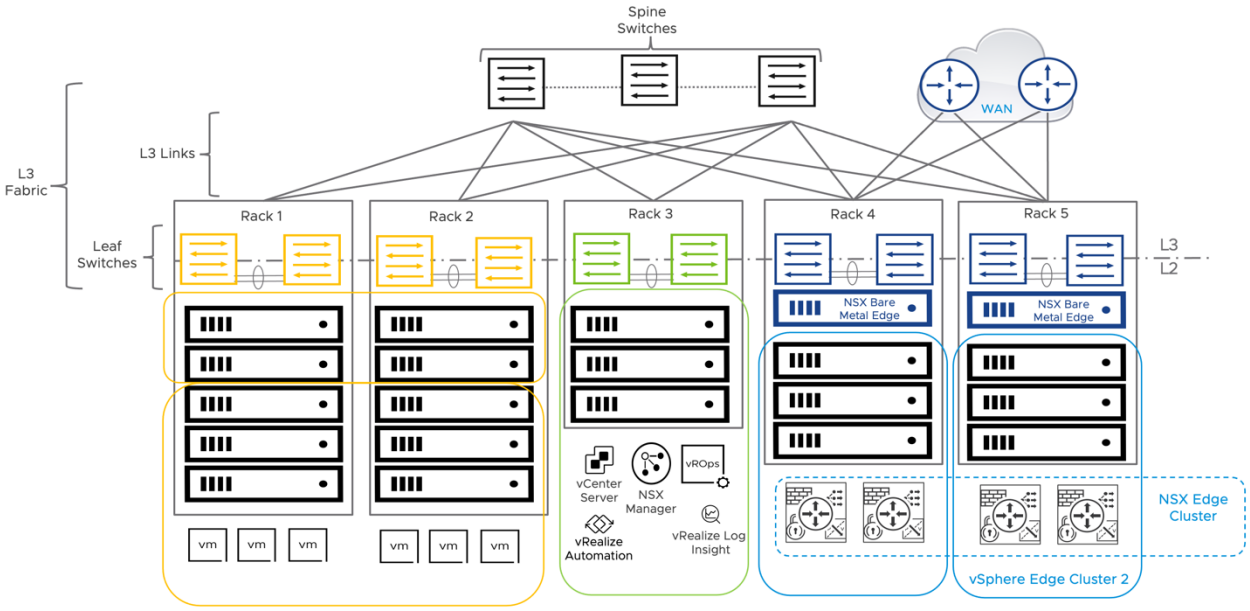


Figure 7-7: NSX Network and Security Deployment – L3 fabric

FIGURE 7-8 below outlines a sample VLAN and IP design for an NSX networking and security deployment over a layer three fabric. Because each rack represents a unique layer two domain, we can use the same VLAN IDs across racks to improve consistency.

The compute racks generally require only four VLANs (ESXi Management, vMotion, Storage, ad NSX Overlay), and they can share the same VDS. Management racks may or may not be prepared for NSX. When they are not, they do not require the NSX Overlay VLAN.

The management racks require a VM Management VLAN to place the management infrastructure components such as the vCenter Server and NSX Manager. This VLAN is generally separate from the ESXi management network as it may be stretched between racks to provide rack high availability for the management VMs. The stretching of the VM management VLAN requires layer two links or physical fabric overlays. The ESXi management VLAN never has this requirement.

The hosts in the edge racks are never prepared for NSX (unless edge and compute blocks are merged) and require two VLANs dedicated to layer three

peering between the physical fabric and the NSX edges (VMs or Bare Metal). Edge vSphere clusters rarely need to be stretched across racks. The edge node VMs themselves can participate in the same NSX edge cluster (and, for example, support the same T0 Gateway) even if they are deployed on different vSphere clusters in different racks. Edges in different racks can use layer 3 peering VLANs local to each rack, so those VLANs do not need to be stretched. The layer three peering subnets should support ECMP to 8 edge nodes, so /28 subnets or larger are recommended.

VLANs & IP Subnets For Compute Racks

SVI Interface	VLAN ID	IP Subnet
ESXi Management	101	10.101.Rack_ID.0/24
vMotion	102	10.102.Rack_ID.0/24
Storage	103	10.103.Rack_ID.0/24
Overlay	104	10.104.Rack_ID.0/24

VLANs & IP Subnets For Management Racks

SVI Interface	VLAN ID	IP Subnet
ESXi Management	101	10.101.Rack_ID.0/24
vMotion	102	10.102.Rack_ID.0/24
Storage	103	10.103.Rack_ID.0/24
VM Management (May be stretched between racks)	105	10.105.Rack_ID.0/24

VLANs & IP Subnets For Edge Racks

SVI Interface	VLAN ID	IP Subnet
ESXi Management	101	10.101.Rack_ID.0/24
vMotion	102	10.102.Rack_ID.0/24
Storage	103	10.103.Rack_ID.0/24
VM Management	105	10.105.Rack_ID.0/24
Routed P2V-1	106	10.254.Rack_ID.1/28
Routed P2V-1	107	10.254. Rack_ID.16/28

Figure 7-8: Sample VLAN and IP Subnets schema for NSX Network Virtualization deployment on a Layer 3 fabric

7.2.2.2 Layer 2 fabrics

In a pure layer two fabric, all the connections between the spine and the leaf switches are layer two links and carry all or a subset of the VLANs in the fabric. Redundant links between the spine and leaf layers are handled by spanning tree, a proprietary implementation of multi-chassis link aggregation protocol (MLAG), or less frequently a routed layer two protocol (e.g., Trill or Cisco FabricPath). The type of layer two fabric does not impact the NSX design. The L2/L3 demarcation point is generally at the spine level, where the WAN or core network blocks are usually connected.

The following considerations apply to this topology in an **NSX security only deployment** (FIGURE 7-9):

- At minimum, VLANs are extended between the switches in the same rack, meeting the IP and MAC mobility requirement of vSphere networking. VLANs can be extended across racks, increasing VM mobility and allowing the possibility to stripe vSphere clusters across racks for increased high availability.
- While possible, extending VLANs across multiple racks increases the size of a layer two fault domain.
- In a dynamic environment requiring frequent provisioning of new networks to support the applications lifecycle, the network administrator may need to perform frequent and widespread changes to the physical network configuration.

- The network architect will be required to balance the above considerations. Maximum flexibility in VM placement via a larger VLAN span will negatively impact the size of failure domains and the overall manageability. A more segmented approach will instead limit VM mobility and the overall elasticity of the design.

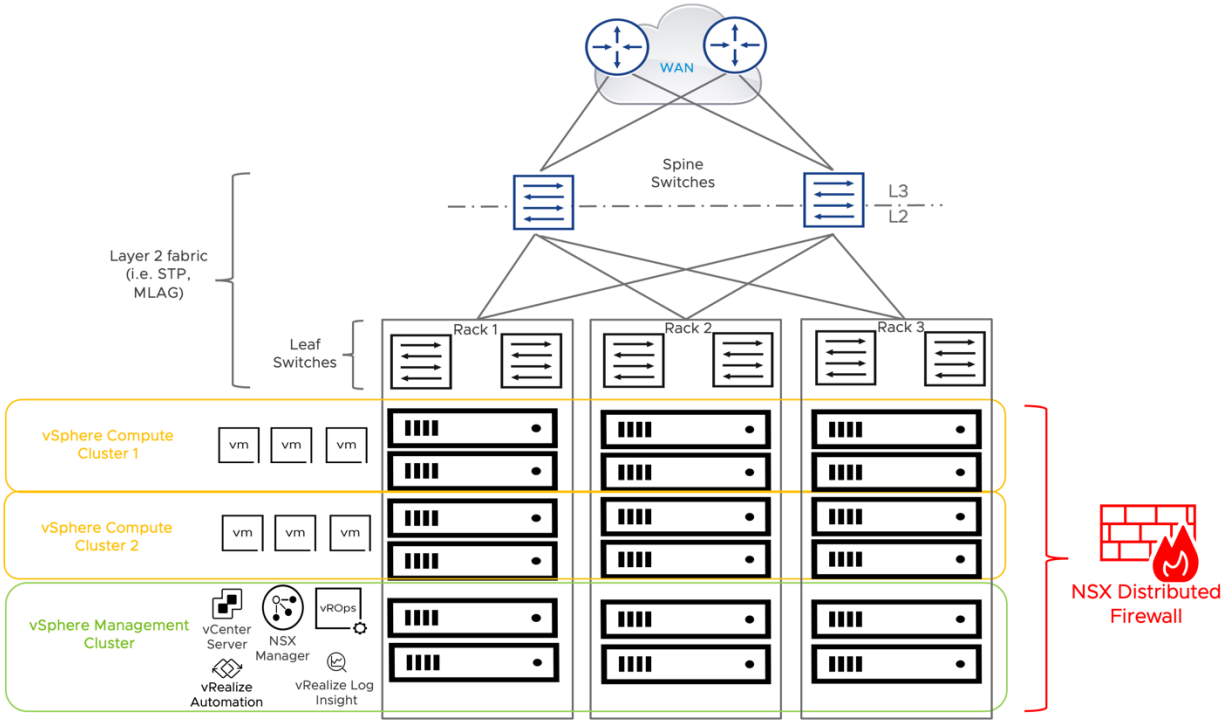


Figure 7-9: Security Only Deployment – L2 Fabric

The following considerations apply to this topology in an **NSX network and security deployment** (FIGURE 7-10 AND FIGURE 7-11):

- The network architect can limit the size of the layer two domains to a single rack by pruning the VLANs appropriately. VM mobility is not impacted as NSX overlays extend VM networks across layer three boundaries. This configuration makes the layer two physical fabric properties very similar to those of a layer three fabric concerning the NSX design.
- Management VMs VLAN can easily be extended between racks enabling the striping of the management cluster.
- Small environments may benefit from a simpler VLAN/IP schema where the infrastructure VLANs (management, vMotion, storage, overlay) are unique across the fabric.
- The configuration of the physical devices becomes standardized and static. New networks and topologies to support the applications lifecycle are provisioned in NSX transparently to the physical fabric.

- Bare metal edge nodes, or the ESXi hosts where edge node VMs reside, should be connected to layer three capable devices. It usually means the spine or aggregation layer switches in a layer two fabric. When implementing such a design is not desirable, dedicated layer 3 ToR switches may be deployed to interconnect the edge nodes.

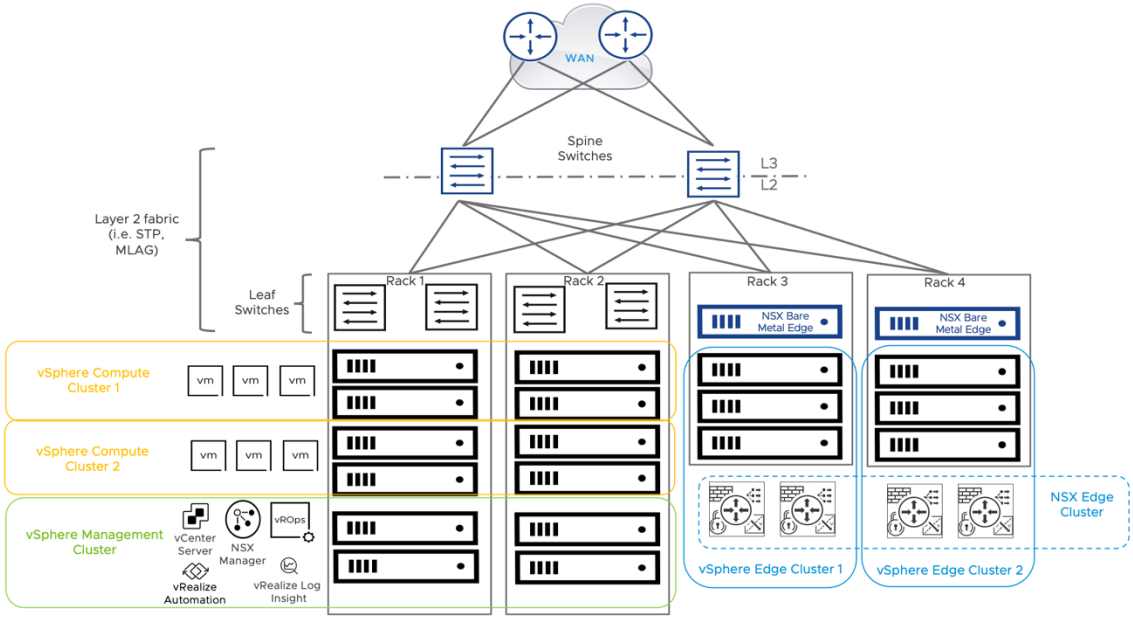


Figure 7-10: Network and Security Deployment – L2 fabric – Edge Nodes connected to spine switches

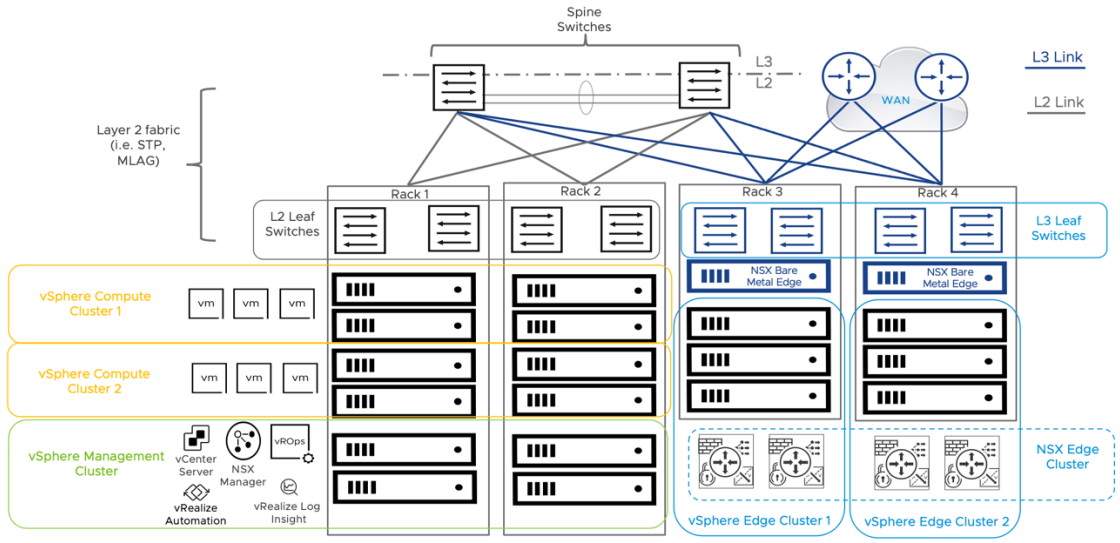


Figure 7-11: Network and Security Deployment – L2 fabric – Edge Nodes connected to dedicated L3 ToRs

FIGURE 7-12 below outlines a sample VLAN/IP schema for a layer two fabric where VLANs have been extended across the whole physical network. This design is appropriate for smaller deployments. A large implementation may benefit from the segmentation of layer two domains. In such cases, the VLAN/IP schema may approach the one outlined for the layer three fabric example.

VLANs & IP Subnets For the Spine/Aggregation Switches		
SVI Interface	VLAN ID	IP Subnet
ESXi Management	101	10.100.101.0/24
vMotion	102	10.100.102.0/24
Storage	103	10.100.103.0/24
Overlay	104	10.100.104.0/24
VM Management	105	10.100.105.0/24
Routed P2V-1	106	10.100.106.1/24
Routed P2V-1	107	10.100.107.1/24

VLANs For Compute Racks Access Switches (L2 Only)	
VLAN Name (no SVI, L2 Only)	VLAN ID
ESXi Management	101
vMotion	102
Storage	103
Overlay	104
VM Management	105

Figure 7-12: Sample VLAN and IP Subnets schema for NSX Network Virtualization deployment on a Layer 2 fabric

7.2.2.3 Layer 3 spine and leaf fabric with Overlay

A layer three spine and leaf fabric with overlay consists of a pure spine and leaf topology where leaf switches are only connected to the spine layer via routed point-to-point links. No connection exists between leaf switches in the same rack. On top of this routed topology, the physical fabric implements an overlay (independent and transparent to vSphere and NSX), making the same layer two segments available on any desired leaf switch. Examples of such fabrics are EVPN based fabrics and Cisco ACI.

From an NSX and vSphere perspective, a layer three fabric with overlay is equivalent to a layer two fabric. As such, similar considerations apply. For **NSX Security only deployments** (FIGURE 7-13), the points to emphasize are:

- VLANs can be extended across racks via the physical fabric overlays, increasing the span of VM mobility and providing the possibility to stripe vSphere clusters across racks for increased high availability.
- The span of the VLANs is less of a concern regarding fault domains size compared to Layer 2 fabrics because network overlays are in use.
- In a dynamic environment requiring frequent provisioning of new networks to support the applications lifecycle, the network administrator may need to perform frequent and widespread changes to the physical network configuration.

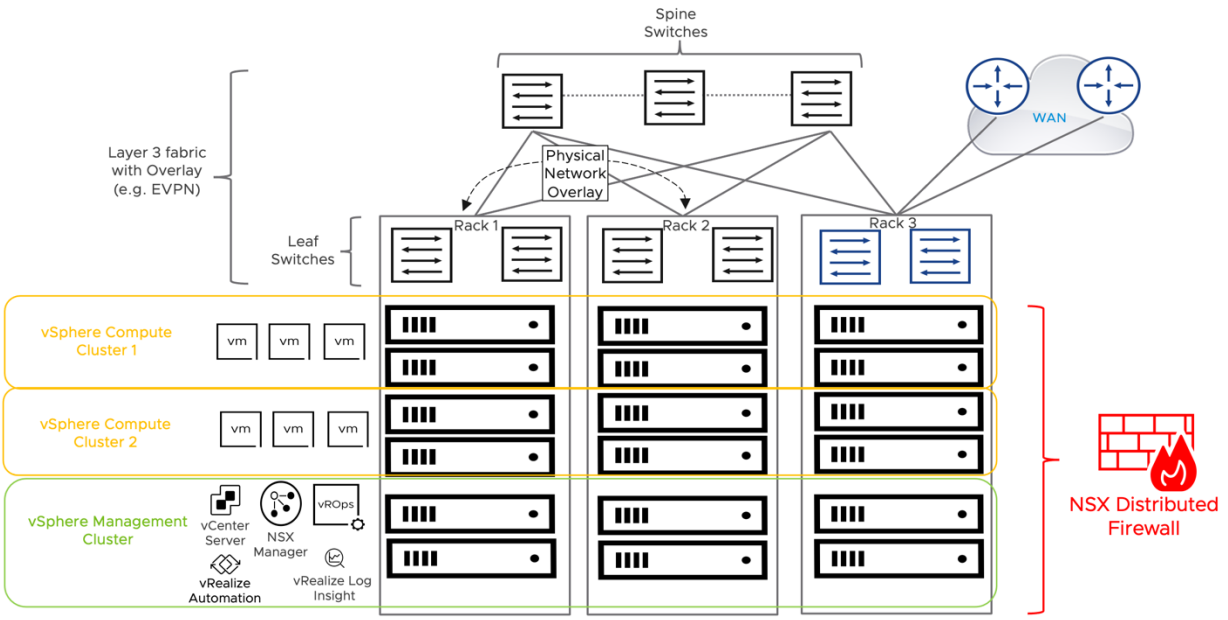


Figure 7-13: Security Only Deployment – L3 fabric with Overlays

For **NSX Network and Security only deployments** (FIGURE 7-14), the points to emphasize are:

- Larger physical fabrics can adopt a simplified VLAN/IP schema where unique VLANs are common across the environment.
- NSX Geneve encapsulation will work over the physical fabric overlay encapsulation without any problem as long as the virtual machines and physical fabric MTU account for the additional headers.
- The configuration of the physical devices becomes standardized and static. New networks and topologies to support the application lifecycle are provisioned in NSX transparently to the physical fabric.
- Edge nodes are connected to edge leaf switches with access to the WAN network.
- Edge node VMs should not be deployed on clusters striped across racks as it would require extending the L3 peering VLANs via the physical fabric overlay. Edge nodes should peer to the leaf switches in the same rack on dedicated local VLANs. Edges in different racks and vSphere clusters can be grouped in the same NSX edge cluster.

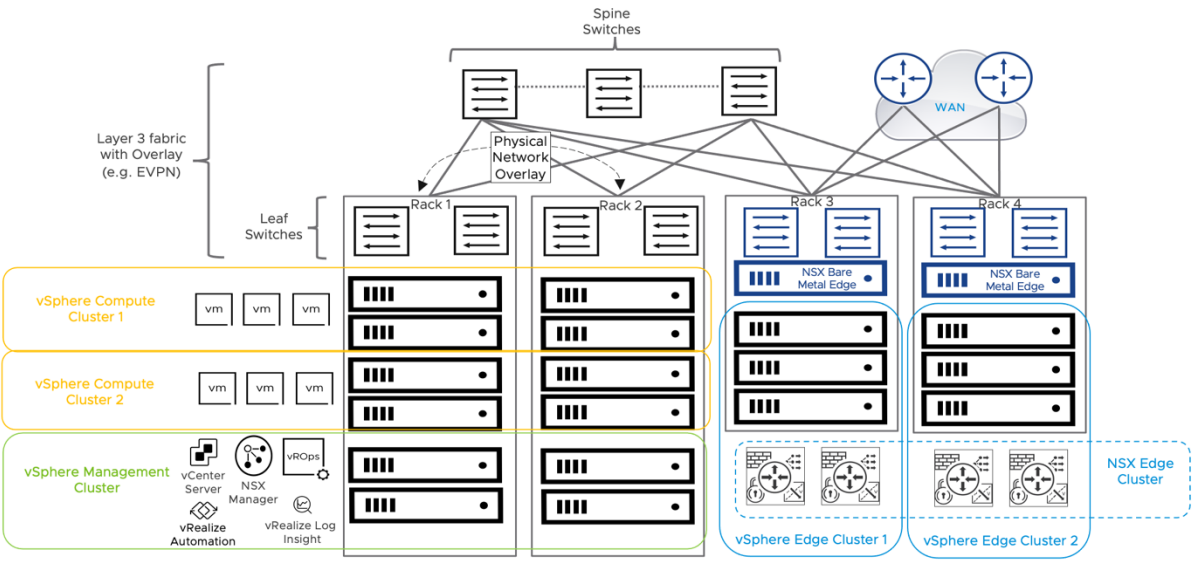


Figure 7-14: Network and Security Deployment – L3 fabric with Overlays

FIGURE 7-15 outlines a sample VLAN/IP schema for a layer three fabric with an overlay where NSX network overlays are also in use. The number of IP segments is minimized as the physical networks overlay allows each IP network to be available on every rack. Note the four physical to virtual transit segments (VLAN 106-108) in the edge racks, two per rack. They are not extended between racks and only provide connectivity to the local edge nodes.

VLANs & IP Subnets For Compute Racks

SVI Interface	VLAN ID	IP Subnet
ESXi Management	101	10.100.101.0/24
vMotion	102	10.100.102.0/24
Storage	103	10.103.103.0/24
Overlay	104	10.104.104.0/24

VLANs & IP Subnets For Management Racks

SVI Interface	VLAN ID	IP Subnet
ESXi Management	101	10.100.101.0/24
vMotion	102	10.100.101.0/24
Storage	103	10.100.103.0/24
VM Management	105	10.100.105.0/24

VLANs & IP Subnets For Edge Rack 1

SVI Interface	VLAN ID	IP Subnet
ESXi Management	101	10.100.101.0/24
vMotion	102	10.100.101.0/24
Storage	103	10.100.103.0/24
VM Management	105	10.100.105.0/24
Routed P2V-1	106	10.100.106.0/28
Routed P2V-2	107	10.100.107.0/28

VLANs & IP Subnets For Edge Rack 2

SVI Interface	VLAN ID	IP Subnet
ESXi Management	101	10.100.101.0/24
vMotion	102	10.100.102.0/24
Storage	103	10.100.103.0/24
VM Management	105	10.100.105.0/24
Routed P2V-3	108	10.100.108.0/28
Routed P2V-4	109	10.100.109.0/28

Figure 7-15: Sample VLAN and IP Subnets schema for NSX Network Virtualization deployment on a Layer 3 fabric with overlays

7.3 NSX Infrastructure Component Connectivity

7.3.1 Overview

NSX Manager Appliances (bundling manager and controller functions) are mandatory NSX infrastructure components. Their networking requirement is basic IP connectivity with the other NSX components. The detailed requirements for these communication are listed at HTTPS://PORTS.VMWARE.COM/HOME/NSX-DATA-CENTER.

NSX Manager Appliances are typically deployed on a hypervisor and connected to a VLAN backed port group on a vSS or vDS; there is no need for the colocation of the three appliances in the same subnet or VLAN. There are no dependencies on MTU or encapsulation requirements as the NSX Manager appliances send management and control plane traffic over the management VLAN only. The NSX Manager appliances ca be deployed on both ESXi or KVM hypervisors.

FIGURE 7-16 shows three ESXi hypervisors in the management rack hosting three NSX Manager appliances.

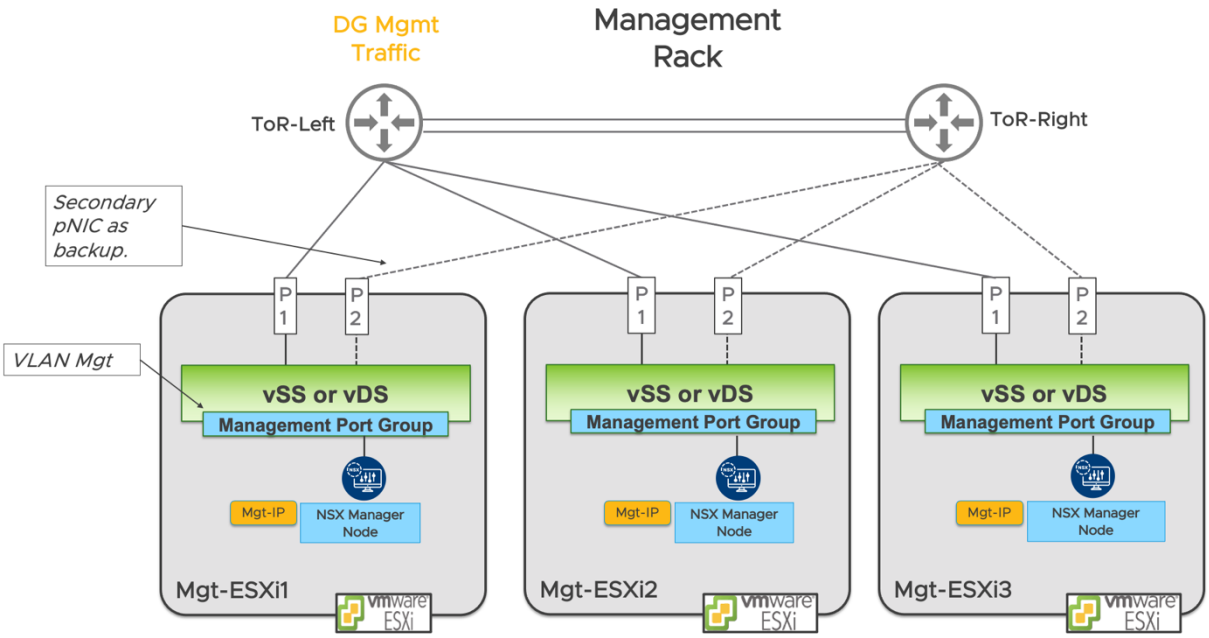


Figure 7-16: ESXi Hypervisor in the Management Rack

The ESXi management hypervisors are configured with a VDS/VSS with a management port group mapped to a management VLAN. The management port group is configured with two uplinks using physical NICs "P1" and "P2" attached to different top of rack switches. The uplink teaming policy has no impact on NSX Manager operation, so it can be based on existing VSS/VDS policy.

FIGURE 7-17 presents the same NSX Manager appliance VMs running on KVM hosts.

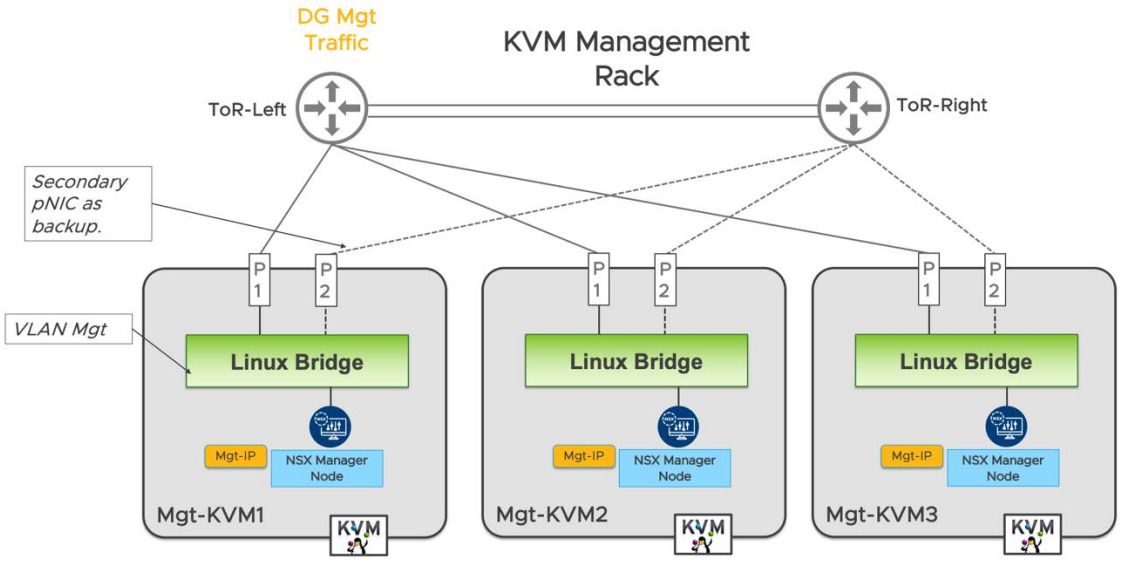


Figure 7-17: KVM Hypervisor in the Management Rack

The KVM management hypervisors are configured on a Linux bridge with two uplinks using physical NICs "P1" and "P2". The traffic is injected into a management VLAN configured in the physical infrastructure. Either active/active or active/standby is fine for the uplink team strategy for NSX Manager since both provide redundancy; this example uses simplest connectivity model with active/standby configuration.

In a typical deployment, the NSX management components should be deployed on a VLAN. This is the recommended best practice. The target compute cluster only need to have the hypervisor switch – VSS/VDS on ESXi and Linux Bridge on KVM. Deploying NSX management components on the software defined overlay requires elaborate considerations and thus beyond the scope of this document.

7.3.2 NSX Manager Node Availability and Hypervisor interaction

The NSX Management cluster represents a scale-out distributed system where each of the three NSX Manager nodes is assigned a set of roles that define the type of tasks that the node can implement. The cluster must have three nodes for regular operation; however, the cluster can operate with reduced capacity in the event of a single node failure. The cluster requires a majority of NSX Manager Nodes (i.e., two out of three) to be fully operational. At a minimum, it is recommended to spread the deployment of the NSX Manager Nodes across separate hypervisors to ensure that the failure of a single host does not cause the loss of a majority of the cluster. More generally, it is recommended to spread the deployment of the NSX Manager Nodes across separate failure domains to ensure that a single failure does not cause the loss of a majority of the cluster. A failure domain at a minimum should address the failure of a single host and may extend to a data store, vSphere cluster, or even cabinet or rack. NSX does not natively enforce this design practice.

When a single vSphere-based management cluster is available, deploy the NSX Managers in the same vSphere cluster and leverage the native vSphere Distributed Resource Scheduler (DRS) and anti-affinity rules to avoid instantiating more than one NSX nodes on the same ESXi server. For more information on how

to create a VM-to-VM anti-affinity rule, refer to the VMware documents on VM-to-VM and VM-to-host rules. For a vSphere-based design, it is recommended to leverage vSphere HA functionality to ensure single NSX Manager node can recover during the loss of a hypervisor. Furthermore, NSX Manager should be installed on shared storage. vSphere HA requires shared storage so that VMs can be restarted on another host if the original host fails. A similar mechanism is recommended when NSX Manager is deployed in a KVM hypervisor environment.

Additional considerations apply for management Cluster with respect to storage availability and IO consistency. A failure of a datastore should not trigger a loss of Manager Node majority, and the IO access must not be oversubscribed such that it causes unpredictable latency where a Manager node goes into read only mode due to lack of write access. If a single VSAN data store is being used to host NSX Manager cluster, additional steps should be taken to reduce the probability of a complete data store failure some of which are addressed in Physical placement considerations for the NSX Manager Nodes below. It is also recommended to reserve resources in CPU and memory according to their respective requirements. Please refer to the following links for details

NSX Manager Sizing and Requirements:

https://docs.vmware.com/en/VMware-NSX-Data-Center/3.2/installation/GUID-AECA2EE0-90FC-48C4-8EDB-66517ACFE415.html

NSX Manager Cluster Requirements with HA, Latency and Multi-site:

https://docs.vmware.com/en/VMware-NSX-Data-Center/3.2/installation/GUID-509E40D3-1CD5-4964-A612-0C9FA32AE3C0.html

7.3.3 Physical placement considerations for the NSX Manager Nodes

This section covers the NSX Manager cluster deployment options in relation to the underlying vSphere infrastructure. The NSX Manager Nodes part of the NSX Management Cluster are not required to exist in a single vSphere Cluster or even under a single vCenter as long as the IP Connectivity Requirements between the nodes are met. Predominantly, however, the most common placement of the NSX Manager Nodes is into a single vSphere cluster. More rarely, when high availability requirements demand it, they are spread across three vSphere clusters to increase availability.

7.3.3.1 Single vSphere Cluster for all NSX Manager Nodes in a Manager Cluster

When all NSX Manager Nodes are deployed into a single vSphere cluster, it is important to design that cluster to meet the needs of the NSX Managers with at least the following:

1. At least four vSphere Hosts. This is to adhere to best practices around vSphere HA and vSphere Dynamic Resource Scheduling (DRS) and allow all three NSX Manager Nodes to remain available during proactive maintenance or a failure scenario.
2. The hosts should all have access to the same data stores hosting the NSX Manager Nodes to enable DRS and vSphere HA.
3. Each NSX Manager Node should be deployed onto a different data store (this is supported as a VMFS, NFS, or other data store technology supported by vSphere)

4. DRS Anti-Affinity rules should be put in place to prevent, whenever possible, two NSX Manager VMs from running on the same host.
5. During lifecycle events of this cluster, each node should be independently put into maintenance mode, moving any running NSX Manager Node off the host prior to maintenance, or any NSX Manager Node should be manually moved to a different host.
6. If possible, a rack-level or critical infrastructure (e.g., Power, HVAC, ToRs) should also be taken into account to protect this cluster from any single failure event taking down the entire cluster at once. In many cases, this means spreading this cluster across multiple cabinets or racks and connecting the hosts in it to a diverse set of physical switches, etc. Rack level redundancy requires extending the management VLAN between the racks as the NSX Managers generally reside on a vCenter managed VLAN dvpg. Properly planned rack-level redundancy includes a cluster striped across three racks and DRS rules placing the NSX Managers on hosts in different racks because the failure of a cabinet with more than one NSX manager causes the loss of the quorum.
7. NSX Manager backups should be configured and pointed to a location running outside of the vSphere Cluster that the NSX Manager Nodes are deployed on.

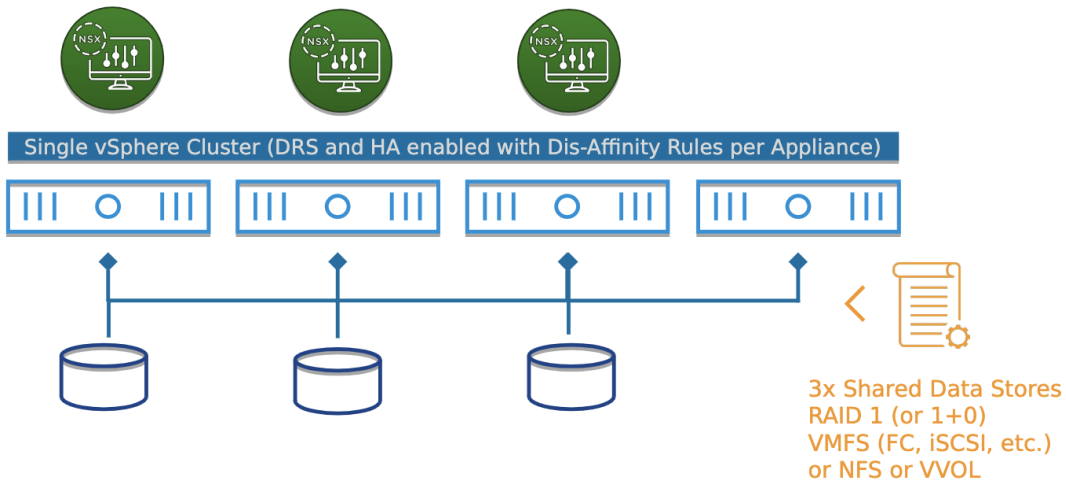


Figure 7-18: Single vSphere Cluster 1:1 Data Store

7.3.3.2 Single vSphere Cluster when leveraging VSAN as the storage technology

When all NSX Manager Nodes are deployed into a single vSphere cluster where VSAN is the storage technology in use, we should take additional steps to protect the NSX Manager Cluster's availability since VSAN will present only a single data store. Few VSAN specific parameters including, Primary Levels of Failures to Tolerate (PFTT), Secondary Levels of Failures to Tolerate (SFTT), and Failure

Tolerance Mode (FTM) (when only a single site in VSAN is configured, PFTT is set to 0 and SFTT is usually referred to as just FTT) govern the availability of the resources deployed on the VSAN datastore. The following configurations should be made to improve the availability of the NSX Management and Control Planes:

- FTT >= 2
- FTM = Raid1
- Number of hosts in the VSAN cluster >= 5

This will dictate that for each object associated with the NSX Manager Node, two or more copies of the data and a witness are available even if two failures occur, allowing the objects to remain in a healthy state. This will accommodate both a maintenance event and an outage occurring without impacting the integrity of the data on the datastore. This configuration requires at least five (5) hosts in the vSphere Cluster.

Cabinet Level Failures can be accommodated as well, by distributing the cluster horizontally across multiple cabinets. No more hosts than the number of failures that can be tolerated (a maximum of FTT=3 is supported by VSAN) should exist in each rack. This means a minimum of five (5) hosts spread across three (3) racks in a 2-2-1 pattern, and DRS rules preferentially placing the NSX Managers in different racks.

Implementing VSAN Stretched Clusters to provide cabinet level protection is not recommended. While appealing because of the reduced number of hosts required (2 hosts and a witness VM in 3 different failure domains are the minimum requirements), the failure of a rack may cause the loss of the NSX manager cluster quorum because two NSX manager appliances are placed in the same rack.

It is strongly recommended that hosts, any time they are proactively removed from service, vacate the storage and repopulate the objects on the remaining hosts in the VSAN cluster.

Providing a VSAN cluster with five hosts is the recommended approach, but fewer hosts are available in some situations. VSAN 7 update 2 introduced the Enhanced Data Durability feature, which may reduce the risk of running the NSX manager cluster with a storage policy with FTT=1. With this feature, if a failure occurs when one of the nodes has been taken out of service for maintenance, VSAN can rebuild the up-to-date data once the host is out of maintenance mode.

Additional VSAN settings and best practices that we should consider are:

- Enable Operations Reserve and Host rebuild reserve. This helps monitor the reserve capacity threshold, generates alerts when the threshold is reached and prevents further provisioning.
- Ensure that the VM storage policy applied to the NSX Managers is compliant.
- Ensure to use the Data migration pre-check tool before carrying out host/cluster maintenance
- Review the vSAN Skyline Health Checks regularly.

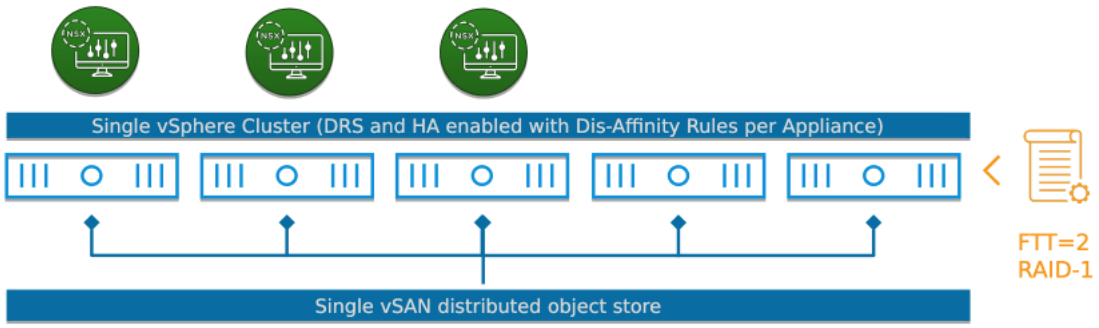


Figure 7-19: Single vSphere Cluster Single VSAN Data Store

7.3.3.3 Placement of NSX Manager Nodes Across Multiple vSphere Clusters

Depending on the design of the underlying infrastructure, it may become necessary or advantageous to place the three NSX Manager Nodes across multiple vSphere Clusters. This type of design tends to become relevant when a single vSphere cluster cannot provide the resiliency or availability for all failure scenarios that are required to be addressed. This can occur, for instance, if the physical infrastructure is preferred to be deployed across multiple facilities or computer rooms or if storage or other underlying infrastructure is required to be diverse. In this scenario:

- If the same IP space is not available to each vSphere Cluster, NSX managers may have IP addresses in different IP subnets . In such a scenario, we cannot provide management plane availability via a VIP, and an external load balancer may be part of the solution. Section DEPLOYMENT OPTIONS FOR NSX MANAGEMENT CLUSTER describes the management plane's high availability options and their requirements.
- If VSAN is part of this design, we recommend avoiding HCI Mesh and use the local VSAN datastore presented to each vSphere Cluster for the NSX Manager Node that is running in it.

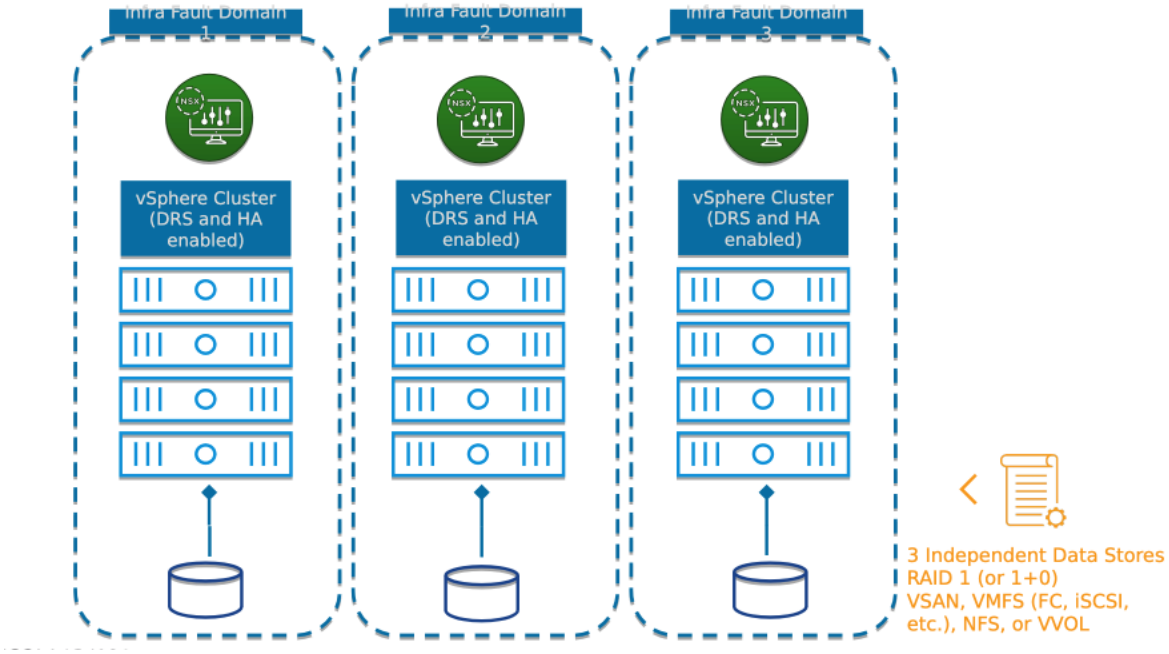


Figure 7-20: 3x vSphere Clusters Each in a Different Infrastructure Failure Domain

7.3.4 Deployment Options for NSX Management Cluster

Three NSX manager appliances form a cluster. The database is replicated to all the nodes in the cluster, so there is a single database cluster instance. The benefit of clustering is that it provides high availability of all services, including GUI and API access to the cluster. Combining the NSX Manager, policy, and central controller reduces the number of appliances deployed and allows greater flexibility with availability models.

NSX Manager appliance serves two critical areas. The first one is the consumption by external systems and user access. Many different endpoints (such as HTTP, vRA, Terraform, Network Container Plugin, custom automation modules) can consume NSX Manager northbound via an IP address. It is the single point of entry for the entire system configuration. The second is the communication with the other NSX components (controllers and transport nodes). The controller communication to the transport node is done via the controller role (element) within the NSX appliance node. We described the interaction between the controller role and the NSX components in chapter 2 MANAGEMENT PLANE AND CONTROL PLANE. The controller availability model is majority based, requiring at least two nodes to be available for normal operations.

The NSX Manager cluster can expose northbound API and GUI access via multiple availability models. There are three different configuration modes available for northbound access to the NSX Manager cluster.

- Default deployment (each node uniquely addressable, no common IP address)
- Cluster VIP based deployment
- External Load Balancer based deployment

7.3.4.1 Default Deployment

Deploying a 3-node cluster without any additional configuration is the default and the simplest option with the least number of IP addresses consumed.

Different endpoints (users and automation systems) can access different nodes. Because the configuration is always synchronized, the choice of the node does not matter. Each node is accessible via a distinct IP address (or FQDN). The availability is driven by the external system by choosing a different IP address (FQDN) in case of a node failure. For example, if vRA or an API script uses the FQDN of node A and node A fails, there has to be some manual intervention to restore the functionality. The same applies to GUI access via a browser. The NSX administrator must repoint the browser to a different node. The options are to change the FQDN in your API script or browser or update the FQDN entry in the DNS. Using round-robin DNS may result in intermittent connectivity in a failure scenario. In terms of topology requirements, as long as there is IP connectivity between all the nodes, this model will work.

7.3.4.2 Cluster VIP based deployment

The second deployment option is based on a simple active/standbys redundancy model, in which we configure a virtual IP address on the management cluster. The cluster VIP configuration option provides node-level redundancy through a virtual IP address on the cluster itself. This virtual IP (like VRRP/HSRP) provides redundancy in accessing the cluster via a single FQDN. In other words: the entire northbound access to the NSX Manager is available via this FQDN (or IP). Since a single FQDN name is available to all endpoints, all the GUI and API requests are accessed through a single node owning the VIP. Essentially, it is a simple availability model, where external intervention is not required in case of a node failure.

The cluster virtual IP address is an IP address floating between the cluster nodes. One of the cluster nodes is assigned as the owner of the cluster VIP. If the case of failure, the system will assign a new owner. Since the cluster VIP must remain the same, it assumes that other nodes are available in the same subnet. The cluster VIP feature uses gratuitous ARP to update the mac-address and the ARP table of the upstream network devices. Thus, it is mandatory to have all NSX Managers in the cluster in the same subnet. From the physical topology perspective, the placement of the NSX Manager nodes can vary. In a L2 physical fabric topology, the NSX Manager appliances can be in the same or different racks. For a L3 topology, all nodes must be in the same rack, assuming that VLAN/subnets are confined to a rack.

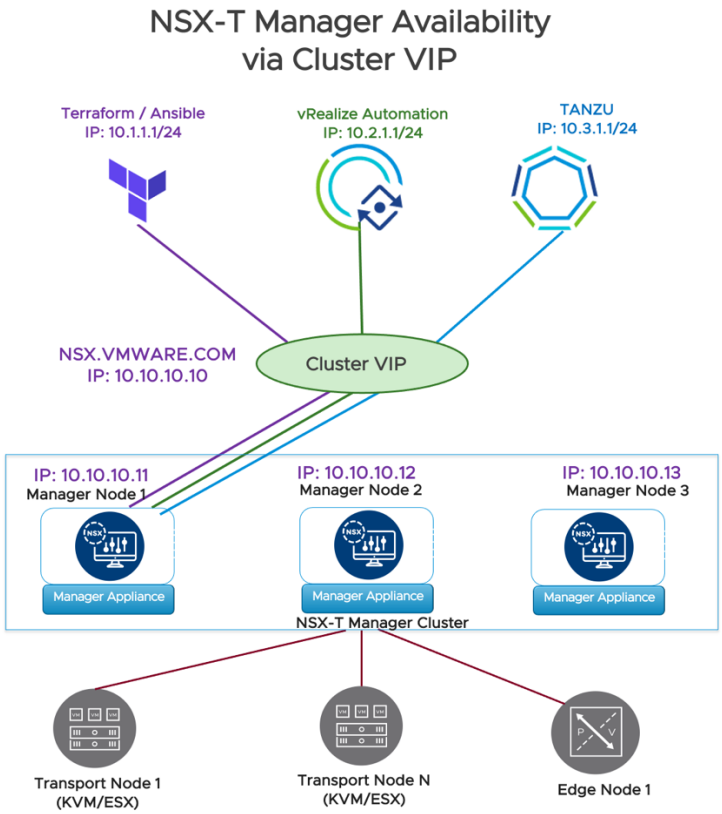


Figure 7-21: NSX Manager Appliances Availability with Cluster VIP

The NSX Manager availability has improved compared to the previous option; however, it is important to clarify the distinction between node availability and load-balancing. In the case of cluster VIP, all the API and GUI requests go to one node, and it's not possible to achieve load-balancing of GUI and API sessions. In addition, the sessions that were established on a failed node will need to be re-authenticated and re-established at the new owner of the cluster VIP. The cluster VIP model is also designed to address the failure of certain NSX Manager services but cannot guarantee recovery for certain corner cases service failures.

It is important to emphasize that the northbound clients' connectivity happened via the cluster VIP. Still, the communication between NSX Manager nodes and the NSX transport nodes components happens on the individual node IPs.

The cluster VIP is the preferred and recommended option for high availability of the NSX Manager appliance nodes.

7.3.4.3 External Load Balancer based deployment

The third deployment option is to keep the same configuration as the first option but add an external load balancer. A VIP on the load balancer will represent the manager nodes as the physical servers in the server pool. Then the UI and API access to the management cluster will go through the VIP on the load balancer. The advantage of this option is that the endpoint access to the NSX Manager nodes will have load-balancing, and the cluster management access is highly available via a single IP address. The external load-balancer option also makes the

deployment of NSX manager nodes independent of underlying physical topology (agnostic to L2 or L3 topology).

Additionally, one can distribute nodes to more than one rack to achieve rack redundancy (in L2 topology, it is the same subnet but a different rack, while in L3 topology, it will be a distinct subnet per rack). The downside of this option is that it requires an external load balancer and additional configuration complexity based on the load-balancer model. The make and model of load-balancer are left to the user preference; however one can also use NSX native load balancer included as part of the system and managed by the same cluster of NSX Manager nodes that are load-balanced or the standalone NSX Advanced load balancer (formally AVI network) offered as a part of NSX Datacenter portfolio.

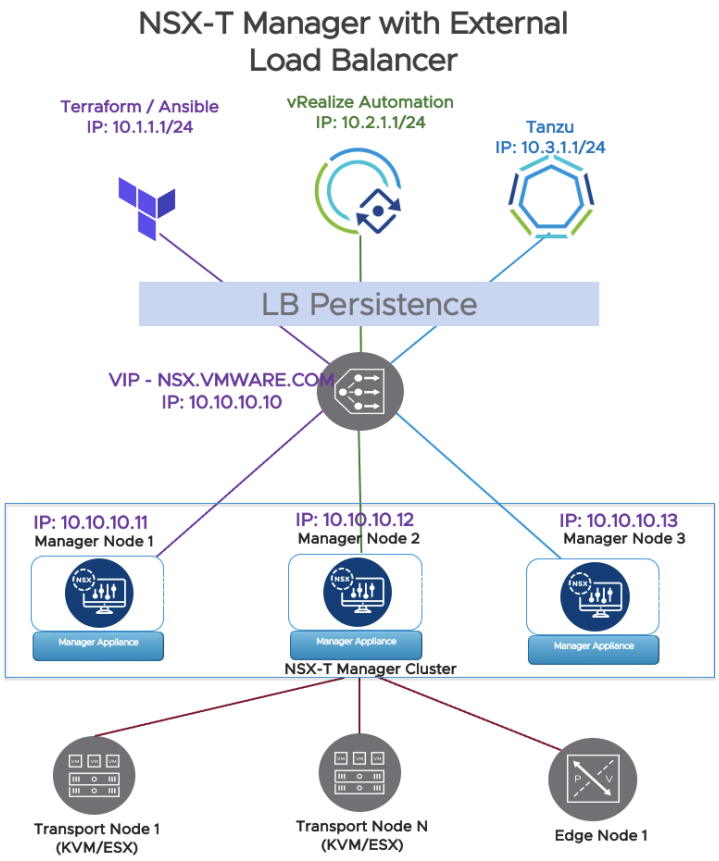


Figure 7-22: NSX Manager Appliances with External Load Balancer

FIGURE 7-22 presents a scenario where the NSX manager nodes are load-balanced by an external load balancer, and session persistency is based on the source IP of the client. With this configuration, the load balancer will redirect all requests from the same client to the same NSX manager node.
NSX Manager can authenticate clients in four ways - HTML basic authentication, client certificate authentication, vIDM, and session. The API-based client can use all four forms of authentication while web browsers use session-based authentication. The session-based authentication typically requires LB persistence configuration, while API-based access does not mandate that. FIGURE

7-22 represents a VIP with LB persistent configuration for both browser (GUI) and API-based access.

While one can conceive an advanced load-balancing schema in which dedicated VIP for browser access with LB persistent while other VIP without LB persistence for API access, this option may have limited value in terms of scale and performance differentiation while complicating the access to the system. For this reason, it is highly recommended to first adopt the basic option of LB persistence based on source IP with a single VIP for all types of access. The overall recommendation is to start with a cluster VIP and move to an external LB if real needs exist.

7.3.4.4 Singleton NSX Manager

The resources required to run a cluster of three NSX Managers may represent a challenge in small environments. Starting with NSX version 3.1, VMware supports deploying a single NSX manager in production environments. This minimal deployment model relies on vSphere HA and the backup and restore procedure to maintain an adequate level of high availability.

vSphere HA will protect against the failure of the physical host where the NSX manager is running. vSphere HA will restart NSX Manager on a different available host. Enough resources must be available on the surviving hosts; vSphere HA admission control can help ensure they are available in case of failure.

Backup and restore procedures help in case of failure of the NSX manager itself. The SFTP server where the backup is stored should not be placed on an infrastructure shared by the single NSX Manager node.

7.4 Compute Cluster Design

This section covers both ESXi and KVM compute hypervisors; unless otherwise specified, discussions and recommendations apply to both. Compute hypervisors host the application VMs in the data center.

In a typical enterprise design where both NSX network and security virtualization are adopted, compute hosts will carry at least two kinds of traffic, typically on different VLANs – management and overlay. The overlay VLAN will carry encapsulated traffic for all the application networks, minimizing the number of VLANs defined on the physical fabric. Because overlay traffic is involved, the uplinks are subject to the MTU requirement mentioned earlier in section PHYSICAL NETWORK REQUIREMENTS. Security-only designs are not impacted by the MTU requirement but will generally include multiple workloads VLANs based on the application requirements.

Additionally, based on the type of hypervisor, compute hosts may carry additional types of infrastructure traffic such as storage (VSAN, NFS, and iSCSI) and vMotion. The ESXi hypervisor defines specific VMkernel interfaces for this infrastructure traffic, typically connected to separate VLANs. Similarly, for the KVM hypervisor, specific interfaces and VLANs are required. Details on particular hypervisor requirements and capabilities can be found in documentation from their respective vendors.

A specific note for the KVM compute hypervisor: NSX uses a single IP stack for management and overlay traffic on KVM hosts. Because of this, both management and overlay interfaces share the same routing table and default gateway. This can be an issue if those two kinds of traffic are sent on different VLANs as the same default gateway cannot exist on two different VLANs. In this case, it is necessary to introduce more specific static routes for the remote overlay networks pointing to a next-hop gateway specific to the overlay traffic.

NSX offers choices for the management of infrastructure and guest VM traffic (overlay or VLAN) through the flexibility of uplink profiles and teaming policies as described in chapter 3 (UPLINK PROFILE). This chapter presents configuration choices and capabilities based on the requirements and best practices prevalent in existing data-center deployments. Compute hosts network requirements are centered around availability, performance, and manageability. Availability is generally the most important design quality and in most cases it is achieved by connecting the hosts to two different switches and ensuring that traffic is redirected on a different path in case of a failure. Performance and manageability may lead the design in different directions based on specific requirements and the network architect preference. In this context two different design patterns emerge:

Optimization across all available physical NICs– With this choice, all traffic types share all available pNICs. The assumption is that by making all pNICs available to all traffic, we can avoid traffic hot spots during a peak/burst, and the probability of contention is reduced due to the number of links and speed offered. This type of traffic management is typically suitable with 25 Gbps or greater speed links. In the case of lower-speed pNIC, it may be necessary to enable traffic management tools such as NIOC to provide adequate bandwidth to critical traffic. One example is VSAN traffic. The teaming type offered in NSX that enables this behavior is called “Load Balanced Source Teaming.”

Deterministic traffic per pNIC– A certain traffic type is only carried on a specific pNIC, thus allowing dedicated bandwidth for a given traffic type. Additionally, it allows deterministic failure as one or more links could be in standby mode. Based on the number of pNICs, one can design a traffic management schema that avoids the contention of two high bandwidth flows (e.g., VSAN and vMotion). The teaming type offered in NSX that enables this behavior is called "Failover Order." We can build a teaming failover policy with a single pNIC or more. When a single pNIC is used, its failure is not covered by a standby pNIC. Such a scenario is not common, but it can be an appropriate choice when other high availability mechanisms like vSphere HA or application-level redundancy are in place.

A scenario where we may leverage deterministic traffic per pNIC is in the case of multiple distinct physical networks, e.g., DMZ, Storage, or Backup Networks, where the physical underlay differs based on pNIC.

Designing a traffic management schema that utilizes both design patterns is possible. This design guide covers both scenarios based on specific requirements and makes generalized recommendations.

7.4.1 Running a VDS prepared for NSX on ESXi hosts

NSX 3.0 introduces the capability of running NSX directly on the top of a VDS (with VDS version 7.0 or later.) This short section introduces the reasons for this feature, its impact on ESXi host design with NSX and some guidelines for moving to this new model.

7.4.1.1 NSX on VDS

Simpler install

The N-VDS introduced on the ESXi platform is very close to the VDS. In fact, the name "N-VDS" was chosen to highlight this tight relationship. Still, the N-VDS is an opaque switch (an opaque switch and an opaque network means it's not instantiated via vCenter and does not have any object existence native to vCenter) that needs to be instantiated separately, or migrated to, when the user wants to start using NSX on ESXi. The capability of running NSX directly on the top of an existing VDS removes this burden and makes the deployment and adoption of NSX must simpler. This is especially critical with the two pNICs design and/or fully collapsed design where management, edge and applications VMs co-exist on the same hypervisor. There is no need for migration of VMkernel interfaces and specific considerations for security and availability.

Compatibility with third party solutions

With the N-VDS, NSX segments appeared as opaque network in vCenter. Even if opaque networks have been available in vCenter for almost a decade, way before NSX was developed, many third-party solutions still don't take this network type into account and, as a result, fail with NSX. When running NSX on VDS, NSX segments are represented as DVPGs (now onward called NSX DVPG) in vCenter. Third party scripts that had not been retrofitted for opaque networks can now work natively with NSX.

7.4.1.2 Impact on ESXi host design

As we have seen earlier in this chapter, the traffic in/out an ESXi host can be classified in two broad categories:

- Infrastructure traffic. Any traffic that is originating from the ESXi host and its functionality to support application, storage, availability and management. This traffic is typically initiated and received on VMkernel interfaces defined on the host and includes traffic for management, vMotion, storage, high availability etc.
- VM and/or application traffic: This is the traffic between virtual machines running on the host. This traffic might be local to the hypervisor but can also extend to VMs on remote hosts. NSX is all about providing advanced networking and security features for VM traffic.

Until now, when the administrator introduced NSX, they needed to deploy a new N-VDS virtual switch on their hosts. The ESXi platform can run multiple separate virtual switches (VSS/VDS/N-VDS) side-by-side with no problem, however, those virtual switches cannot share physical uplinks. The administrator is thus left with two main options:

1. Deploy an N-VDS for VM traffic along with the existing virtual switch (VSS/VDS) handling infrastructure traffic. This solution is very easy to deploy as it does not impact the current host operations. However, it requires separate uplinks for the N-VDS, meaning additional connectivity to the physical infrastructure.
2. The second, an efficient option is to consolidate both infrastructure and VM traffic on a single N-VDS. Deploying this model is more complex as it implies a virtual switch migration: VMKernel interfaces and physical uplinks need to be moved from VSS/VDS to the N-VDS. When the hosts only have two high-speed uplinks, which is increasingly the case with modern servers, migrating to a single N-VDS becomes mandatory to maintain uplink redundancy for the virtual switch.

Because of those considerations, the NSX design guide traditionally addresses a 4 (or more) pNICs design, corresponding to the first option, and a two pNIC design for the second. The introduction of NSX on VDS changes all this. NSX can be installed on a VDS without incurring migration of infrastructure traffic VMkernel, making the second option just as simple as the first one.

The next sections, showing the ESXi compute node design options, will thus focus on installing NSX on VDS in a two pNIC scenario. This scenario will in fact be unchanged if there are more than two pNICs on the host. The use-cases will still cover a four pNICs design, for those who plan on keeping the N-VDS for now, and for the cases when multiple virtual switches are a requirement for other reasons.

7.4.1.3 When to run NSX on VDS

As mentioned earlier, in order to run NSX on VDS, you need NSX 3.0 or later and a VDS version 7.0 or later.

- For a greenfield deployment that meet those requirements, we recommend starting with NSX on VDS. This is the simpler approach and in future N-VDS shall converge on the VDS.
- VMkernel should remain on DVPG with VDS with NSX

- For those already running NSX on N-VDS, or for those with a new install that cannot meet the version requirements, the recommendation is to stay on N-VDS for now.
    - The N-VDS remains fully supported. In this case one must migrate and keep VMkernel on NSX unless one has 4 pNICs configurations
    - There is almost no functionality difference between N-VDS and VDS with NSX. In fact, one could even mix VDS and N-VDS in the same NSX network. For a given compute cluster do not mix and match the NSX virtual switch type. For a given host with more than 2 pNICs, coexistence of all NSX virtual switch (not the third party) allowed except N-VDS and VDS with NSX in the same host.
    - One can introduce new compute cluster or vCenter within the brownfield deployment, for which the recommendation is to deploy VDS with NSX with properly supported software on compute and NSX. In this case VMkernel can remain on DVPG.
    - The goal is to eventually convert every NSX deployment to the VDS model. Future conversion tool will automate this conversion and make it straightforward.
    - One can also migrate their current N-VDS deployment into a VDS one right now. However, the recommendation is to wait for the conversion tool as a manual migration would be unnecessarily complex.

7.4.1.4 NSX on VDS and Interaction with vSphere & Other Compute Domains

The relationship and representative difference between N-VDS vs NSX on VDS is subtle and requires consideration in designing the cluster and operational understanding. The N-VDS has a complete independence to underlying compute manager (vSphere or any other compute domains like or in cloud like AWS, Azure or Google Cloud). Unlike deployment of VDS via vCenter, N-VDS deployment is managed via NSX manager. Because of this decoupling it allows consistent connectivity and security with multiple compute domains. This consistent connectivity is shown in below FIGURE 7-23 first part with N-VDS and its relation to vCenter. With NSX 3.0, the NSX can be enabled on traditional VDS allowing same level of flexibility, however now there is a mandatory requirement of having vCenter to instantiate a VDS. This capability when enabled, depicted in below FIGURE 7-23 center part as a NSX DVPG. The center part of the figure below depicts a case of single VDS with NSX which is logically similar to first one, where only difference is representation in vCenter - opaque vs NSX DVPG. The third part of the figure represent multiple VDS. This is possible either with single vCenter with each cluster having dedicated VDS or with multiple vCenters with VDS. In this later case of multiple VDSs, the same segment of NSX is represented via unique NSX DVPG under each VDS or vCenters. This might represent a challenge operationally identifying VM connectivity to VDS and the automation that relies on the underlying assumption; however, future releases shall make this identification easier with unique names. Typically, it is a good practice to invoke a single VDS per compute domain and thus have a consistent view and operational consistency of the VM connectivity. However, there are cases where single VDS invocation may not be ideal from separation of workload for security, automation, storage

policy, NIOC control and provisioning boundary. Thus, it is acceptable to have a multiple VDS per given vCenter.

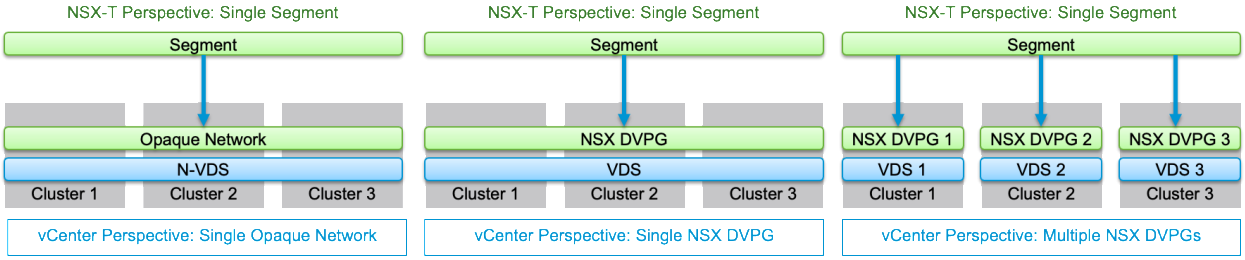


Figure 7-23: N-VDS vs. VDS with NSX – Representation

The details of these differences and additional considerations are documented with following KB articles

https://kb.vmware.com/s/article/79872

Co-existence with existing NSX, KVM and New vSphere Clusters

In practical deployment, where one has existing NSX deployed with N-VDS (ESXi) or OVS (KVM). The core realization of consistent networking and security does not change with advent of VDS with NSX. FIGURE 7-24 describes such configuration. Of course, for the VDS with NSX based cluster requires appropriate vSphere and ESXi version and NSXDVPG port for a VM connectivity.

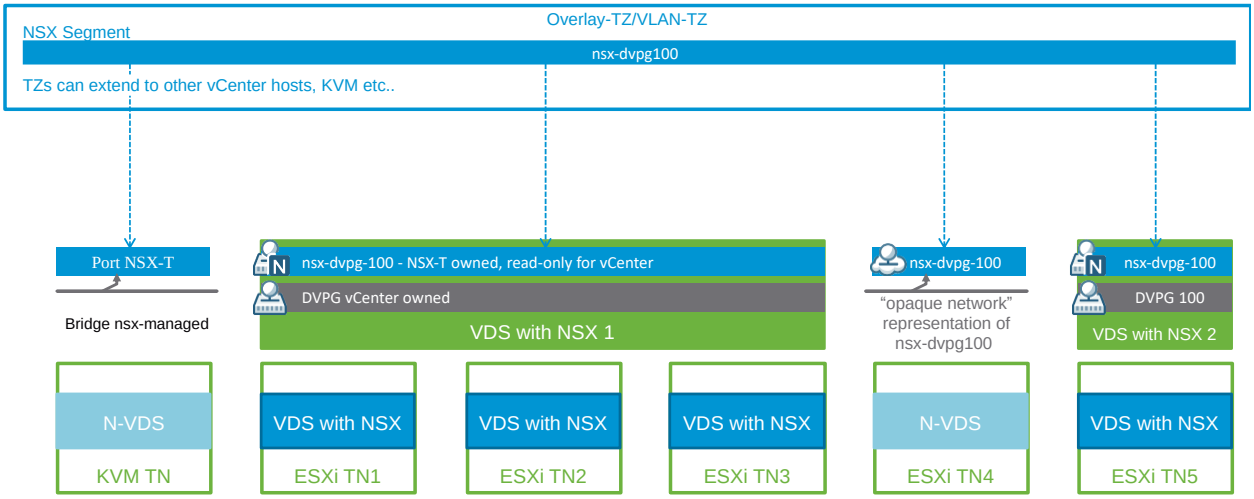


Figure 7-24: Multi-domain compute co-existence

The key concept here is that NSX abstract the connectivity and security via segment. This segment is represented via various realization like NSX DVPG in VDS with NSX, Port-NSX in KVM or an opaque network in N-VDS. Regardless of the underlying switch DFW can be realized either on VLAN or overlay however only with NSX DVPG and not via vSphere DVPG.

7.4.2 ESXi-Based Compute Hypervisor with two pNICs

This section provides recommendations for deployments of computing hypervisors with the following parameters:

- Two pNICs
- All host traffic (VM and infrastructure traffic) shares the same two pNICs
- Each host traffic type has a dedicated IP subnet and VLAN

We will cover the configuration options for two kinds of designs in the next sections, one that enforces deterministic traffic per pNIC the second that optimizes link utilization by leveraging all links for all types of traffic.

7.4.2.1 Deterministic Traffic per pNIC

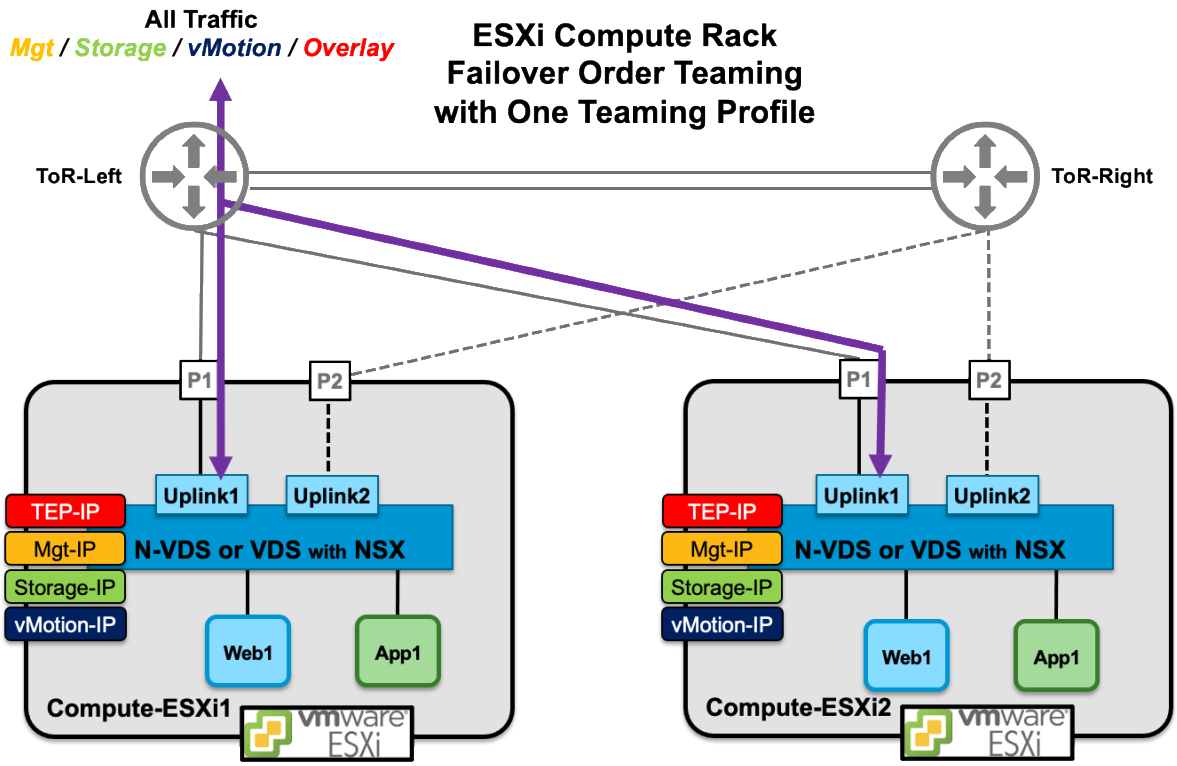


Figure 7-25: ESXi Compute Rack Failover Order Teaming with One Teaming Policy

In FIGURE 7-25Figure 7-25, a single virtual VDS or N-VDS is used with a two pNICs design and carries both infrastructure and VM traffic. Physical NICs "P1" and "P2" are attached to the different top of rack switches. The teaming policy selected is failover order active/standby; "Uplink1" is active while "Uplink2" is standby.

- If the virtual switch is an N-VDS, all traffic on the host is carried by NSX segments (VLAN or overlay), and the redundancy model can be achieved with a single default NSX teaming policy.

- If the virtual switch is a VDS, only NSX DVPG traffic will follow the NSX teaming policy. The infrastructure traffic will follow the teaming policy defined in their respective VDS DVPGs configured in vCenter.

The top-of-rack switches are configured with a first-hop redundancy protocol (e.g., HSRP or VRRP), providing the active default gateway for all the VLANs on "ToR-Left ." The VMs are attached to overlay or VLAN segments defined in NSX, and they will follow the default teaming policy regardless of the type of segment they are connected to. With the use of a single teaming policy, the above design allows for a simple configuration of the physical infrastructure and simple traffic management at the expense of leaving an uplink completely unused.

It is, however, easy to load balance traffic across the two uplinks while maintaining the deterministic nature of the traffic distribution. The following example shows the same hosts, this time configured with two separate failover order teaming policies: one with P1 active, P2 standby, and the other with P1 standby, P2 active. Then, individual traffic types can be assigned a preferred path by mapping it to either teaming policy.

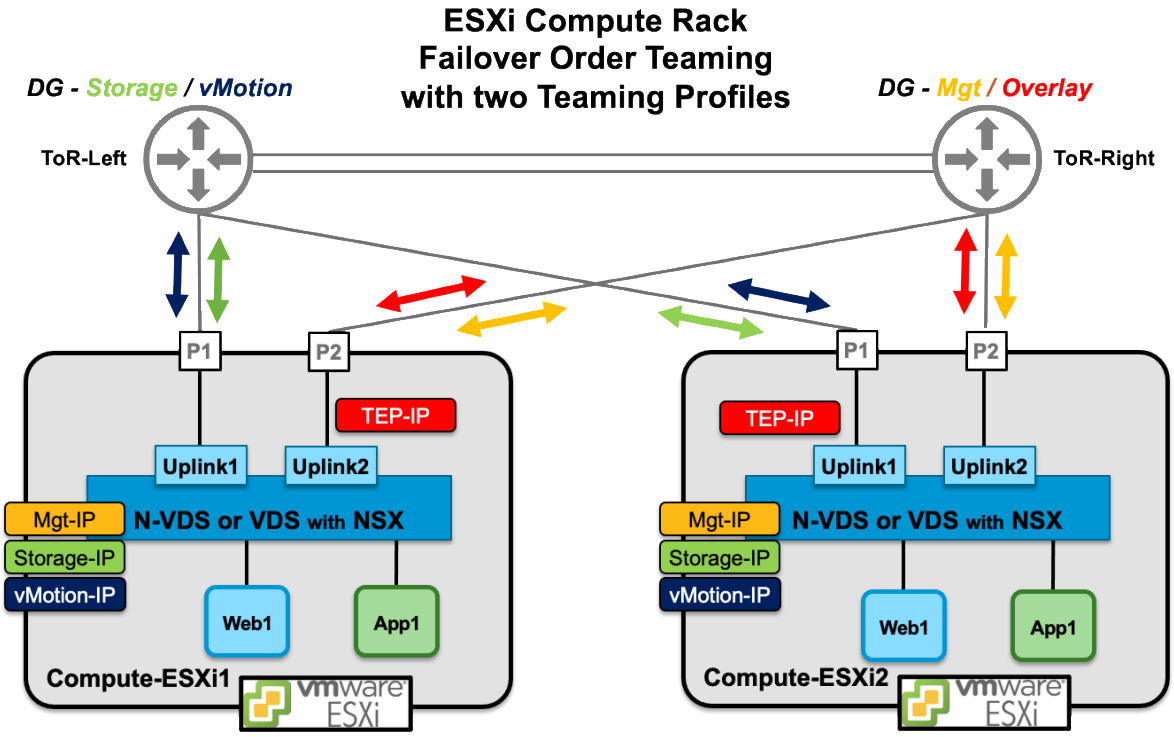


Figure 7-26: ESXi Compute Rack Failover Order Teaming with two Teaming Policies

In FIGURE 7-26, storage and vMotion traffic follow a teaming policy setting P1 as primary active, while management and VM traffic are following a teaming policy setting P2 as primary active.

- When the virtual switch is an N-VDS, all segments follow the default teaming policy by default. VLAN segments can, however, be associated with additional teaming policies (identified by a name and thus called "named" teaming policies). We can thus achieve the above design with a default

teaming policy (P2 active/P1 standby) and an additional named teaming policy (P1 active/P2 standby) to which NSX VLAN segments for storage and vMotion traffic are mapped. Overlay networks always follow the default teaming policy, and it is impossible to associate them with a named teaming policy.

- When the virtual switch is a VDS with NSX, the above design is achieved with a default teaming policy P2 active & P1 standby. Then, the DVPGs for infrastructure traffic need to be configured individually: storage and vMotion will have a failover order teaming policy setting P1 active & P2 standby, while the management DVPG will be configured for P2 active & P1 standby.

The ToR switches are configured with a first-hop redundancy protocol (FHRP), providing an active default gateway for storage and vMotion traffic on "ToR-Left," management, and overlay traffic on "ToR-Right" to limit interlink usage. Multiple teaming policies allow the utilization of all available pNICs while maintaining deterministic traffic management. This is a better recommended approach when adopting a deterministic traffic per pNIC design approach.

7.4.2.2 Optimization of traffic across all the available physical NICs

NSX supports two source teaming policies providing load balancing, load balancing based on the source port and load balancing based on the source mac address. Those are the exact equivalent to the corresponding teaming policies available in vSphere.

FIGURE 7-27, shows a two-pNIC design as in the previous example, with load balance based on source port teaming policy. Notice that NSX instantiates one TEP on every uplink specified in the teaming policy configuration to be able to send overlay traffic on all the corresponding physical ports. With this kind of policy, potentially both uplinks are utilized based on the hash value generated from the source port originating the traffic or the source MAC address of the traffic. In case of a pNIC failure, the virtual switch teaming policy moves the corresponding TEP interface to a still operational uplink. That means that in a failure scenario multiple TEPs are still operational but send and receive traffic over the same physical link.

Load balancing based on source port is generally sufficient and recommended over a MAC-based hash because each VM usually generates traffic from a single MAC address. Both infrastructure and guest VM traffic benefit from this policy, allowing the use of all available uplinks on the host. Infrastructure traffic is generally handled by different VMK interfaces dedicated to each traffic type, making it possible for the hashing algorithm to place different kinds of traffic on different links. Each traffic type will flow over a single link, though. vSphere features such as multi-nic vMotion can address the requirement for load balancing the same kind of traffic across different physical NICs.

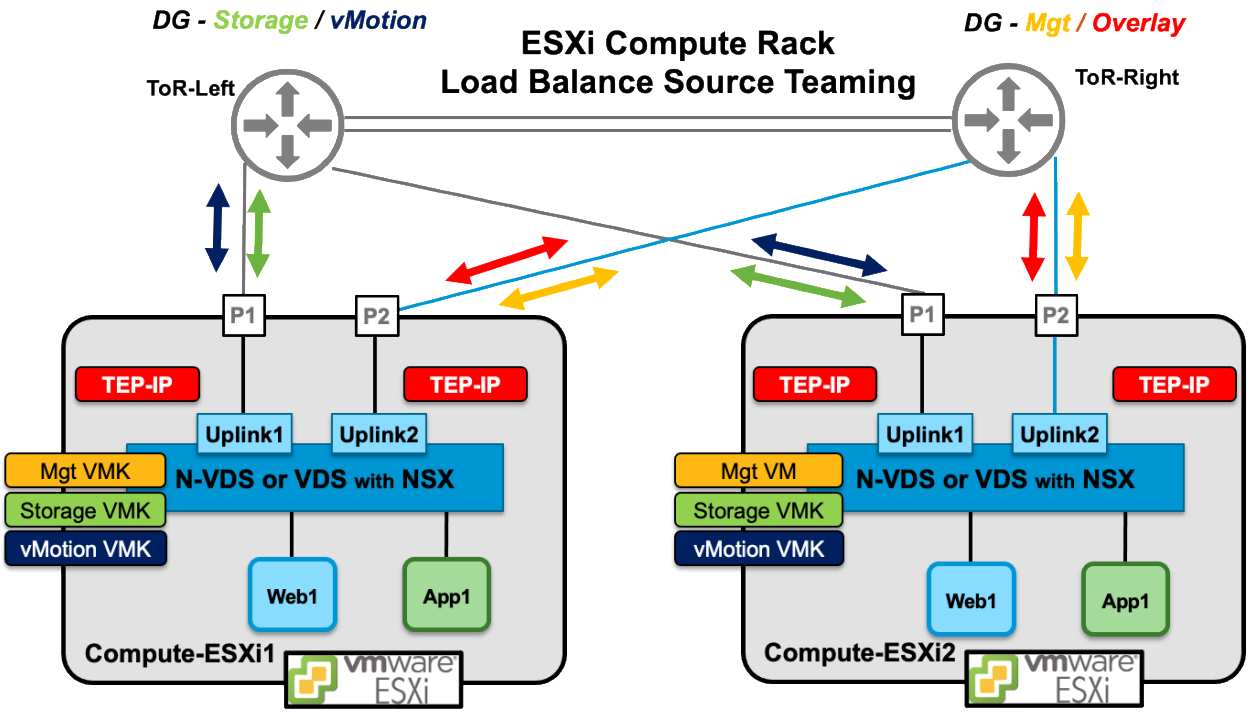


Figure 7-27: ESXi Compute Rack Load Balanced Source Teaming

Additionally, one can utilize a mix of the different teaming policy types together such that infrastructure traffic (VSAN, vMotion, Management) leverage "failover order" enabling deterministic bandwidth and failover, while VM traffic use some "load balance source" teaming policy, spreading VM traffic across both pNICs.

- On a host running NSX with an N-VDS, the default teaming policy will have to be configured for "load balance source," as it's the only policy that overlay traffic follows. Then, we can map individual VLAN segments to named teaming policies steering the infrastructure traffic to the desired uplink.
- On a host running VDS with NSX, overlay traffic will follow the default teaming policy defined in NSX. Infrastructure traffic will follow the individual teaming policies configured on their DVPGs in vCenter.

Based on the requirement of underlying applications and preference, one can select a type of traffic management as desired for the infrastructure traffic. However, for the overlay traffic, it is highly recommended to use one of the "load balanced source" teaming policies as they're the only ones allowing overlay traffic on multiple active uplinks and thus provide better throughput in/out of the host for VM traffic.

7.4.2.3 LAG based traffic distribution

An alternate method to distribute traffic over multiple pNICs is by implementing LAG. This scenario requires the ESXi hosts to be connected to separate ToRs forming a single logical LAG based on a proprietary multi-chassis link aggregation solution (MLAG, VPC, etc.) to provide ToR redundancy. This introduces

troubleshooting complexity and potential support coordination challenges across multiple vendors. Relying on this kind of technology is not recommended because it's vendor dependent and typically involves significant vendor-specific limitations that VMware has not validated. The private cloud fundamental relies on decoupling of physical network to NSX SDN topology, driving flexibly life cycle on both domains, and thus use of LAG from Host is strongly discouraged and if possible avoid it.

However, existing deployment may carry this type of teaming, and the operation team may have the knowledge and have accepted the risk to support LAG-based teaming. In such situations, LAG is an acceptable solution for compute only hosts.

If the compute hosts carry edge VMs for North-South traffic, and the topology requires peering over LAG, a change of the design is highly recommended, as support of the routing peer over the LAG is not supported and not validated by VMware. One option is to decouple the edge and compute hosts in a separate vSphere cluster and not implement LAG on the hosts where the edge node resides. When a separate vSphere edge cluster is not desirable, providing additional pNICs not part of the LAG for the edge VMs L3 peering is the most preferred option.

7.4.3 ESXi-Based Compute Hypervisor with Four (or more) pNICs

In most cases, a host with four or more pNICs can be configured the same way as a 2-pNIC host. There are just more uplinks available for consumption by the different teaming policies. We will focus on a few scenarios where four pNICs or more are necessary.

7.4.3.1 Simple install or traffic on separate uplinks for policy reason

With a minimum of four pNICs, an ESXi host can run two virtual switches, each with redundant uplinks. This type of installation has the following benefits:

- the design with 4 pNIC design provides a non-disruptive and straightforward installation of NSX on existing ESXi hypervisors. We can deploy an N-VDS (or a new VDS prepared for NSX) with two dedicated uplinks for VM traffic while leaving an existing VSS/VDS running on two separate uplinks to handle the infrastructure traffic.
- On top of the benefits from this simple NSX installation, the model also provides a strict separation between VM and infrastructure traffic. There are scenarios where this separation is mandated by policy. The fact that NSX is deployed on its dedicated virtual switch ensures that no misconfiguration could ever lead to VM traffic being sent on the uplinks dedicated to infrastructure traffic.

The following FIGURE 7-28 presents an example of a host with 4 uplinks and two virtual switches:

- A VDS is dedicated to infrastructure traffic.
- A second NSX prepared VDS or N-VDS handles the VM traffic.

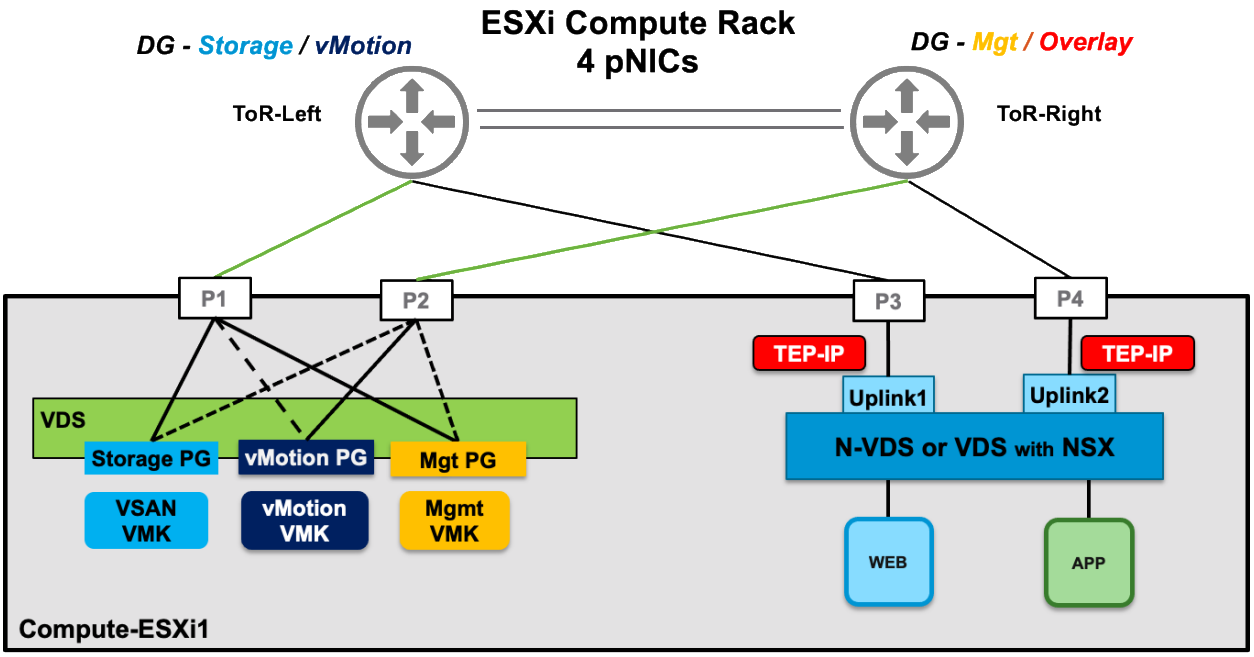


Figure 7-28: ESXi Compute Rack 4 pNICs – VDS and NSX virtual switch

The VDS is configured with pNICs "P1" and "P2". And each port group is configured with different pNICs in active/standby to use both pNICs. However, the choice of teaming mode on VDS is left to the user based on the considerations we outlined in the two pNIC design section. The virtual switch running NSX owns pNICs "P3" and "P4". To leverage both pNICs, the N-VDS is configured in load balance source teaming mode. Each type of host traffic has dedicated IP subnets and VLANs.

The model in FIGURE 7-28 still make sense when running NSX on N-VDS. However, now that we can deploy NSX directly on a VDS, we can easily achieve the same functionality with a single VDS running NSX and owning the four uplinks, as represented below.

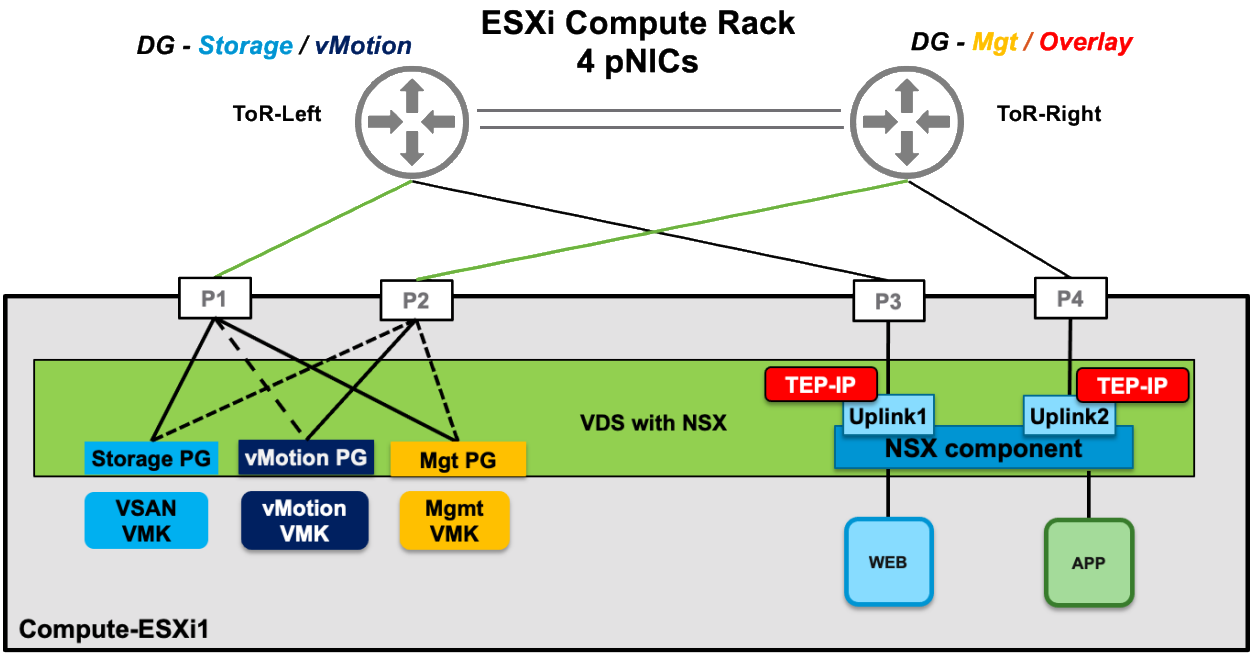


Figure 7-29: ESXi Compute Rack 4 pNICs – VDS and NSX virtual switch

You can achieve this configuration by simply mapping the four pNICs of the host to 4 uplinks on the VDS. Then, install NSX using an uplink profile that maps its uplinks (Uplink1/Uplink2 in the diagram) to the VDS uplinks P3 and P4. This model still benefits from the simple, non-disruptive installation of NSX. At the same time, the NSX component can only work with the two uplinks mapped to the uplink profile. This means that VM traffic can never flow over P1/P2.

The final added benefit of this model is that the administrator can manage a single virtual switch and has the flexibility of adding additional uplinks for overlay or infrastructure traffic.

7.4.3.2 Multiple virtual switches as a requirement

Specific scenarios still call for multiple virtual switches on a host. For example:

- Compliance-driven topologies, e.g., PCI, HIPPA, etc., often necessitate separate and dedicated infrastructure components (e.g., pNIC, operational controls, etc.)
- Some cloud provider models require internal or external-facing infrastructure on separate virtual switches.
- There is a specific use case for NFV (Network Function Virtualization) where two pNICs are dedicated to a standard virtual switch for overlay and the other two pNICs for an "enhanced datapath" virtual switch. The "enhanced mode" is not discussed here. Please refer to VMware NFV documentation.

In the following scenario, FIGURE 7-30 shows that each virtual switch is built to serve specific topology or provide traffic separation based on enterprise requirements. The virtual switches on a given host can be either VDS or N-VDS (note, however, that if both virtual switches are prepared for NSX, they can either

be both VDS with NSX or both N-VDS, not a mix of N-VDS and VDS with NSX.). The segments cannot extend between two virtual switches as each is tied to a different transport zone.

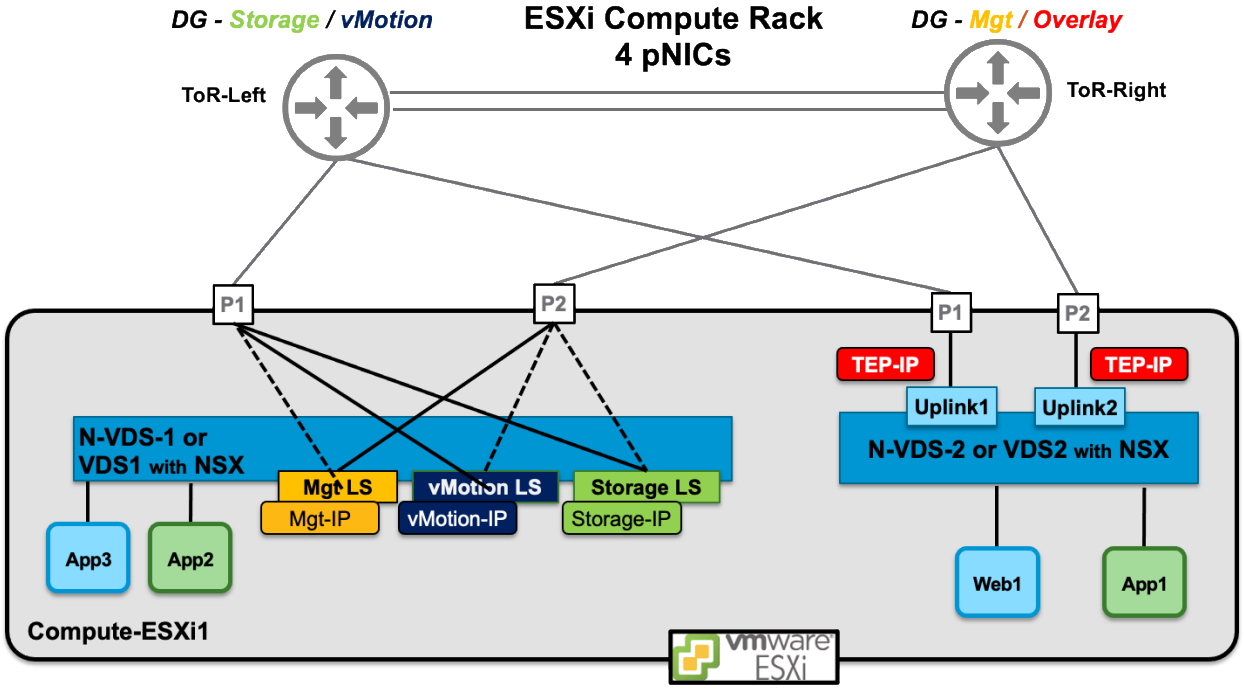


Figure 7-30: ESXi Compute Rack 4 pNICs- Two N-VDS (Or VDS with NSX)

Below, we list a series of use cases and configurations where the dual VDS design is relevant. In any of those cases, the infrastructure traffic will be carried on the first virtual switch. Here are some examples:

- The first two pNICs are exclusively used for infrastructure traffic, and the remaining two pNICs are for overlay VM traffic. This allows dedicated bandwidth for overlay application traffic. One can select the appropriate teaming mode as discussed in the above two pNICs design section (ESXi-Based Compute Hypervisor with two pNICs)
- The first two pNICs are dedicated to "VLAN only" micro-segmentation, and the second one is for overlay traffic.
- Building multiple overlays for separation of traffic. Starting with NSX 3.1, a host can have virtual switches part of different overlay transport zones and the TEPs on each virtual switch can be on different VLAN/IP subnets (still, all the TEPs for an individual switch must be part of the same subnet). When planning such configuration, it is important to remember that while hosts can be part of different overlay transport zones, edge nodes cannot. The implication is that when implementing multiple overlay transport zones, multiple edge clusters are required, one per overlay transport zones as a minimum.
- Building regulatory compliant domains with VLAN only or overlay.

- Building traditional DMZ type isolation.

The two virtual switches running NSX must attach to different transport zones. See detail in section SEGMENTS AND TRANSPORT ZONES

==========

Note: The second virtual switch could be of enhanced datapath type, for the NFV use case. This type is beyond the scope of this design guide.

==========

7.4.4 KVM-Based Compute Hypervisor with two pNICs

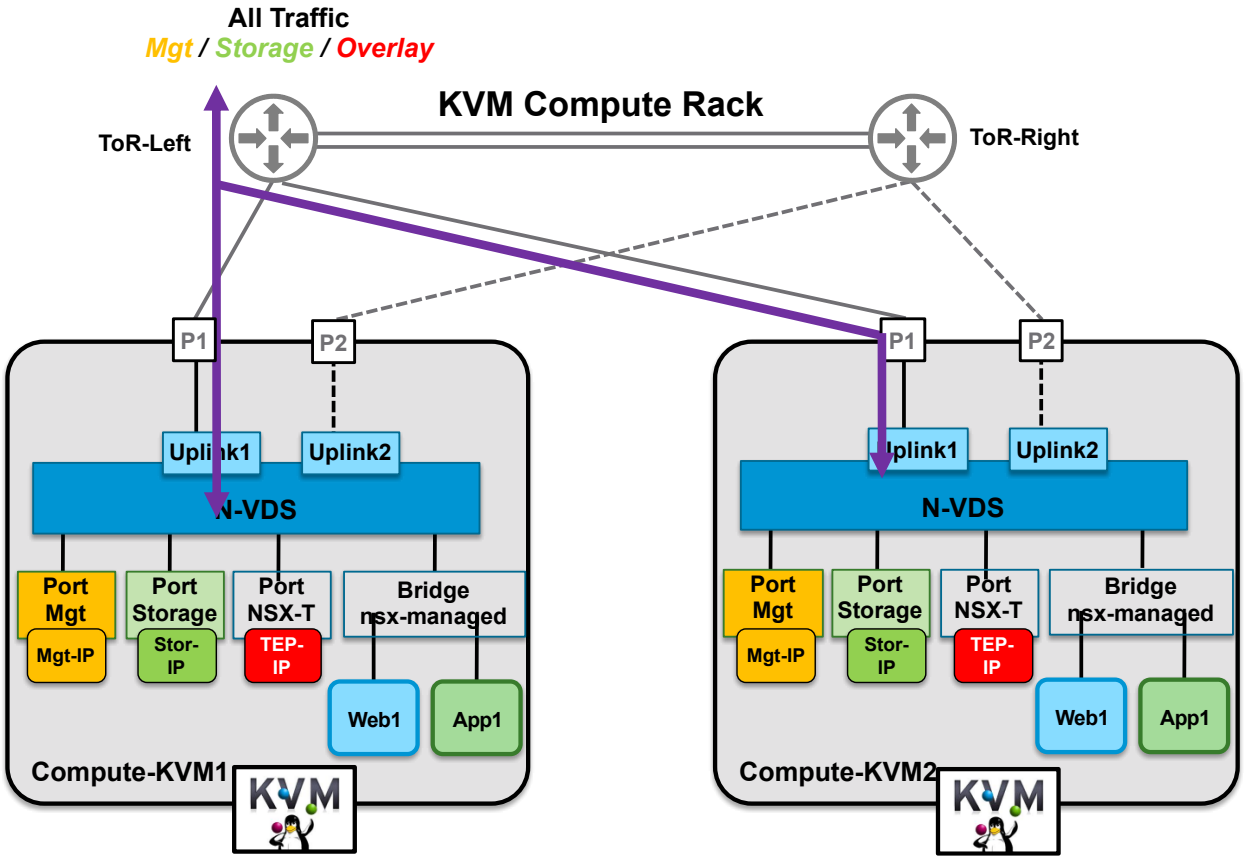


Figure 7-31: KVM Compute Rack Failover Teaming

In FIGURE 7-31 the design is very similar to ESXi Failover Order Teaming Mode.

A single host switch is used with a 2 pNICs design. This host switch manages all traffic – overlay, management, storage, etc. Physical NICs "P1" and "P2" are attached to different top of rack switches. The teaming option selected is failover order active/standby; "Uplink1" is active while "Uplink2" is standby. As shown logical switching section, host traffic is carried on the active uplink "Uplink1", while "Uplink2" is purely backup in the case of a port or switch failure. This teaming policy provides a deterministic and simple design for traffic management.

The top-of-rack switches are configured with a first hop redundancy protocol (e.g. HSRP, VRRP) providing an active default gateway for all the VLANs on "ToR-Left".

The VMs are attached to segments/logical switches defined on the N-VDS, with the default gateway set to the logical interface of the distributed Tier-1 logical router instance.

Note about N-VDS ports and bridge:

NSX host preparation of KVM creates automatically the N-VDS and its "Port NSX" (with TEP IP address) and "Bridge nsx-managed" (to plug the VMs). The other ports like "Port Mgt" and "Port Storage" have to be created outside of NSX preparation.

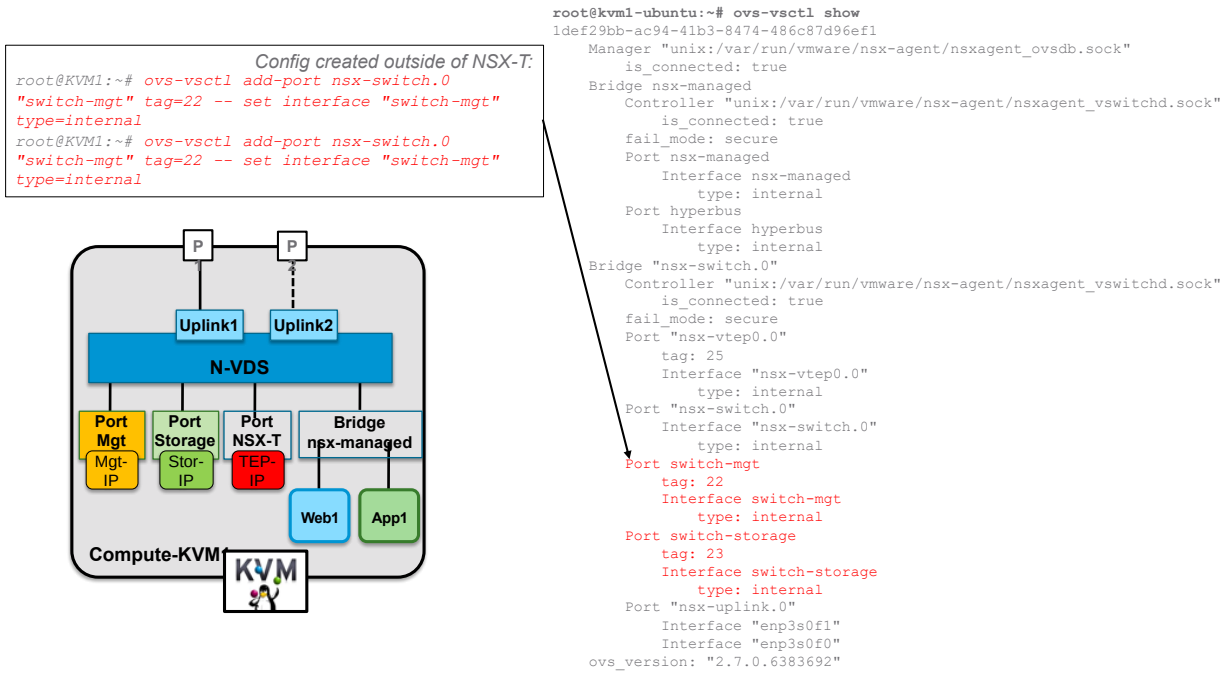
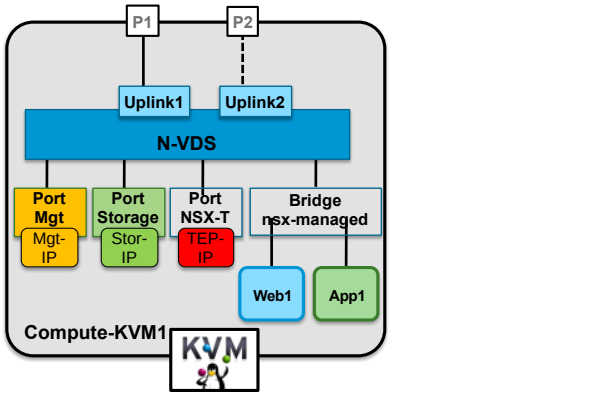


Figure 7-32: Creation of the N-VDS Mgt and Storage ports

And IP addresses for those ports have to be created outside of NSX preparation.




```
root@kvm1-ubuntu:~# vi /etc/network/interfaces
auto enp3s0f0
iface enp3s0f0 inet manual
    mtu 1700

auto enp3s0f1
iface enp3s0f1 inet manual
    mtu 1700

auto switch-mgt
iface switch-mgt inet static
        pre-up ip addr flush dev switch-mgt
        address 10.114.213.86
        netmask 255.255.255.240
        gateway 10.114.213.81
        dns-nameservers 10.113.165.131
        down ifconfig switch-mgt down
        up ifconfig switch-mgt up

auto switch-storage
iface switch-storage inet static
        pre-up ip addr flush dev switch-storage
        address 10.114.214.86
        netmask 255.255.255.240
        down ifconfig switch-storage down
        up ifconfig switch-storage up

auto nsx-vtep0.0
iface nsx-vtep0.0 inet static
    pre-up ip addr flush dev nsx-vtep0.0
    address 172.16.213.101
    netmask 255.255.255.240
    mtu 1700
    down ifconfig nsx-vtep0.0 down
    up ifconfig nsx-vtep0.0 up
    up route add -net x.x.x.x/x gw x.x.x.x dev nsx-vtep0.0
```

Figure 7-33: Creation Mgt and Storage IP addresses

7.5 Edge Node and Services Design

Edge nodes are available in two form factors – VM and bare metal server. While both form factors offer the same functionality, their physical infrastructure connectivity is quite different, however they have common requirements around the different types of IP networks that must be provided:

- Management – Accessing and controlling the Edge node
- Overlay (TEP) - Creating tunnels with peer transport nodes
- External (N-S) - Peering with the physical networking infrastructure to provide connectivity between the NSX virtual components and the external network

Edge nodes provide a pool of capacity for running centralized services in NSX. Edge nodes are always active in the context of connectivity and control plane. They host Tier-0 and Tier-1 routing services, installed in either active/active or active/standby mode. If a Tier-0 or Tier-1 router enables stateful services (e.g., NAT, Load Balancer, Gateway Firewall & VPN) it can only be deployed in active/standby mode. The status of active or standby mode is within the context of a gateway data plane forwarding rather than related to the Edge node itself. The Edge node connectivity options discussed in this section are independent of type of services enabled on any given node.

The design consideration for Edge node has changed since NSX 2.5 with two critical areas of enhancements

- Multi-TEP support for Edge – Just like an ESXi transport node supporting multiple TEP, Edge node has a capability to support multiple TEP, each associated to a different uplink providing the following advantages:
    - Removes topology restriction with bare metal – straight through LAG
    - Allowing the use of multiple pNICs for the overlay traffic in both bare metal and VM form factor.
    - An Edge VM supporting multiple TEP allows it to have two uplinks from the same N-VDS, allowing utilization of both pNICs. Efficient load sharing among host to Edge VM.
- Multiple teaming policy per N-VDS – Default and Named Teaming policies (TEAMING POLICYTEAMING POLICY)
    - Allows specific uplink to be designated or pinned for a given VLAN
    - Allowing uplinks to be active/standby or active-only to drive specific behavior of a given traffic types while co-existing with other traffic type following entirely different paths on the same N-VDS
- Normalization of N-VDS configuration – All the edge node form factors deployments can use a single N-VDS like a host hypervisor. Single teaming policy for overlay – Load Balanced Source. Multiple policies for N-S peering – Named teaming Policies

This section is divided into four major areas:

- Bridge use case
- Layer 3 peering use case
- Stateful Services use case
- Gateway Firewall for VLAN networks use case
- Edge services and resources design considerations

7.5.1 Bridging Use Case

The chapter 3 section BRIDGING OVERLAY TO VLAN WITH THE EDGE BRIDGE covers the basic functionality and availability model requirements. The next section covers the bridging design guidelines.

The Edge Bridge is available on both Edge form factors - bare metal or Virtual Machine (VM). The use of the Bridge in the bare metal form factor is relatively straightforward: the bridged traffic is sent on the uplinks of the N-VDS selected by VLAN transport zone specified on the Bridge Profile. There is no Bridge-specific configuration necessary on the physical infrastructure where the bare metal Edge attaches.

7.5.1.1 Bridging Design Considerations for VM Edge

7.5.1.1.1 Edge VM vNIC Configuration Requirement with Bridging

For the VM form factor, it is important to remember that the Edge Bridge will end up sourcing traffic from several different mac addresses on its VLAN vNIC. This means, that the uplink vNIC must be connected to a DVPG port group allowing:

- Forged transmit
- Mac learning or promiscuous mode

Both above capabilities are not supported on VSS while supported on VDS. This means it's a strong recommendation is to use VDS when deploying Edge node. Mac learning is available on the VDS as of vSphere 6.7. However, there is no GUI capability on vCenter to configure mac-learning as of this writing but it can be enabled via API or using powerCLI (See https://www.virtuallyghetto.com/2018/04/native-mac-learning-in-vsphere-6-7-removes-the-need-for-promiscuous-mode-for-nested-esxi.html.)

If deployment is running vSphere 6.5 where mac learning is not available, the only other way to run bridging is by enabling promiscuous mode. Typically, promiscuous mode should not be enabled system wide. Thus, either enable promiscuous mode just for DVPG associated with bridge vNIC or it may be worth considering dedicating an Edge VM for the bridged traffic so that other kinds of traffic to/from the Edge do not suffer from the performance impact related to promiscuous mode.

7.5.1.1.2 Edge VM: Virtual Guest Tagging

The Edge Bridge will be sending bridged traffic with an 802.1Q tag on its VLAN uplink. That means that this Edge VM vNIC will have to be attached to a port

group configured for Virtual Guest Tagging (VGT, i.e. the DVPG shows as VLAN Trunk in the vCenter UI.) Refer VLAN TAG Requirements for more information.

7.5.1.1.3 Edge VM dedicated to Bridging example

The following FIGURE 7-34 represents an Edge VM dedicated to bridging and following the rules enunciated earlier in this section. The Edge VM has four vNICs, but this design only uses 3:

- vNIC1 is dedicated to management traffic
- vNIC2 is the uplink of N-VDS1, the N-VDS that will be used for overlay traffic. On the ESXi host, it is attached to a Transport DVPG using active/standby both pNICs of the host for redundancy.
- vNIC3 is the uplink of N-VDS2 that is attached to the VLAN transport zone "N-VDS2" where the bridged traffic will be sent. On the ESXi host, the "Bridge VLAN" DVPG has the following configuration:
    - Virtual Guest Tagging is enabled so that it is possible to bridge to several segments to different VLAN IDs
    - Forged transmit and mac learning, so that the bridge can send traffic sourced from different mac addresses. If mac learning is not possible, the promiscuous can be configured instead at the expense of degraded performance.
    - Active standby teaming policy leveraging the same pNICs (but not necessarily in the same order) as the overlay DVPG. That last point is important and will be justified in the next part.

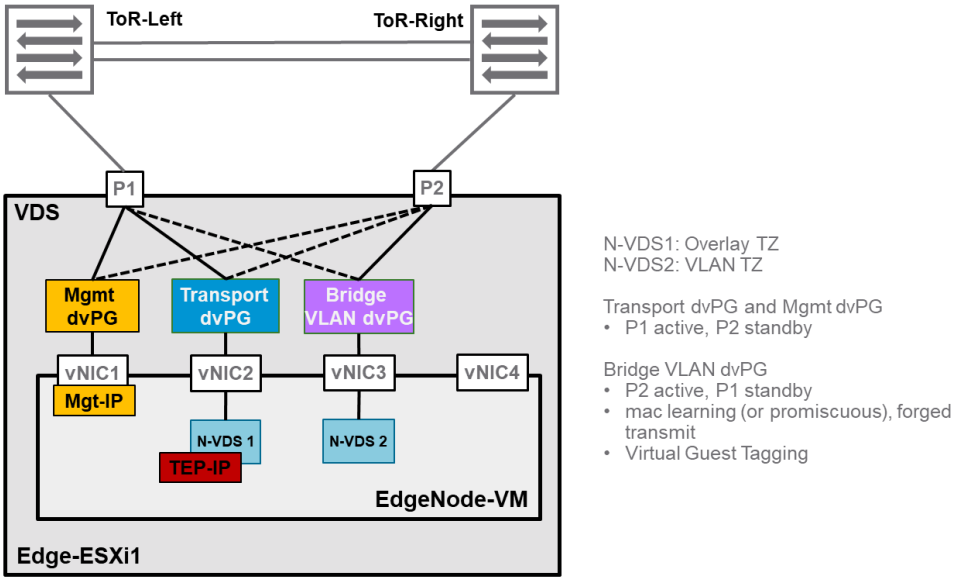


Figure 7-34: Dedicated Edge VM for Bridging

7.5.1.1.4 Edge VM: Edge uplink protection

As we have seen, the Edge Bridge sends/receives two kinds of traffic on its uplinks: overlay traffic and VLAN traffic. This part discusses how to protect both against failure in the data path on the host.

The Edge HA mechanism is exchanging BFD hellos over the tunnels between the different Edges in the Edge Cluster. As a result, overlay traffic is protected against failure in the data path. In FIGURE 7-34 above, if both P1 and P2 went down on the host, all the tunnels between this Edge VM and its peers would go down. As a result, this Edge VM would be considered as failed by Edge HA and another Edge would take over the services it was running (including, but not limited to, the bridge service.)

Let us now focus on the VLAN uplink redundancy. There is no keepalive mechanism protecting the VLAN traffic and the edge VM does not detect when the physical uplinks are going down. The "Bridge VLAN" dvPortGroup has a teaming policy that will protects against a single physical uplink of the host going down. Should both physical uplinks go down on the host, then overlay traffic would be affected, the edge itself would be considered as down and the standby edge would takeover. In a way, the VLAN traffic benefits from the keepalive mechanism run on the overlay. For this to happen, it is important to ensure that the VLAN traffic can use all the physical uplinks where overlay traffic is forwarded.

7.5.1.2 Bridging Design Considerations for both VM and Bare Metal

7.5.1.2.1 Redundant VLAN connectivity

The Edge Bridge HA mechanism does not protect against connectivity problem in the VLAN infrastructure beyond the Edge physical uplink.

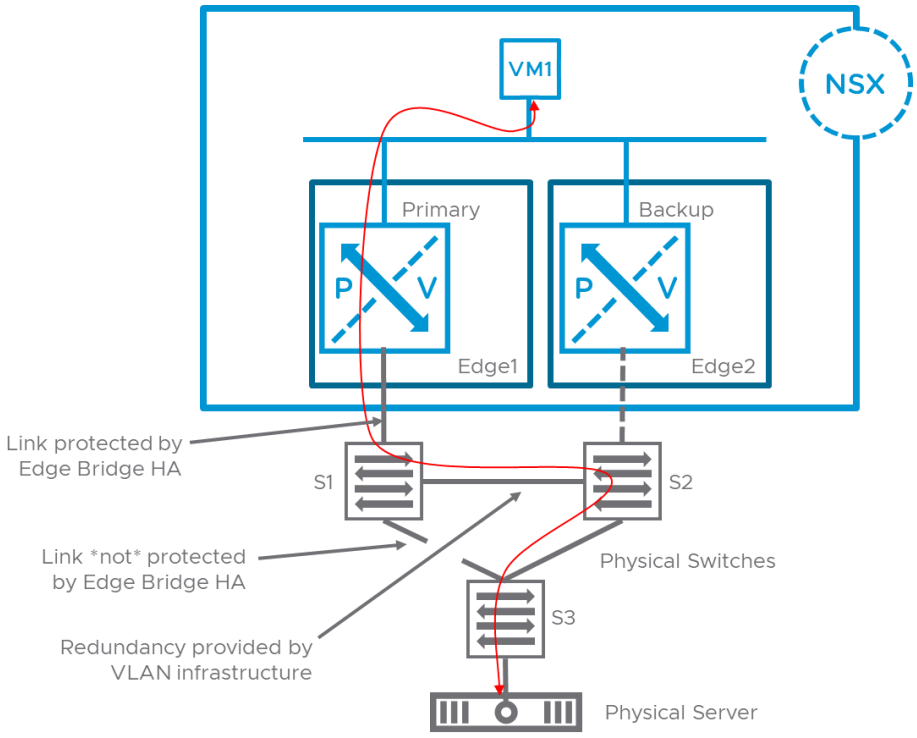


Figure 7-35: Physical bridging infrastructure must be redundant

In the above scenario, the failure of the uplink of Edge 1 to physical switch S1 would trigger an Edge Bridge convergence where the Bridge on Edge 2 would become active. However, the failure of the path between physical switches S1 and S3 (as represented in the diagram) would have no impact on the Edge Bridge HA and would have to be recovered in the VLAN L2 domain itself. Here, we need to make sure that the alternate path S1-S2-S3 would become active thanks to some L2 control protocol in the bridged physical infrastructure.

7.5.1.2.2 Preemptive vs. non-preemptive

The choice is between precise bandwidth allocation on the uplinks and minimum disruption. It also allows one to decide the placement of the bridge for capacity control of the either host or Edge Node.

The preemptive model allows making sure that, when the system is fully operational, a Bridge is using a specified uplink for its VLAN traffic. This is required for scaling out the solution, precisely distributing the load across several Edge Bridges and getting more aggregate bandwidth between virtual and physical by doing Segment/VLAN load balancing.

The non-preemptive model maximizes availability. If the primary fails then recovers, it will not trigger a re-convergence that could lead to unnecessary packet loss (by preempting the currently active backup.) The drawback is that, after a recovered failure, the bridged traffic remains polarized on one Edge, even if there was several Bridge Profiles defined on this pair of Edges for Segment/VLAN load balancing. Also note that, up to NSX release 2.5, a failover cannot be triggered by user intervention. As a result, with this design, one cannot assume that both Edges will be leveraged for bridged traffic, even when they are both available and several Bridge Profiles are used for Segment/VLAN load

balancing. This is perfectly acceptable if availability is more important than available aggregate bandwidth.

7.5.1.2.3 Performance: scaling up vs. scaling out

The performance of the Edge Bridge directly depends on the Edge running it. NSX thus offers the option to scale up the Edge Bridge from a small form factor Edge VM running several other centralized services to a powerful bare metal Edge node dedicated to bridging.

Scaling out is also possible, as a complement to or instead of scaling up. By creating two separate Bridge Profiles, alternating active and backup Edge in the configuration, the user can easily make sure that two Edge nodes simultaneously bridge traffic between overlay and VLAN. The diagram below shows two Edges with two pairs (numbered 1 and 2) of redundant Edge Bridges. The configuration defines the Primary 1 on Edge 1 and Primary 2 on Edge 2. With preemption configured, this ensures that when both Edges are available, both are active for bridging traffic.

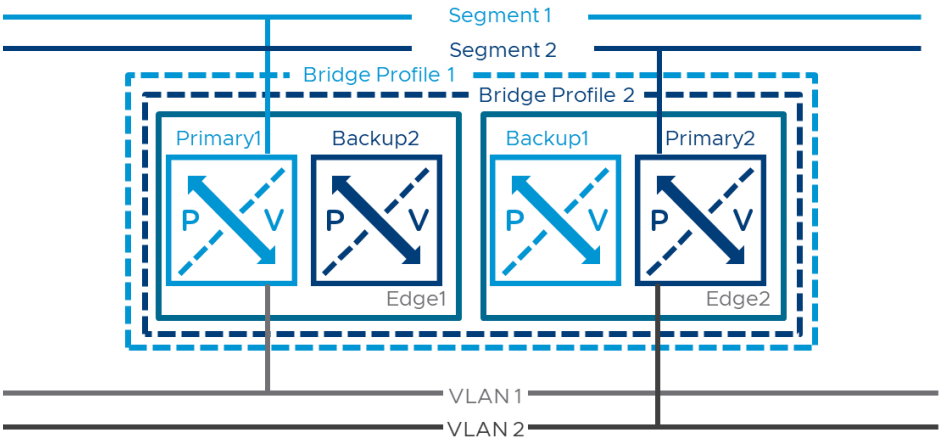


Figure 7-36: Load-balancing bridged traffic for two Logical Switches over two Edges (Edge Cluster omitted for clarity.)

Further scale out can be achieved with more Edge nodes. The following diagram shows an example of three Edge Nodes active at the same time for three different Logical Switches.

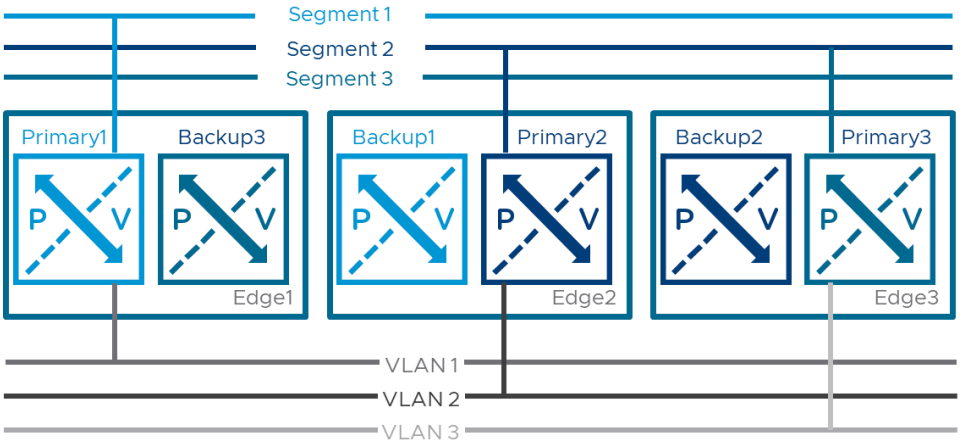


Figure 7-32: Load-balancing example across three Edge nodes (Bridge Profiles not shown for clarity.)

Note that if several Bridge Profiles can be configured to involve several Edge nodes in the bridging activity, a given Bridge Profile cannot specify more than two Edge nodes.

7.5.1.3 Recommended bridging designs

The following sections present the recommended design for the edge bridge use case for both the edge node VM and bare metal form factor in the two or four pNICs configurations. The designs have been selected based on the following characteristics:

- Availability: the designs protect against the failure of a pNIC, edge node, ESXi host, or ToR switch
- Performance:
  - traffic is distributed across all the available paths to maximize throughput
  - the design can be scaled up by adding host resources (CPU, pNICs) or scaled out via additional nodes
- Manageability:
  - Configurations are agnostic to the physical network and do not require any specific configuration of the ToR switches (e.g., LAG).
  - The Layer 3 peering use case is compatible with the presented designs
  - Avoid server cabling specific to the bridge use case.

7.5.1.3.1 4 pNICs server Bare Metal Form Factor

A 4 pNIC design is preferred over a two pNIC design to maximize the server resource utilization and avoid the physical NICs representing the limiting performance factor. The bridging design for a four pNICs server in a bare metal form factor follows the following guidelines:

- Overlay traffic is load-balanced across two physical links leveraging multi-tep edge capability. Multi-tep provides load balancing and better edge node resource utilization compared to a single TEP.
- Because for each bridging instance, VLAN traffic can be forwarded over a single uplink at the time, load balancing is achieved by pinning different VLANs to different pNICs.
- VLAN and overlay traffic are forwarded over different pNICs. While sharing the pNICs between VLAN and overlay traffic may lead to better performances and high availability (e.g., configuring 4 TEPs and connecting the edge to 4 ToRs), we chose a simpler design with a deterministic traffic pattern.

The design presented in FIGURE 7-38 below includes a single NVDS per edge node. The NVDS manages both the overlay and VLAN traffic. The diagram presents two bridging instances, for VLAN and overlay X, and VLAN and overlay Y, but more can be configured. The VLAN traffic is pinned to pNIC 2 or pNIC 3. Additional VLANs could be pinned to either interface. In this case, the pinning is achieved via a bridging teaming policy, implying that any bridging instance must be associated with a named teaming policy. Overlay traffic for any bridging instance is load-balanced across pNIC0 and pNIC1 via the default teaming policy.

We should pay special attention to the cabling and VLAN to pNIC association, which is not the same for the two bare metal edge nodes. The reason for this asymmetry is that while overlay traffic can failover between the two pNICs on the same edge node, the same is not true for VLAN traffic, which is pinned to a single uplink because of the limitation discussed in chapter 3. If that single pNIC or the ToR where it is connected fails, the bridge instance should fail over to the second edge, which must have a viable path for the associated VLANs. We want to maintain a standard cabling configuration, where two pNICs are connected to one ToR and two to the other consistently while at the same time allowing for the recovery of the traffic after the failure of a ToR. For this reason, we crossed the mapping between pNIC2 and pNIC3 and uplink2 and uplink3 on the two bare metal edges.

FIGURE 7-37 shows a sample configuration. If this configuration is not desirable, connecting pNIC2 to ToR2 and pNIC3 to ToR1 for bare metal edge-2 will have the same effect.

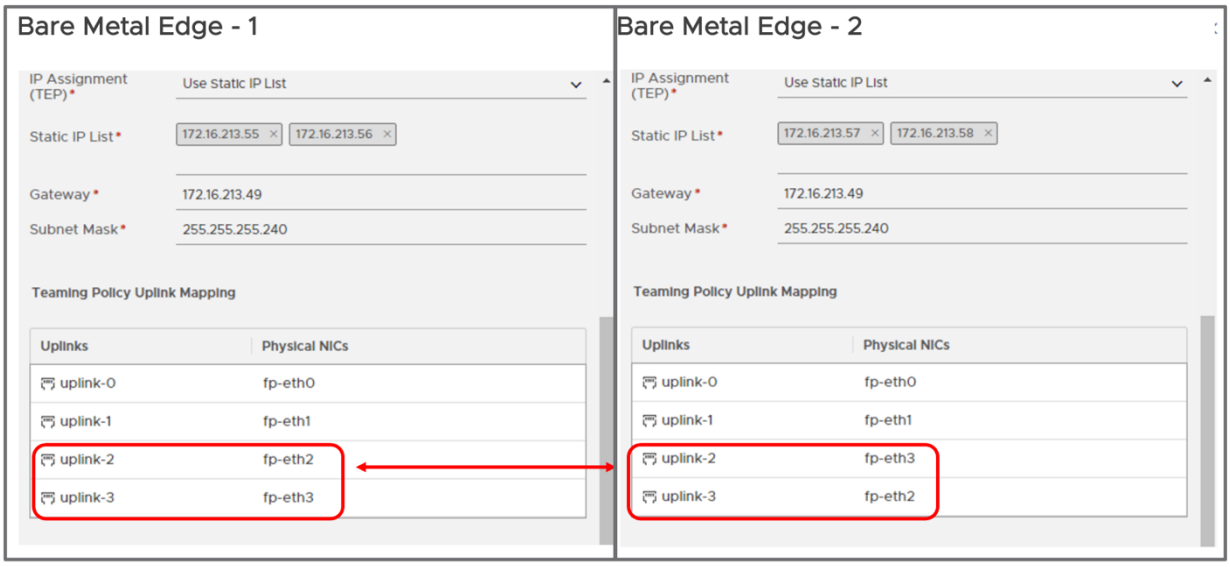


Figure 7-37: Bridge Design – 4 pNIC BM edge – Single NVDS – uplink to pNIC mapping

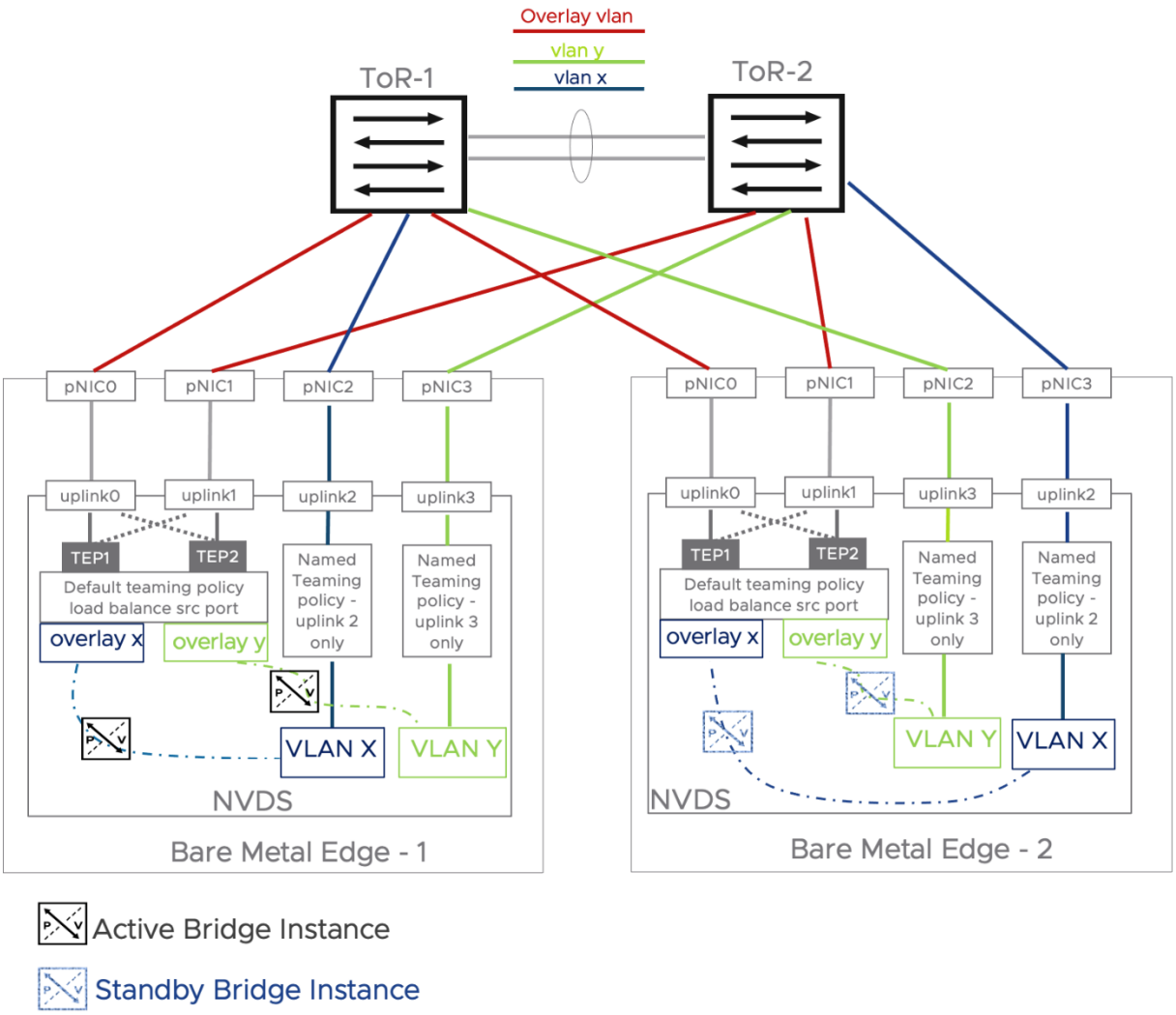


Figure 7-38: Bridge Design – 4 pNIC BM edge – Single NVDS

We can achieve the same design without named teaming policies and deploying multiple NVDSs on the same bare metal edge. The functionalities provided are the same, and the decision between the two options may be based on the preferred configuration workflows for new bridging instances. This configuration option is presented in the diagram below.

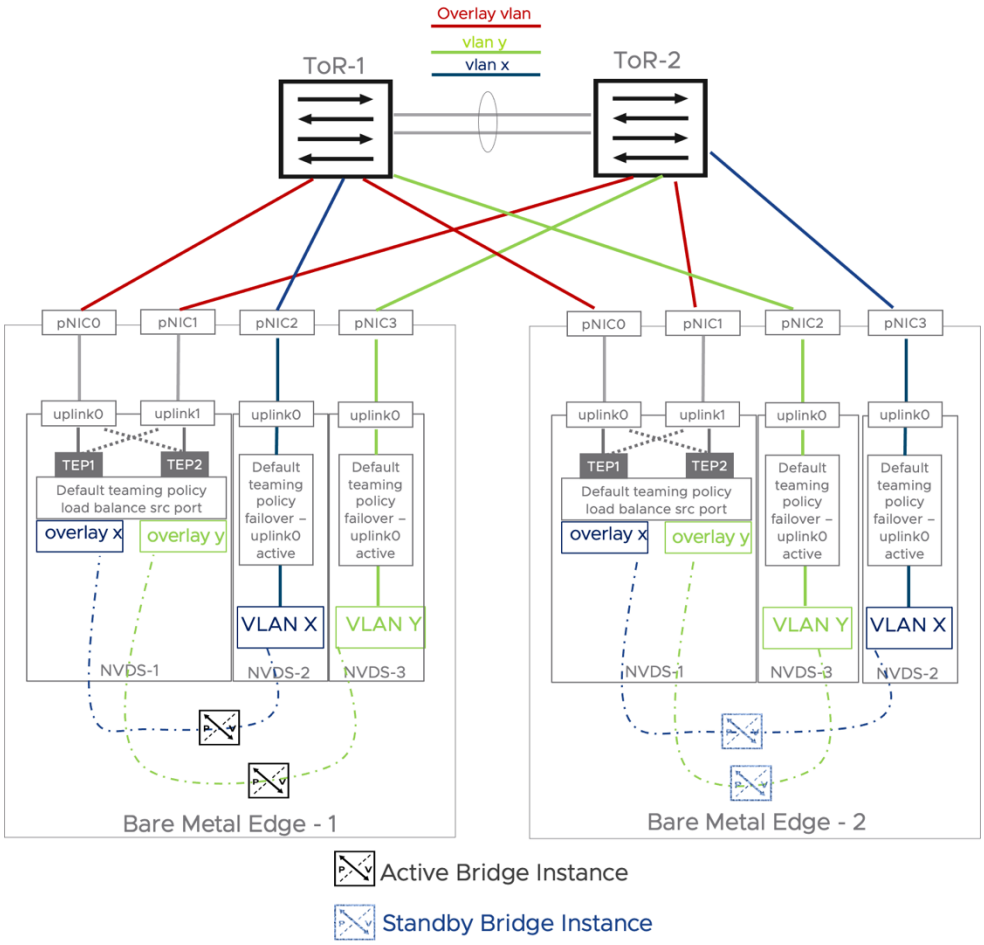


Figure 7-39: Bridge Design – 4 pNIC BM edge – 3 NVDSs

So far, we have shown how to achieve optimal high availability, CPU, and link utilization across a single bare metal edge. The previous diagrams show Bare Metal Edge 2 running standby bridging instances only. It is possible to spread the active bridging instances across multiple bare metal edges via bridge profiles. Because the traffic for each bridging instance is always handled by a single edge node at the time, and VLAN traffic is pinned to a single pNIC, at least four bridging instances are required to utilize all available links. This configuration is depicted in FIGURE 7-40.

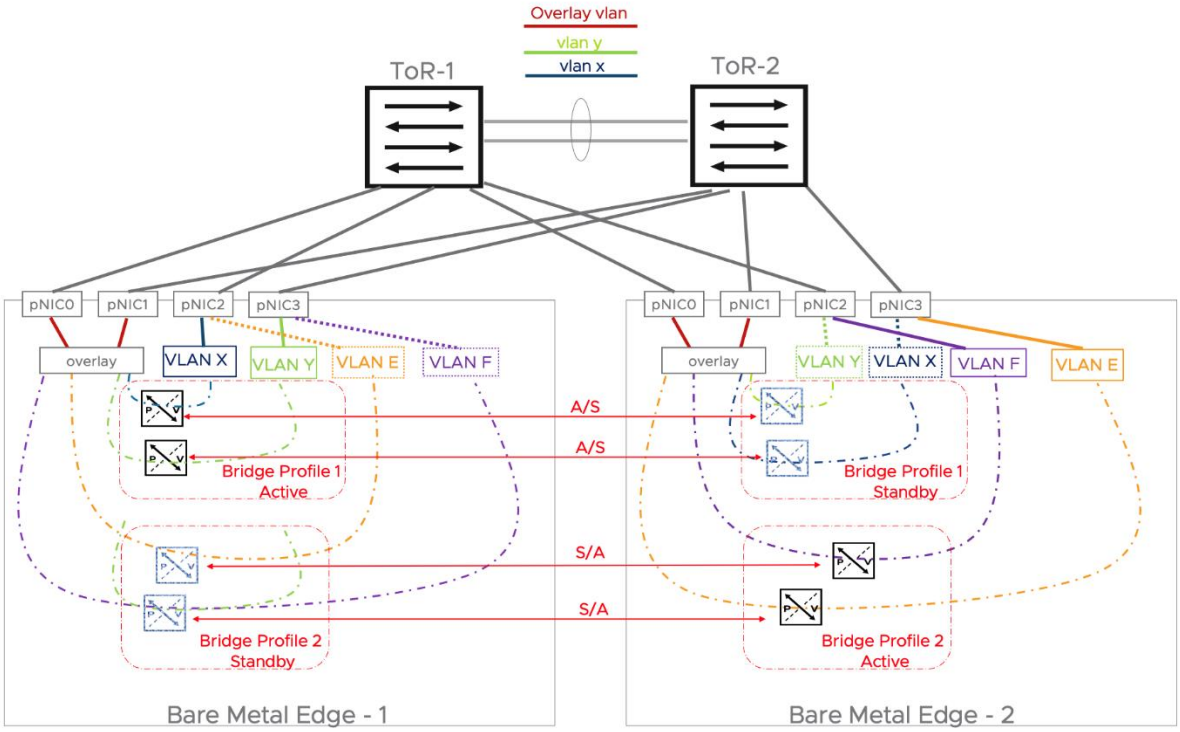


Figure 7-40: Bridge Design – 4 pNIC BM edge – Bridge profiles

7.5.1.3.2 2 pNICs server Bare Metal Form Factor

The bridging design for a 2 pNICs server follows the same principles of the 4 pNIC server design. Specific considerations for this design are:

- The number of pNICs may represent the limiting factor from a performance perspective.
- Crossing the uplink to pNIC mapping or the cabling configuration must be performed to avoid that VLAN traffic is interrupted because of a single ToR failure.
- Because the servers have 2 pNICs only, a singe NVDS must be deployed and named teaming policies are require for the VLAN traffic.

Both the overlay and the VLAN traffic flow over the same set of pNICs in this case.

FIGURE 7-41 depicts the recommended bridging design for a 2 pNIC bare metal edge.

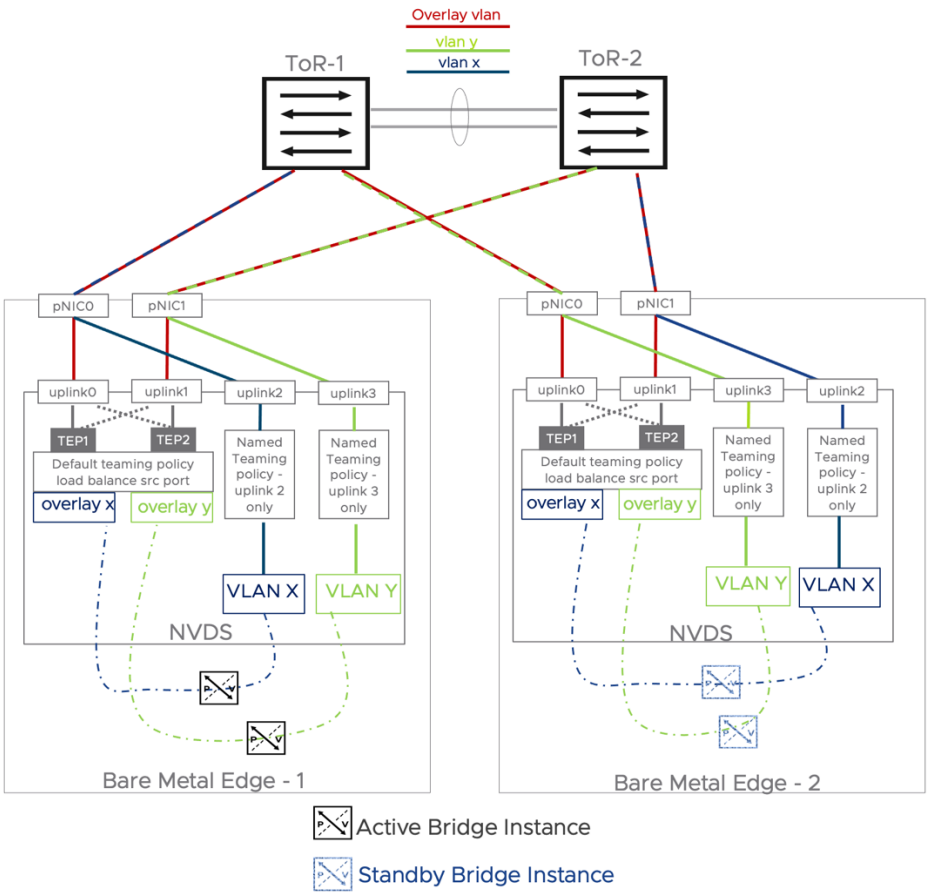


Figure 7-41: Bridge Design – 2 pNIC BM edge

7.5.1.3.3 2 pNICs server VM Form Factor

The bridging design for a 2 pNICs server running edge nodes in a VM form factor follows the following guidelines:

- Overlay traffic is load-balanced across two physical links leveraging multi-tep edge capability. Multi-tep provides load balancing and better edge node resource utilization than a single TEP.
- VLAN traffic is load-balanced across the same two physical links leveraging the VDS load balancing based on MAC teaming capability.

From a configuration perspective, overlay traffic is load-balanced over vNIC1 and vNIC2 on the edge node VM by the default teaming policy set to load-balance based on the source port. vNIC1 and vNIC2 are connected to different VDS dvpg configured with a mirrored active/standby teaming policy. These dvpgs may or may not be configured as trunk as they carry a single VLAN for overlay traffic (this is different than in the L3 peering design where the dvpg must carry both overlay traffic and the routing peering traffic to the physical network). The bridge VLAN traffic is carried by the edge VM vNIC3 only. VLAN traffic presents different source MAC addresses (the MACs of all the VMs on the overlay) that can be used by the upstream VDS for the load-balancing hash.

MAC learning and forged transmit must be enabled on the bridge dvpg carrying VLAN traffic. It is not required to configure MAC learning on the Overlay1 and Overlay2 dvpgs as the failures of vNIC1 or vNIC2 are not realistic occurrences and should not be part of a test plan.

This configuration can be achieved with a single N-VDS and a named teaming policy as depicted in FIGURE 7-42.

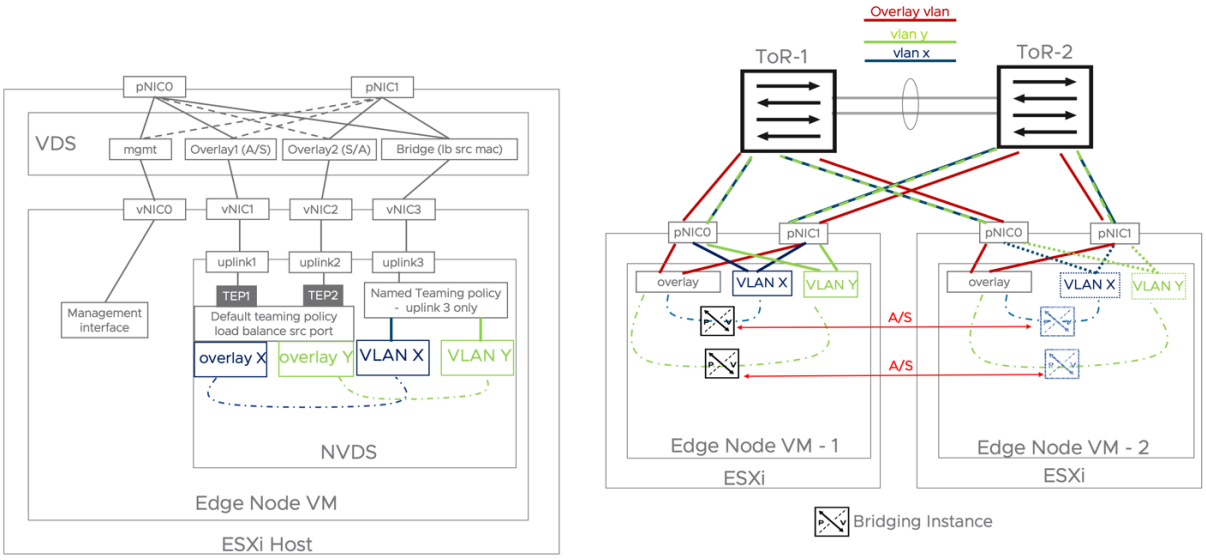


Figure 7-42: Bridge Design – 2 pNIC VM Edge – 1 NVDS

It can also be achieved without named teaming policies and two NVDSs per edge VM. The resulting designs are equivalent, they only differ in the configuration workflow. The 2 NVDS design may be preferable to avoid the need to associate a teaming policy to each bridging instance. Doing so may lower the probability of a misconfiguration.

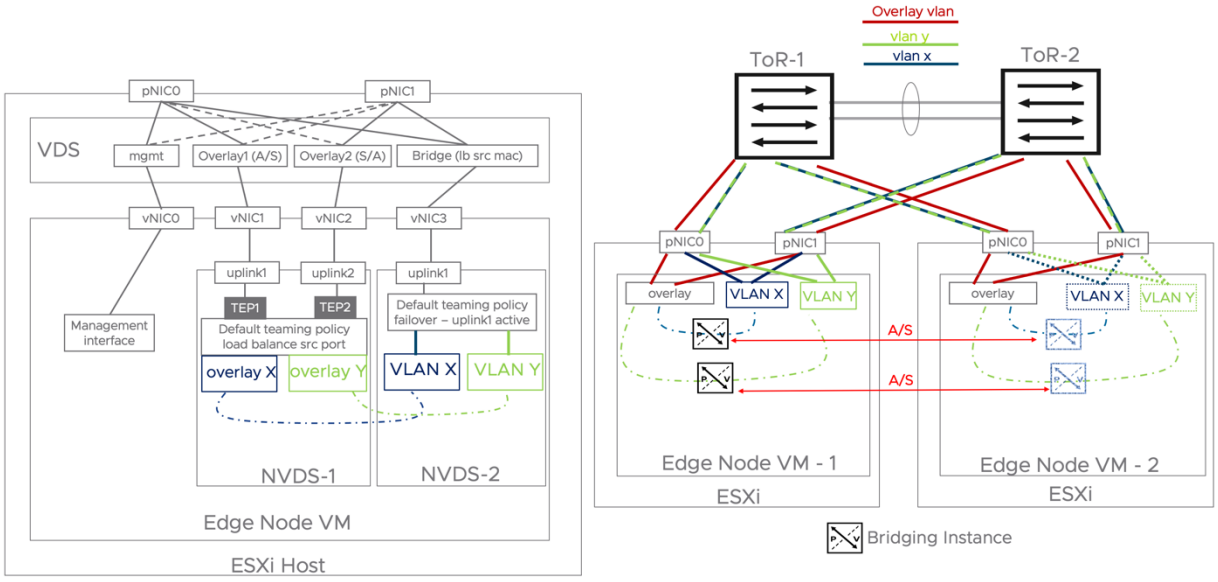


Figure 7-43: Bridge Design – 2 pNIC VM Edge – 2 NVDSs

7.5.1.3.4 4 pNICs VM Form Factor

A 4 pNICs design is preferred over a 2 pNICs design to maximize the server resource utilization and avoiding that the physical NICs represents the limiting performance factor.

When the edge node is in an edge VM form factor, a single VM is not likely to fully utilize the resource of the underlying hardware. Two edge node VMs on a 4 pNIC physical host (4 x 25G recommended) generally represent a more optimal configuration.

The 4 pNIC design is an extension of the two pNIC design for the edge VM for factor. The four pNICs are managed by a single VDS that provides connectivity to both edge node VM. Wiring and teaming policies configuration are the same as in the 2 pNIC model, the design is fundamentally replicated for the second edge VM and the two additional pNICs. The diagram below presents the recommended configuration. The two edge node VMs do not share the same dvpg for the management interfaces. This choice enforce fate sharing between the management and overlay network of the two edge VMs (i.e. if pNIC 2 and pNIC 3 go down the two edges are completely isolated for both overlay and management). While not required, this configuration may provide additional protection in case of specific failure scenarios in certain topologies (Refer to the Edge HA section in Chapter 4).

The design proposed in this section requires the careful planning and configuration of bridging profiles so that the active bridging instances are distributed across different edge VMs. Specifically:

- Active/Standby instances for the same bridge profiles should not be associated to edge VMs running on the same host.
- A deployment incorporating two hosts with 4 edge VMs (2 per host) requires two bridge profiles for an active/standby configuration (1 host active, 1 host standby), and 4 bridge profiles for an active/active configuration (both hosts active).

Note: nor hosts or edge VMs are active or standby, it's the bridging instances running on top of the edge VMs that are. Designating hosts as active or standby represents a design abstraction helpful when building a logical model hiding implementation complexities.

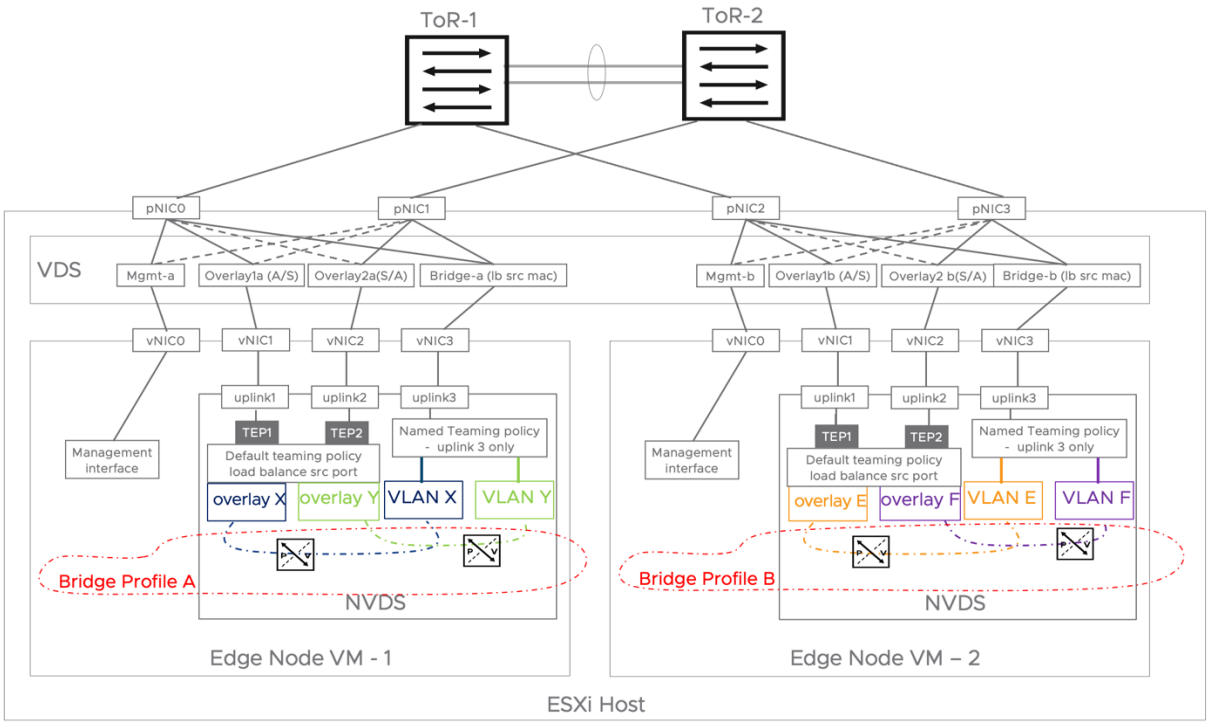


Figure 7-44: Bridge Design – 4 pNIC VM Edge – 1 NVDS

FIGURE 7-44 presents a 4 pNICs ESXi server hosting two edge VMs dedicated to bridging. The two edge VMs are associated with different bridge profiles so that they will never run an active/standby pair for any VLAN-Overlay pair. The failure of the host should not cause both the active and standby instance to fail at the same time. Edge VMs running on a different host provide high availability for the bridging instances depicted in the diagram. vSphere DRS rules should be implemented to achieve the desired design and placement.

7.5.2 Edge Connectivity Guidelines for Layer 3 Peering Use Case

The goal of the layer 3 N-S connectivity design is simple: deterministic and redundant configuration without incurring any dependencies on Spanning-tree related configuration. This means the peering VLAN is confined to a specific ToR and do not span across ToR. This topology choice also allows direct mapping from Edge node uplinks to physical NIC of the devices (bare metal or ESXi host) and eventually to ToR interface. This creates 1:1 mapping of physical connectivity and logical peering connectivity from Edge node to physical router, resulting into a simple, operationally deterministic connectivity of N-S traffic forwarding and troubleshooting. The operator always knows the which peering is impacted during the failure of a pNIC, ToR or Edge-Uplink.

In the typical enterprise design, Edge nodes in FIGURE 7-45 are assigned to a Tier-0 router. This Tier-0 router peers with the physical infrastructure using eBGP. Two adjacencies per Edge node with two logical switch connects to distinct "External-VLAN" per ToR. FIGURE 7-45 represents the logical view of the BGP peering. FIGURE 7-45 represents the logical view of the BGP peering.

One common misconception prevails that separation of TEP and BGP on two different switches somehow increases the resiliency or availability of the system. That is not the case at all. In fact, when both TEP and BGP are configured on the same virtual switch, enables consistent failover of both TEP and BGP at the same time. The dual VTEP and BGP combination allows consistent configuration either with Edge VM or bare metal Edge. However, there are corner use cases where asymmetric bandwidth ration of E-W vs N-S exist in that case specific optimization of design may be appropriate.

From the perspective of the physical networking devices, the Tier-0 router looks like a single logical router; logically, the adjacency to "Router1" is hosted on Edge node "EN1", while "EN2" is implementing the peering to "Router2".

Those adjacencies are protected with BFD, allowing for quick failover should a router or an uplink fail. See the specific Graceful Restart and BFD Interaction with Active/Standby interaction based on type of services – ECMP or stateful services – enabled in the Edge node and type of physical routers supported.

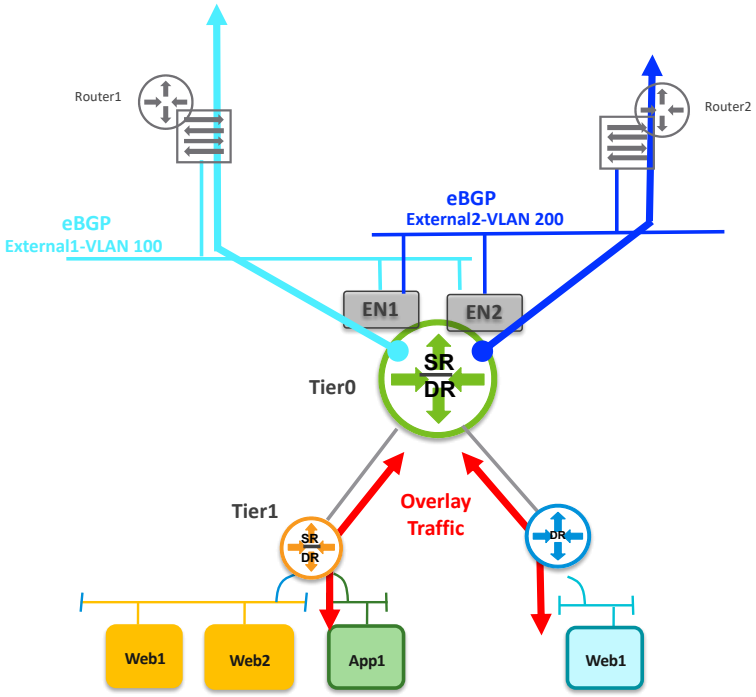


Figure 7-45: Typical Enterprise Bare metal Edge Note Logical View with Overlay/External Traffic

7.5.2.1 Bare Metal Edge Design

The bare metal Edge node is a dedicated physical server that runs a special version of the NSX Edge software. The bare metal Edge node requires a NIC supporting DPDK. VMware maintains a LIST OF THE COMPATIBILITY WITH VARIOUS VENDOR NICS.

This design guide covers commonly deployed bare metal configuration.

7.5.2.1.1 Bare metal Design with 2 Datapath pNICs

FIGURE 7-46 shows two pNIC bare metal Edges using a single N-VDS design for the data plane.The left side of the diagram shows the bare metal Edge with four physical NICs where management traffic has two dedicated physical NICs (P1 & P2) configured in active/standby mode.

This topology uses a single N-VDS "Overlay and External N-VDS" that carries both overlay and External traffic. Overlay traffic to and from different overlay segments/logical switches gets pinned to TEP IP1 or TEP IP2 and gets load balanced across uplinks, Uplink1, and Uplink2. Notice that both TEP IPs use the same transport VLAN, i.e., VLAN 200, configured on both top of rack switches.

Two VLANs segments, i.e., "External VLAN Segment 300" and "External VLAN Segment 400," provide northbound connectivity to Tier-0 gateway. External traffic from these VLAN segments is load-balanced across uplinks using a named teaming policy, which pins a VLAN segment to a specific uplink.

We can also use the same VLAN segment to connect Tier-0 Gateway to TOR-Left and TOR-Right. However, it is not recommended because of inter-rack VLAN dependencies leading to spanning tree-related convergence and the inability to load-balance the traffic across different pNICs.

This topology provides redundancy for management, overlay, and external traffic, in the event of a pNIC failure on Edge node/TOR and a complete TOR Failure.

The right side of the diagram shows two pNICs bare metal edge configured with the same N-VDS "Overlay and External N-VDS" for carrying overlay and external traffic as in the example above, but that it is also leveraging in-band management.

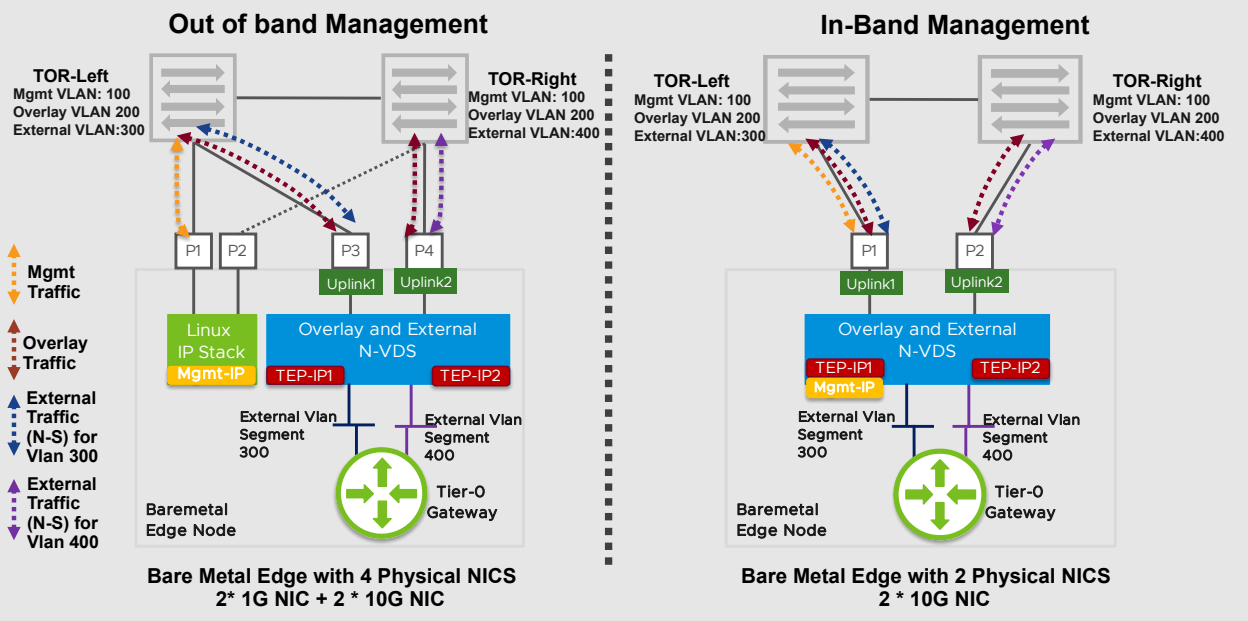


Figure 7-46: Bare metal Edge configured for Multi-TEP - Single N-VDS for overlay and external traffic

Both topologies use the same transport node profile, as shown in FIGURE 7-47. This configuration shows a default teaming policy that uses both Uplink1 and Uplink2. This default policy is used for all the overlay segments/logical switches created on this N-VDS.

Two additional teaming policies, "Vlan300-Policy" and "Vlan400-Policy," have been defined to override the default teaming policy and send traffic to "Uplink1" and "Uplink2" only, respectively.

"External VLAN segment 300" is configured to use the named teaming policy "Vlan300-Policy" that sends traffic from this VLAN only on "Uplink1". "External VLAN segment 400" is configured to use a named teaming policy "Vlan400-Policy" that sends traffic from this VLAN only on "Uplink2".

Based on these teaming policies, TOR-Left will receive traffic for VLAN 100 (Mgmt.), VLAN 200 (overlay) and VLAN 300 (Traffic from VLAN segment 300). Similarly, TOR-Right will receive traffic for VLAN 100 (Mgmt.), VLAN 200 (overlay) and VLAN 400 (Traffic from VLAN segment 400). A sample configuration screenshot is shown below.

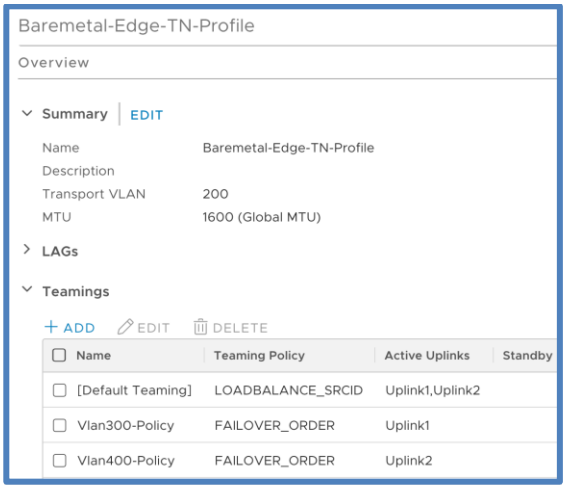


Figure 7-47: Bare metal Edge Transport Node Profile

FIGURE 7-48 shows a logical and physical topology where a Tier-0 gateway has four external interfaces. External interfaces 1 and 2 are provided by bare metal Edge node "EN1", whereas External interfaces 3 and 4 are provided by bare metal Edge node "EN2". Both the Edge nodes are in the same rack and connect to the TOR switches in that rack. Both the Edge nodes are configured for Multi-TEP and use named teaming policy to send traffic from VLAN 300 to TOR-Left and traffic from VLAN 400 to TOR-Right. Tier-0 Gateway establishes BGP peering on all four external interfaces and provides 4-way ECMP.

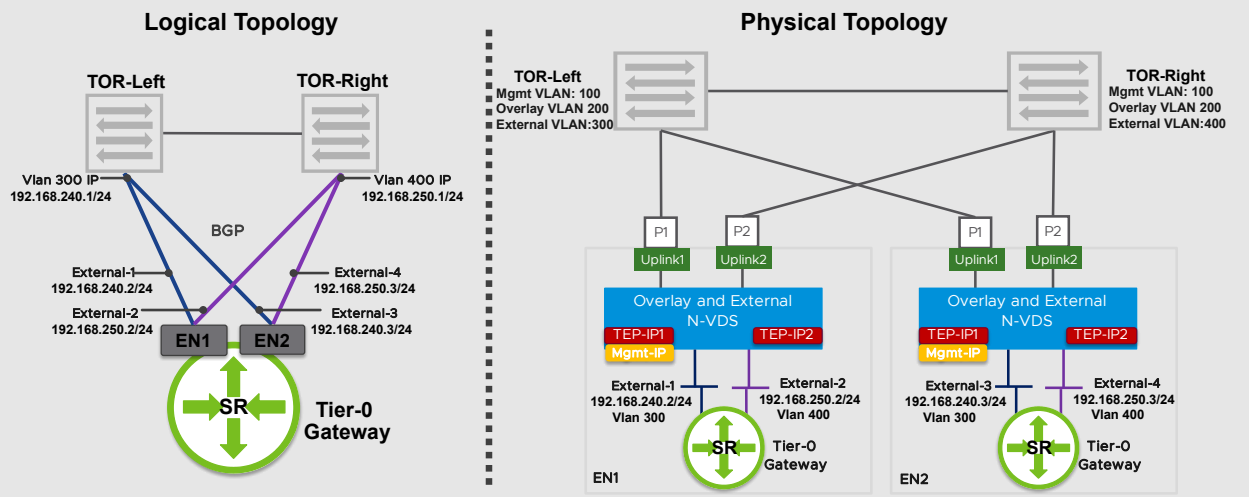


Figure 7-48: ECMP using multiple bare metal edges

For most environments the performance provided by a pNIC bare metal edge are comparable to an edge node VMs when the physical server has only 2 pNICs, if those interfaces are 10Gbps or 25Gbps. The pNIC bandwidth usually represents the bottleneck and the VM form factor is generally recommended for the easier lifecycle management. Situation when a bare metal edge node should be considered are:

- Requirement for line rate services
- Higher bandwidth pNIC (25 or 40 Gbps)

- Traffic profile is characterized by small packet size (e.g., 250 Bytes)
- Sub-second link failure detection between physical network and the edge node
- Network operation team retains responsibility for the NSX edges and has a preference for an appliance-based model
- In a multiple Tier-0 deployment model where the top Tier-0 is deployed on bare metal edges and drives the throughput with higher speed (40 Gbps) pNICs.
- Management interface redundancy is not always required but a good practice. In-band option is most practical deployment model when a limited number of interfaces is available.

7.5.2.1.2 Bare metal Design with more than 2pNICs

FIGURE 7-49 shows NSX bare metal Edge with six physical NICs. Management traffic has two dedicated pNICs configured in Active/Standby. Two pNICs, P3 and P4 are dedicated for overlay traffic and two pNICs (P5 and P6) are dedicated for external traffic.

This topology uses a single N-VDS which carries both overlay and External traffic. However, different uplinks are used to carry overlay and external traffic. Multi-TEP is configured to provide load balancing for the overlay traffic on Uplink1 (mapped to pNIC P3) and Uplink2 (mapped to pNIC P4). Notice that, both TEP IPs use same transport VLAN i.e. VLAN 200 which is configured on both top of rack switches.

FIGURE 7-49 also shows a configuration screenshot of named teaming policy defining two additional teaming policies, “Vlan300-Policy” and “Vlan400-Policy”.

"External VLAN segment 300" is configured to use a named teaming policy “Vlan300-Policy” that sends traffic from this VLAN only on Uplink3 (mapped to pNIC P5). "External VLAN segment 400" is configured to use a named teaming policy “Vlan400-Policy” that sends traffic from this VLAN only on Uplink4 (mapped to pNIC P6). Hence, BGP traffic from Tier-0 on VLAN 300 always goes to TOR-Left and BGP traffic from Tier-0 on VLAN 400 always goes to TOR-Right.

This topology provides redundancy for management, overlay and external traffic. This topology also provides a simple, high bandwidth and deterministic design as there are dedicated physical NICs for different traffic types (overlay and External traffic).

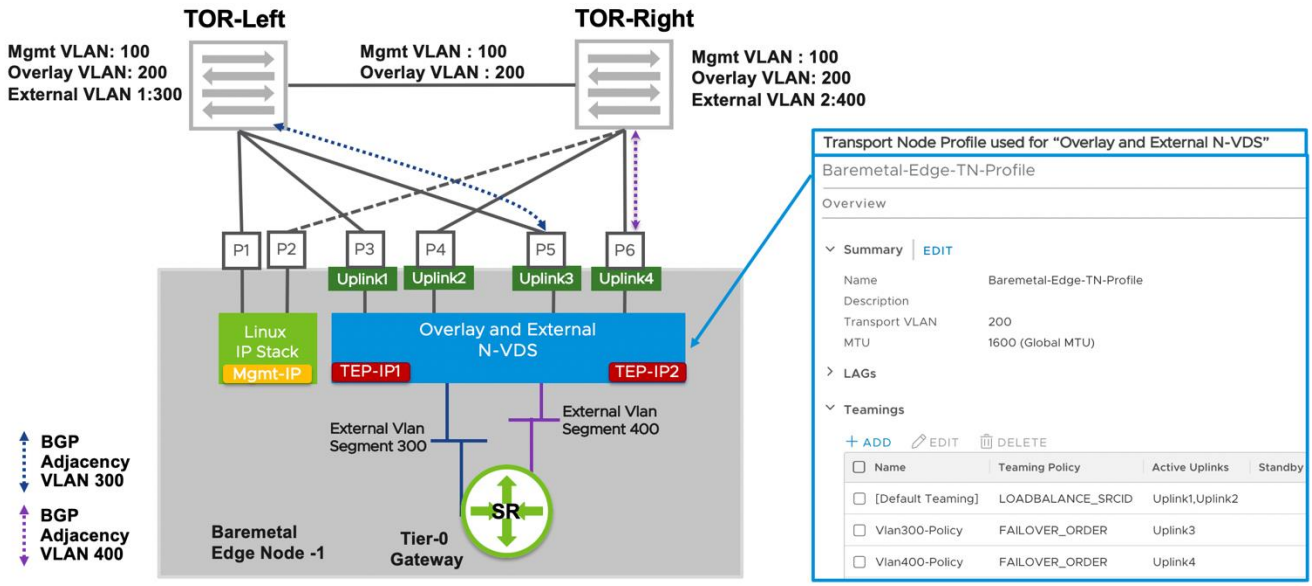


Figure 7-49: Bare metal Edge with six pNICs - Same N-VDS for Overlay and External traffic

The bare metal configuration with greater than two pNICs is the most practical and recommended design. This is because four or more pNICs configurations substantially offer more bandwidths than the equivalent Edge VM configurations. The same reasons for choosing bare metal apply as in the two pNICs configurations discussed above.

7.5.2.2 Edge Node VM

Starting with NSX 2.5 release, Edge nodes support Multi-TEP configuration to load balance overlay traffic. Similar to the bare metal Edge one N-VDS design, we recommend configuring the Edge VM with a single N-VDS for overlay and external traffic.

FIGURE 7-50 shows an Edge VM with one N-VDS to carry both overlay and external traffic. Multi-TEP is configured to provide load balancing for overlay traffic on "Uplink1" and "Uplink2". "Uplink1" and "Uplink2" are mapped to use vNIC2 and vNIC3 respectively. Based on this teaming policy, overlay traffic will be sent and received on both vNIC2 and vNIC3 of the Edge VM. Load balancing across the two TEPs is based on a segment hash. Traffic to and from the same overlay segment is forwarded by the same TEP. Both TEP IPs use same transport VLAN (i.e. VLAN 200 in the diagram) which is configured on both top of rack switches.

Similar to the bare metal edge design, the Tier-0 Gateway connects to the physical infrastructure leveraging VLAN segments "External VLAN Segment 300" and "External VLAN Segment 400" respectively. The routing adjacency between the T0 Gateway and the TOR switches happens on those two external VLANs. In this example, "External VLAN Segment 300" and "External VLAN Segment 400" are configured with a VLAN tag, 300 and 400 respectively. External traffic received on VDS port groups "Trunk1 PG" and Trunk2 PG" is VLAN tagged and hence, these port groups should be configured in VGT (Virtual guest tagging) mode and allow those specific VLANs.

Named teaming policy is also configured to force external traffic on specific edge VM nNICs. FIGURE 7-50 also shows named teaming policy configuration used for this topology. "External VLAN segment 300" is configured to use a named teaming policy “Vlan300-Policy” that sends traffic from this VLAN on “Uplink1” (vNIC2 of Edge VM). "External VLAN segment 400" is configured to use a named teaming policy “Vlan400-Policy” that sends traffic from this VLAN on “Uplink2” (vNIC3 of Edge VM). Based on this named teaming policy, external traffic from “External VLAN Segment 300” will always be sent and received on vNIC2 of the Edge VM. North-South or external traffic from “External VLAN Segment 400” will always be sent and received on vNIC3 of the Edge VM.

Overlay or external traffic from Edge VM is received by the VDS DVPGs “Trunk1 PG” and “Trunk2 PG”. The teaming policy used on the VDS port groups defines how this overlay and external traffic coming from Edge node VM exits the hypervisor. For instance, “Trunk1 PG” is configured to use active uplink as “VDS-Uplink1” and standby uplink as “VDS-Uplink2”. “Trunk2 PG” is configured to use active uplink as “VDS-Uplink2” and standby uplink as “VDS-Uplink1”.

This configuration ensures that the traffic sent on “External VLAN Segment 300” (i.e., VLAN 300) always uses vNIC2 of Edge VM to exit the Edge VM. This traffic then uses “VDS-Uplink1” (based on “Trunk1 PG” configuration) and is sent to the left TOR switch. Similarly, traffic sent on VLAN 400 uses “VDS-Uplink2” and is sent to the TOR switch on the right.

In case of a failure of an ESXi host pNIC, the active/standby teaming policy on the trunk dvpg will redirect the traffic to the surviving pNIC. Overlay traffic (VLAN 200 in the diagram) will flow over the operational ESXi pNIC. This failover event is entirely transparent to the edge node VM N-VDS, which keeps load balancing overlay traffic over the two TEPs and vNICs. Traffic originated or destined to both TEPs flows over a single ESXi pNIC. External VLAN traffic will not failover to the surviving ESXi pNIC because, while the overlay VLAN is defined on both switches, the two external VLANs are defined on TOR-Left or TOR-Right. A routing protocol reconvergence will redirect the traffic to the surviving peering VLAN and corresponding pNIC. The TOR switches configuration mustn't define the two peering VLAN on both switches; otherwise, it would be possible to have the failed neighborship restored over the inter-switch link, which is not desirable because of the spanning tree dependency and the lack of univocal mapping between the physical links and the routing paths.

This design does not consider the failure of the edge vNICs because it does not represent a realistic failure being the vNIC a virtual component. Disabling the edge vNICs will cause a black hole of the traffic. It is not recommended to enable MAC learning and/or forged transmit on the upstream VDS dvpg to support this failure scenario.

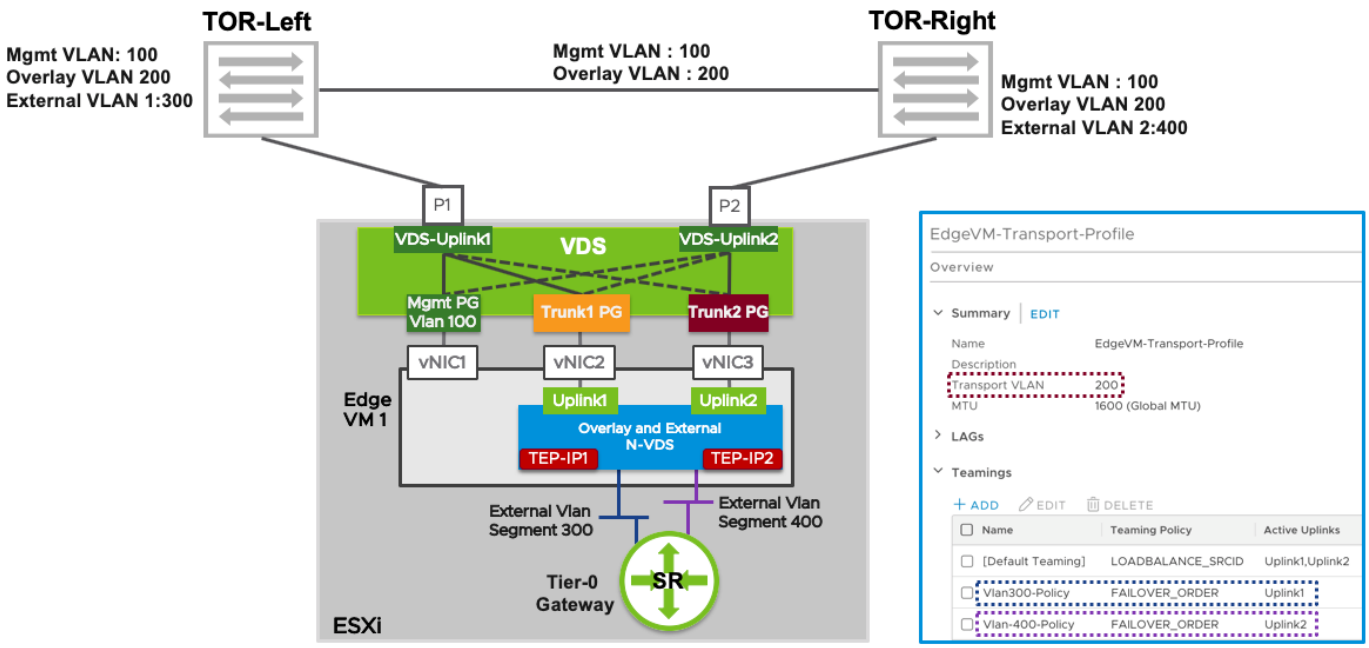


Figure 7-50: VLAN tagging on Edge node

Starting with NSX release 2.5, single N-VDS deployment mode is recommended for both bare metal and Edge VM. Key benefits of single N-VDS deployment are:

- Consistent deployment model for both Edge VM and bare metal Edge with one N-VDS carrying both overlay and external traffic.
- Load balancing of overlay traffic with Multi-TEP configuration.
- Ability to distribute external traffic to specific TORs for distinct point to point routing adjacencies.
- No change in DVPG configuration when new service interfaces (workload VLAN segments) are added.
- Deterministic North South traffic pattern.

7.5.2.3 VLAN Backed Service Interface on Tier-0 or Tier-1 Gateway

A service interface is an interface connecting VLAN backed segments/logical switch to provide connectivity to VLAN backed physical or virtual workloads. This interface acts as a gateway for these VLAN backed workloads and is supported both on Tier-0 and Tier-1 Gateways configured in active/standby HA configuration mode.

Service interface is realized on Tier-0 SR or Tier-1 SR. This implies that traffic from a VLAN workload needs to go to Tier-0 SR or Tier-1 SR to consume any centralized service or to communicate with any other VLAN or overlay segments. Tier-0 SR or Tier-1 SR is always hosted on Edge node (bare metal or Edge VM).

In the recommended single N-VDS design for layer 3 peering, no change in the DVPGs configuration is required when new service interfaces (workload VLAN segments) are added.

FIGURE 7-51 shows a VLAN segment "VLAN Seg-500" that is defined to provide connectivity to the VLAN workloads. "VLAN Seg-500" is configured with a VLAN tag of 500. Tier-0 gateway has a service interface "Service Interface-1" configured leveraging this VLAN segment and acts as a gateway for VLAN workloads connected to this VLAN segment. In this example, if the workload VM, VM1 needs to communicate with any other workload VM on overlay or VLAN segment, the traffic will be sent from the compute hypervisor (ESXi-2) to the Edge node (hosted on ESXi-1). This traffic is tagged with VLAN 500 and hence the DVPG receiving this traffic ("Trunk-1 PG" or "Trunk-2 PG") must be configured in VST (Virtual Switch Tagging) mode. Adding more service interfaces on Tier-0 or Tier-1 is just a matter of making sure that the specific VLAN is allowed on DVPG ("Trunk-1 PG" or "Trunk-2 PG").

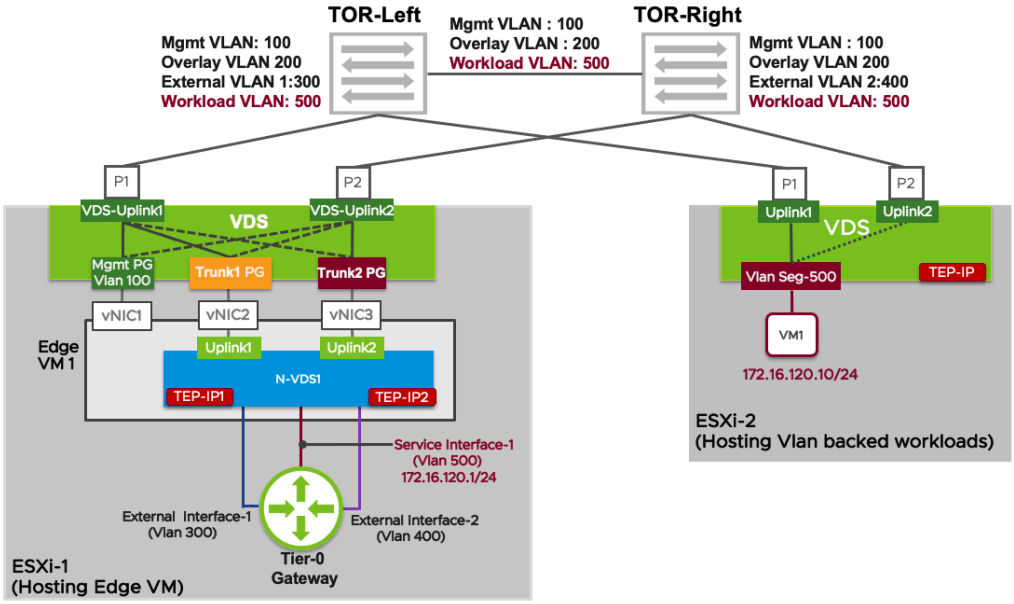


Figure 7-51: VLAN tagging on Edge node with Service Interface

Service interface on a Tier-1 Gateway can also be connected to overlay segments for standalone load balancer use cases. This is explained in Load balancer CHAPTER 6. Connecting a service interface to an overlay segment to act as the default gateway for the VMs on that segment is supported but not recommended. The service interface is residing on the edge node SR, meaning that any East-West traffic will traverse the edge node and distributed routing capabilities are not available. Overlay segments should always be connected to downlink interfaces (default behavior when creating a segment and connecting it to a Gateway).

In some corner case scenarios, service interfaces belonging to different Gateways (Tier-0 or Tier-1) must be connected to the same segment. This configuration is supported, with the caveat that if the segment is a VLAN the edges node hosting

the different gateways must belong to different edge clusters. This limitation does not apply to overlay segments.

7.5.2.4 Edge VM Design

This section covers the Edge VM design in various combinations. This design is solely based on single N-VDS per Edge VM for basic overlay and N-S connectivity. This design is consistent with the design that has been discussed for bare metal edge and remains the same for 2 pNIC or 4 pNIC design. The design pattern benefits in following ways:

- Symmetric bandwidth offered to both overlay and N-S
- Ability to accommodate asymmetric bandwidth requirement (i.e., 90% South to North Traffic)
- Deterministic layer 3 peering with univocal 1:1 mapping between logical and physical paths
- Repetitive design that can scale with pNIC number growth

7.5.2.4.1 Dedicated Host for Edge VM Design - 2 pNICs

FIGURE 7-52 below shows the Edge VM connectivity with 2 pNIC ESXi host. The basic configuration for overlay and N-S is described in detail in the section: EDGE NODE VM. The diagram shows how two edge VMs can share the same 2 host pNIC.

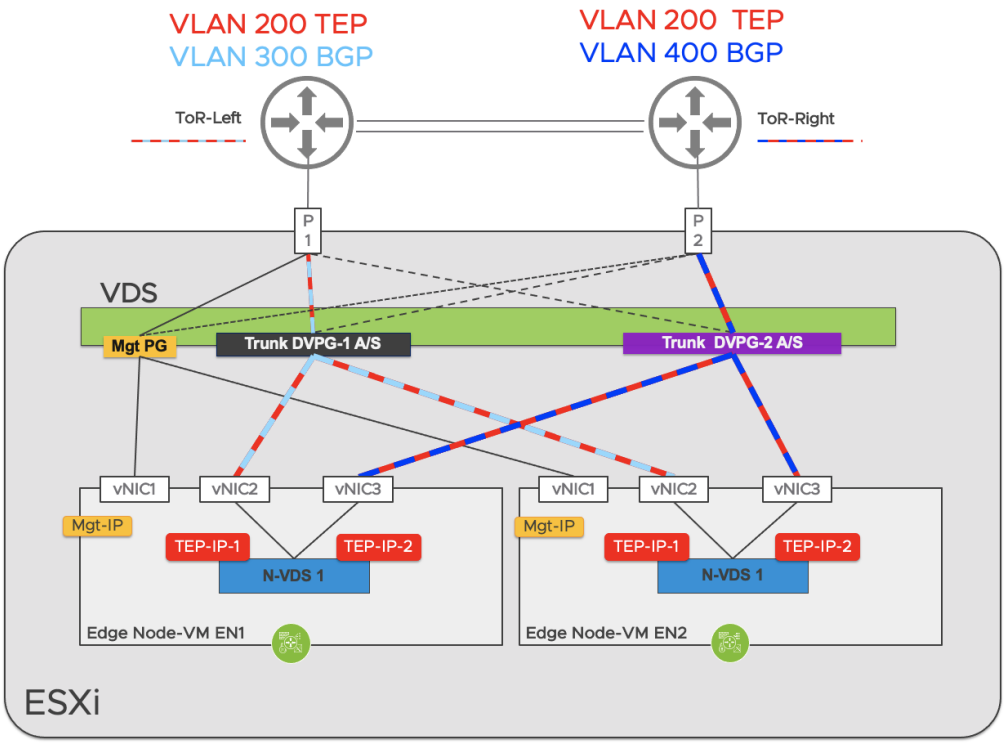


Figure 7-52: Single N-VDS per Edge VM - Two Edge Node VM on Host

The key design considerations are as follows:

- For pure North/South packets forwarding configurations (Edge VM running T0 Gateway only with no services) and common traffic profiles, the pNICs are generally the most evident performance-limiting factor. In most situations, placing two edge nodes on the same pNIC pair has a limited performance benefit.
- For shared edge clusters configurations, where the same edge node VM provides L3 forwarding and stateful services e.g., T0 or T1 Gateway firewall or NAT), CPU might be the limiting performance factor. Connecting multiple edges on the same two pNICs might better utilize the underlying hardware without saturating the pNIC capacity.
- Because of the tight relationship between edge VM performance and underlying hardware (specifically pNICs), we must carefully plan the placement of the edge VMs on the vSphere host. We should influence the automatic placement by vSphere DRS via DRS rules or disabled it altogether. DRS can help in balancing CPU intensive edge VMs (i.e. running TLS decryption or other CPU intensive services) across the vSphere hosts in a cluster, but when pNICs are the main source of contention, manual placement is generally more appropriate.
- In some designs placing multiple edge node VMs on the same ESXi host is driven by topology requirements and not performance or high availability. An example would be a multi-tenant design where each tenant receives a dedicated T0 Gateway (Dedicated edge VM). Higher consolidation ratios can be obtained by placing multiple edge VMs on the same host. Oversubscription is a risk to be mitigated by monitoring pNIC and CPU resource utilization. DRS can help mitigate CPU oversubscription in dense and/or dynamic environments with high churn.

The key implementation best practices are as follows.

- Transport zone – one overlay and VLAN – consistent compared to three N-VDS designs where external VLANs have two specific VLAN transport zone due to unique N-VDS per peering
- N-VDS(derived from matching transport zone name – both overlay and VLAN) defined with dual uplinks that map to unique vNICs, which maps to unique DVPG at VDS – duality is maintained end-to-end
- N-VDS carries multiple VLANs per vNIC – overlay and BGP peering
- The overlay VLAN must be the same on both N-VDS uplink with source ID teaming.
- BGP Peering VLAN is unique to each vNIC as it carries 1:1 mapping to ToR with a named teaming policy with only one active pNIC in its uplink profile
- VDS DVPG uplinks are active-standby (Failover Order teaming for the trunked DVPG) to leverage faster convergence of TEP failover. The failure of either pNIC/ToR will force the TEP IP to register (via GARP) on an alternate pNIC and TOR. The BGP peering recovery is not needed as alternate BGP peering is alive. The BGP peering over the failed pNIC will be timed out based on either protocol timer or BFD detection.

- The recommendation is not to use the same DVPG for non-edge VM traffic. VLAN allowed should be those of the routing peering, overlay, and any optional service interface.
- BFD configuration is recommended for fast link failure detection.
- Without BFD, recommended routing timer is set to 1/3 Sec. (Hello/Dead)

When adding bridging services to an edge VM configured for North/South peering, it is straightforward to layer the bridging topology presented in the bridging use case section: 2 pNICs server VM Form Factor. The third edge VM vNIC can be enabled and leveraged for the VLAN side of the bridge. A dedicated NVDS or the one already used for the routing peering together with a named teaming policy can be used. Figure 7-53 presents a scenario with a dedicated NVDS.

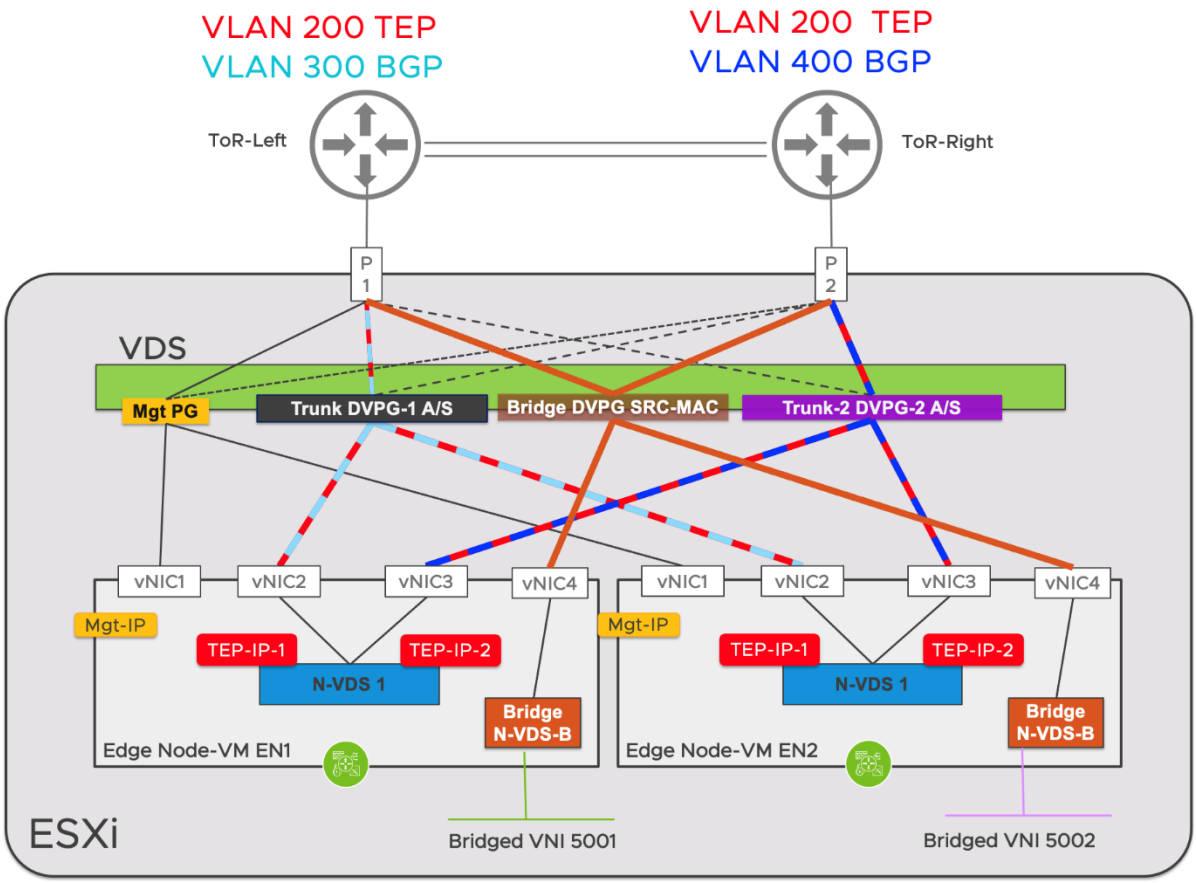


Figure 7-53: Two Edge VMs per host with 2 pNICs – Routing and bridging

7.5.2.4.2 Dedicated Host for Edge VM Design - 4 pNICs

The design choice with four pNICs is optimal for having multiple Edge nodes per host and minimizing pNIC oversubscription. In most cases (except when a host is oversubscribed with other VMs or resources like management, multi-tenant edges, services intense edges, etc.), it is not the host CPU, but the number of pNICs available on the host that determines the optimal number of Edge node per

host. We should customize the physical hardware dedicated to the edge node VMs considering the specific traffic profile of the environment. A good starting point for a balanced hardware configuration across the different relevant dimensions (CPU, pNICs, number, and size of the edge node VMs) and optimized for L3 forwarding only (Single T0 Gateway only, no services) is:

- 4 x 25Gbps pNICs (RSS capable)
- 32 X CPU Cores
- 2 X Large Edge Node VMs

More information about the rationale for this design is provided in the performance chapter 8.

From a configuration perspective, the 2 PNIC design is replicated "side-by-side" with dedicated pNICs and dvpgs. We create a separate management dvpg carrying traffic for the same VLAN and IP space of the first edge VM, but with a different teaming policy that forces the management traffic for the second edge VM over P3 and P4. We perform this configuration to enforce fate sharing across management, overlay, and VLAN peering networks for each edge VM. Refer to the edge HA section in chapter 4 for the rationale behind this recommendation.

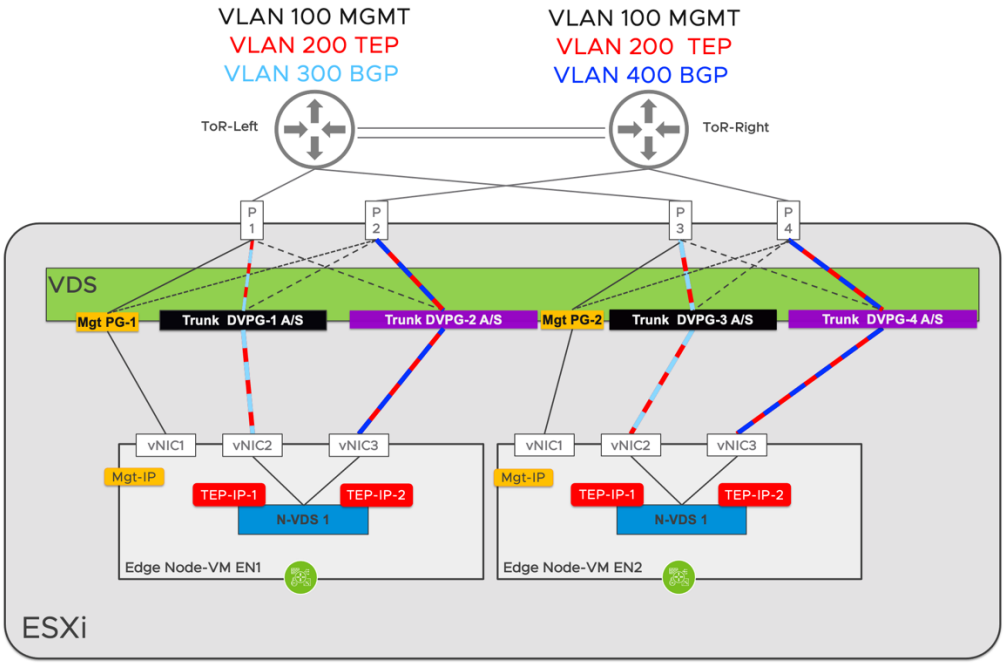


Figure 7-54: Two Edge VMs per host with 4 pNICs

7.5.2.5 Edge node VM connectivity to N-VDS (or VDS prepared for NSX)

In all the previous scenarios, we connected the edge node VMs to a vSphere VDS not prepared for NSX, which is the most common scenario when a dedicated vSphere edge cluster is available. When edge node VMs share the ESXi host with regular workload VMs requiring NSX services (Overlay and/or DFW), it may be desirable to connect the edge VMs and the workload VMs to the same virtual switch (NVDS or VDS prepared for NSX).

This type of design is the only available option when only two pNICs are available on the host. When four or more pNICs are available, it is possible to have multiple virtual switches on the ESXi host and connect edge VMs and workloads VMs to different ones.

Before NSX 3.1, it was required to place the ESXi TEPs and the Edge TEPs in different subnets and VLANs when leveraging the same two pNICs for both. FIGURE 7-55 below presents such a design. We can transport the edge overlay traffic over trunk NSX segments or regular vCenter managed trunk dvpgs. If the ESXi host virtual switch consists of an NVDS, NSX VLAN segments are the only choice.

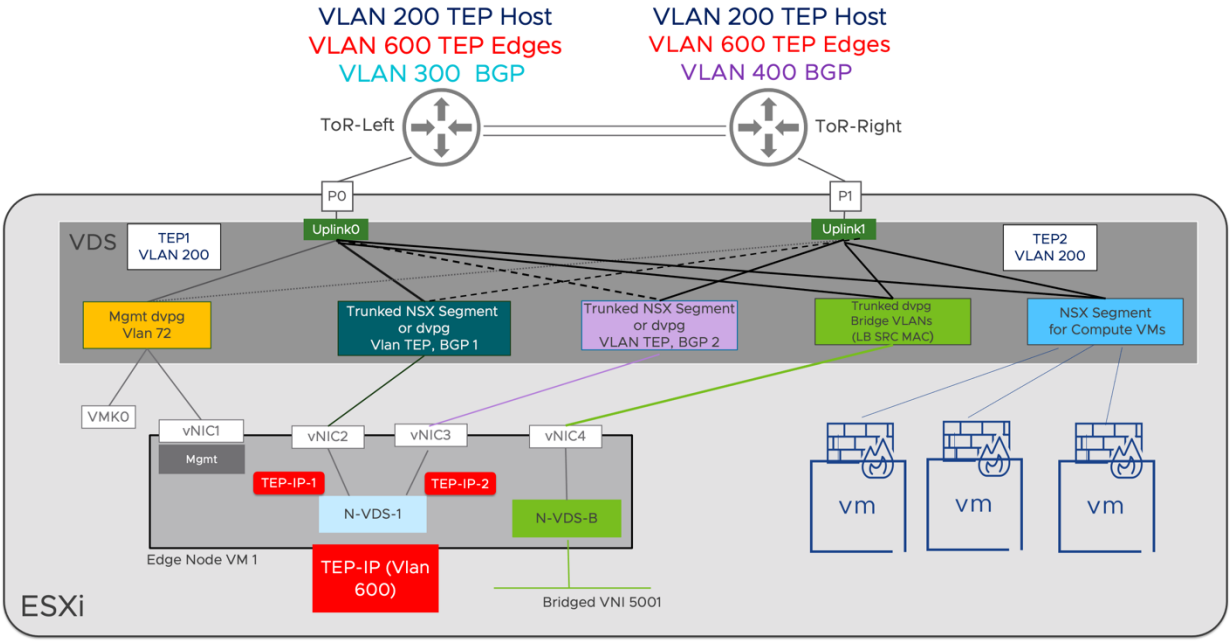


Figure 7-55: Edge VM connected to NSX Prepared VDS – Host and Edge TEP on different IP/VLAN

Starting with NSX version 3.1, edge and host TEPs can reside on the same VLAN because the host now can process Geneve traffic internal to the host itself. We must transport edge VM overlay traffic over an NSX Segment in this case. If the edge TEPs are connected to a vCenter managed dvpg, tunnels between the host and the edge will not come up. This design is presented in FIGURE 7-56 below:

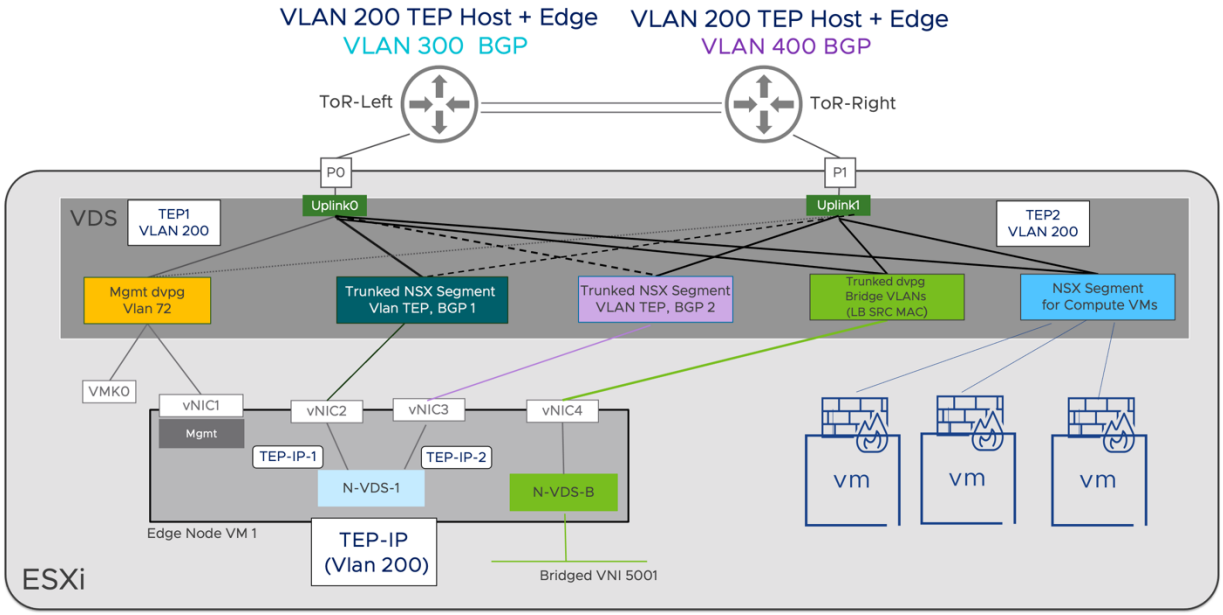


Figure 7-56: Edge VM connected to NSX Prepared VDS – Host and Edge TEP on the same IP/VLAN - NSX 3.1 or later

When connecting edge VMs to an NSX prepared VDS or NVDS, please keep in mind the following recommendations:

- Host and Edges should be part of different VLAN Transport Zones. This ensures a clear boundary between the transport segments on the host and those used for the routing peering on the edges. The edge segment traffic is transported by the host segments, configured as a trunk.
- When implementing a single TEP VLAN design like in FIGURE 7-56, the VDS trunk port groups transporting the edge TEP traffic must be NSX managed segments and cannot be created in vCenter.
- Follow the canonical recommendations regarding VLAN trunking and teaming policy configuration for overlay and VLAN peering traffic described in section: 7.5.2.2.

The design with different VLAN/IP subnets per TEP is still valid and can be used with any NSX version, including 3.1 or later. In most cases, the single TEP design is preferred for its simplicity, however for most deployments having separate VLANs for Edge and Host TEP is recommended due to following considerations:

- When Edges and hosts share the same TEP VLAN, they also share the span of that VLAN. While it is usually desirable to limit the host TEP VLAN to a rack, edge VMs may require mobility across racks or even sites (e.g., in the VCF stretched cluster design). Separate VLANs allow to manage the span of host and edge TEP networks individually.
- An edge and the host where the edge is running will never lose TEP connectivity if they share the same TEP network, regardless of a pNIC failure. This means that the edge node VM will never incur in an all tunnels down HA condition, limiting its ability to react to specific failures. Please refer to the EDGE HA SECTION IN CHAPTER 4 for more information. (Note: a

design that matches FIGURE 7-56 should not incur any issue as management, overlay, and VLAN peering networks share the same pNICs).

7.5.3 Edge Connectivity Guidelines for Services Only Use Case

Some network virtualization deployments may require deploying edge nodes for services only. Those are edge nodes where no L3 peering to the physical network is not required, see FIGURE 7-57. They are leveraged for stateful services, usually running on tier-1 gateways, but at times on tier-0 gateway in back to back tier-0 configurations (see TOPOLOGY CONSIDERATION). In such situations, we keep the configuration of the edge connectivity consistent with the layer three peering and bridging use cases but with fewer required configurations. The edge nodes dedicated for stateful services require only connectivity to the overlay network, so no VLAN transport zone or VLAN segments must be provisioned for the BGP peering.

Key implementation guidelines that apply to both bare metal and VM form factors are:

- Single NVDS deployment with multi-TEP configuration
- The edge node is only part of the overlay transport zone
- No VLAN connectivity is required (no VLAN TZ, no VLAN segments)

FIGURE 7-57 shows an example of bare metal edge node configuration where edge node 1 is a shared node where both Layer 3 peering and services are implemented. Bare metal edge node 2 is dedicated for services only. You can notice the absence of the VLAN transport zone and VLAN connectivity on the NVDS of bare metal edge 2.

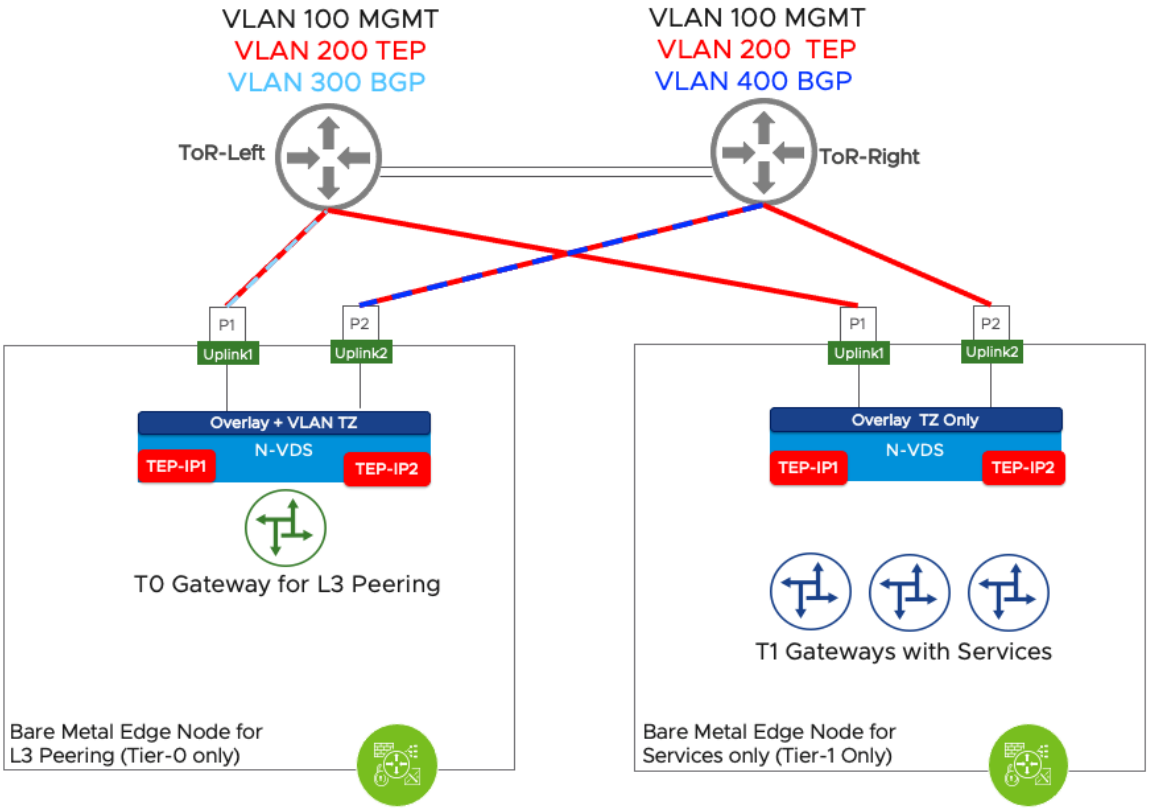


Figure 7-57: Services Only BM Edge

FIGURE 7-58 shows a detailed example of services only edge node VM connectivity. Again, you can notice the absence of VLAN connectivity requirements for the services only edges deployed on the ESXi host on the right.

Specific implementation considerations apply to the edge node VM form factor:

- VLAN tagging is not required on edge NVDS as the upstream VDS dvpg only carries overlay traffic and could be configured for a specific VLAN rather than as a trunk.
- Tagging overlay traffic on the edge node VM NVDS and configuring the dvpgs as trunks is recommended for consistency with the other use cases and possibly adopting them at a later stage on the same edge.

Specific design considerations apply to the edge node VM form factor:

- In a layer 2 physical fabric (or layer 3 with overlays), edge node VMs dedicated to T1 Gateway services can have a larger mobility span than edges providing peering functionalities and can be deployed on clusters striped across multiple racks.
- While the ESXi pNICs are usually the limiting performance factor for edge node VMs running T0 Gateway for layer three peering, the host CPU may represent the bottleneck for service-intensive edges. In such cases

connecting multiple edge VMs to the same pNIC may be appropriate (see the right ESXi in FIGURE 7-58)

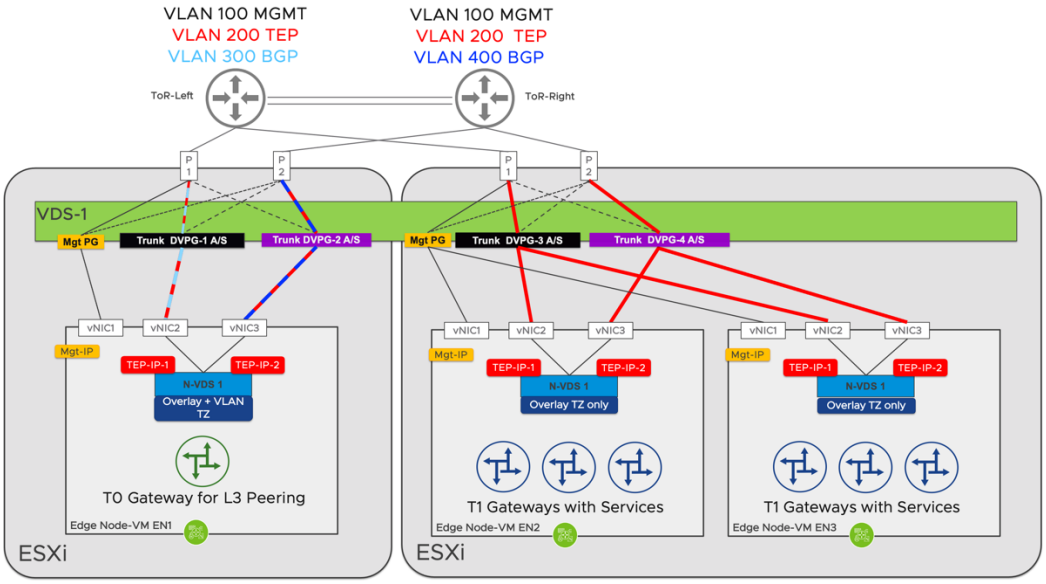


Figure 7-58: Services only Edge VM

7.5.4 NSX Edge Resources Design

The previous section covered the Edge node wiring for bare metal and VM form factors. It explains how overlay, N-S peering, and bridging connectivity can be implemented based on design considerations regarding the number of pNICs and the availability model. This section provides details about building services (e.g., ECMP, FW, NAT, LB, and VPN) with either bare metal or VM edge nodes. In addition, we include several considerations about optimizing the resource footprint for such services. There are two major design areas affecting the implementation of NSX service clusters: the type of services enabled and the clustering model.

7.5.4.1 Edge Services Requirements

The first step in designing and edge cluster is gathering the requirements for the solution. We should group those requirements in functional and non-functional requirements.

Functional requirements describe what the system should do, and in the context of the edge cluster design, they should list, as a minimum the services in scope. Ideally, they should be mapped to the relevant workloads that will consume such services. An example of a weak functional requirement is:

- *The edge cluster will provide NAT, gateway Firewall, and VPN services*

Stronger functional requirements are:

- *Tenant 1 requires SNAT functionalities*
- *Tenant 2 requires layer 3 VPN services to redundant remote peers*
- *Tenant 3 requires access to the hosted workloads without any NAT*

We use the word tenant here to represent a group of workloads. The considerations are general and valid for both an enterprise and a service provider design. To be noted is that such requirements are derived from higher level design decisions or business requirements, for example:

- *Tenant 1 will migrate their current application to the cloud platform without change to the IP addresses, so IP conflict with other tenants should be managed*
- *Tenant 2 considers direct private links to the cloud provider too expansive and opted for IPSEC VPN*
- *Tenant 3 users deploys VMs in the private cloud environment with the expectation that they are fully accessible from any internal location with minimal complexity*

Non-functional requirements describe how the edge services cluster work as a system and the level of service it will provide. Non-functional requirements can be grouped in five categories: Availability, Manageability, Performance, Recoverability, and Security. We will provide some examples for each category.

Availability:

- General SLA for the service itself. We could have an SLA for all the services, e.g., X for all services to be available, or we could have a different SLAs for different services (e.g., X for layer3 peering, Y for VPN, Z TLS Decryption),

or even a specific one per tenant and service (e.g., tenant 1 requires A for TLS decryption, tenant 1 requires B for Gateway IPS functionality, tenant 2 requires C for TLS decryption). SLA requirements affect the edge cluster design in term of the number of edge nodes deployed, where they are deployed (same rack, different rack, different rooms), and the availability requirements of the underlying infrastructure (e.g., the vSphere cluster or the datastore)

- Max recovery time in case of a network failure. While the general SLA considers the total downtime over an extending time, usually a year, requirements should outline the expectation for fast recovery in case of common hardware network failure such as the failure of a pNIC, a host, or a top of rack switch. The reason for such requirements is that why multiple short interruption may not violate the SLA for the year, they may represent a problem for applications with short timeouts or incapable of reconnecting automatically. This type of requirements affects, among other things, the edge node form factor, routing protocols and BFD timer's implementation. These requirements should always be derived by specific application needs, and not provided as blanket statements.

Manageability:

- Lifecycle. Different edge services should be able to undergo maintenance at different times. Requirements in this area may lead to deploy edges in different edge clusters, different vSphere clusters, and even different NSX Manager domains.
- Scalability. How elastic should the service cluster be? This consideration may not apply to all the services in the same way. For example, I may need to increase the throughput for the TLS decryption service based on on-boarding new tenants, but the VPN service will not be affected by this change. Requirements in this area may lead to dedicate edge clusters to specific services, and to design the underlying infrastructure with different characteristics depending on the services (e.g., hosts with different pNIC, CPU resources, memory)
- RBAC. Who is entitled to configure each service? Is it a self-service offering? This area has impacts on the integration with directory services and user privileges.
- Object based RBAC. Do I need different users to be able to manage different objects of the same type? For example, each tenant should be able to manage the Gateway Firewall rules for their dedicated T0 or T1 Gateway. This area may lead to considerations around separating the NSX deployment in multiple NSX Managers domains, or the integration with the most appropriate cloud management platform.
- Monitoring. What are the metrics that I need to collect to ensure I am proactive in addressing performance and scalability issues? Separating services on different edge VMs may provide granular reporting around the resource consumption of specific services.

Performance:

- North/South throughput. The amount of traffic that the NSX edges should support between the physical and the virtual environments is commonly

estimated in bits per second (bps), e.g., 20Gbits. Such requirements tend to be misleading as the throughput is highly influenced by the packet size. So, the traffic profile should be taken into consideration when stating the expected North/South throughput for the system. A more general and accurate characterization of the capability of the provided design would be stating North/South throughput is packets per second (PPS). Requirements in this are affect edge node form factor, the number of edges deployed in ECMP, the host resources in terms of pNIC and CPU, and others.

- Latency and jitter. Some application may have stringent latency and jitter requirements. Those requirements should be evaluated and may lead to the provisioning of dedicated resources such as T0 Gateways, pNICs or hosts to avoid the resource contention between mice and elephant flows.
- New connections per second (CPS). This parameter has impact on the stateful services design, it may require spreading services across multiple edge nodes.
- Throughput per service. This is different from the north/south throughput which is generally and aggregate of all the services traffic. Different services may be more or less resource intensive and may require the allocation of dedicated resources. For example, NAT and Gateway firewall are in most case light on resources and can coexist on the same edge dedicated to the layer 3 peering without a noticeable performance impact. Other services (e.g. VPN and TLS decryption) may have noticeable impact if deployed at scale. Requirements in this area will impact the segmentation of services in different edge clusters, and the choice of the edge form factor for some services.

Recoverability:

- Recovery Time Objective (RTO). This is different than the requirements for high availability in the sense that we want to specify how much time we have to restore the services after a major outage. The outage in scope depends on the context of the design and may refer for example to a rack failure, a datacenter room failure, or an entire site failure. Requirements in this area may affect the striping of an edge cluster across different availability zones, the protection mechanisms we want to rely on for the recovery (e.g., edge native HA vs vSphere HA), and the recovery procedure in place (fully automatic, vs. manual, vs. scripted)
- Recovery Point Objective (RPO). While RPO usually refers to workload data, in the context of NSX services it may help us formalize how much configuration it is acceptable to lose because of a failure. While this requirement may not be stringent in a manually provisioned environments, NSX is often a core building block of self-service platform where applications and the supporting infrastructure services are continuously provisioned. Requirements in this area may affect the NSX Manager cluster deployment model, the backup settings, and the underlying storage technology.

Security:

- Compliance. Compliance requirements varies and may lead to various degrees of separation. When distributed firewall is an acceptable

segregation mechanism, compliance has no impact on edge services and network topologies. When an organization requires stronger "physical" boundaries, compliance may lead to segment the environment with multiple gateways, edge clusters, transport zones, or even NSX Manager domains.

7.5.4.2 Edge Services Technology rule set

Once the edge services requirements have been outlined, it is important to identify the relevant technology constrains. A technology constrain is not a negative attribute of the technology in use, in this case NSX, it's simply a description of how the system behave based on the way it has been designed. We need to consider the boundaries of the product capabilities to produce a valid design. Chapters one to six provide detail explanation of how NSX works. In this section we will emphasize functionalities and their limits that are important for the edge services design.

- A single T0 Gateway can be deployed on each edge node.
- A T0 or T1 Gateway cannot span multiple NSX edge clusters (NSX Edge VM clusters can span multiple vSphere clusters)
- ECMP. ECMP provides load sharing across multiple layer 3 next-hops. ECMP happens on each T0 Gateway component. It means that the Distributed Router (DR) component on the ESXi host is capable of forwarding the outbound traffic to eight edge nodes. The T0 service router (SR) component on each edge node is capable of load sharing the traffic across eight upstream physical devices.
- Multiple Edge clusters can be deployed within a single NSX Manager, allowing for the creation of pool of capacity that can be dedicated to specific services (e.g., NAT at Tier-0 vs. NAT at Tier-1)
- Within a single Edge cluster, all Edge nodes should be the same type – either bare metal or VM. Edge node VMs of different size can be mixed in the same Edge cluster, as can bare metal Edge nodes of different performance levels based on pNICs, however those combination is discouraged but supported for upgrades and other lifecycle reasons. A mixture of different sizes/performance levels within the same Edge cluster can have the following effects:
    - With two Edge nodes hosting a Tier-0 configured in active/active mode, traffic will be spread evenly. If one Edge node is of lower capacity or performance, half of the traffic may see reduced performance while the other Edge node has excess capacity.
    - For two Edge nodes hosting a Tier-0 or Tier-1 configured in active/standby mode, only one Edge node is processing the entire traffic load. If this Edge node fails, the second Edge node will become active but may not be able to meet production requirements, leading to slowness or dropped connections.
- Services available on a T1 Gateway and not on a T0 Gateway. Some services are available on T1 Gateway only and are not available on T0 Gateway. Designs requiring: L7 Gateway Firewall, Gateway IDS/IPS/Malware Prevention, Gateway Identity Firewall, URL Filtering, TLS Inspection and NSX Load Balancer, need to include T1 Gateways.

- Services on T1 implies the hair pinning of the traffic between tenants connected to different T1 Gateways.
- VPN remote peer redundancy. VPNs can be enabled on T0 or T1 Gateways with exactly the same functionalities except for the ability to establish BGP peering over an IPSEC routed VPN. If remote peer redundancy is required VPNs on T0 Gateways are the only option.
- Because Edge nodes can be part of a single overlay transport zone, if the NSX deployment has been segmented in multiple overlay transport zone, as a minimum, a corresponding number of edge clusters should be deployed to provide physical to virtual connectivity to each overlay transport zone.

7.5.4.3 Preemptive vs Non-preemptive Mode with Active/Standby Services

Each stateful services can be configured for either preemptive or non-preemptive mode. The design choice is between deterministic balancing of services among available resources (bandwidth and CPU) verses reducing disruption of services.

- The preemptive model allows making sure that, when the system is fully operational, pool of edge resources (bandwidth and CPU) always get balanced after the restoration of host or Edge VM. However, preemptive mode triggers the switchover of Edge VM running services, leading secondary disruption causing packet loss. Operationally this may not be acceptable triggering intentional switchover.
- The non-preemptive model maximizes availability and is the default mode for the service deployment. If the active fails then recovers; it will be in standby mode, it will not trigger a re-convergence that could lead to unnecessary packet loss (by preempting the currently active.) The drawback is that, after a recovered failure, the services remains polarized on a host or an edge cluster. This leads to an oversubscription of a host or uplink bandwidth. If availability is more important than oversubscription (bandwidth and CPU), this mode is perfectly acceptable. Typically, one can restore the rebalancing of the services during off-peak or planned maintenance window.

7.5.4.4 Edge cluster deployment options, shared vs dedicated

Services can be enabled either in shared or dedicated Edge node.

Shared Mode: Shared mode is the most common deployment mode and a starting point for building the edge services model. Tier-0 or Tier-1 not enabled with stateful service, by default runs in distributed mode. In shared mode, typically Tier-0 runs ECMP service while Tier-1 can runs stateful services (aka either active or standby mode for that services). If the Edge node fails, all the services within that nodes fails. Below FIGURE 7-59: SHARED SERVICE EDGE NODE CLUSTER shows the flexibility in deploying ECMP services along with variable services model for Tier-1. Tier-0, represented by the green node, running the ECMP service on all four Edge nodes providing aggregated multi-Gbps throughput for combined Tier-1 services. The Tier-1 services are deployed based on tenants or workload needs. For an example few Tier-1 services (red, black and blue) are stateful over a pair of Edge nodes where many services are spread over four nodes. In other words, they do not have to be deployed on the same nodes so the services can scale. The stateful services have a SR component running on the Edge nodes. The active SR component is shown with solid color Tier-1 router icon while the standby

on in light faded icon. The Tier-1 routing function can be entirely distributed (aka stateless yellow Tier-1) and thus does not have SR component and thus they exist on each Edge node by its nature of distributed router component, below figure depiction of for it is for illustration purpose only. Note that active/standby services all have distributed routing (DR) component running in all Edge nodes but traffic will be always be going through active services. This services configuration can be applicable to both bare metal and Edge VM.

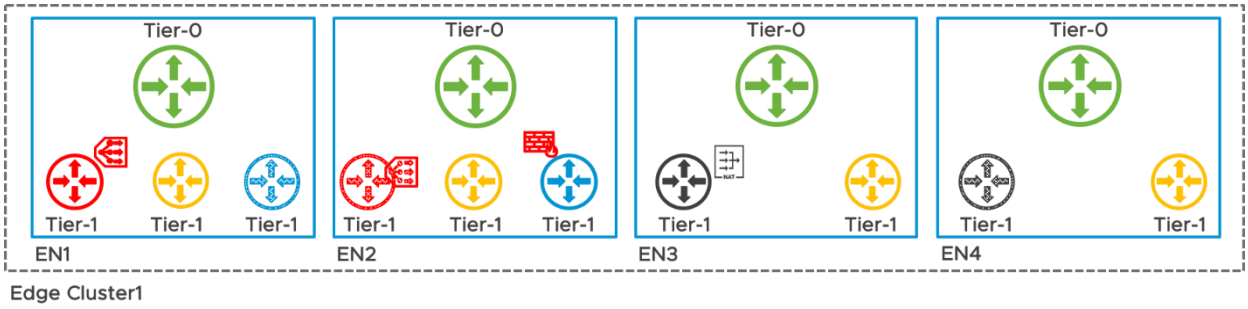


Figure 7-59: Shared Service Edge Node Cluster

The shared mode provides simplicity of allocating services in automated fashion as NSX tracks which Edge node is provisioned with service and reduced that Edge node as potential target for next services deployment. However, each Edge node is sharing CPU and thus bandwidth is shared among services. In addition, if the Edge node fails, all the services inside Edge nodes fails together. Shared edge mode if configured with preemption for the services, leads to only service-related secondary convergence. On the other hand, it provides optimized footprint of CPU capacity per host. If the high dedicated bandwidth per service and granular services control is not a priority, then use shared mode of deployment with Edge services.

Dedicated Mode: In this mode, Edge node is either running ECMP or stateful services but not both. This mode is important for building scalable and performance-based services edge cluster. Separation of services on dedicated Edge node allows distinct operational model for ECMP vs stateful services. The choices of scaling either ECMP or stateful services can be achieved via choice of bare metal or multiple of Edge VMs.

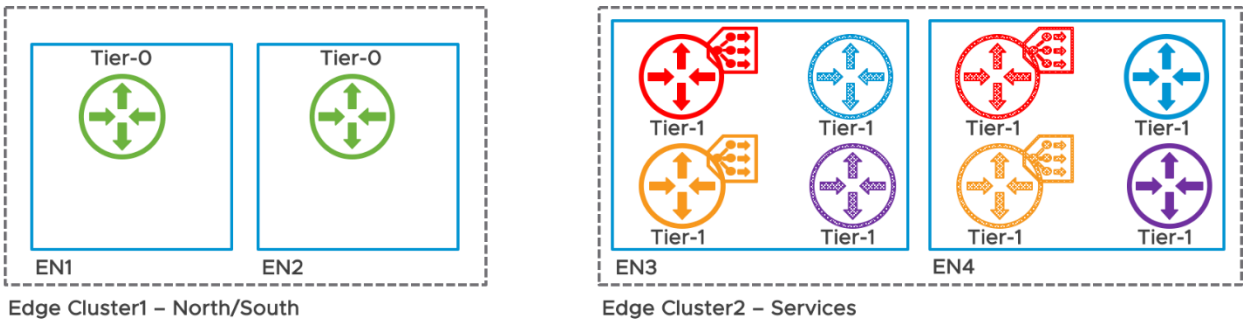


Figure 7-60: Dedicated Service Edge Node Cluster

FIGURE 7-60: DEDICATED SERVICE EDGE NODE CLUSTER described dedicated modes per service, ECMP or stateful services. One can further enhanced configuration by

deploying a specific service per Edge node, in another word each of the services in EN3 and EN4 gets deployed as an independent Edge node. It's the most flexible model, however not a cost-effective mode as each Edge node reserves the CPU. In this mode of deployment one can choose preemptive or non-preemptive mode for each service individually if deployed as a dedicated Edge VM per services. In above figure if preemptive mode is configured, all the services in EN3 will experience secondary convergence. However, if one segregate each service to dedicated Edge VM, one can control which services can be preemptive or non-preemptive. Thus, it is a design choice of availability verses load-balancing the edge resources. The dedicated edge node either per service or grouped for set of services allows deploying a specific form factor Edge VM, thus one can distinguish ECMP based Edge VM running larger form (8 vCPU) allowing dedicated CPU for high bandwidth need of the NSX domain. Similar design choices can be adopted by allowing smaller form factor of Edge VM if the services do not require line rate bandwidth. Thus, if the multi-tenant services do not require high bandwidth one can construct a very high density per tenant Edge node services with just 2 vCPU per edge node (e.g. VPN services or a LB deployed with DevTest/QA). The LB service with container deployment is one clear example where adequate planning of host CPU and bandwidth is required. A dedicated edge VM or cluster may be required as each container services can deploy LB, quickly exhausting the underlying resources.

Another use case to run dedicated services node is multi-tier Tier-0 or having a Tier-0 level multi-tenancy model, which is only possible with running multiple instances of dedicated Edge node (Tier-0) for each tenant or services and thus Edge VM deployment is the most economical and flexible option. For the startup design one should adopt Edge VM form factor, then later as growth in bandwidth or services demands, one can lead to selective upgrade of Edge node VM to bare metal form. For Edge VM host convertibility to bare metal , it must be compatible with BARE METAL REQUIREMENT. If the design choice is to immunize from most of the future capacity and predictive bandwidth consideration, by default going with bare metal is the right choice (either for ECMP or stateful services). This decision to go with VM versus bare metal also hinges on operational model of the organization in which if the network team owns the lifecycle and relatively want to remain agnostic to workload design and adopt a cloud model by providing generalized capacity then bare metal is also a right choice.

7.5.4.5 Services Availability Considerations with Edge Node VM

The availability and service placement for Edge node VM depends on multiple factors due to nature of its flexibility as a VM. Design choices revolves around shared vs dedicated services deployment, number of Edge nodes, in-rack vs multi-rack availability and number of pNIC available in the host, growth, capacity and finally bandwidth required as a whole and per services. In addition, restrictions exists when the Edge node VM coexist with compute VMs in the same host. In this section the focus is mostly in the context of dedicated edge cluster and availability with two pNICs.

There are two forms of availability to consider for Edge VM. First is Edge node availability as a VM and second is the service that is running inside Edge VM. Typically, the Edge node VM availability falls into two models. In-rack verses multi-rack. In-rack availability implies minimum two hosts are available (for both ECMP and stateful services) and failure of a host will trigger either re-deployment of Edge node to available host or a restart of the Edge VM depending on the

availability of the underlying hypervisor. For multi-rack availability, the recommendation is to keep availability model for Edge node recovery/restoration/redeployment restricted to rack avoiding any physical fabric related requirement (independent of L2 or L3 fabric).

For the simplest and most common form of Edge deployment is shown in below FIGURE 7-61: ECMP BASE EDGE NODE CLUSTER GROWTH PATTERN, in which entire NSX domains is only requiring on Tier-0 active/active (ECMP) services. In this case, it is important to remember that all Tier-1 are distributed by default and thus does not require any specific consideration. The services enablement assumes single rack deployment, with minimum two Tier-0 (ECMP only services) Edge Nodes. The growth pattern starts with two hosts to avoid single point of failure, addition of two additional Edge nodes per host, leading to four hosts with eight Edge nodes delivering eight ECMP forwarding paths. Notice the edge cluster striping is vertical and not much impact how it is striped. If one has requirement to support multi-tenant with each tenant requiring dedicated Tier-0, one must stripe more than one cluster and vertically for which minimum unit of deployment is two hosts with two Edge nodes (this is not shown in below diagram, but one can imagine that arrangement of ECMP services)

.

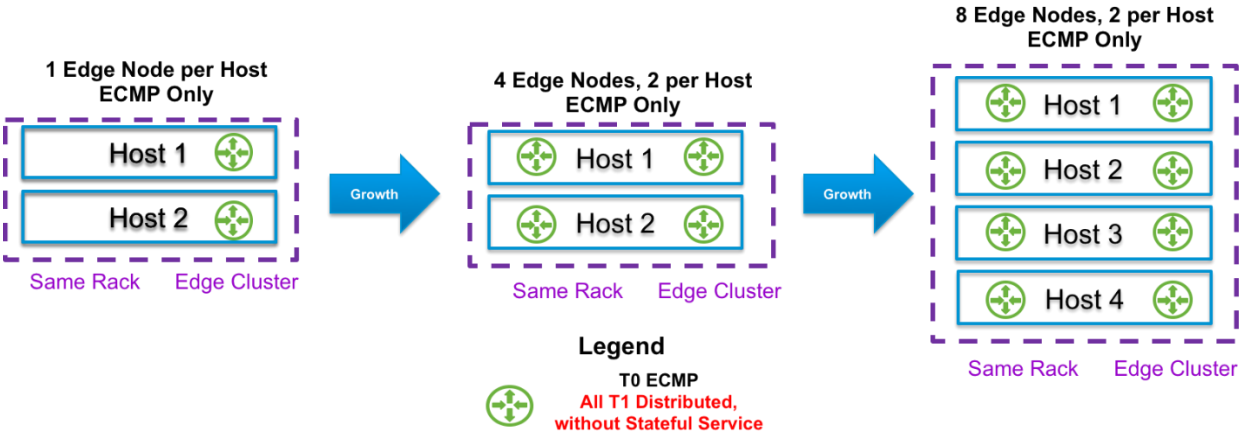


Figure 7-61: ECMP Base Edge Node Cluster Growth Pattern

For the deployment that requires stateful services the most common mode of deployment is shared Edge node mode (see FIGURE 7-59: SHARED SERVICE EDGE NODE CLUSTER) in which both ECMP Tier-0 services as well stateful services at Tier-1 is enabled inside an Edge node, based on per workload requirements. The FIGURE 7-62: SHARED SERVICES EDGE NODE CLUSTER GROWTH PATTERNS below shows shared edge not for services at Tier-1, red Tier-1 is enabled with load-balancer, while black Tier-1 with NAT. In addition, one can enable multiple active-standby services per Edge node, in other word one can optimize services such that two services can run on separate host complementing each other (e.g. on two host configuration below one can enable Tier-1 NAT active on host 2 and standby on host 1) while in four hosts configuration dedicated services are enabled per host. For the workloads which could have dedicated Tier-1 gateways, are not shown in the figure as they are in distributed mode thus, they all get ECMP service from Tier-0. For the active-standby services consideration, in this case of in-rack deployment mode one must ensure the active-standby services instances be deployed in two different host. This is obvious in two edge nodes deployed on

two hosts as shown below as NSX will deploy them in two different host automatically. The growth pattern is just adding two more hosts so on. Note here there is only one Edge node instances per host with assumption of two 10 Gbps pNICs. Adding additional Edge node in the same host may oversubscribed available bandwidth, as one must not forget that Edge node not only runs ECMP Tier-0 but also serves other Tier-1s that are distributed.

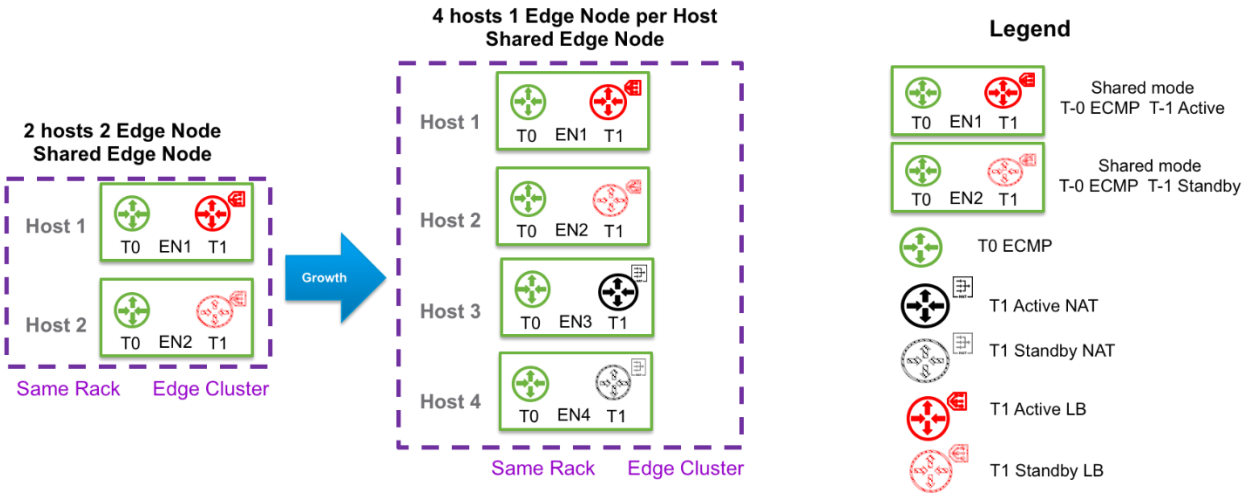


Figure 7-62: Shared Services Edge Node Cluster Growth Patterns

The choices in adding additional Edge node per host from above configuration is possible with higher bandwidth pNIC deployment (25/40 Gbps) or, even better, a higher number of pNICs. In the case four Edge node deployment on two hosts, it is required to ensure active-standby instances does not end up on Edge nodes on the same hosts. One can prevent this condition by building a horizontal Failure Domain as shown in below FIGURE 7-63: TWO EDGE NODES PER HOST – SHARED SERVICES CLUSTER GROWTH PATTERN. Failure domain in below figures make sure any stateful services.

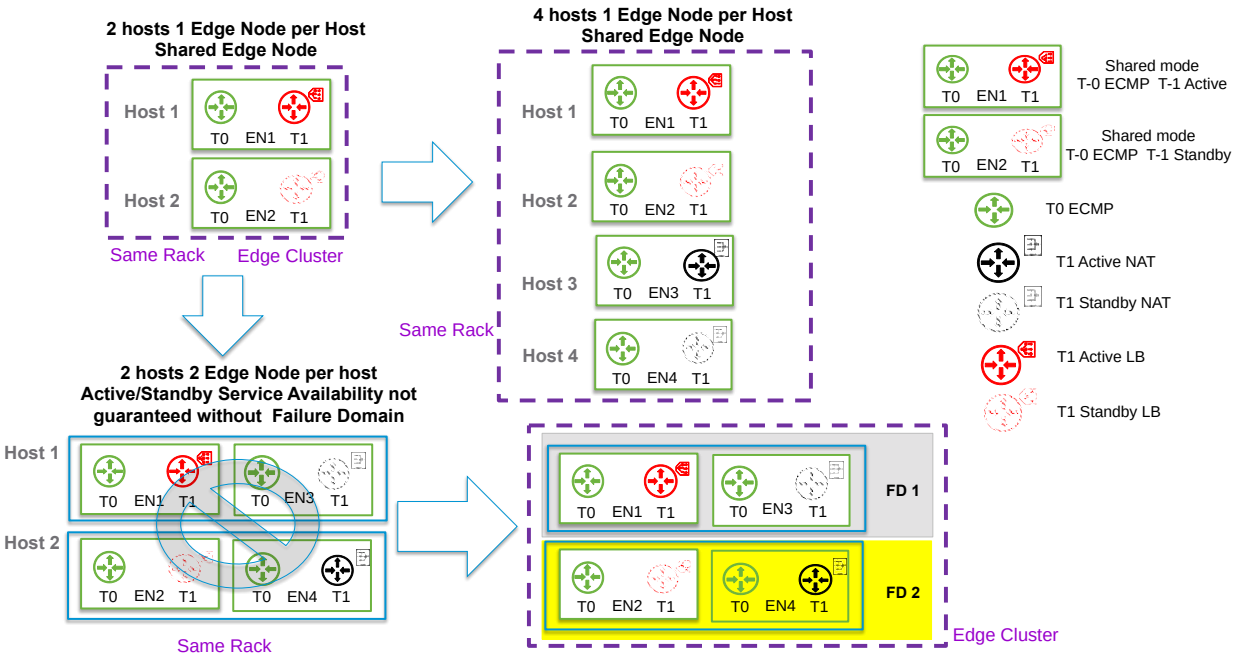


Figure 7-63: Two Edge Nodes per Host – Shared Services Cluster Growth Pattern

An edge cluster design with dedicated Edge node per services is shown in below FIGURE 7-64: DEDICATED SERVICES PER EDGE NODES GROWTH PATTERN. In a dedicated mode, Tier-0 is only running ECMP services belongs to first edge cluster while Tier-1 running active-standby services on second edge cluster. Both of this configuration are shown below.

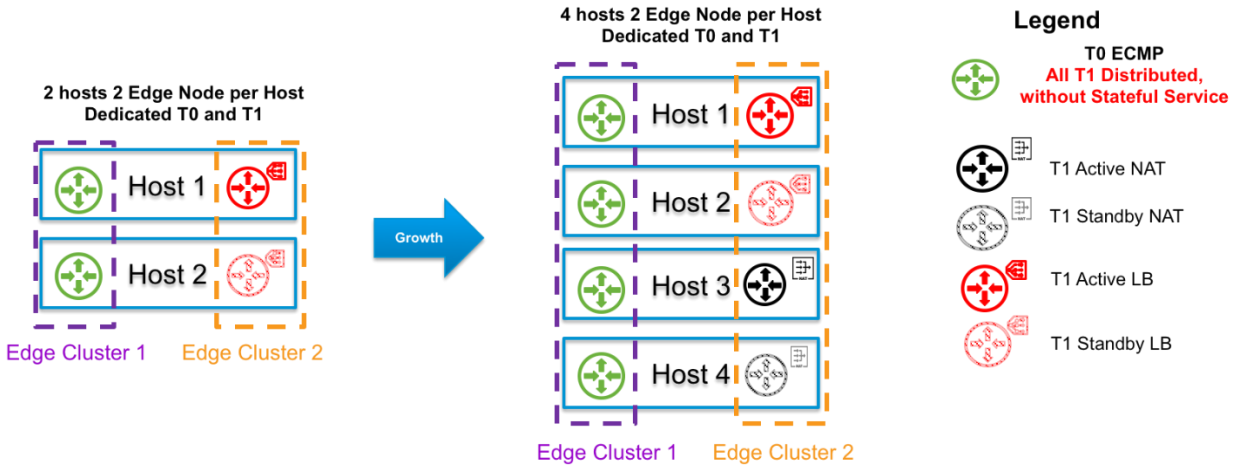


Figure 7-64: Dedicated Services per Edge Nodes Growth Pattern

Notice that each cluster is striped vertically to make sure each service gets deployed in separate host. This is especially needed for active/standby services. For the ECMP services the vertical striping is needed when the same host is used for deploying stateful services. This is to avoid over deployment of Edge nods on

the same host otherwise the arrangement shown in FIGURE 7-61: ECMP BASE EDGE NODE CLUSTER GROWTH PATTERN is a sufficient configuration.

The multi-rack Edge node deployment is the best illustration of Failure Domain capability. It is obvious each Edge node must be on separate hypervisor in a separate rack with the deployment with two Edge nodes.

The case described below is the dedicated Edge node per service. The figure below shows the growth pattern evolving from two to four in tandem to each rack. In the case of four hosts, assuming two Edge VMs (one for ECMP and other for services) per host with two hosts in two different rack. In that configuration, the ECMP Edge node is stripped across two racks with its own Edge cluster, the placement and availability are not an issue since each node is capable of servicing equally. The Edge node where services is enabled must use failure domain vertically striped as shown in below figure. If the failure domains are not used, the cluster configuration will mandate dedicated Edge cluster for each service as there is no guarantee that active-standby services will be instantiated in Edge node residing in two different rack. This mandate minimum two edge clusters where each cluster consist of Edge node VM from two racks providing rack availability.

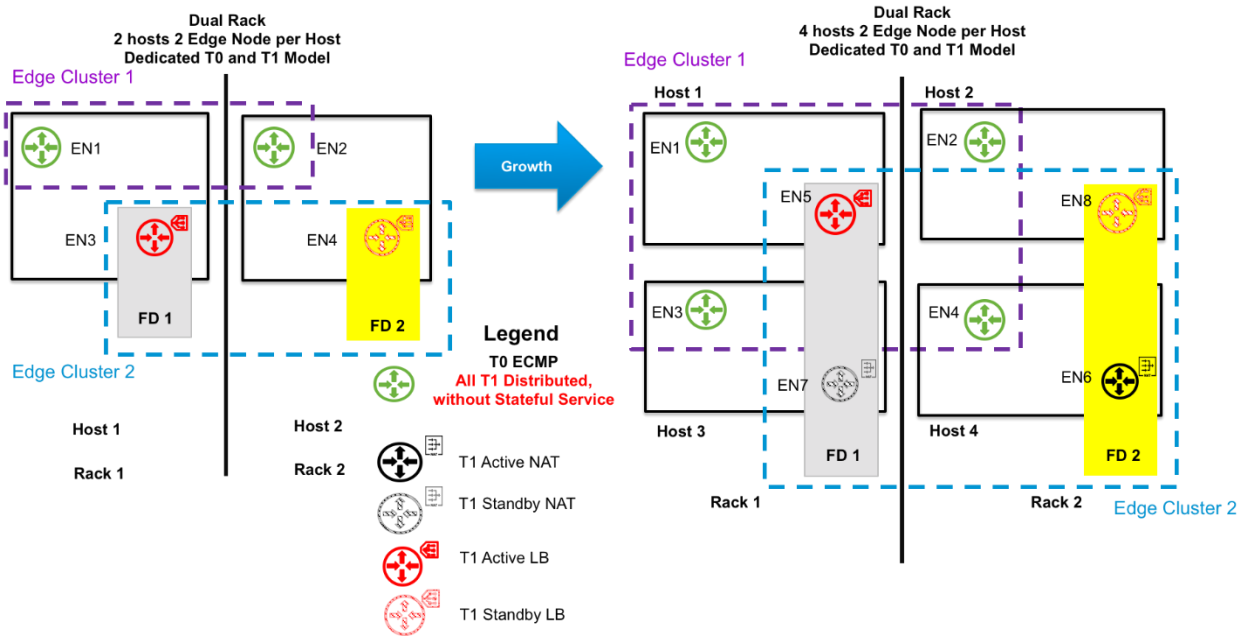


Figure 7-65: Dedicated Services per Edge Nodes Growth Pattern

Finally, the standby edge reallocation capability (only available to Tier-1 gateways) allows a possibility of building a multiple availability zones such that a standby edge VM can be instantiated automatically after minimum of 10 minutes of failure detection. If the Edge node that fails is running the active logical router, the original standby logical router becomes the active logical router and a new standby logical router is created. If the Edge node that fails is running the standby logical router, the new standby logical router replaces it.

There are several other combinations of topologies are possible based on the requirements of the SLA as described in the beginning of the section. Reader can build necessary models to meet the business requirements from above choices.

7.5.4.6 Edge Node VM Physical Host Considerations for Resiliency

As discussed above, there are numerous technical solutions in place to ensure that if/when an Edge Node fails that the services running on that Edge Node are recovered and continued in a reasonable amount of time. That said, there are steps that can be taken when designing the environment hosting the Edge Node VMs to limit the impact of a failure event. An Edge Node VM inherits the resiliency from the underlying platform that is hosting it, it is imperative that the hosting platform is configured in a way to remove single points of failures whenever possible such as:

- Redundant Power and UPS infrastructure
- Redundant Physical pNICs
- Redundant northbound switches
- Fault tolerant storage platform for hosting the Edge Node VM

In addition to removing the single points of failures from the individual host, limiting the potential for cascading failures across a single NSX Edge Cluster is also a key design consideration. Some of the important steps that should be taken to achieve this are:

- Deployment of Edge Nodes in an NSX Edge Cluster across multiple northbound switches preventing a single set of ToRs from taking down all NSX Edge Services.
- Deployment of Edge Node VMs across multiple datastores to prevent a single datastore event (datastore corruption, array availability, etc.) from bringing down multiple Edge Nodes at the same time.
- Configuration of NSX Failure domains (as described in the previous section) where some Edge Nodes do share some single points of failure to ensure the availability of the services in the event of a failure.
- Deployment of additional NSX Edge Node capacity to address the recovery of NSX Services from a failed Edge Node to other Edge Nodes.
- Proactively choose IP Subnets for N/S Peering that can accommodate addition additional T0 Peering IPs in the event that additional Edge Nodes must be brought online to restore additional T0 connectivity.
- Enable Standby relocation for T1 Gateways with services. If the Edge node where the active or standby logical router is running fails, a new standby logical router is created on another Edge node to maintain high availability. If the Edge node that fails is running the active logical router, the original standby logical router becomes the active logical router and a new standby logical router is created. If the Edge node that fails is running the standby logical router, the new standby logical router replaces it.

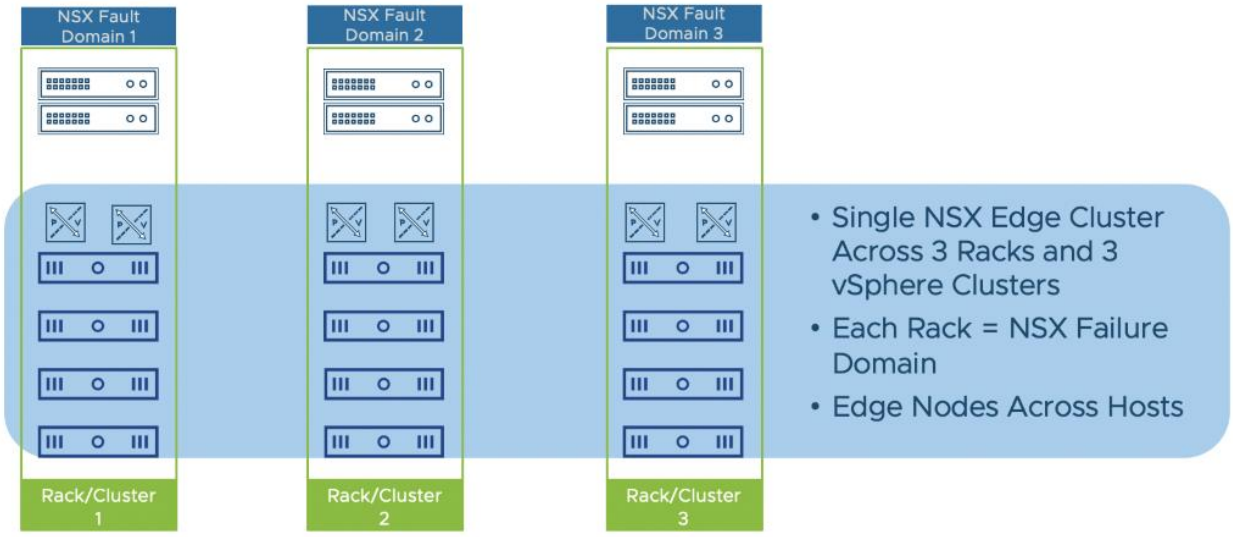


Figure 7-66: Single Edge Cluster

7.5.4.7 Edge Node VMs on VSAN storage

While most of these recommendations can be implemented in any environment, in Hyper Converged Infrastructure or VSAN based environments item number 2 (Deployment of Edge Node VMs across multiple datastores) may become challenging. With VSAN, it is strongly recommended that the following configurations are made:

- Edge Node VMs in a single NSX Edge Cluster should be placed across a minimum of two (2) vSphere Clusters and therefore two (2) or more VSAN datastores. When the NSX Edge cluster is placed across multiple VSAN datastores, it is acceptable, even if not optimal, to implement FTT=1.
- Each Edge Node VM hosted in a single vSphere Cluster using VSAN would then be configured as a Failure Domain in NSX to ensure placement of Active/Standby Services across the two datastores.
- The number edge nodes VMs in a vSphere Cluster used to host the NSX Edges should provide sufficient throughput for the T0 Gateways in the event of a failure of one of the vSphere Clusters.

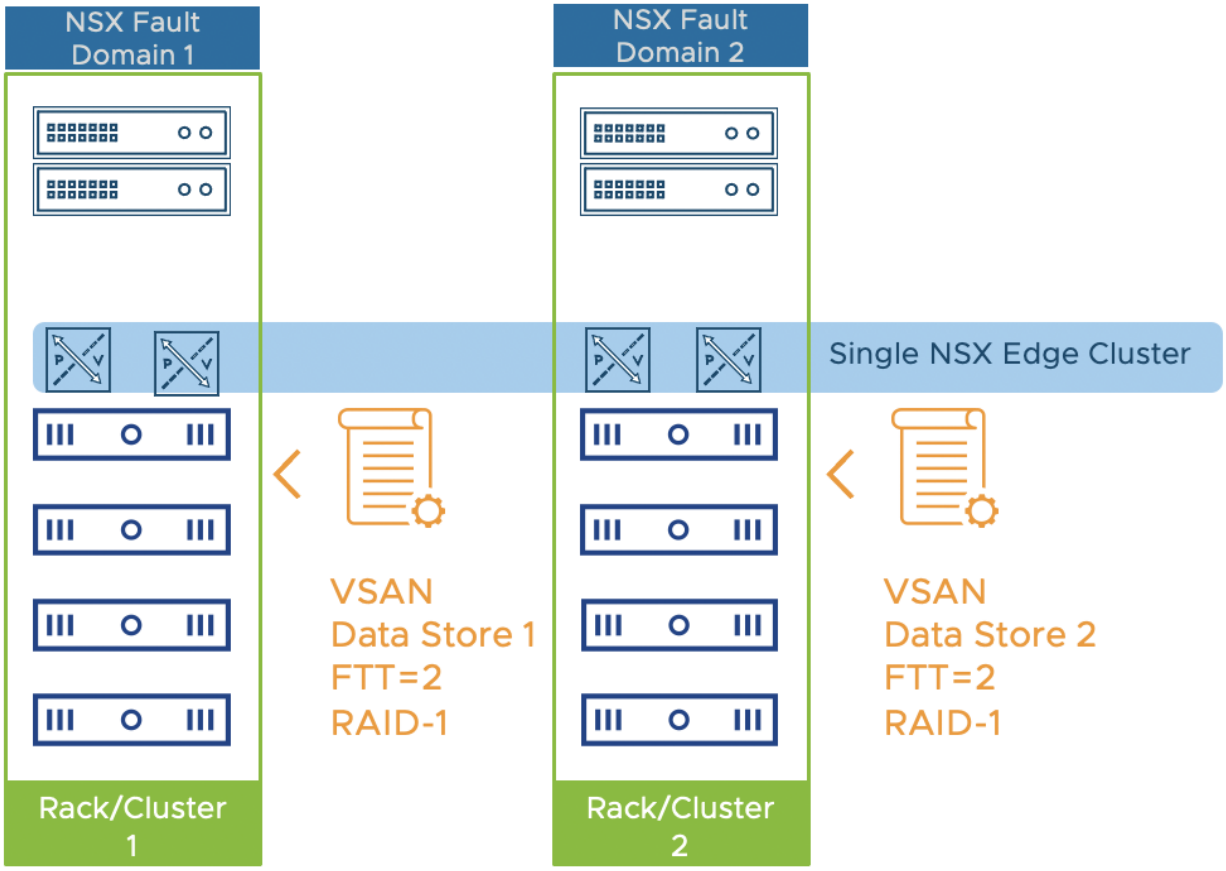


Figure 7-67: Single NSX Edge, FTT=2

If deployment of the NSX Edge Cluster across multiple VSAN datastores is not feasible then the following configurations can be made to minimize the potential of a cascading failure due to the VSAN datastore:

1. At a minimum, the vSphere Cluster should be configured with an FTT setting of at least 2 and an FTM of Raid1
   This will dictate that for each object associated with the NSX Edge Node VMs, a witness, two copies of the data (or component) are available even if two failures occur, allowing the objects to remain in a healthy state. This will accommodate both a maintenance event and an outage occurring without impacting the integrity of the data on the datastore. This configuration requires at least five (5) hosts in the vSphere Cluster.
2. ToR and Cabinet Level Failures can be accommodated as well, this can be accomplished in multiple ways; either through leveraging VSAN's PFTT capability commonly referred to as Failure Domains or VSAN Stretched Clusters and leveraging a Witness VM running in a third failure domain or by distributing the cluster horizontally across multiple cabinets where no more hosts exist in each Rack or Cabinet than the number of failures that can be tolerated (a maximum of FTT=3 is supported by VSAN).
3. When workload is spread across multiple cabinets or ToRs, it is also important to potentially set Host affinities for workloads to better disburse the NSX Edge Nodes across the cabinets and have predictability for which

one is running where and Anti-Affinity rules to minimize Edge Node VMs running on the same host unless specifically designed to do so.

4. It is strongly recommended that Hosts, any time they are proactively removed from service, vacate the storage and repopulate the objects on the remaining hosts in the vSphere Cluster.

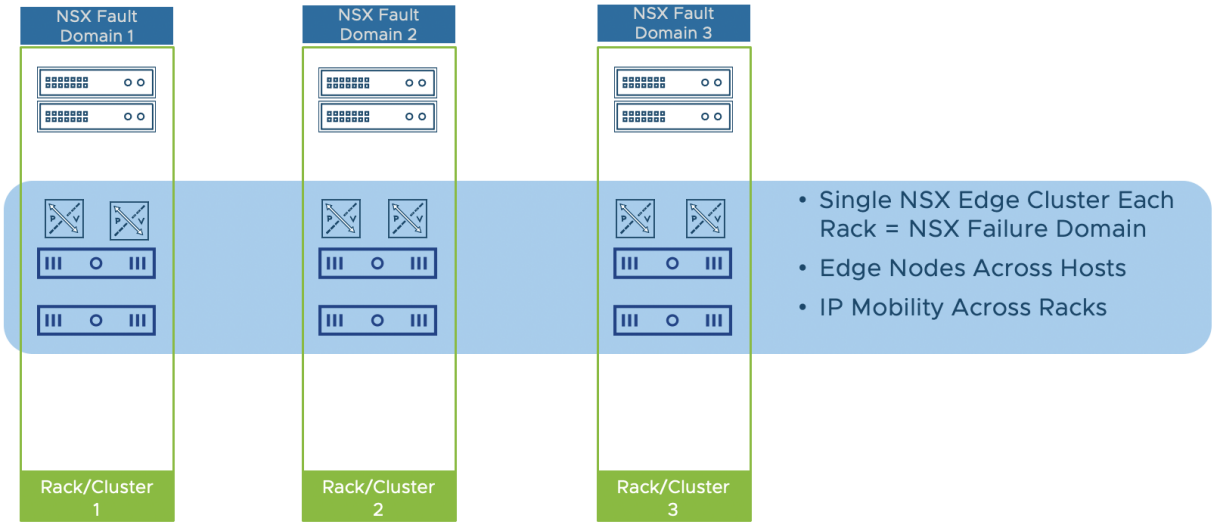


Figure 7-68: Single NSX Edge Cluster, Each Rack NSX Failure Domain

For a realistic example of edge nodes deployed on a single vSphere cluster striped across multiple racks refer to section USE CASE: IMPLEMENTING A REPEATABLE RACK DESIGN FOR A SCALABLE PRIVATE CLOUD PLATFORM IN A LAYER 3 FABRIC

While VSAN Stretched Cluster and other Metro-Storage Cluster technologies provide a very high level of storage availability, NSX Edge Nodes provide an "Application Level" availability through horizontal scaling and various networking technologies. If a dedicated vSphere Cluster is planned to host the Edge Node VMs, using two independent clusters that are in diverse locations as opposed to a single vSphere Cluster stretched across those locations should seriously be considered to simplify the design.

7.6 NSX Platform Design Consideration

NSX enables an operational model that supports compute domain diversity, allowing for multiple vSphere domains to operate alongside as part of the same networking and security solution. A variety of different use cases can be implemented as part of the same architecture, including IaaS (e.g., via vRealize Automation), VDI (via Horizon), and CaaS (e.g., via TKG), or PaaS (e.g., via TAS or OpenShift).

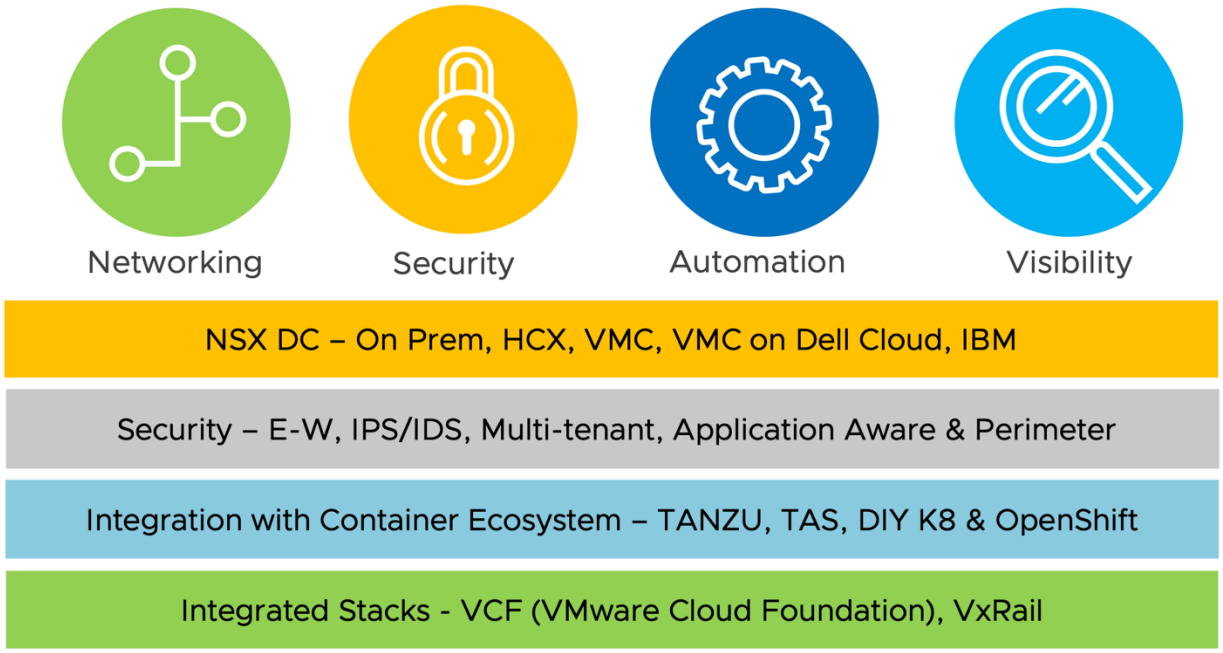


Figure 7-69: Single Architecture for Heterogeneous Compute and Cloud Native Application Framework

There are essential factors to consider when evaluating how to best design these workload domains, as well as how the capabilities and limitations of each component influence the arrangement of NSX resources. Designing multi-domain compute requires considering the following key factors:

- Use Cases
  - IaaS
  - CaaS
  - VDI
  - PaaS
  - Other
- Type of Workloads
  - Enterprise applications, QA, DevOps
  - Regulation and compliance
  - Performance requirements
  - Security
- Management
  - RBAC
  - Inventory of objects and attributes controls

    - Lifecycle management
    - Ecosystem support – applications, storage, and staff knowledge
- Scale and Capacity
    - Compute hypervisor scale
    - NSX Platform Scale limits
    - Network services design
    - Bandwidth requirements, either as a whole compute or per compute domains
- Availability and Agility
    - Cross-domain mobility
    - Cross-domain connectivity

7.6.1 NSX domains design guidelines

It is important to understand that NSX is a component of the overall enterprise architecture. Its role and design are influenced by the requirements of the solutions that consume its services. This section covers the design considerations for the NSX platform as part of a larger private cloud infrastructure initiative, single and multi NSX domain designs, and the different levels of multitenancy.

7.6.1.1 Single NSX Domain

FIGURE 7-70 shows a single NSX installation providing services to multiple independent solutions served by dedicated vCenter servers and vSphere resources. This design provides a unified networking and security model with the following benefits:

Single pane of glass for networking and security

Easy to create security policy spanning multiple compute domains, for example, a policy allowing access from the VDI environment to an application deployed on a Tanzu Kubernetes cluster.

Higher efficiency and resource utilization. Resources can be shared if required.

Easier troubleshooting (e.g., end to end trace flow)

Unified lifecycle management

The single NSX domain solution is the most appropriate for the majority of the real-world customer implementations.

The single NSX domain solution has the following implications:

- ✖ Unified lifecycle management. While it may benefit many situations, specifically smaller deployments, having independent lifecycle management per use case (VDI vs. Tanzu) or environment (Test vs. Prod) allows shorter and less risky maintenance operations. Upgrades motivated by features or bug fixes required by a specific solution (e.g., TKG-S) now affect the whole infrastructure (e.g., VDI), requiring more planning, testing, and an overall higher burden on the operations team.
- ✖ Separation of management responsibility. Different teams may be responsible for different solutions. The organization may be segmented per solution rather than for technology (e.g., the VDI team has NSX expertise, same as the

team managing the Tanzu platform. No single NSX team manages NSX across the different internal IT offerings)

- ✖ Scalability. The NSX platform scalability limits affect all the solutions. Capacity planning becomes more complex as the growth of all the different environments must be taken into consideration.
- ✖ The mixing of manual and programmatic consumption (potentially by multiple plug-ins, integrations, or tools) on the same NSX manager creates the risk of configuration conflicts and errors.

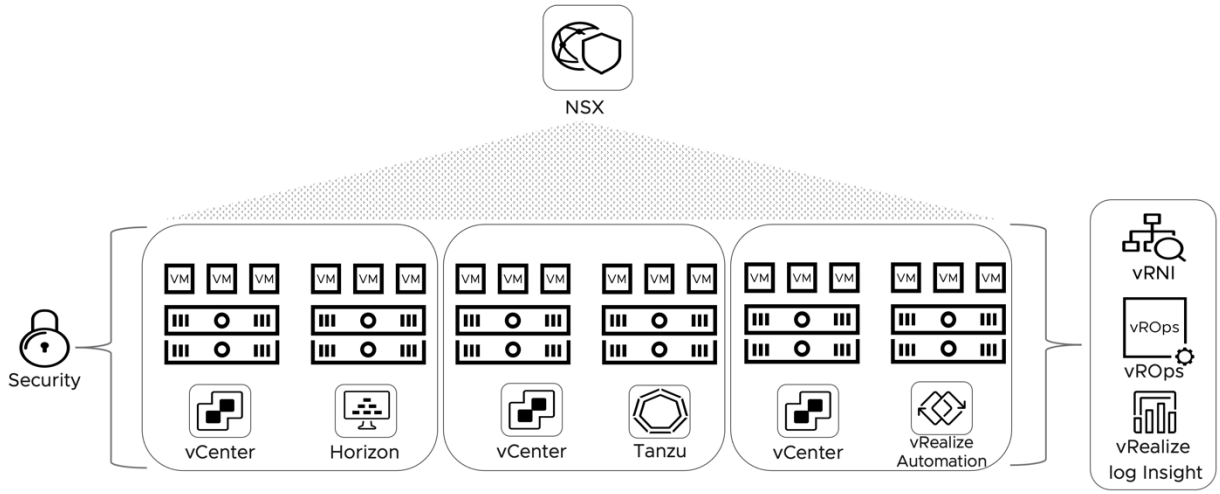


Figure 7-70: Single NSX Domain Design

7.6.1.2 Multiple NSX Domains

This section presents a few designing incorporating multiple NSX domains. In each scenario, the separation is motivated by different drivers. It is essential to carefully evaluate and document the requirements that lead to the break of the NSX design in multiple domains. An excessive level of separation leads to inefficiencies and management overhead when we underestimate the implications on day-to-day operations and lifecycle management.

7.6.1.2.1 NSX Domain per Use Case

FIGURE 7-71 shows a design where one NSX domain is allocated per use case. Examples of reasons for this decision are:

Tanzu environment requires the latest release, and frequent and timely software upgrades are expected

VDI environment runs an internally certified version of NSX and does not require frequent updates. Each new NSX version needs to go through an internal certification cycle.

The IaaS environment includes dynamically created configuration objects (Gateways, segments, DFW rules, Groups) by vRealize Automation. The frequent manual configuration changes required by the security policies in the

VDI environment should be carefully planned to avoid conflicts, and manual configuration errors may lead to applications downtime.

Examples of design implications for the NSX domain separation are:

- ✖ When the administrator creates security policies in the VDI environment to provide access to applications in the IaaS environments, groups based on IP sets must be used. Intelligent grouping is not available for resources outside of the NSX domain. Policies may have to be duplicated in the destination NSX Domain (The duplication may be avoided by creating a wider allow rule on the destination environment permitting the entire VDI pool range. Granular policies can be enforced at the source only).
- ✖ Troubleshooting may be more complex as native tools such as trace flow or live traffic analysis have a single NSX domain scope.
- ✖ Micro-segmentation planning may be more complex as NSX Intelligence have a single NSX domain scope.

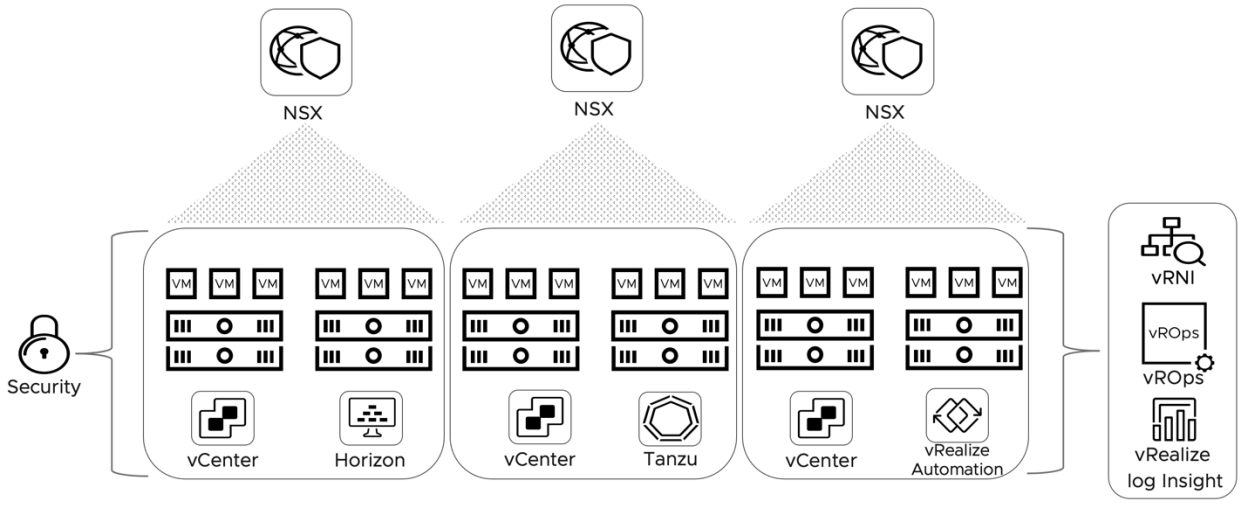


Figure 7-71: NSX Domain per Use Case

7.6.1.2.2 NSX Domain per Environment

FIGURE 7-72 shows a design where the different NSX domains support the same use case, in this example: Horizon VDI. Examples of drivers for the separation decision are:

Test and Production environments have different lifecycle management. New versions and features are validated in the test environment before being adopted in production.

More relaxed change control management in the Test environment

The production environment has been segmented in multiple PODs for scalability reasons.

Examples of design implications for the NSX domain separation in this scenario are:

- ✖ VDI to VDI communication across PODs may require implementing IP Based Groups and not intelligent grouping.
- ✖ Consistent security policies across the entire production environment must be enforced manually, via custom automation, or by implementing NSX Federation.

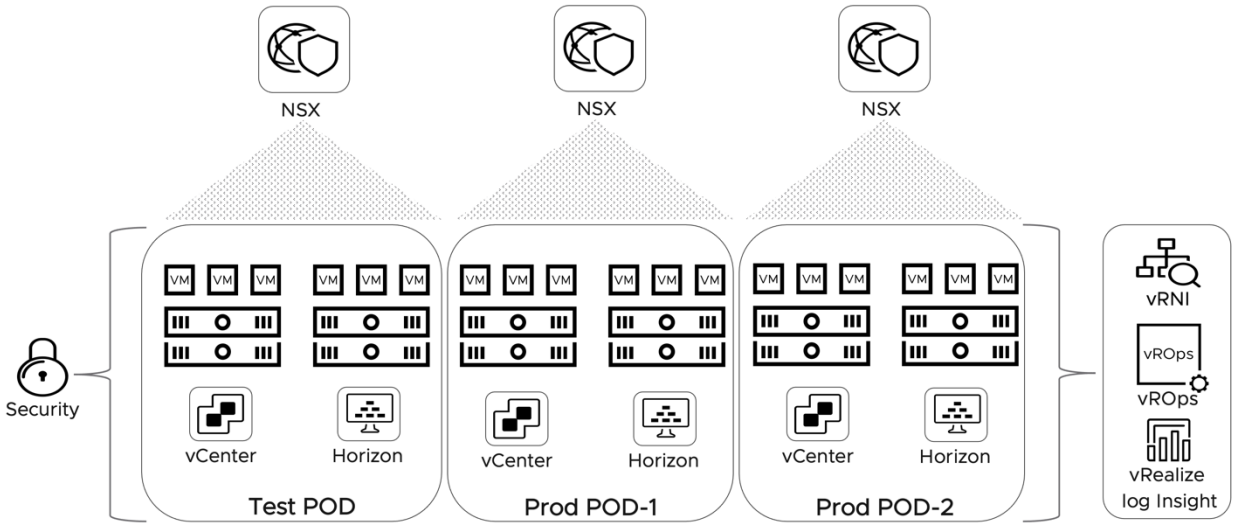


Figure 7-72: NSX Domain Per Environment

7.6.1.2.3 NSX Domain per Tenant

FIGURE 7-73 presents a design where the NSX domain separation is motivated by a multi-tenancy requirement. Different levels of multi-tenancy can be achieved within a single NSX installation with a variety of tools. We can segment different environments very easily by implementing distributed firewall security policies, or we can separate the data plane altogether via tenant dedicated segments, Tier-1s, Tier-0s gateways, or VRFs. Physical resources such as vSphere clusters and ESXi hosts pNICs can be separated by implementing multiple transport zones.

What we cannot provide within a single NSX installation is an effective separation of the management plane in a proper multi-tenant solution. Relying on the native multi-tenancy capability of an interoperable cloud management solution such as VMware vRealize Automation or VMware vCloud director is an option, but such solutions may not expose all the required NSX functionalities or direct access to the NSX GUI/API may be preferred. In such situations dedicating an NSX manager per tenant may be an appropriate solution. Typical examples are VMware Cloud designs such as VMConAWS.

Examples of drivers for the separation decision are:

Management plane multi-tenancy

Independent lifecycle management per tenant

Self-service IaaS solution

Examples of design implications for the NSX domain separation in this scenario are:

- ✖ NSX Manager sprawl. Hard to manage at scale without custom automation.
- ✖ Infrastructure resources overhead in small environments. NSX manager VMs must be deployed for each tenant, even if the environment includes few hosts, at times one in DRaaS use cases. The singleton NSX Manager deployment may alleviate the issue.

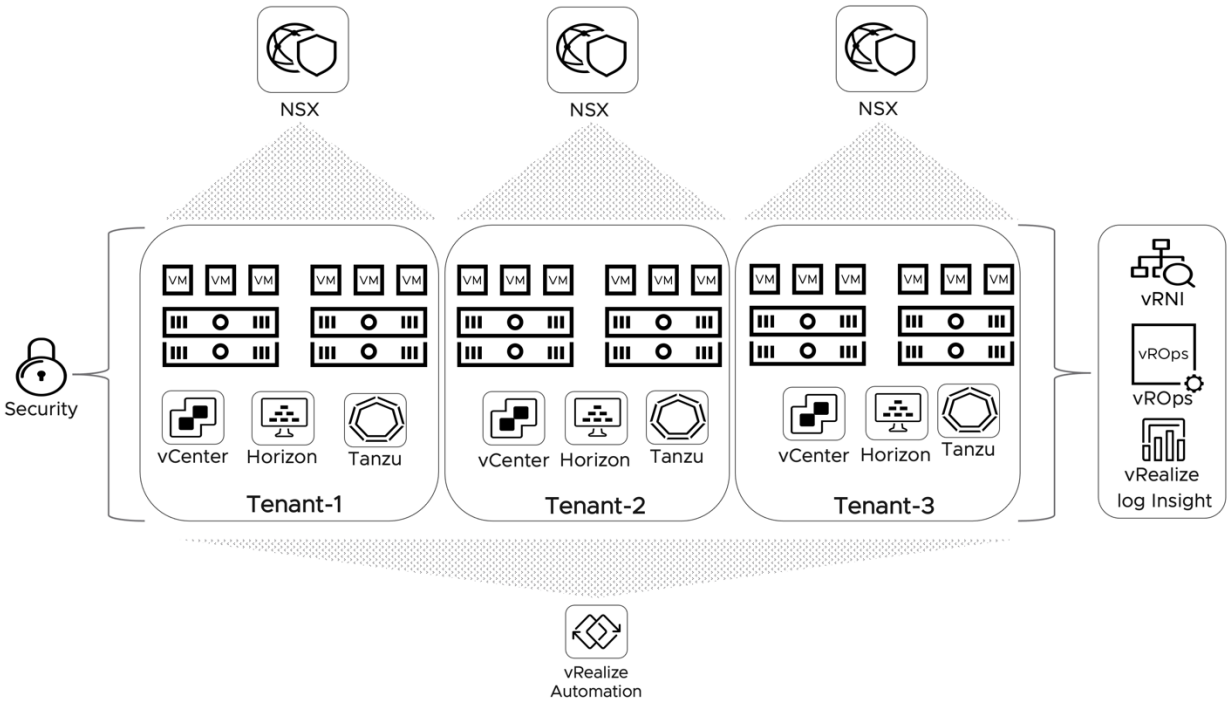


Figure 7-73: NSX Domain per Tenant

7.6.1.3 Single NSX Domain with multiple transport zones

This section presents examples of how the NSX transport zone can be used to segment the infrastructure for different use cases (VDI vs. IaaS) or environments (Test vs. Prod) while providing a unified networking and security consumption model at the management plane level.

Transport zones are not security boundaries and should not be implemented for the purpose of separating workloads at the data plane level. Distributed firewall policies and dedicated routed topologies are the appropriate NSX tools to achieve segmentation.

Transport zones provide a hard boundary for the span of NSX objects such as segments and gateways. They provide a mean to avoid mixing objects specifics to a use case or environment in compute managers or clusters serving other solutions. The implication is that vCenter server domains, or as a minimum, vSphere clusters, should be dedicated to specific use cases or environments and not shared between them. This type of design leads to potential inefficiencies and lower consolidation ratios, but it may be motivated by compliance or risk management reasons.

Edge nodes can be part of a single overlay transport zones. It implies that dedicated edge clusters and corresponding peering configuration to the physical network must be deployed if multiple overlay transport zones are part of the design. If a concern, the multiple peering configuration problem can be solved via a two levels Tier-0 architecture, but the edge node sprawl cannot be mitigated.

7.6.1.3.1 Single NSX Transport Zone Design

FIGURE 7-74 presents a single overlay transport node design catering to multiple use cases. Compared to the multiple transport zone solution, this design has the following advantages:

- Single shared edge cluster deployed on dedicated vSphere resources. Dedicated edge clusters per use case are possible.
- Single Tier-0 gateway and peering configuration to the physical network. Multiple Tier-0 Gateways are possible too.
- Ability to share compute resources between the use cases.

The drawback of the single overlay transport zone design is:

- ✖ Segments and gateways are available for all the use cases or environments. It means that the VDI administrator could mistakenly connect a VDI to a network dynamically created by TKG or vRA. If transport zones separate environments (Test vs Prod), production segments are visible on test vSphere clusters.

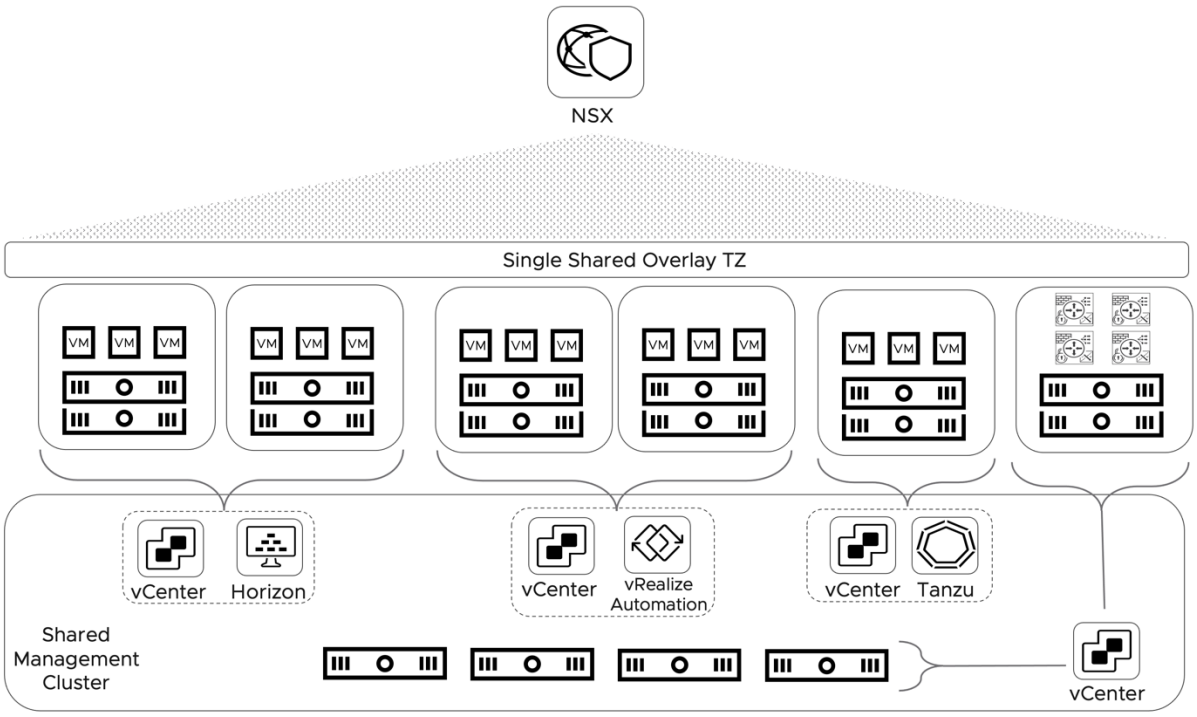


Figure 7-74: NSX Design with Single Overlay Transport Zone Design

7.6.1.3.2 Separate NSX Transport Zones Per Use Case Design

FIGURE 7-75 presents a design where vSphere cluster are allocated to each use case. The rationale for the separation may be based on requirements driving the separation for compliance or lifecycle management. From an NSX perspective, we can map the same separation segmented the vSphere clusters in different overlay transport zones based on the associated use case. The consumption model of cloud automation platforms (e.g. vRA), container plug-in (e.g., NCP in the Tanzu use case), or automation tools (e.g., Terraform) usually assume the existence of a pre-created Tier-0 Gateway and an associated overlay transport zone. When setting up the integration, the administrator pass the Tier-0 gateway, the edge cluster, and Transport zone where dynamically created object should be placed. The automation integration will create segments as part of the specified overlay transport zone and will connect the dynamically created Tier-1 Gateways to the specified Tier-0. This behavior matches the desired resource placement based on vSphere clusters and NSX edge cluster dedicated to each use cases

It's important to emphasize that the design choice to use multiple overlay transport zones is derived by a higher level design decision to segment the compute resources per use case. Implementing multiple transport zones without such requirements leads to unnecessary segmentation of resources, lower consolidation ratio, and it is in general a poor choice.

Each use case requires a dedicated NSX edge cluster because edge nodes can be part of a single overlay transport zone only. The edge nodes can be deployed on the vSphere clusters dedicated to the use case as depicted in FIGURE 7-75. Or they can be deployed on a shared vSphere cluster as depicted in FIGURE 7-76.

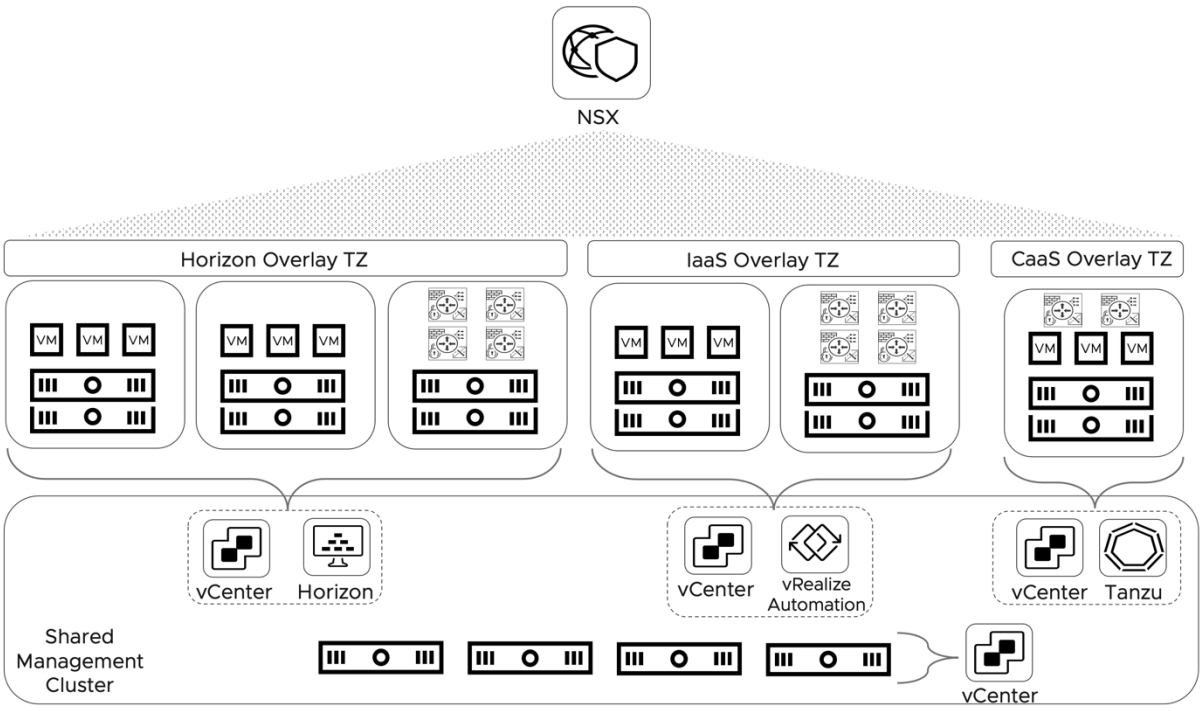


Figure 7-75: NSX Design with Overlay Transport Zone per use case - Edges on Compute Clusters

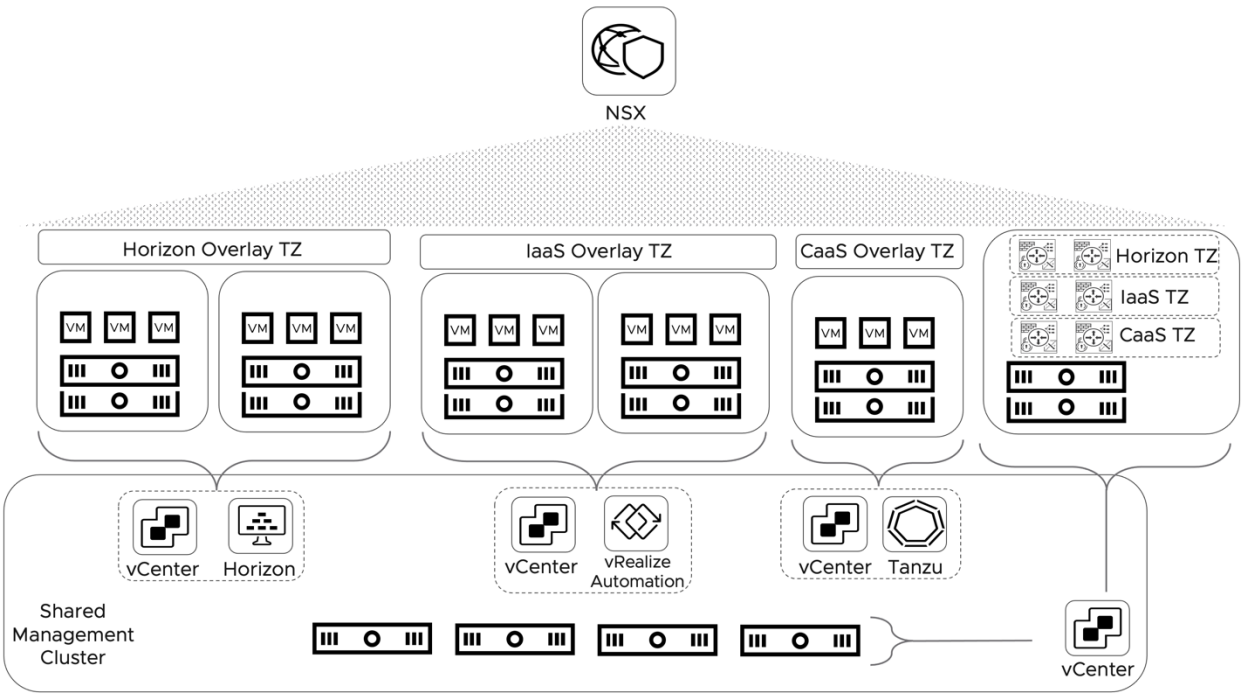


Figure 7-76: NSX Design with Overlay Transport Zone per use case - Edges on Shared Cluster

7.6.1.3.3 Separate NSX Transport Zones Per Physical fabric

Another use case when multiple overlay transport zone can be used is that of dedicated physical fabrics. Again, we have a requirement that dictates the inefficient separation of physical resources, in this case, the physical network, while NSX could efficiently separate workloads and network topologies at the logical level. In the uncommon situation when the separation is dictated at the physical network level but not on the compute host, we can implement the design depicted in FIGURE 7-77. We can map the different physical fabrics to different overlay transport zones and VDSs on the compute hosts. VMs connected to segments on the Internal VDS will use the Internal physical fabric; those on segments on the DMZ VDS will use the DMZ physical fabric. An edge cluster per transport zone is required for physical to virtual connectivity.

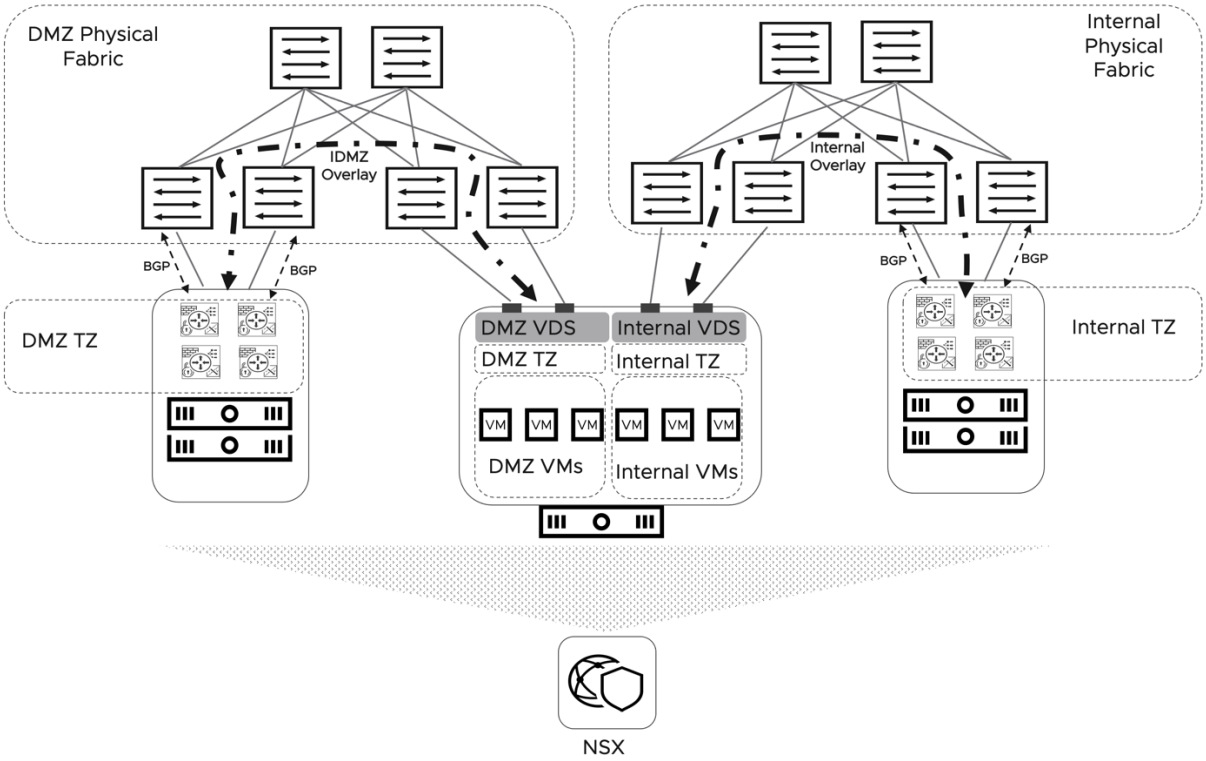


Figure 7-77: Multiple Overlay Transport Zones on the same compute cluster

7.6.2 Network Topologies and Logical Design Guidelines

Outlining an appropriate logical design and the corresponding network topology is dependent on a clear understanding of the platform requirements. The logical design should be derived from a high-level conceptual design that does not include any technological components but presents how the workloads are logically grouped, their dependencies, and the desired level of separation. A simple example of a conceptual design for NSX is presented in FIGURE 7-78. The three different orgs may represent different business units within the same company, different environments (PROD, TEST, QA), different use cases (IaaS, VDI, Cloud Native Apps), or different external customers. Separations based on other criteria are legitimate as long as they are rooted in the platform requirements and the desired operational model.

As we mentioned in the previous section, NSX DOMAINS DESIGN GUIDELINES, excessive levels of separation lead to inefficiencies, lower consolidation ratios, and a more complex operational model. The tenancy model should reflect the required level of separation without limiting the elasticity and the flexibility of the overall solution.

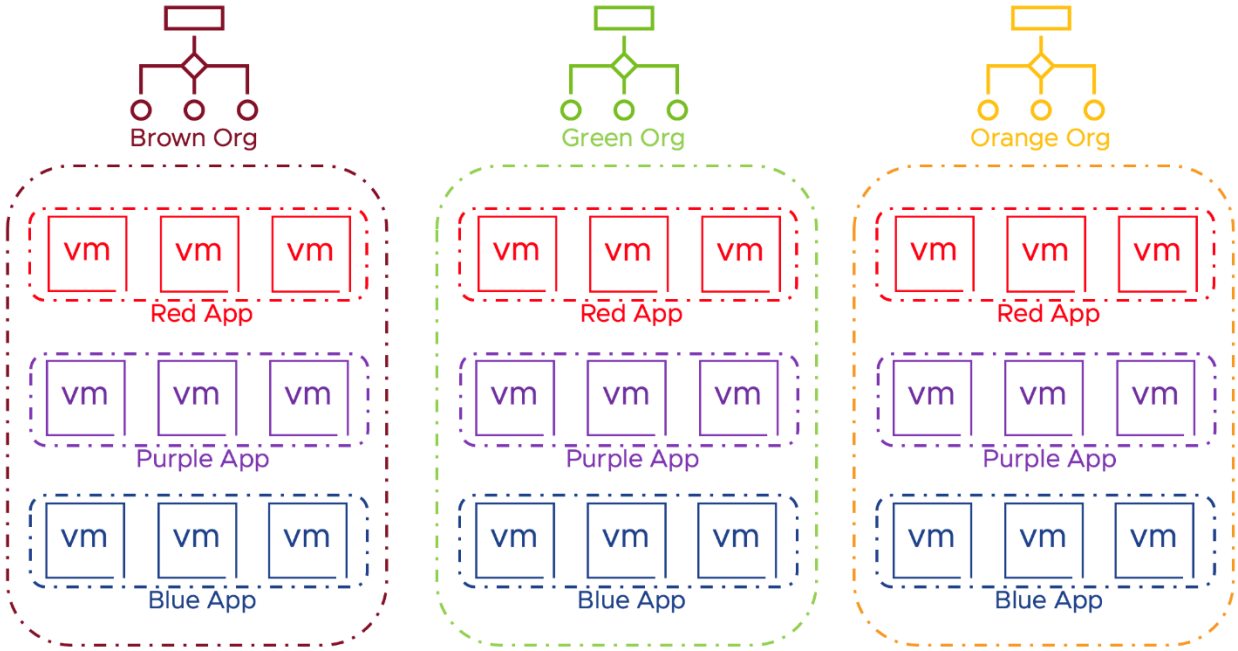


Figure 7-78: Single-Tier Tenancy Model

FIGURE 7-79 presents a more complex multi-tenancy model incorporating a two-tier architecture. This two-tier architecture is embedded in some VMware products (i.e., vRealize Automation) and may be helpful in larger multi-environment, multi-use case deployments. For example, the first level of tenancy (tenant/org) may represent the environment (TEST, PROD, QA), while the second (Project) the use case (IaaS, VDI, etc.).

Note that the complexity of the conceptual design is not dependent on the size of deployment or the number of sites it will encompass. We do not mention larger multi-site environments at this stage. We are still planning our design at a more abstract level. The decision to include a disaster recovery site or multiple active

data centers will depend on non-functional requirements such as the high availability of SLA and RPO/RTO. The design may cater to a single application with very stringent non-functional requirements. In such a case, the conceptual/logical design can be very simple, while the complexity resides in implementing resilient physical components across multiple sites.

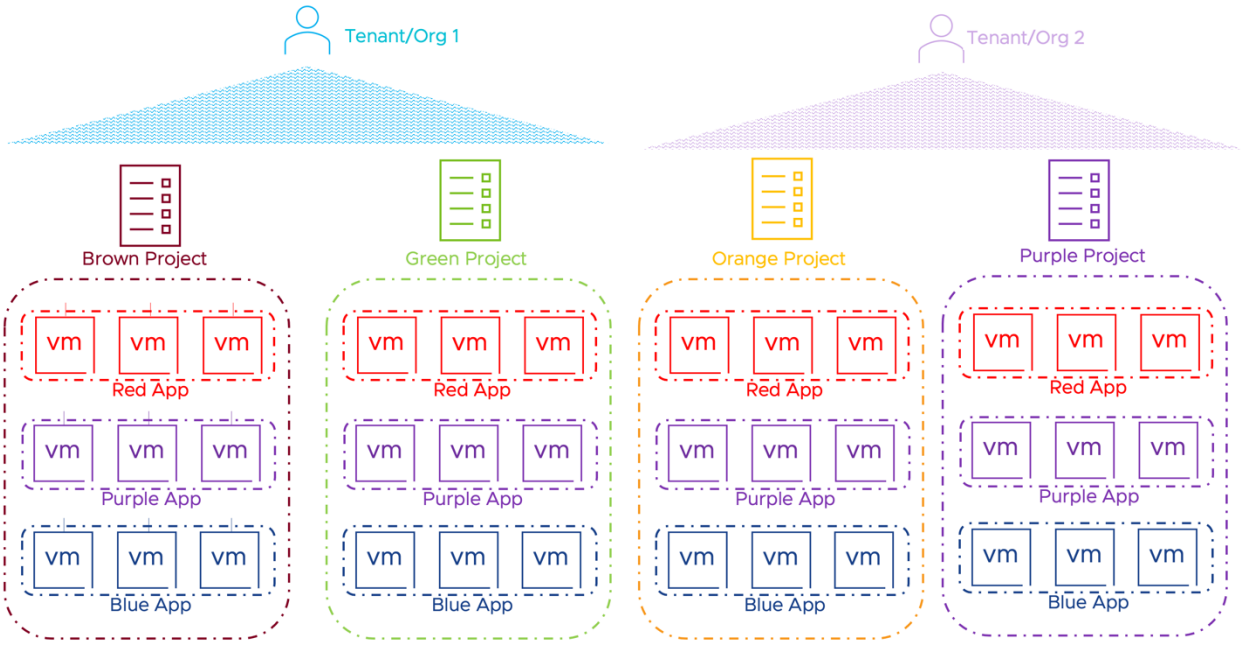


Figure 7-79: Two-Tier tenancy model

Once we have our conceptual model in place, we can now map the abstract elements of the conceptual design to actual NSX components. FIGURE 7-80 presets an example of such mapping, where each tenant is mapped to an independent NSX Domain, projects to dedicated compute managers (vCenter servers) and Tier-0 Gateway, and individual applications, VDI pools, and Tanzu Namespaces to segments and/or Tier-1 Gateways.

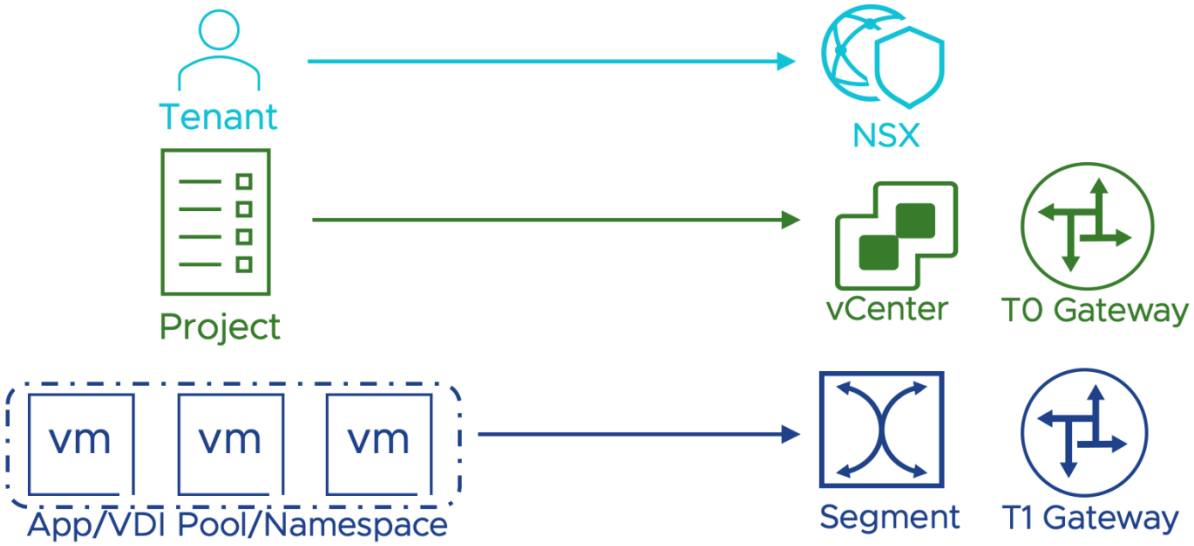


Figure 7-80: conceptual model to logical objects mapping

Let's consider the implications of such mapping in a multi-environment multi-use case design where the first level of tenancy (tenant/org) is mapped to the environment while the second (project) is to the use case. An example of such mapping is presented in FIGURE 7-81.

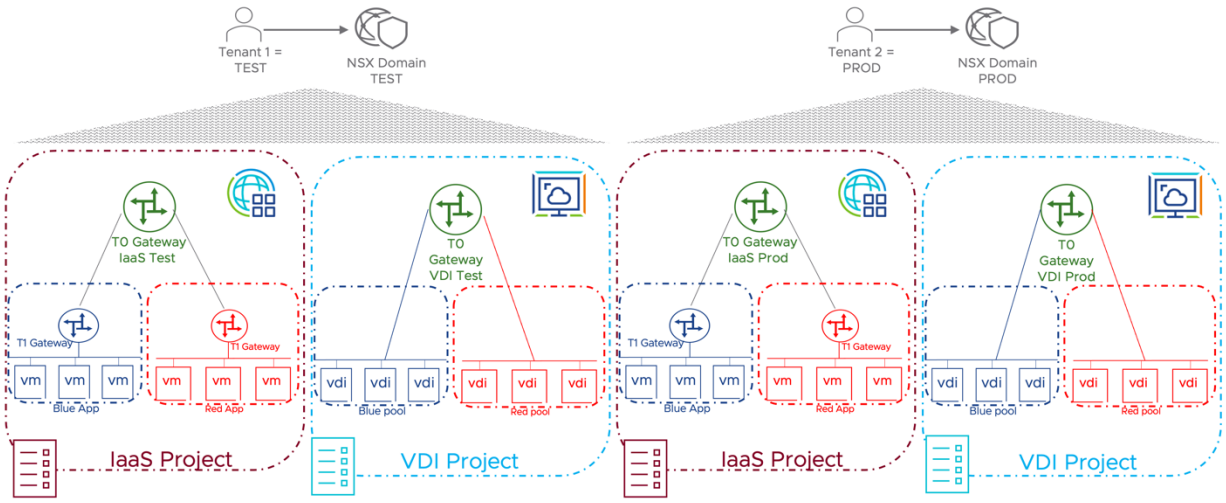


Figure 7-81: Two Tier Multi tenancy design with mapping to NSX constructs

Tenant level considerations:

- The management and lifecycle of each environment are independent
- Complete physical resource separation between environments
- IP ranges are dedicated to each environment.

- Environments may still be able to communicate with each other through the physical network, a gateway or distributed firewall may be used for segmentation. Because the inventory is not shared, IP ranges must be used in the rules providing segmentation.
- Communication between environments is possible, but the management of cross-environment rules is cumbersome because of the lack of a shared inventory (IP-based rules are required). If a high number of flows must be allowed between environments is required, it may mean that the conceptual separation in different tenants was not appropriate. NSX Federation may mitigate the issue.

Project level considerations:

- Each use case has dedicated north/south edge resources, limiting contention.
- Edge nodes are dedicated to each use case.
- Compute managers' and hypervisors' lifecycle is independent between use cases.
- NSX lifecycle is common across use cases
- Tier-0 gateways are created as a day one operation and do not require a frequent lifecycle. They are commonly deployed in Active/Active, and they may be scaled out if needed.

Application-level considerations:

- Each application is deployed on a dedicated segment. This may lead to segment sprawl if the number of applications is high.
- Applications expected to communicate intensively with each other should be placed on segments connected to the same gateway to leverage distributed routing (If the Tier-1 gateways are different, but no services are enabled, routing is fully distributed)
- Segments should be connected to a Tier-1 gateway with services only when they require such services (i.e., per application NAT)
- Segments and Tier-1 Gateways may be dynamically created via automation. A cloud automation platform may manage its lifecycle.
- Services are deployed on the Tier-1 Gateways, not on the Tier-0 Gateway. The only service not available in this topology is L3VPN with redundant remote peering with failover managed by BGP. Non-redundant remote VPN peers are supported on Tier-1 gateways. Dedicated Tier-0 VPN Gateways can be deployed if needed.

Because the previous section's NSX domains design guidelines delved already in the design choices regarding NSX domains, the next few examples cover designs including a single NSX domain.

FIGURE 7-82 presents a design with the following properties:

- A dedicated Tier-0 Gateway for each use case: IaaS and VDI. Performance, scalability, and high availability requirements can be met independently for the two use cases.
- The IaaS Tier-0 gateway is referenced in the cloud management platform (e.g., vRealize Automation), which dynamically creates Tier-1 Gateways and segments for each application deployment. The dynamically created Tier-1 gateways are connected to the IaaS Tier-0 gateway.
- VDI pools are deployed on dedicated segments rather than on a shared one. Distributed firewall security policies can be applied to the VDIs at the deployment time based on the network where they reside without human intervention or custom automation (Segments can be part of groups, both via direct membership and based on tags)
- VDI does not require edge network services. VDI segments are directly connected to an Active/Active Tier-0 gateway.

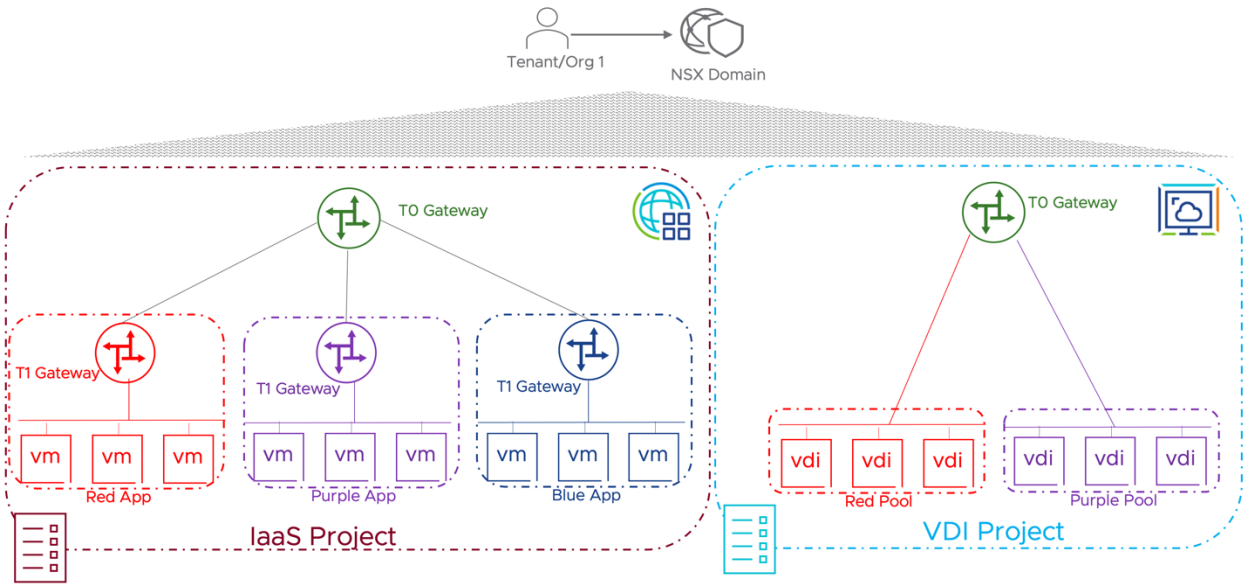


Figure 7-82: Dedicated Tier-0 Gateway per use case

FIGURE 7-83 presents an example where the same use cases are implemented, but the chosen topology is different. The design has the following properties:

- The two use cases share the same Tier-0 gateway. North/South traffic for both goes through the same edge nodes. The edge design must cater to the requirements of both use cases at the same time.
- The shared Tier-0 should be deployed in Active/Active (no stateful services) to allow for scaling out the North/South bandwidth.

- A Tier-1 Gateway is deployed for each use case, but no stateful services are required on them. The separation is mostly logical. It has no impact on the data plane as the routing is fully distributed across the entire deployment.
- Each application or VDI pool is deployed on a dedicated segment
- Centralized networking services are not required. Security is provided via distributed firewall.
- If load balancer services are needed, they can be deployed in one-arm mode.

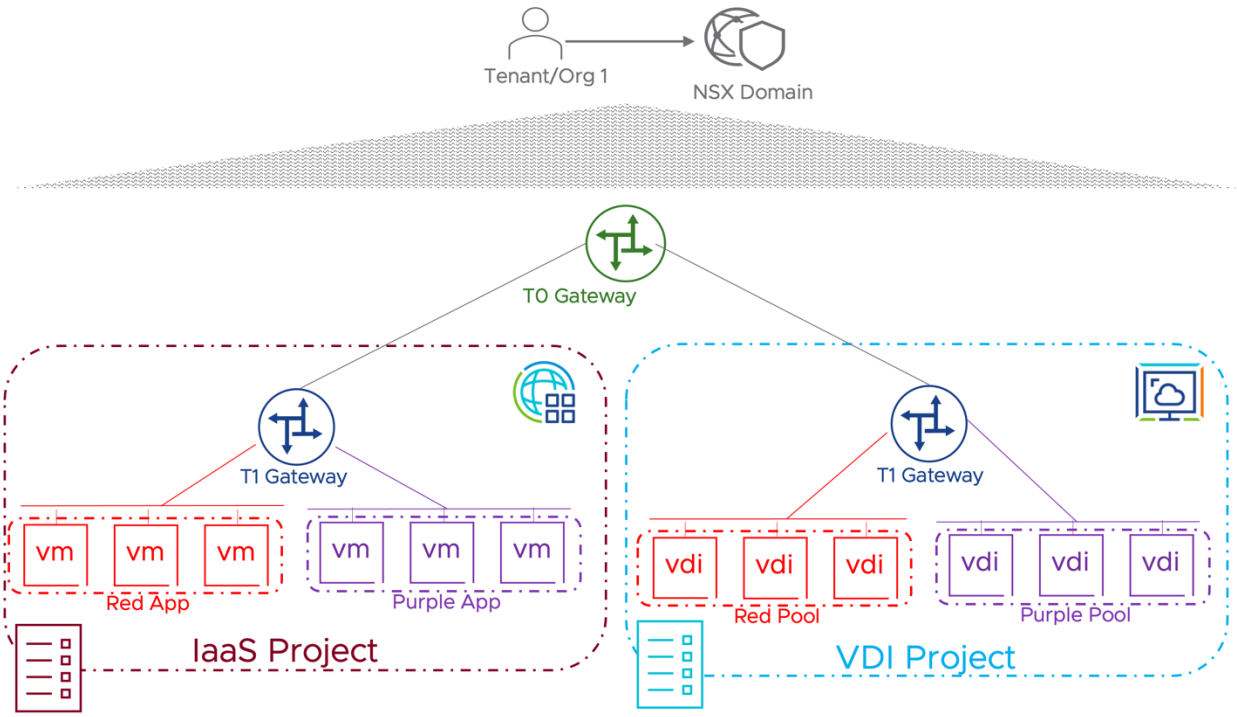


Figure 7-83: Tier-0 Gateway shared across different use cases

FIGURE 7-84 shows an example where the entire NSX Domain is dedicated to a single use case (IaaS). Within the same NSX domain, two environments are present: Prod and Test. The design has the following properties:

- The two use cases share the same Tier-0 gateway. North/South traffic for both goes through the same edge nodes. The edge design must cater to the requirements of both use cases at the same time.
- The shared Tier-0 are deployed in Active/Active (no stateful services) to allow for scaling out the North/South bandwidth.
- Centralized networking services are not required. Security is provided via distributed firewall.
- If load balancer services are needed, they can be deployed in one-arm mode.
- Each environment has a dedicated segment and corresponding IP range.

- When an environment grows beyond a single segment limit, segments can be added and connected to the same Tier-0 gateway.
- Segmentation between environments is provided via DFW based on the IP ranges associated with each environment. Such policies are manually created by the administrator.
- The CMP (e.g., vRA) sees the segments as external networks. It means that those segments must be pre-create by the NSX admin, referenced in the vRA configuration, and mapped to the two different projects. vRA will place the dynamically created VMs on those segments based on the environment. Because of the segment placement, the VMs will inherit the environment segmentation based on IP ranges implemented via DFW.
- The separation between applications within an environment can be achieved via dynamically created groups and security policies managed by the CMP.
- Larger, shared, and manually configured segments facilitate the integration with disaster recovery orchestration tools (e.g., VMware Sire Recovery Manager) compared to segments dynamically created by the CMP. Network mappings between sites can be configured in advance and do not necessitate frequent updates.

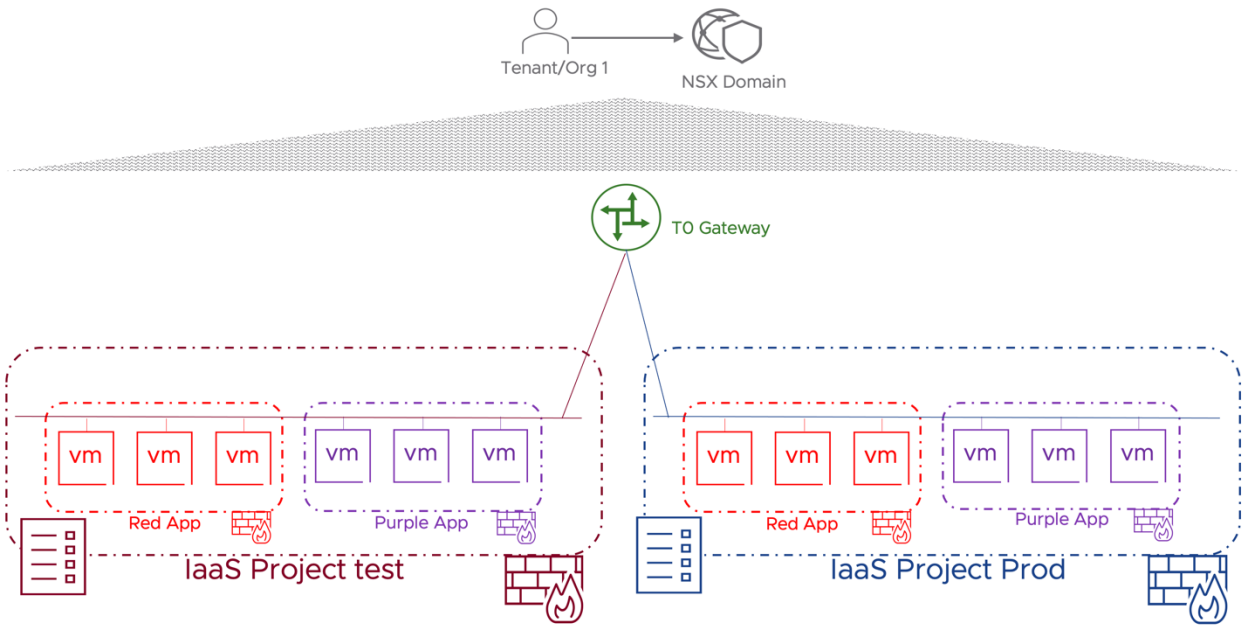


Figure 7-84: Pre-created segment per environment

7.6.3 NSX Components Placement in a vSphere Deployment

This section covers the different possible arrangements of the NSX Components across a vSphere deployments. An NSX deployment may include multiple NSX Domains which may follow different models regarding the components' placement. Each NSX domain has its own specific requirement that will drive the design decisions around component placement.

7.6.3.1 Dedicated management, edge, and compute vSphere clusters

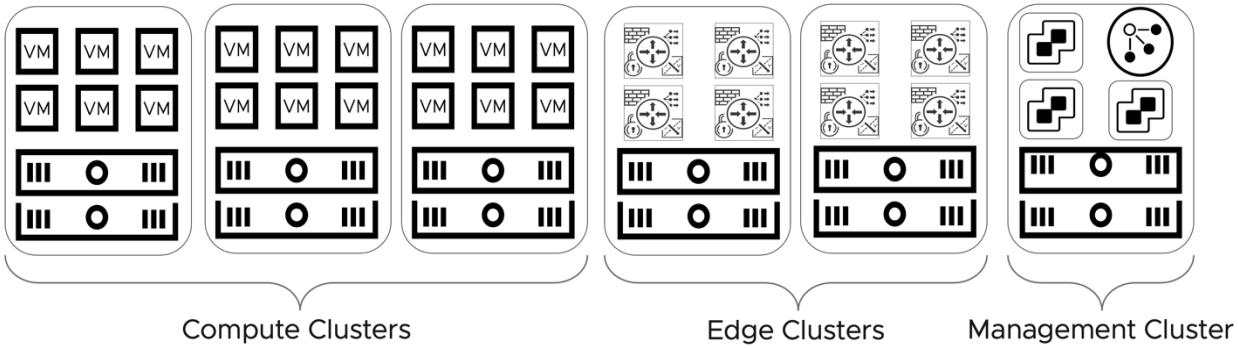


Figure 7-85: Dedicated management, edge, compute clusters

This model is the most scalable and provides less resource contention for the NSX components.

7.6.3.1.1 The Management Cluster

The management cluster host the NSX manager appliances and all other infrastructure management components like the vCenter for all the managed compute domains, the cloud automation platform, and the monitoring and operations components. These components have predictable resource requirements that should not dramatically change over time if the design accounts for the environment's expected growth. Some of the components in the management cluster (e.g., vCenter, NSX Manager) require management VLAN connectivity as they cannot be placed on the overlay. For this reason, management clusters are confined to a single rack if the physical fabric doesn't allow the stretch of VLANs across racks. Multiple management clusters can be deployed, and the three NSX Manager appliances can be distributed across them for extra redundancy even when layer three boundaries exist between racks. The vCenter server deployment, even when deployed in HA, is confined to a single layer two domain allowing vSphere HA to recover it in case of a failure. Most of the other components generally placed in the management cluster can be deployed on overlays to allow for rack mobility but check the documentation for each management component you are considering placing on an overlay network.

The management cluster can be prepared for NSX if NSX Distributed firewall should protect management workloads. NSX Manager itself cannot be protected via DFW and should be protected via an external firewall (NSX edge can be used). NSX Manager should be placed on a dvpg managed by vCenter and not on a VLAN segment to avoid circular dependencies. When the management cluster is prepared for NSX, overlay networks can be consumed by the management appliances part of the vRealize suite, but not by vCenter and NSX Manager.

7.6.3.1.2 The Edge Cluster

The vSphere edge clusters only host NSX edge nodes and in most cases, they are not prepared for NSX (The ESXi are not transport nodes). NSX Edge nodes are purpose-built appliances with unique resource requirements compared to other workloads in the data center. Dedicating hardware resources to the edge nodes is highly recommended, and in particular, it is recommended to have a specific hardware design for the ESXi hosting the NSX Edge nodes. This kind of customization has the drawback of having a non-uniform hardware footprint in the data center and should only be embraced based on strong and well-defined requirements for the edge services (see EDGE SERVICES REQUIREMENTS).

Servers hosting edge nodes dedicated to North/South traffic are often constrained by their pNIC capabilities and configuration before the CPU or memory constrains them. Using the same generic hardware platform for both workload cluster and edge cluster (e.g., 2x25G pNICs with 512GB memory and 64 core count) will most likely lead to underutilized edge vSphere hosts from a CPU and memory perspective (e.g., a balanced host for N/S edge services would have 4x25Gbs pNICs, 32 cores, 128 GB Memory running two Large Edge Node VMs). In all practical deployments 4 pNICs design brings the most flexibility and economics to start a design which may use limited Edge resources initially and eventually lead to dedicated host for Edge.

Additionally, Edge nodes performing advanced security services are often limited by the CPU resources of the ESXi host. If the edge VM vCPU is maxed out, we can deploy additional edge VMs on the same host assuming available resources are available. Deploying a mix of north/South and service edge node VM on the same set of hosts may lead to better CPU utilization, but it's hard to overcome pNIC and memory designs that were not planned for the specific requirements of the edge nodes. At the same time, planning a dedicated edge cluster without well-defined requirements for the edge services will lead to future resource constrain and lack of flexibility and elasticity to address the likely asymmetric growth of the workload requirements. Requirements are the most important part of any design. When they are weak or completely absent, a generic platform incorporating some level of sharing for the edge resources can be the most proper course of action. One such starting point is a collapsed edge/compute model which optimizes the available resources and yet remains open to expansion based on the above guidelines.

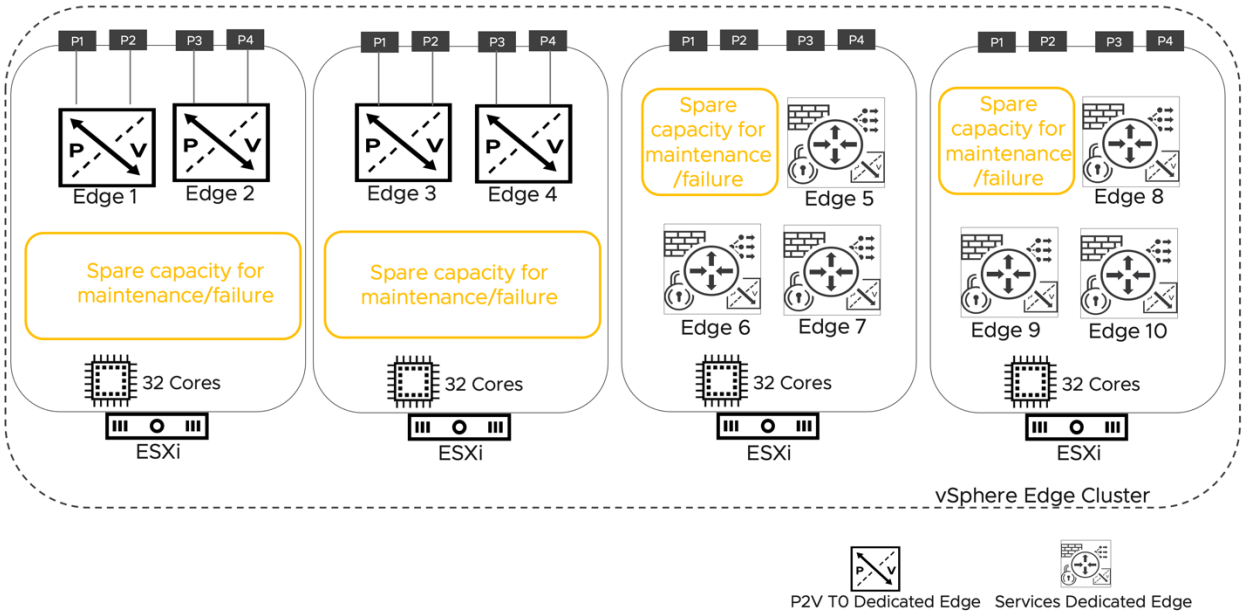


Figure 7-86: vSphere Edge Cluster - Dedicated Hosts to P2V and Services edges

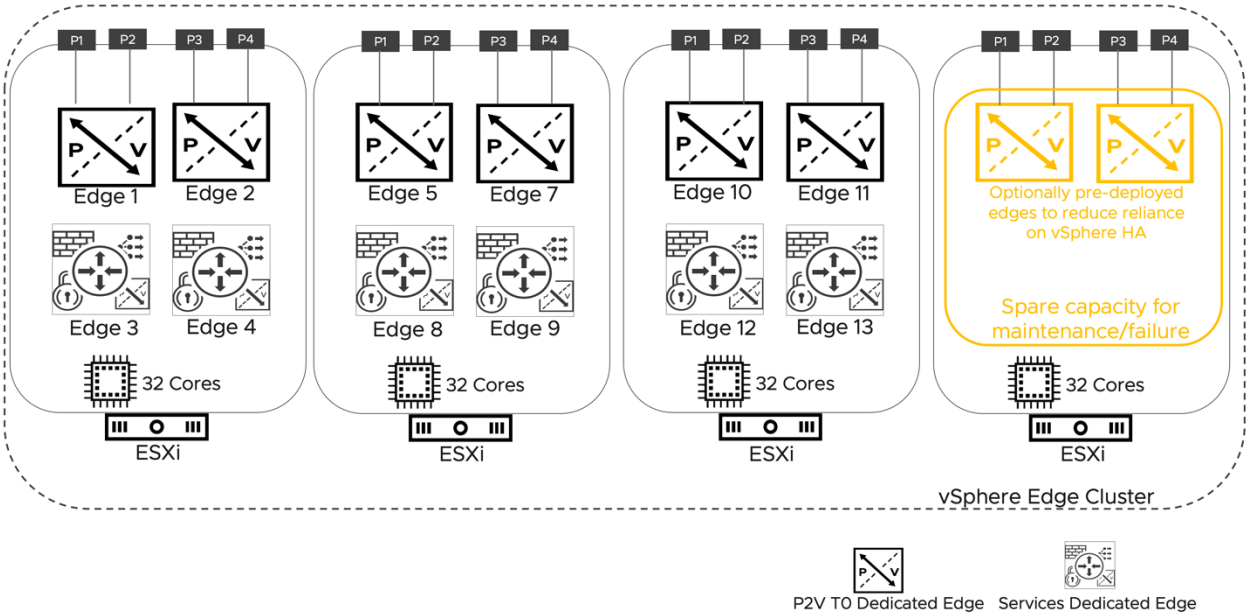


Figure 7-87: vSphere Edge Cluster - Shared Hosts to P2V and Services Edges

vSphere edge clusters demand specific configurations on the leaf switches to where they connect. Specifically, they require VLANs for edge management, edge overlay, and edge routing peering. If we stripe the vSphere edge cluster across different racks, we must extend those VLANs as well. NSX edge clusters can be deployed across multiple vSphere clusters, and they do not require layer two connectivity between them. In a layer 3 fabric, it is recommended to keep the vSphere edge cluster confined to a single rack and deploy NSX Edges in different vSphere clusters if rack availability is required. See FIGURE 7-88.

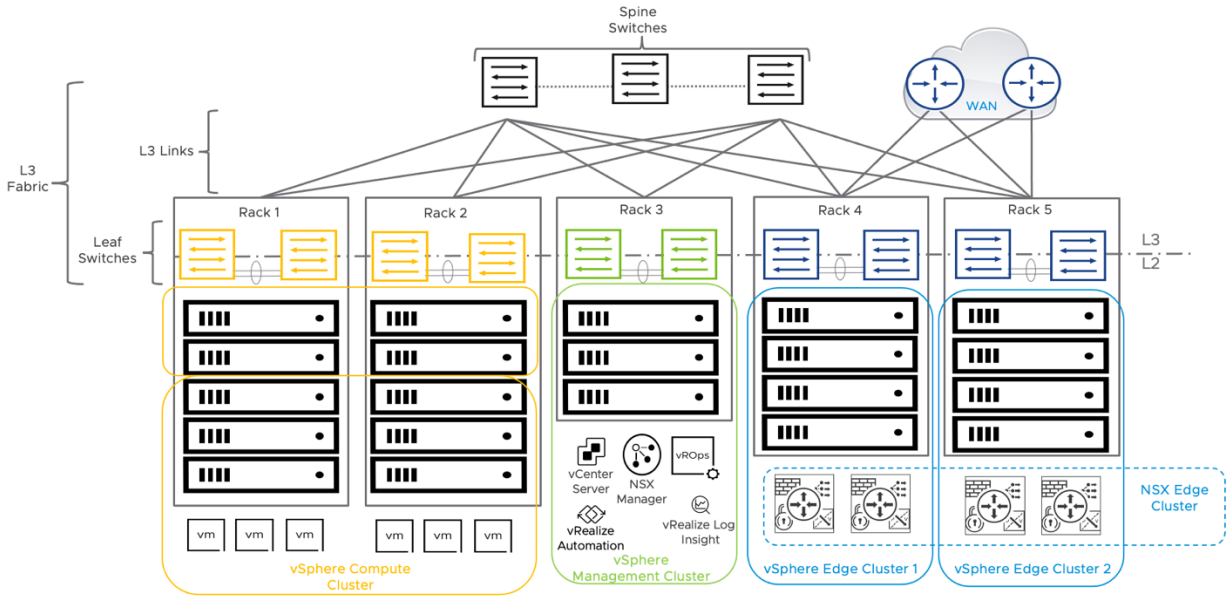


Figure 7-88: NSX Edge Cluster deployed across 2 different vSphere clusters in different racks

7.6.3.1.3 The Edge Cluster (DRS, vMotion, vSphere HA)

Edge Node VMs support vSphere vMotion, DRS, and vSphere HA. While those vSphere functionalities provide great value to the NSX infrastructure in general, the benefit they provide to edge node VMs depends on the use case implemented and the preferred operational model.

ECMP Tier-0 Only Edges

vSphere HA

FIGURE 7-89 presents a scenario where the edge VMs are deployed for North/South traffic only, with a single Tier-0 Gateway deployed across them (Top of the diagram). If the ESXi hosting Edge 1 and edge 2 fails, the two edge VMs will be recovered on the second ESXi that is already hosting Edge 3 and Edge 4 (Bottom of the diagram). The four edge nodes VMs are now competing for the host resources (CPU and pNICs) with potentially little or no benefit from a performance perspective. If the hosts are designed to support four edge node VMs, wouldn't it make more sense to deploy four edge node VMs per host in the first place? If the design already includes eight edge node VMs and the ESXi hosts have available spare capacity, vSphere HA provides a benefit as we can achieve the same level of performance when the system is in a degraded state (One ESXi fails or is in maintenance mode).

Protecting edge node VMs is generally not recommended when the appropriate resiliency level is already in place at the edge cluster level (additional edge VMs running on different ESXi hosts), but it provides the benefit of maintaining the same vSphere HA configuration across the entire compute environment and a consistent failure recovery mechanism.

DRS and vMotion

We recommend setting DRS to fully automated and use DRS should rules to have deterministic placement of the edge VMs during normal operations while allowing edge VMs to move during ESXi maintenances.

From an operational perspective, disabling DRS (or setting it to anything other than Fully Automated) requires manually moving or powering off the edge VMs before placing the ESXi host in maintenance mode. The design choice is between control and predictability vs. automation of the lifecycle management that comes with vSphere. When DRS is enabled for edge node VMs hosting a T0 gateway, we want to limit unnecessary migrations that can cause interruptions to the north south traffic. Implementing VMs to Host should rules allows to specify on which host the edge VM will run. DRS will not migrate the edge VM unless the host is placed in maintenance mode or extreme resource contention occurs.

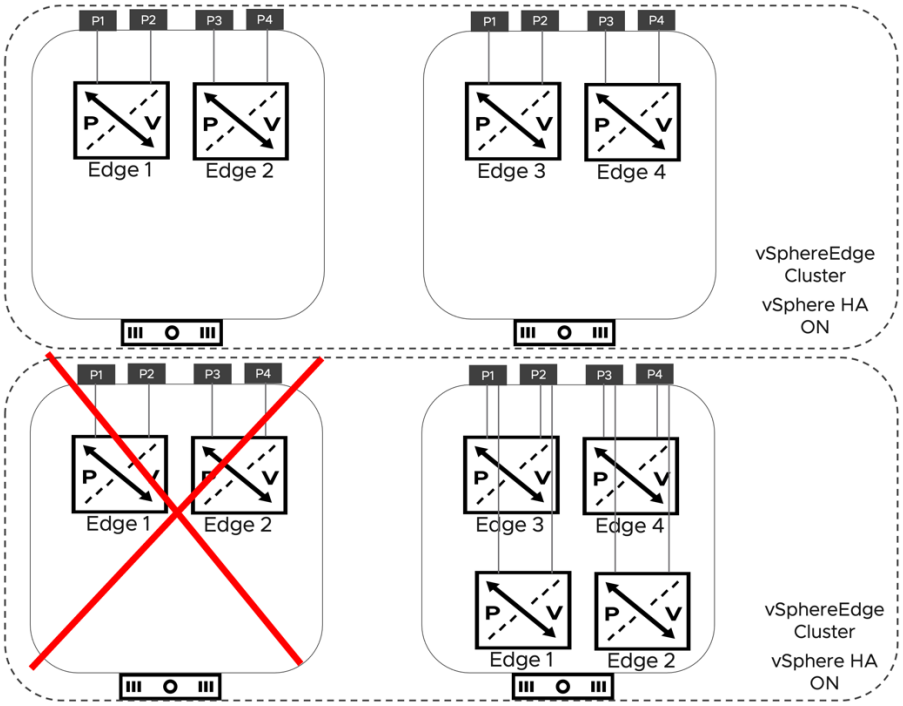


Figure 7-89: vSphere Edge cluster and vSphere HA for P2V Edges

Services Only Edges

When edge nodes are dedicated to services, the CPU is the most critical host resource. The underlying vSphere cluster should be designed to provide enough resources to support the required availability model (e.g., N+1, N+2,…). vSphere HA admission control can be used to ensure that the vSphere cluster is not overprovisioned and unable to support a host failure without experiencing degraded performances. vSphere HA should always be enabled for edges running services, even when in conjunction with an Active/Active Tier-0 gateway, as it allows to recover the lost capacity and redeploy the standby SRs before the standby relocation timeout kicks in.

DRS in fully automated mode provides efficient utilization of the edge cluster resources. NSX places the T1 Gateways with services across different edge node VMs, but it does not consider the traffic or the services they provide. As a consequence, some edge nodes can run hotter than others. DRS can help balance the utilization across the cluster. As mentioned, Edge node VMs support vMotion, but it is recommended to set the DRS threshold to a conservative value to avoid

too frequent vMotion operations. The network interruption caused by vMotion, even if it is extremely brief and hardly disruptive, will impact many workloads when affecting an edge VM. Some applications may not be affected at all, and others may be. DRS rules, in conjunction with NSX failure domains, should be in place so that Active/Standby Gateways do not reside on the same ESXi host during normal operations and maintenance windows (Anti-affinity rules).

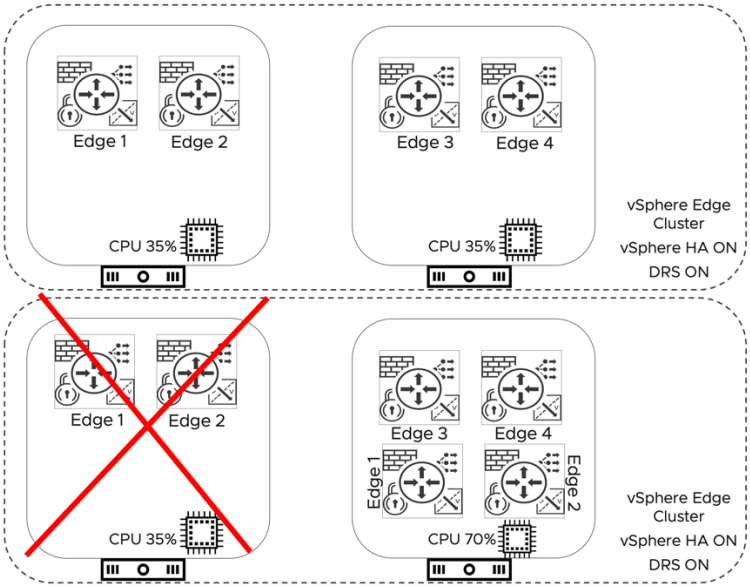


Figure 7-90: vSphere edge cluster - vSphere HA and DRS for Service Edge VMs

With a single vSphere edge cluster hosting independent NSX Edge cluster dedicated to different use cases (e.g., p2v and services), it's possible to customize the vSphere HA and DRS setting per edge VM to achieve different behavior per use case. When the NSX edge cluster is shared (e.g., the same edge node VMs provide p2v and services), the design decisions around vSphere HA and DRS setting are more complex and should be based on what design properties are the most important.

7.6.3.2 Collapsed Management/Edge cluster

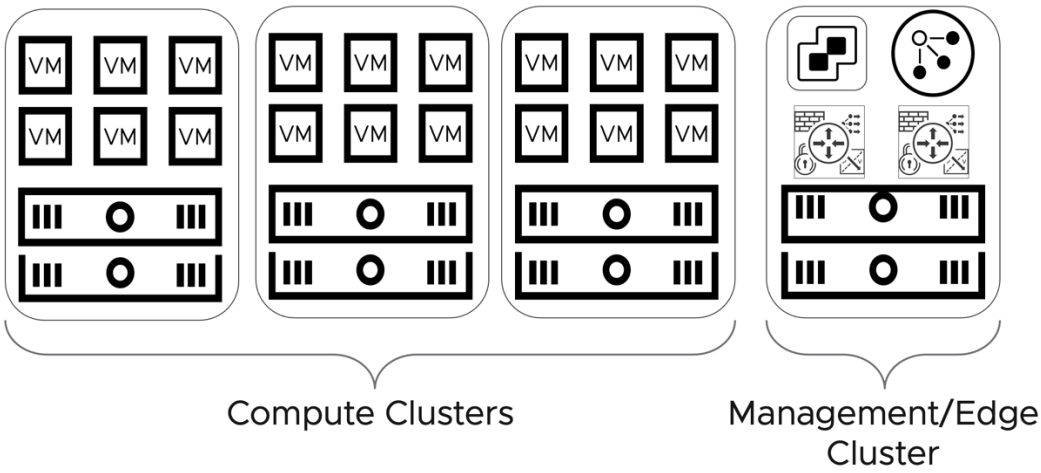


Figure 7-91: Collapsed Edge/Management cluster

When limited compute resources are available, or the footprint of the management and edge components is limited, collapsing those components in the same vSphere cluster is an option. When dedicated clusters are not an option, collapsing edge and management functions is usually the preferred option because of the predictable footprint and growth of the management components compared to those of the application workloads.

Mixing management and edge components in smaller or simpler deployments is not an issue as long as the requirements for both workload types are taken into consideration when designing the hardware resources in the cluster. For example, the four nodes management/edge cluster depicted in FIGURE 7-92 can be adequate to host essential infrastructure management components and edge nodes dedicated to North/South traffic and minimal services. Having dedicated pNICs for the edge node VMs hosting the tier-0 Gateway is always a good idea. Host traffic (vMotion, storage) should share the pNICs used by the management appliances rather than those used by the edge VMs. Deploying a management/edge cluster with two pNIC hosts is supported, but it will most likely represent a compromise in terms of north/south performances.

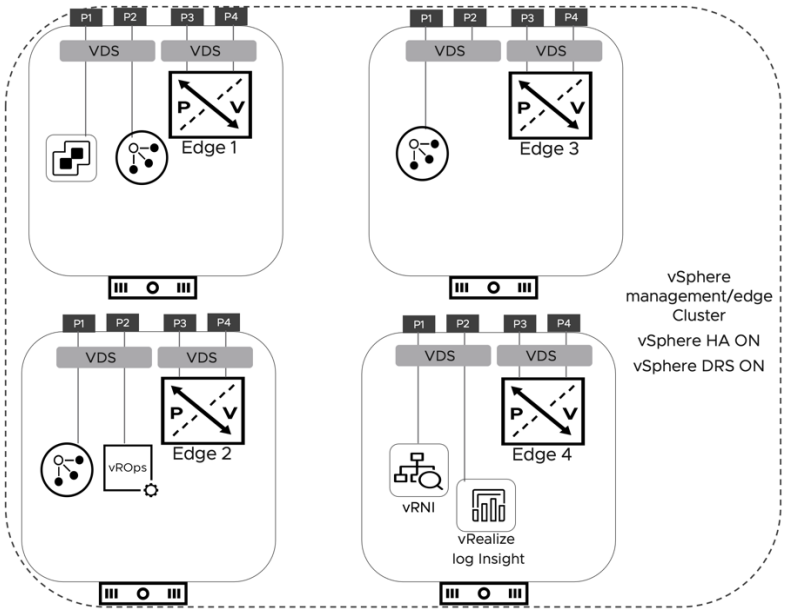


Figure 7-92: Dedicating pNIC resources to edge VMs in management/edge cluster

Key considerations when deploying a collapsed management/edge cluster:

- The management/edge cluster must support management and edge VLAN requirements
- Expected mobility across racks might be different for management and edge workloads
- Resource contention should be managed by resource planning and vSphere reservations
- Separating management and edge traffic on different pNICs should be considered
- vSphere HA is most likely enabled globally for the cluster to support management workloads. If vSphere HA for T0 edge VM is not desired, it must be disabled manually (In case the recovery of the edge VM on a different host will not lead to gains in high availability or performances). vSphere HA for T1 only service or T0/T1 mixed edges is a good practice in most situations.
- DRS is most likely enabled for the cluster to support management workloads. VM to Host Should rules should be implemented to enforce the correct pNICs to T0 edge VM placement and limit the T0 edge VM mobility to host maintenance events.
- Limit over reservation of CPU/memory resources for the management workloads as it may lead to DRS moving the edge VMs regardless of the preferential rules.
- DRS rules, in conjunction with NSX edge failure domains, should be in place so that Active/Standby Gateways do not reside on the same ESXi host during normal operations and maintenance windows (Anti-affinity rules).

Refer to the dedicated sections for management components and edge node connectivity for detailed guidelines in term of virtual switch design and teaming policies. The diagrams below present a detailed view of such configuration settings in a two and four pNIC designs.

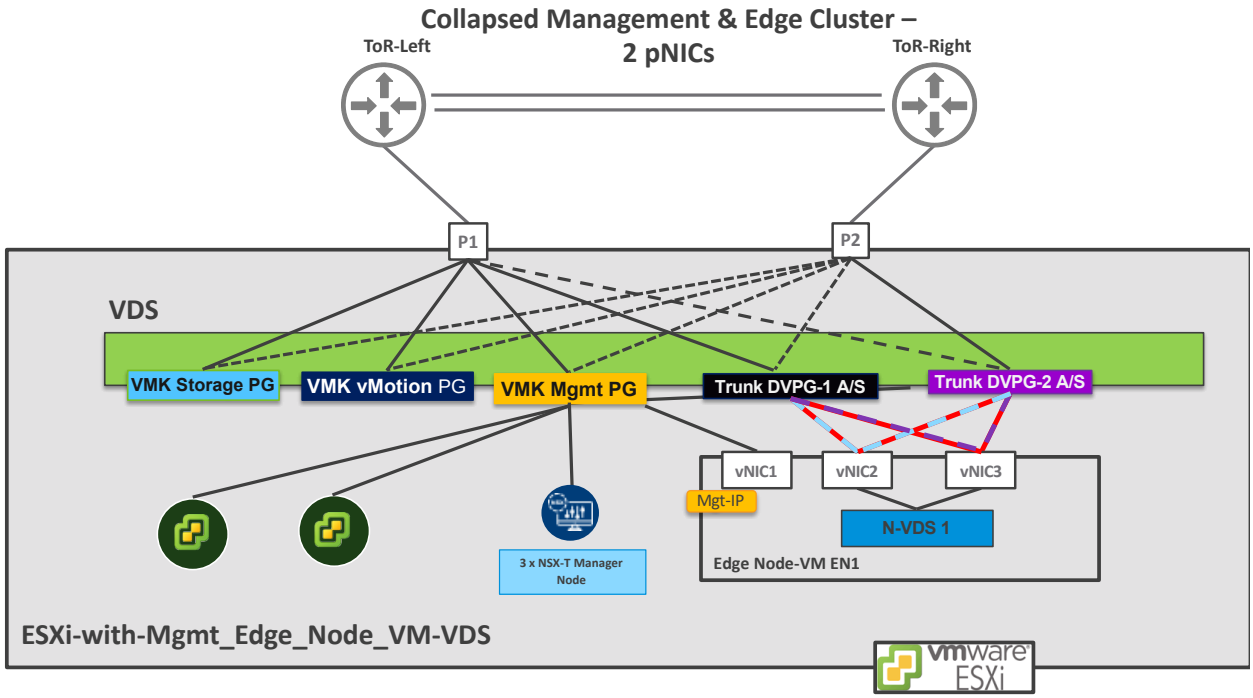


Figure 7-93: Collapsed Management and Edge on VDS with 2 pNICs

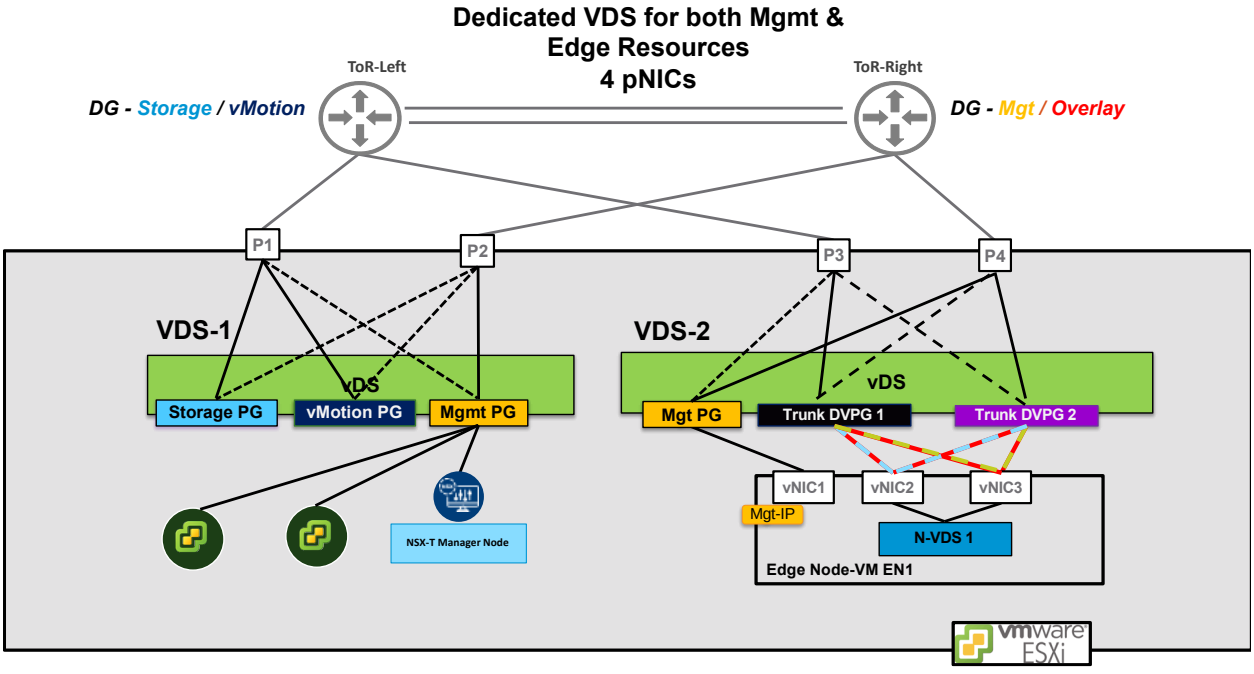


Figure 7-94: Collapsed Management and Edge VM on Separate VDSs with 4 pNICs

7.6.3.3 Collapsed Compute/Edge cluster

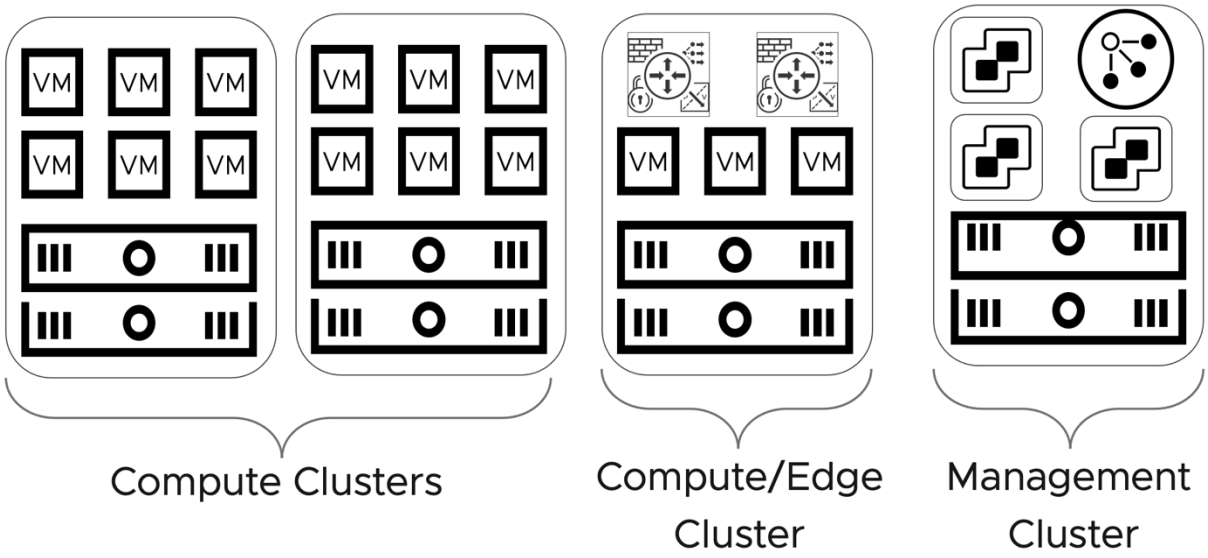


Figure 7-95: Collapsed Edge/Compute cluster

Some designs incorporate a collapsed edge/compute cluster design. This model may be appealing from a resource planning perspective and allows for a simple chargeback model. It allows the operator to dedicate ESXi resources to a use case or a tenant and use those resources for both workloads and network services.

A dedicated vSphere edge cluster may mean that multiple use cases or tenants will share the same compute resources. Sharing the compute hosts dedicated to edge services across multiple use cases or tenants makes resource planning and measuring consumption more complex.

Splitting the edge services in dedicated vSphere clusters may be unfeasible from a cost perspective. The co-placement of the edge nodes with the associated workloads may become appealing to preserve the resource segmentation across the use cases. A collapsed compute/edge deployment model presents the following challenges:

- Resource contention and planning. Application workloads demands may vary over time and may affect edge services performance even when the appropriate capacity planning was performed as part of the initial implementation.
- North/South edge services benefit from dedicated pNICs. Compute clusters are usually deployed with a two pNIC model. We should not deploy more than one edge per host to avoid excessive oversubscription.
- Compute hosts require minimal configuration on the physical fabric: four VLANs local to each rack, management, vMotion, Overlay, and storage, even when the cluster is striped across racks. Edge nodes require additional VLANs that must be available on each host in the cluster. A layer 2 fabric is now required to stripe the compute cluster across racks if the edge node VMs are co-located with compute workloads. Edge VMs running T1 Services

and no T0 can run without problems on a cluster striped across racks in a layer 2 fabric.

- Edge nodes with T0s should always create routing adjacencies with switches directly connected to the host where the edge node VMs reside. This may be unfeasible in an edge/compute cluster spanning multiple racks. An edge VM moving between racks may lose the BGP peering adjacencies unless the network fabric extends the peering VLANs between racks (not recommended). A vSphere cluster hosting T0 Edge VMs should be confined to a single rack. Rack availability can be provided via another vSphere cluster hosting edge node VMs part of the same NSX edge cluster and T0 Gateway. In case the cluster must be striped across racks, mobility of the T0 Gateway should be limited to a single rack even during maintenance operations (Placing the edge node in maintenance mode and powering it off is an option if adequate compute resources are not available in the same rack).
- vSphere HA is most likely enabled globally for the cluster to support compute workloads. If vSphere HA for T0 edge VM is not desired, it must be disabled manually (In case the recovery of the edge VM on a different host will not lead to gains in high availability or performances). vSphere HA for T1 only service or T0/T1 mixed edges is a good practice in most situations.
- vSphere HA does not honor preferential rules (should) but respects mandatory rules. This behavior should be considered when the placement of edge node VMs is critical (T0 Edge VM with vSphere cluster striped across racks)
- DRS is most likely enabled for the cluster to support compute workloads. VM to Host Should rules should be implemented to enforce the desired host to T0 edge VM placement and limit the T0 edge VMs mobility to host maintenance events.
- Limit over reservation of CPU/memory resources for the compute workloads as it may lead to DRS moving the T0 edge VMs regardless of the preferential rules.
- DRS rules, in conjunction with NSX edge failure domains, should be in place so that Active/Standby Gateways do not reside on the same ESXi host during normal operations and maintenance windows (Anti-affinity rules).
- In a two pNIC host design, the edge node VMs connect to an NVDS or a VDS prepared for NSX. Connectivity guidelines presented in "Figure 7-55: Edge VM connected to NSX Prepared VDS – Host and Edge TEP on different IP/VLAN" or "Figure 7-56: Edge VM connected to NSX Prepared VDS – Host and Edge TEP on the same IP/VLAN - NSX 3.1 or later" should be followed.

7.6.3.4 Collapsed Compute/Edge cluster

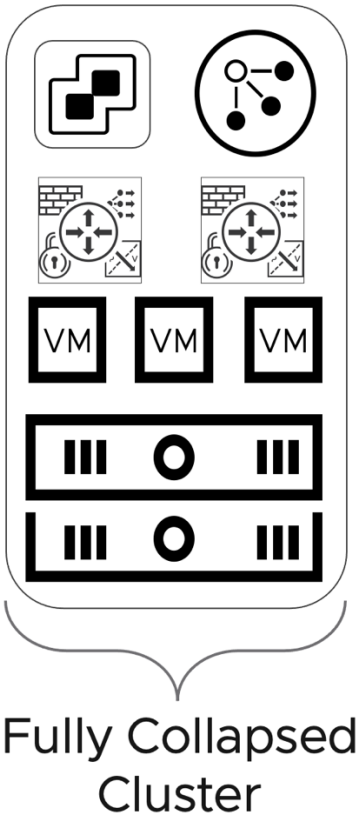


Figure 7-96: Collapsed Edge/Management/Compute cluster

In a fully collapsed cluster design, compute workloads, edge node VMs, and management components all reside in the same vSphere cluster. The NSX EASY ADOPTION DESIGN GUIDE presents an example of this deployment model in great detail. Please refer to the DC in A Box section. A collapsed cluster design presents the challenges of both the compute/edge, and the management/edge collapsed models, but those concerns are usually mitigated by the smaller size of the deployment, usually confined to a single rack. Key considerations for a fully collapsed cluster are:

- In a two pNIC host design, the edge node VMs connect to an NVDS or a VDS prepared for NSX. Connectivity guidelines presented in "FIGURE 7-55: EDGE VM CONNECTED TO NSX PREPARED VDS – HOST AND EDGE TEP ON DIFFERENT IP/VLAN" or "FIGURE 7-56: EDGE VM CONNECTED TO NSX PREPARED VDS – HOST AND EDGE TEP ON THE SAME IP/VLAN - NSX 3.1 OR LATER" should be followed.
- NSX Manager should not be placed on a VLAN segment but on a vCenter managed dvpg.

The diagrams below outline virtual switch and teaming policies design guidelines for four pNICs and two pNICs hosts.

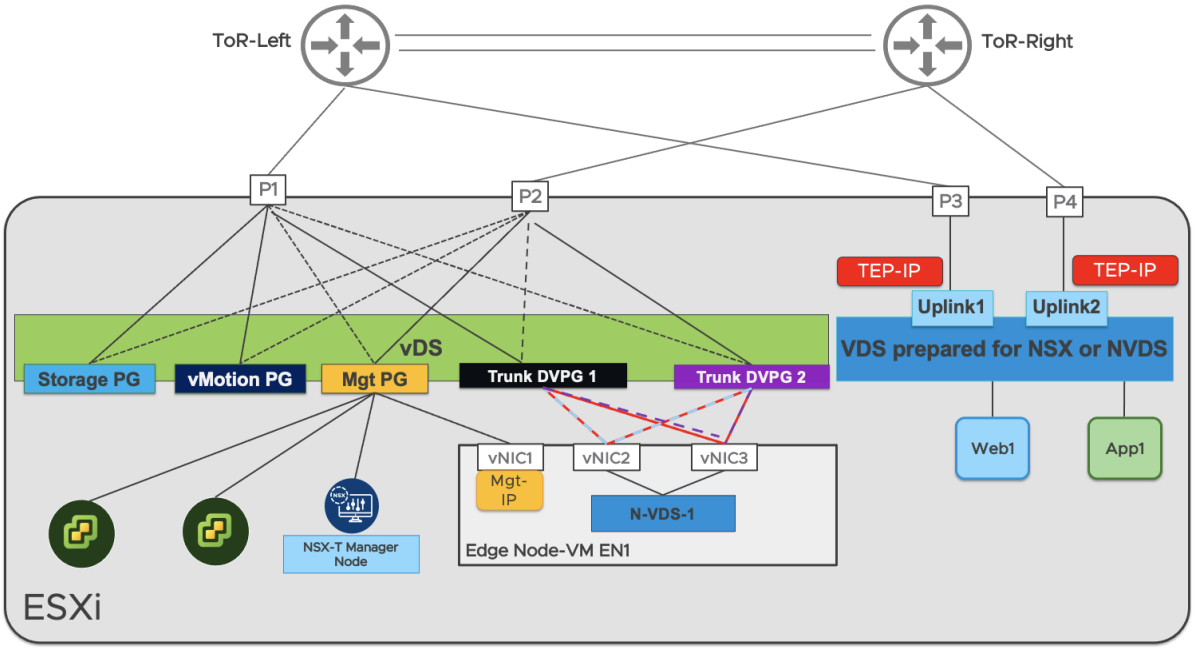


Figure 7-97: Collapsed Edge/Management/Compute cluster - 4 pNICs

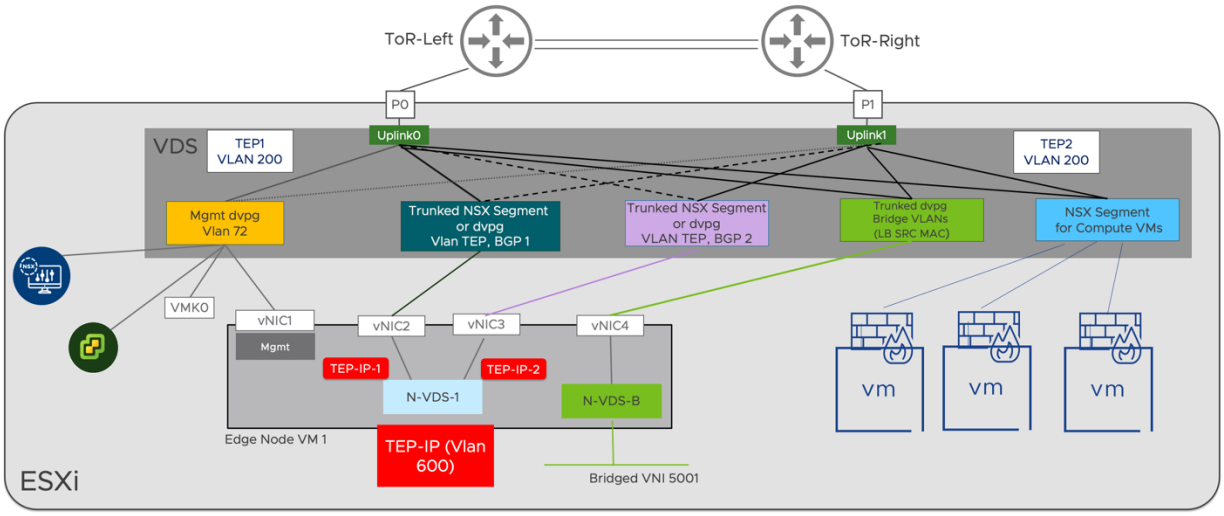


Figure 7-98: Collapsed Edge/Management/Compute cluster - 2 pNICs - separate VLANs for Host and Edge TEPs

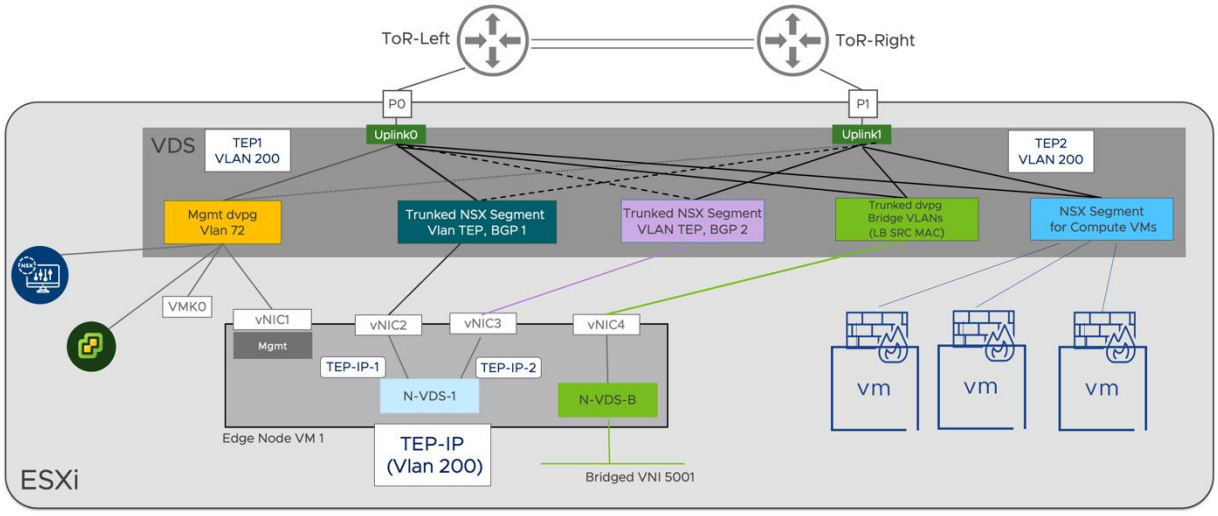


Figure 7-99 Collapsed Edge/Management/Compute cluster - 2 pNICs - single VLAN for Host and Edge TEPs

7.6.3.5 Use case: implementing a repeatable rack design for a scalable private cloud platform in a layer 3 fabric

We have seen that vSphere clusters dedicated to management, edge, and compute workloads have different connectivity and resource requirements. It is possible to cater to such specific requirements by customizing the hardware design for each class of workloads (e.g., CPU core count and pNIC design specific to edges hosts) and dedicating racks to a specific cluster function (e.g., racks dedicated to ESXi hosts running edges with ToR switches configured with the appropriate VLANs). While such a design may be the most effective at leveraging the available resources, it may lack the repeatability and consistency required in large-scale deployments. Large-scale private clouds require a high degree of automation, from the automatic deployment and configuration of the ToR switches and the provisioning of the ESXi hosts to the dynamic allocation of resources to different compute pools (e.g., a vSphere cluster or a VCF workload domain).

This section presents different options to create a repeatable rack layout for computing and edge workloads. We assume that management workloads can be centralized and do not require to follow the same repeatable pattern because of their predictable resource requirements.

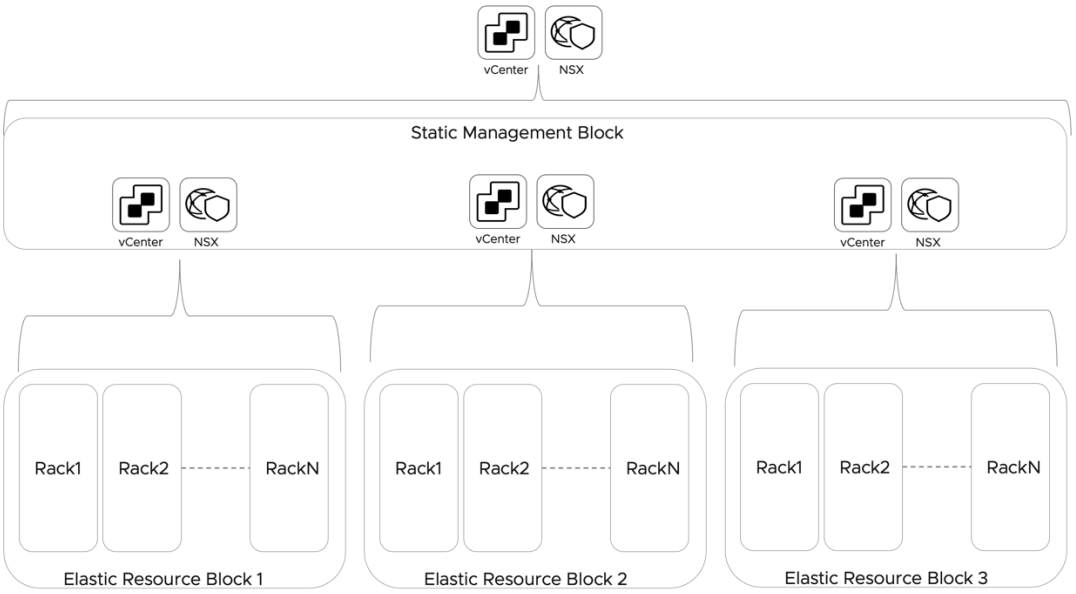


Figure 7-100: Example of conceptual design for a repeatable infrastructure

7.6.3.5.1 Dedicated edge cluster, one host per rack

The diagram in FIGURE 7-101 presents a three racks layout where vSphere clusters are striped across the different racks. Each rack represents a failure domain. If VSAN is the adopted storage technology, the three failure domains imply a storage policy with PFTT=1. We should adopt a five-racks deployment to achieve PFTT=2. We present a three-rack design for simplicity, but we can apply the same design principles to five or more rack designs common in critical infrastructure deployments.

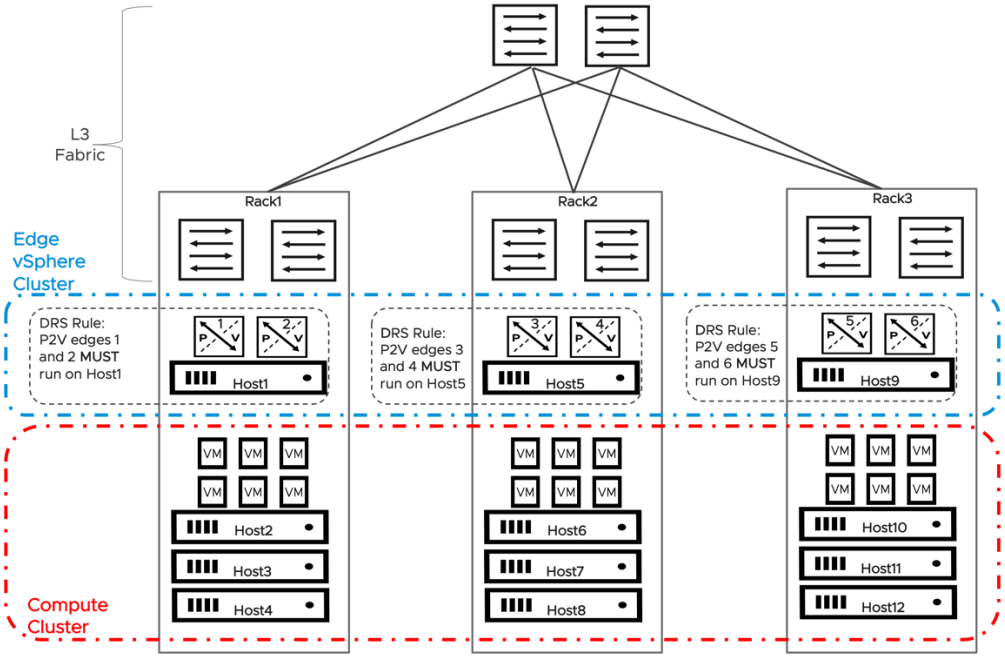


Figure 7-101: Resource Block - Dedicated vSphere edge cluster with one host per rack

In FIGURE 7-101, all hosts have the same hardware configuration (e.g., 2x25G pNICs, 32 cores, 512G Memory). One host in each rack is dedicated to NSX edge services. The host runs two or more NSX edge node VMs where both a T0 Gateway in ECMP and T1 Gateways with services are hosted. While this configuration may not be optimal, it allows for a repeatable deployment model that enhances the scalability and the manageability of the overall platform. This model can be extended to a design where T0 and T1 gateways run on different edge nodes part of different edge clusters. Such model is not discussed in this section.

In FIGURE 7-101, the edge hosts are grouped in a dedicated vSphere edge cluster. The cluster is striped across racks. We mentioned already that, in general, it's not a good practice to stripe vSphere edge clusters across racks, but in this case, the repeatability of the deployment model, which dictates one edge host per rack, takes precedence. The three hosts part of the edge cluster have access to the same VSAN Datastore, but they cannot provide VM mobility across racks for the edge VMs because each rack has specific management, overlay, and layer 3 peering VLANs. If an edge VMs migrates to a different rack, it will lose connectivity immediately. Design guidelines and implications for the model are:

- We should map edge node VMs to NSX failure domains based on the rack where they are deployed.
- Edge Node VMs cannot move across racks because the appropriate management, overlay, and L3 peering networks are not available on every rack.
- If the hosts in the edge vSphere cluster are part of the same VDS and the VLAN IDs are repeated on each rack for consistency, DRS is not aware that the available networks are different and may move the edge VMs anyway.
- DRS MUST rules should enforce the correct rack placement for each edge node VM to avoid any edge VM migration.
- If a single edge ESXi host is present in each rack, as depicted in FIGURE 7-101, the host cannot be evacuated to be placed in maintenance mode. Edge node VMs must be placed in maintenance mode and then shut down before the host can be placed in maintenance mode. When the host exits maintenance mode and the edge VM is powered and brought back online, it is required to manually check the status of its services (e.g., BGP neighbors are up) before proceeding with the maintenance of another ESXi host and the corresponding edge VMs.

7.6.3.5.2 Dedicated edge cluster, two hosts per rack

From an operations perspective, the main drawback of the previous design is the inability to leverage DRS to evacuate the edge ESXi hosts during maintenances. The design in FIGURE 7-102 attempts to alleviate this pain point by deploying two edge ESXi hosts per cluster.

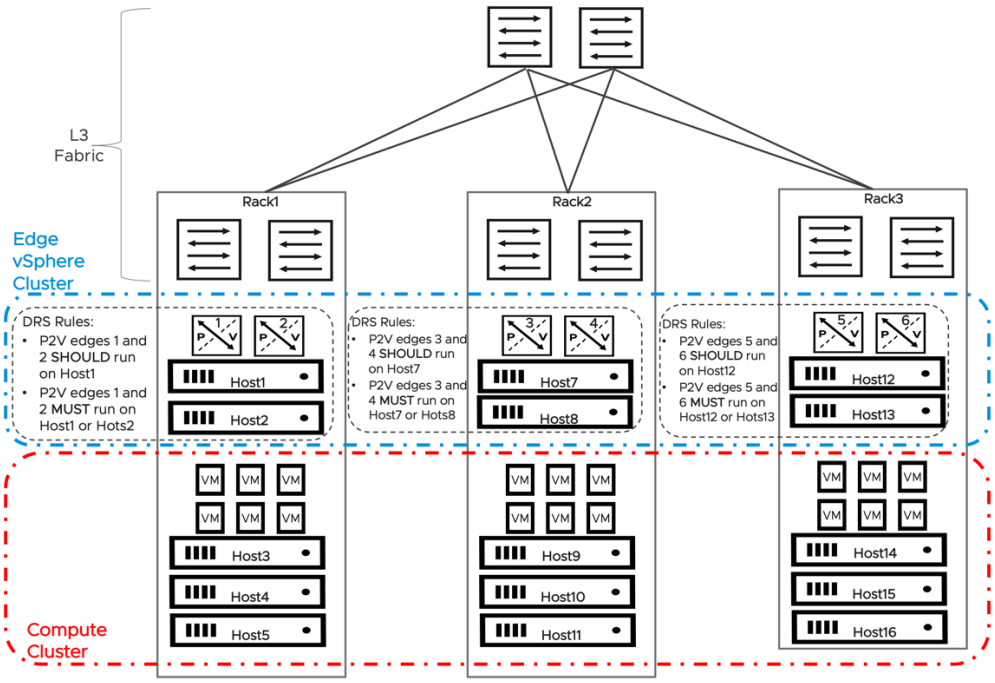


Figure 7-102: Resource Block - Dedicated vSphere edge cluster with two hosts per rack

The design in FIGURE 7-102 has the following guidelines and implications:

- Edge hosts can be evacuated by vMotion to a different host in the same rack during a maintenance. Edge nodes are not shut down.
- Edge Node VMs cannot move across racks because the appropriate management, overlay, and L3 peering networks are not available on every rack.
- If the hosts in the edge vSphere cluster are part of the same VDS and the VLAN IDs are repeated on each rack for consistency, DRS is not aware that the available networks are different and may move the edge VMs anyway.
- DRS MUST rules should enforce the correct rack placement for each edge node VM to avoid any edge VM migration across racks.
- DRS SHOULD rules will dictate the preferred host placement for each edge VMs. Because now multiple hosts are present in the same rack, DRS in fully automated mode could rebalance the edge VMs placement during normal operations. The SHOULD rule will prevent undesired migrations. A SHOULD rule will be violated when placing the ESXi host in maintenance mode. The edge VMs will be moved to another host in the same rack. It won't be moved to a different rack because MUST rules cannot be violated.
- If DRS rules management is considered too much of an overhead, DRS can be disabled or set to partially automated. In this case, edge VMs must be migrated manually during maintenances. This choice carries the risk of human error (e.g., edge node VM migrated to a different rack by mistake)

- Even with six hosts in the edge cluster spread across three racks, the VSAN storage policy has an PFTT=1 when each rack is mapped to a VSAN failure domain.
- Two edge hosts per rack may represent a waste of computing resources, especially in deployments with five racks or more.

7.6.3.5.3 Shared edge/compute cluster with dedicated resources

The design presented in FIGURE 7-103 is a compromise between the two previous designs. All hosts are grouped in a single edge/compute cluster, but host resources are still dedicated. One host in each rack is dedicated to the edge VMs, while the rest is dedicated to computing workloads, but during maintenances, we can move edge node VMs to compute hosts in the same rack. A MUST and SHOULD rules combination guarantees that the edge VMs are not migrated across racks. They run preferentially on a dedicated host but can migrate to a different host in the same rack when the preferred host is placed in maintenance mode.

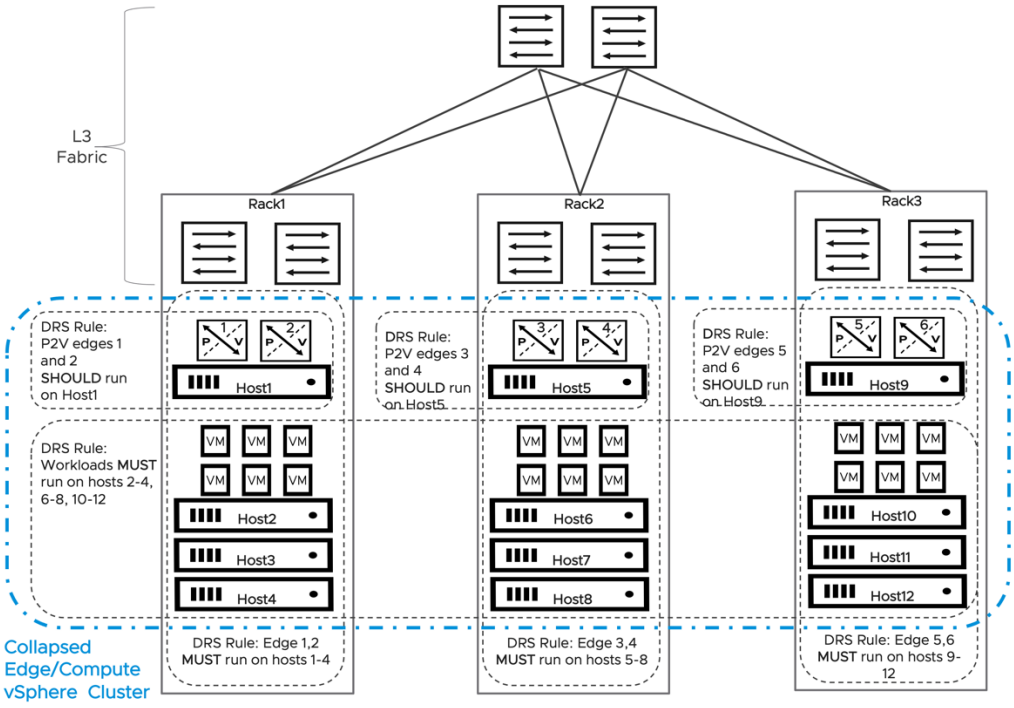


Figure 7-103: Resource Block - Collapsed vSphere edge/compute cluster

Design guidelines and implications for the design in FIGURE 7-103 are:

- A complex set of DRS rules is required.
- One VM to host MUST rule per rack is required to avoid edge VM mobility across racks.
- VM to Host SHOULD rules must be in place to avoid edge VM movements across hosts in the same rack during normal operations.

- DRS rule (SHOULD or MUST depending on the design requirements) to avoid/discourage the placement of computing workloads on ESXi hosts dedicated to edge node VMs. This DRS rule has the most impactful implications in terms of day-2 operations because any new workload deployed on the cluster needs to be added to the corresponding VM group to avoid resource contention with the edge node VMs.

8 NSX Performance & Optimization

8.1 Typical Data Center Workloads

Workloads in the data center are typically TCP-based. Generally, 80% of the traffic flows within the data center are east/west, that is, communication between various compute nodes. The remaining 20% is north/south, that is, communication in and out of the data center. The following FIGURE 8-1: DATA CENTER TRAFFIC PATTERN WITH NSX shows the typical traffic flow distribution:

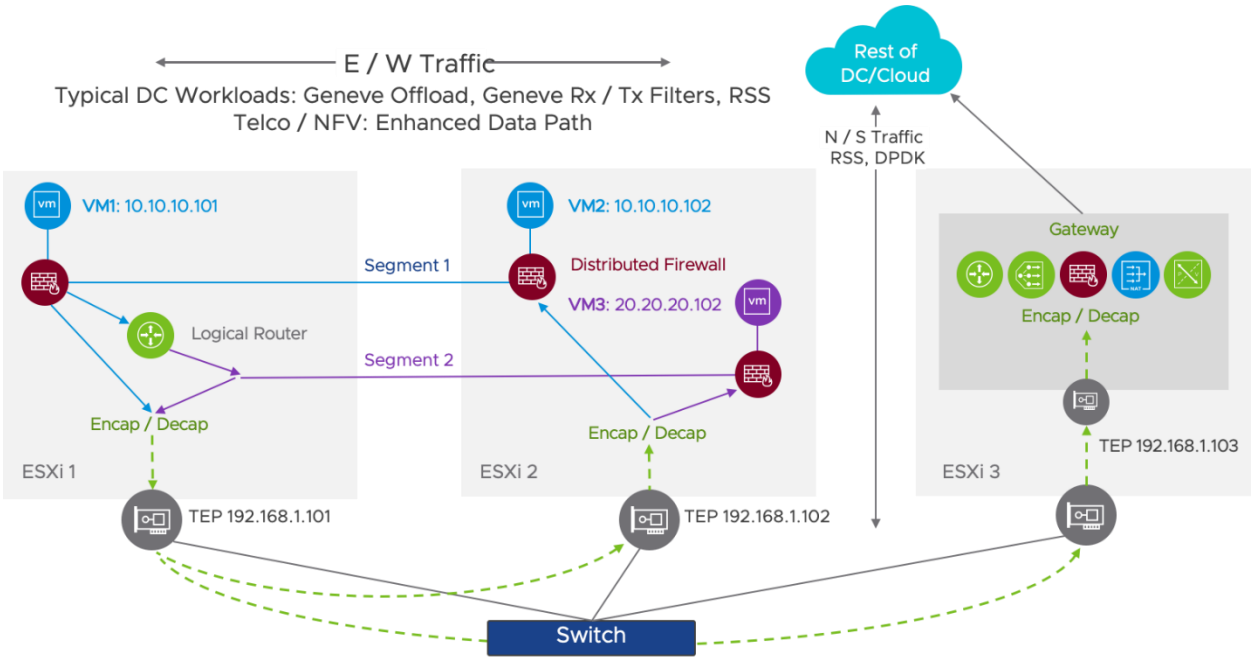


Figure 8-1: Data Center Traffic Pattern with NSX

Because of this underlying data center framework, this chapter primarily focuses on performance in terms of throughput for TCP-based workloads. There are some niche workloads such as NFV, where raw packet processing may be ideal, and the enhanced version of N-VDS called N-VDS (E) was designed to address these requirements. Check out the last part of this section for more details on N-VDS (E).

8.2 Next Generation Encapsulation - Geneve

Geneve, a draft RFC in the IETF standard body co-authored by VMware, Microsoft, Red Hat, and Intel, grew out of a need for an extensible protocol for network virtualization. With Geneve the new framework for overlay/network virtualization, understanding Geneve is foundational to rest of this chapter as the rest of the optimizations to be discussed revolve around Geneve.

With its options field length specified for each packet within the Geneve header, Geneve allows packing the header with arbitrary information into each packet. This flexibility offered by Geneve opens up doors for new use cases, as additional information may be embedded into the packet, to help track the packets path or for in depth packet flow analysis.

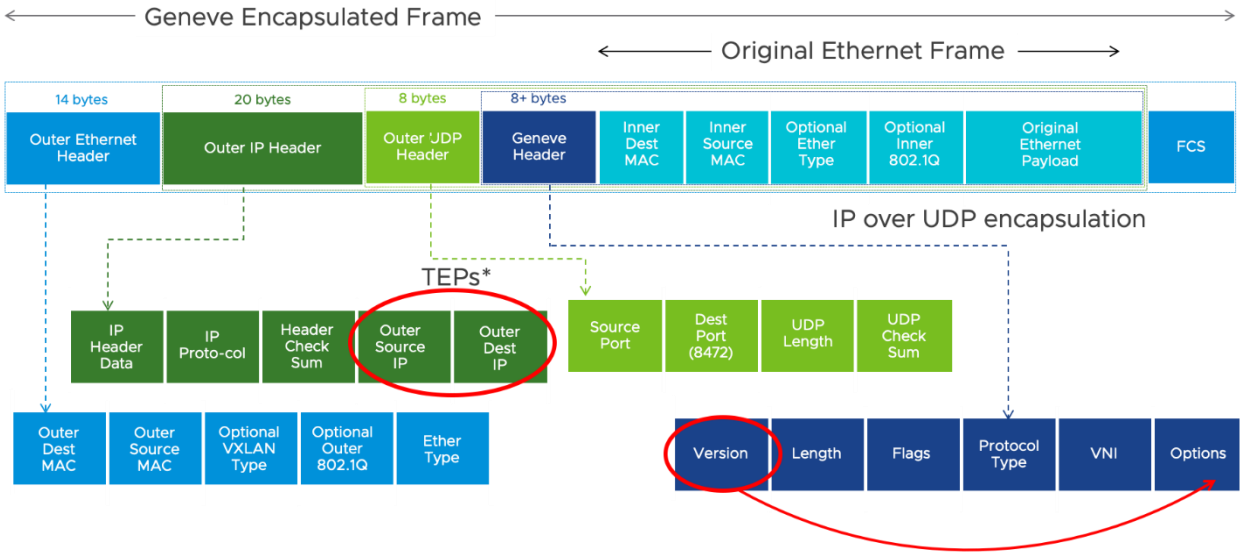


Figure 8-2: Geneve Header

The above FIGURE 8-2: GENEVE HEADER shows the location of the Length and Options fields within the Geneve header and also shows the location of TEP source and destination IP's.

For further insight into this topic, please check out the following blog post: HTTPS://OCTO.VMWARE.COM/GENEVE-VXLAN-NETWORK-VIRTUALIZATION-ENCAPSULATIONS/

8.3 Performance Considerations

In this section we will take a look at the factors that influence performance. Performance of a workload in a virtualized environment, depends on many factors within that environment; hardware used, drivers and features etc. FIGURE 8-3 shows the different areas that may have an impact on performance.

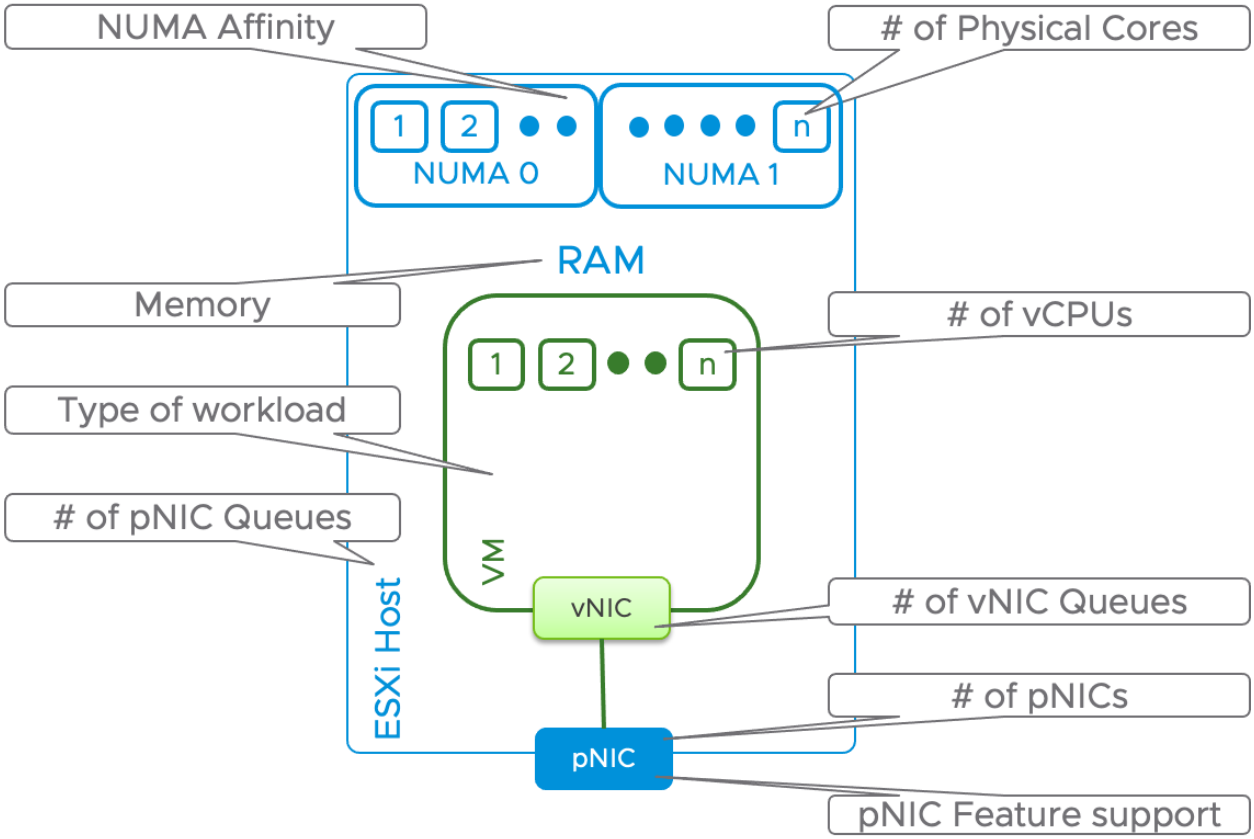


Figure 8-3: Factors that influence performance

8.3.1 pNIC Feature Support

Features supported by the pNIC and the associated firmware and drivers, have a significant impact on the performance. For applications leveraging standard datapath, there are three important features to consider.

- Geneve offload
- RSS
- Geneve Rx / Tx Filters

We will take a closer look at each of these features in the following sections.

8.3.1.1 Geneve Offload

Geneve Offload is simply TCP Segmentation Offload (TSO) tuned to understand Geneve headers. Since Geneve is not TCP traffic, NIC cards need to be aware of Geneve headers to perform TCP Segmentation Offload type functionality on the Geneve segments. In addition, guest VMs need to enable TSO, the default behavior with most modern operating systems.

TCP Segmentation Offload (TSO): TCP Segmentation offload is a well-established TCP optimization scheme of relatively long duration that allows large segments to pass through the TCP stack, instead of smaller packets as enforced by the MTU on the physical fabric.

8.3.1.1.1 (TSO) applicability to Geneve Overlay Traffic

In the context of Geneve, NICs are aware of the Geneve headers and perform TSO taking Geneve headers into consideration. The following FIGURE 8-4: NIC BASED GENEVE OFFLOAD - TSO SHOWS how the VM would transmit 64K segments, such as switching, routing and firewall and the ESX TCP Stack, which go through the NSX Components as 64K segments. NIC cards take care of chopping the segments down to MTU-sized packets before moving them on to the physical fabric.

While TSO's primary benefit is in reducing CPU cycles, this features also helps increase throughput, marginally. However, in cases where the NIC does not have Geneve capability, ESX automatically falls back to software based Geneve offload mode. While NIC-based offload is ideal, software-based offload still helps reduce the CPU cycles spent for NSX components.

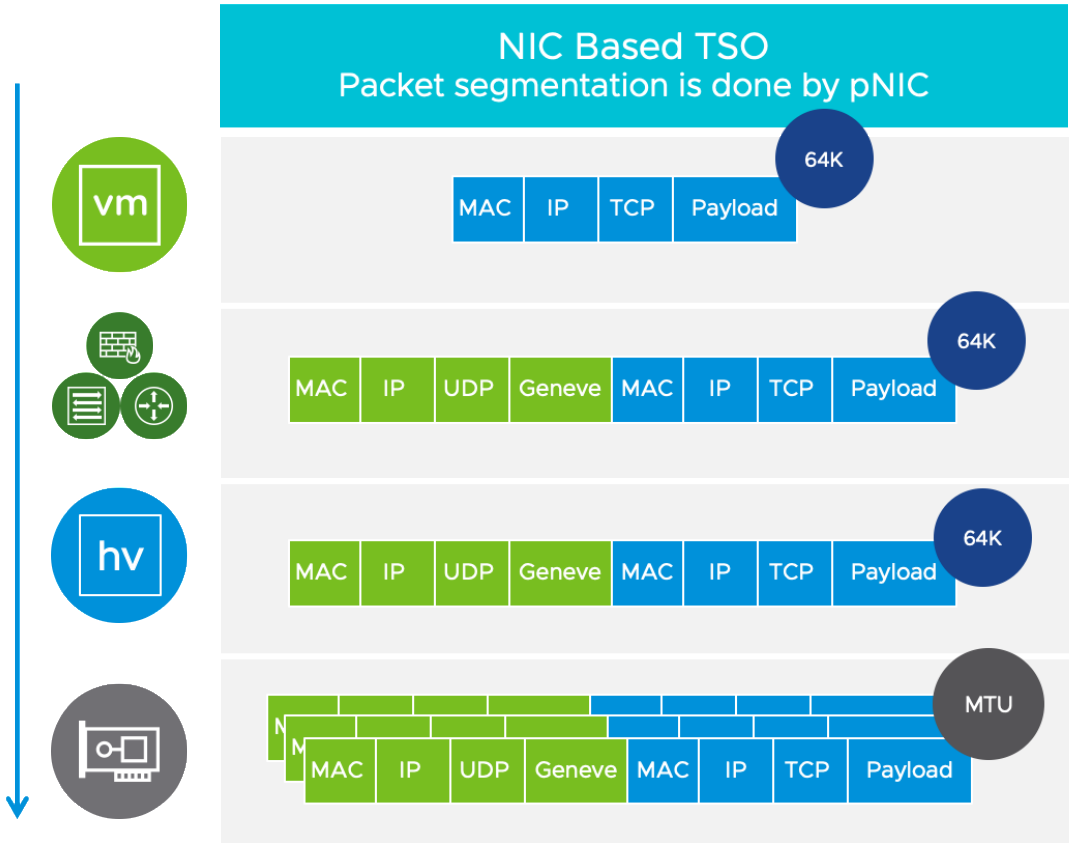


Figure 8-4: NIC Based Geneve Offload - TSO

8.3.1.1.2 NIC Supportability with TSO for Geneve Overlay Traffic

In cases where the NIC card does not support TSO for Geneve overlay traffic, TSO is done in software by the hypervisor just before moving the MTU-sized packets to the NIC card. Thus, NSX components are still able to leverage TSO.

The following FIGURE 8-5: SOFTWARE/CPU BASED GENEVE OFFLOAD - TSO shows the process where the hypervisor divides the larger TSO segments to MTU-sized packets.

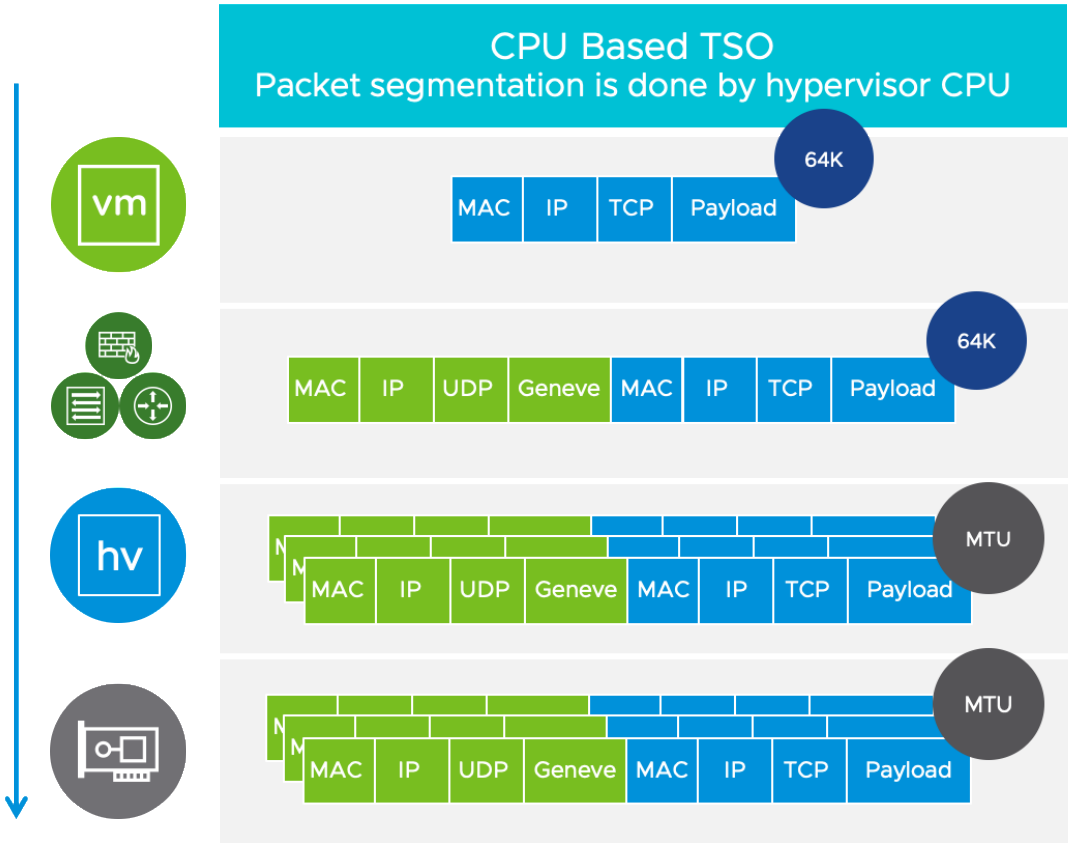


Figure 8-5: Software/CPU Based Geneve Offload - TSO

8.3.1.1.3 Confirming Geneve Offload on ESXi-Based Hypervisor

On an ESXi host with a NIC card supporting Geneve-Offload in Hardware with the appropriate supported driver, the following commands can be used to confirm Geneve-Offload is enabled on a pNIC – in this case pNIC vmnic3:

```
[Host-1] vsish -e get /net/pNics/vmnic3/properties | grep
".*Activated.*Geneve"

   Device Hardware Cap Activated:: 0x793c032b ->
VMNET_CAP_SG VMNET_CAP_IP4_CSUM VMNET_CAP_HIGH_DMA
VMNET_CAP_TSO VMNET_CAP_HW_TX_VLAN VMNET_CAP_HW_RX_VLAN
VMNET_CAP_SG_SPAN_PAGES VMNET_CAP_IP6_CSUM VMNET_CAP_TSO6
VMNET_CAP_TSO256k VMNET_CAP_ENCAP VMNET_CAP_Geneve_OFFLOAD
VMNET_CAP_IP6_CSUM_EXT_HDRS VMNET_CAP_TSO6_EXT_HDRS
VMNET_CAP_SCHED
```

CLI 1 Check Geneve Offload Support

Look for the tag "VMNET_CAP_Geneve_OFFLOAD", highlighted in red above. This verbiage indicates the Geneve Offload is activated on NIC card vmnic3. If the tag is missing, then it means Geneve Offload is not enabled because either the NIC or its driver does not support it.

8.3.1.2 Receive Side Scaling (RSS) and Rx Filters

Readers familiar with software based VXLAN deployment with NSX-V, are likely familiar with the immense performance benefits of RSS, including improving the performance of overlay traffic by four (4) times.

8.3.1.2.1 Benefits with RSS

RSS, another long-standing TCP enhancement, enables use of multiple cores on the receive side to process incoming traffic. Without RSS, ESX by default will use only one core to process incoming traffic. Utilizing only one core has a huge impact on the overall throughput as the receiving node then becomes the bottleneck. RSS on the NIC creates multiple queues to process incoming traffic and efficiently uses a core for each queue, with most NIC cards being able to support at least 4 queues. Hence the 4x benefit of using RSS. See FIGURE 8-6: RSS for a visual representation of how this works.

8.3.1.2.2 RSS for overlay

While RSS itself in general is in fairly common use today, there are NICs which still may not support RSS for overlay. Hence, our recommendation is to confirm with the NIC vendor whether RSS for overlay traffic is available in hardware, then also confirm with the VMware COMPATIBILITY IO GUIDE (HTTPS://WWW.VMWARE.COM/RESOURCES/COMPATIBILITY/SEARCH.PHP?DEVICECATEGORY=IO) whether there is a RSS-certified driver.

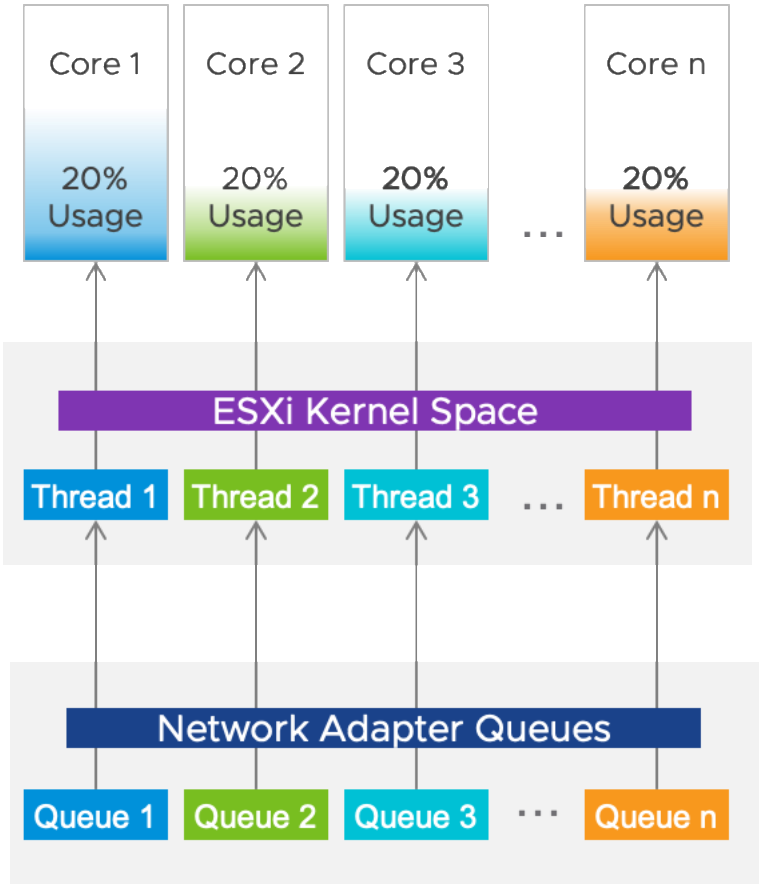


Figure 8-6: RSS

8.3.1.2.3 Enabling RSS for Overlay

Every vendor has their own unique mechanism to enable RSS for overlay traffic. There are also cases where the setting used to change RSS is different based on the driver version. Please refer to the concerned vendor documentation for details on enabling RSS for specific NICs.

8.3.1.2.4 Checking whether RSS is enabled

Use the "vsish" command to check whether RSS is enabled. The following example shows how to check whether RSS (marked blue) is enabled on NIC vmnic (marked in red).

```
[Host-1] # vsish
/> get /net/pNics/vmnic0/rxqueues/info
rx queues info {
# queues supported:5
# filters supported:126
# active filters:0
```

```
Rx Queue features:features: 0x1a0 -> Dynamic RSS Dynamic
Preemptible
}
/>
```

CLI 2 Check RSS

8.3.1.3 Geneve Rx Filters

RSS uses the outer headers to hash flows to different queues. Using outer headers of Geneve overlay, especially between two hypervisors, may not be optimal as the only varying parameter is the source port.

The following image shows the fields used by RSS (circled in red) to hash flows across various CPU cores. Since all the fields are from the outer headers, there is a little variability and in the worst-case scenarios the only variable may be the source port.

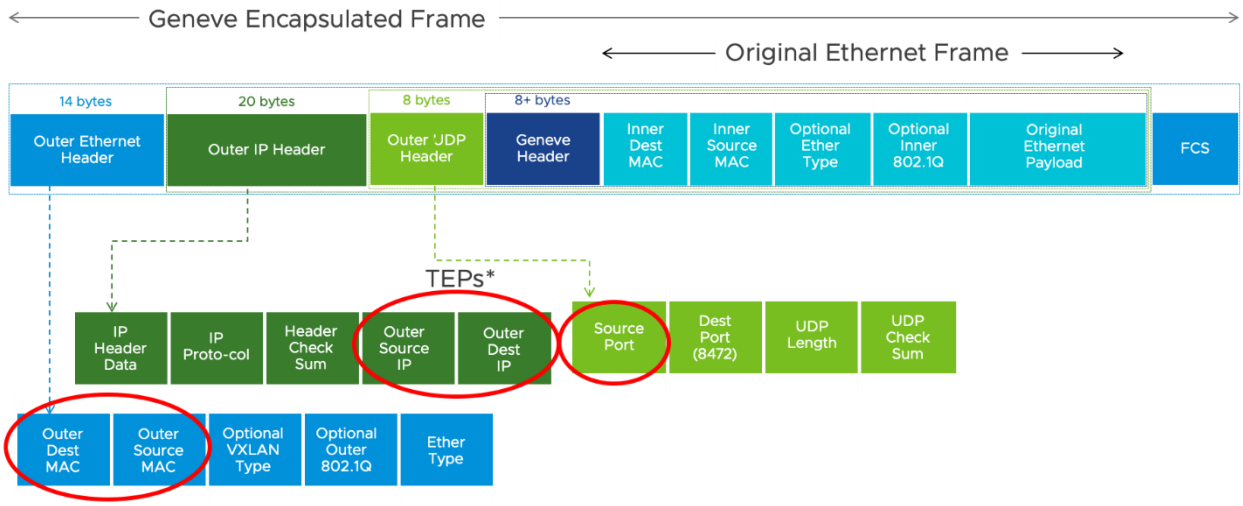


Figure 8-7: RSS: Fields Used for Hashing

To overcome this limitation, the latest NICs (SEE COMPATIBILITY GUIDE) support an advanced feature, known as Rx Filters, which looks at the inner packet headers for hashing flows to different queues on the receive side. In the following FIGURE 8-8: RX FILTERS: FIELDS USED FOR HASHING fields used by Rx Filter are circled in red.

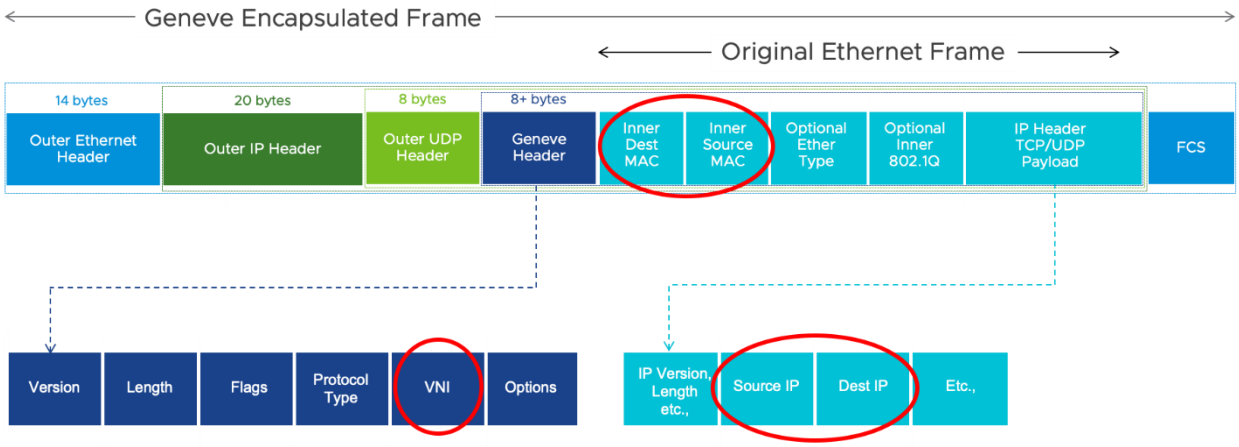


Figure 8-8: Rx Filters: Fields Used for Hashing

Simply put, Rx Filters look at the inner packet headers for queuing decisions. As driven by NSX, the queuing decision itself is based on flows and bandwidth utilization. Hence, Rx Filters provide optimal queuing compared to RSS, which is akin to a hardware-based brute force method.

8.3.1.3.1 Checking whether Rx Filters are enabled

Rx Filters are enabled by default on a NIC that supports Rx Filters in hardware and has a driver to use it. Please use the VMware's COMPATIBILITY GUIDE FOR IO, discussed earlier in this chapter, to confirm whether Rx Filters are available. In VCG for I/O page, select "Geneve-RxFilter", and make sure the right driver is installed on the ESXi host.

On the ESXi host, use the "vsish" command to check whether Rx Filters are enabled. The following example shows how to check whether the NIC vmnic5 (marked in red) has Rx Filters Enabled for Geneve (marked in blue)

Check Whether Rx / Tx Filters are Enabled:

```
[Host-1] vsish
/> cat /net/pNics/vmnic5/rxqueues/info
rx queues info {
   # queues supported:8
   # filters supported:512
   # active filters:0
   # filters moved by load balancer:254
   # of Geneve OAM filters:2
   RX filter classes:Rx filter class: 0x1c -> VLAN_MAC
VXLAN Geneve GenericEncap
   Rx Queue features:features: 0x82 -> Pair Dynamic
```

```
}
/>
```

CLI 3 Check RxFilters Support

8.3.1.3.2 RSS vs Rx Filters for Edge VM

In the case of Edge VMs, the hypervisor does not encapsulate/decapsulate the overlay packets. Instead, the packets are sent along with the overlay headers to the Edge VM's vNIC interfaces. In this case, RSS is the best mechanism available today to hash packets to separate queues. See more on RSS for Edges in the Edge sections.

8.3.1.4 NIC Feature Support

To confirm the features supported by a NIC, use VMware's IO compatibility guide, which is a publicly accessible online tool and the single source truth:

HTTPS://WWW.VMWARE.COM/RESOURCES/COMPATIBILITY/SEARCH.PHP?DEVICECATEGORY=IO

The following section show the steps to check whether a NIC, Intel 810s in this case, supports one of the key features, Geneve offload.

1. Access the online tool: HTTPS://WWW.VMWARE.COM/RESOURCES/COMPATIBILITY/SEARCH.PHP?DEVICECATEGORY=IO

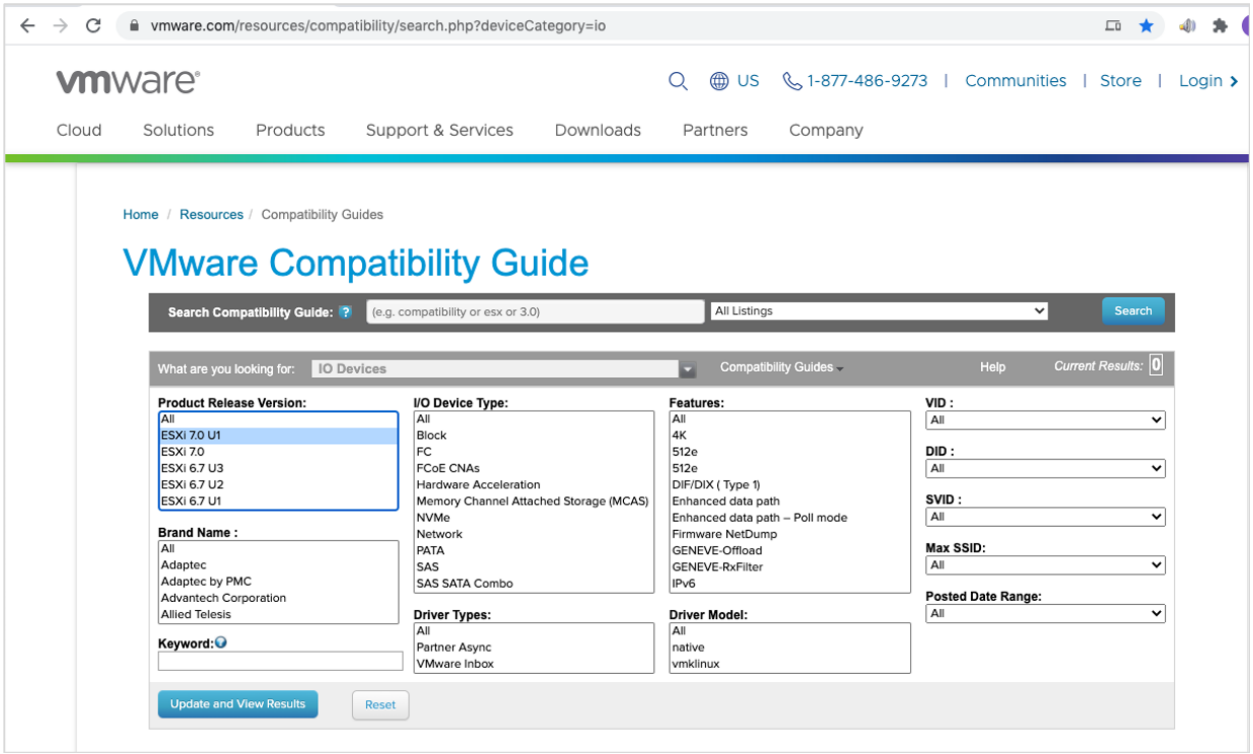


Figure 8-9: VMware Compatibility Guide for I/O

2. Specify the
    1. Version of ESX
    2. Vendor of the NIC card
    3. Model if available
    4. Select Network as the IO Device Type
    5. Select Geneve Offload and Geneve Rx Filters (more on that in the upcoming section) in the Features box
    6. Select Native
    7. Click "Update and View Results"

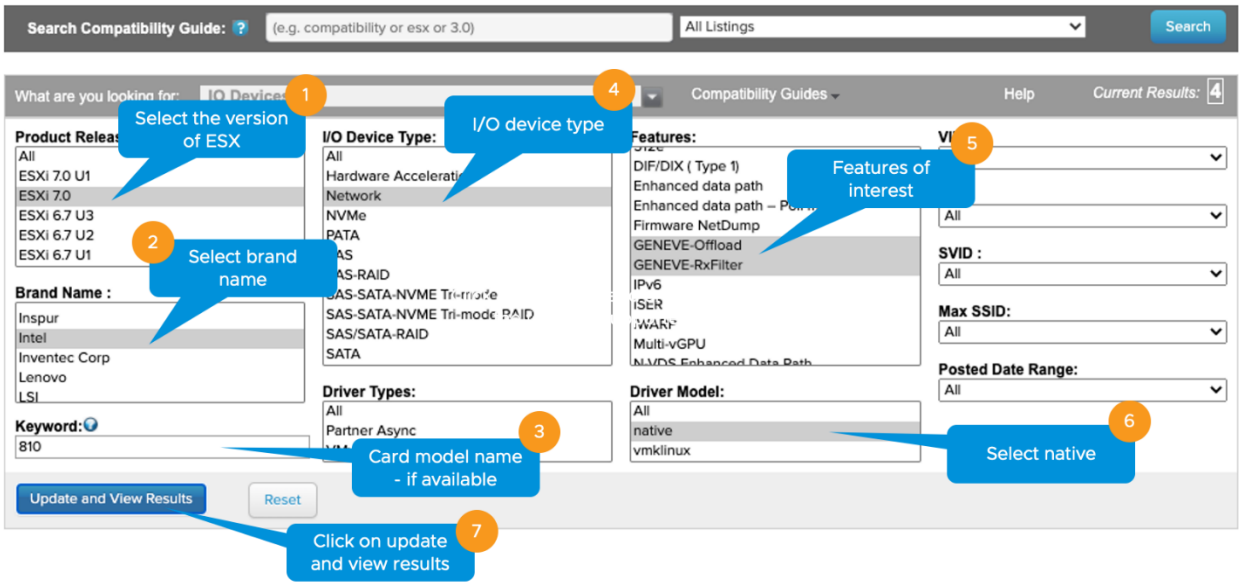


Figure 8-10: Steps to Verify Compatibility – Part 1

3. From the results, click on the ESX version for the concerned card. In this example, ESXi version 7.0 for Intel E810-C with QSFP ports:

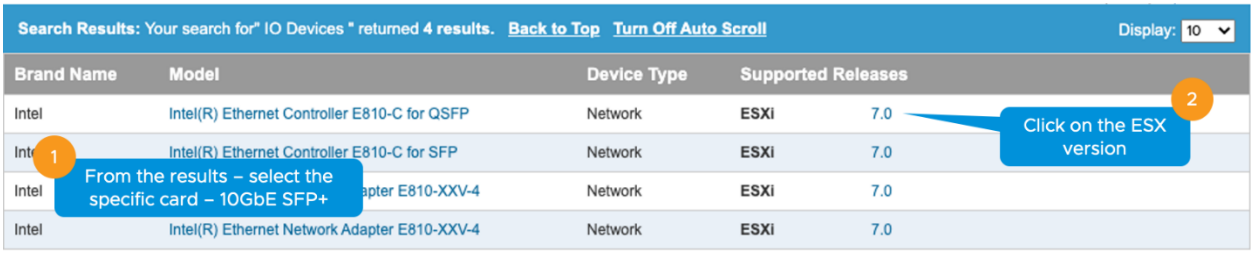


Figure 8-11: Steps to Verify Compatibility - Part 2

4. Click on the [+] symbol to expand and check the features supported.

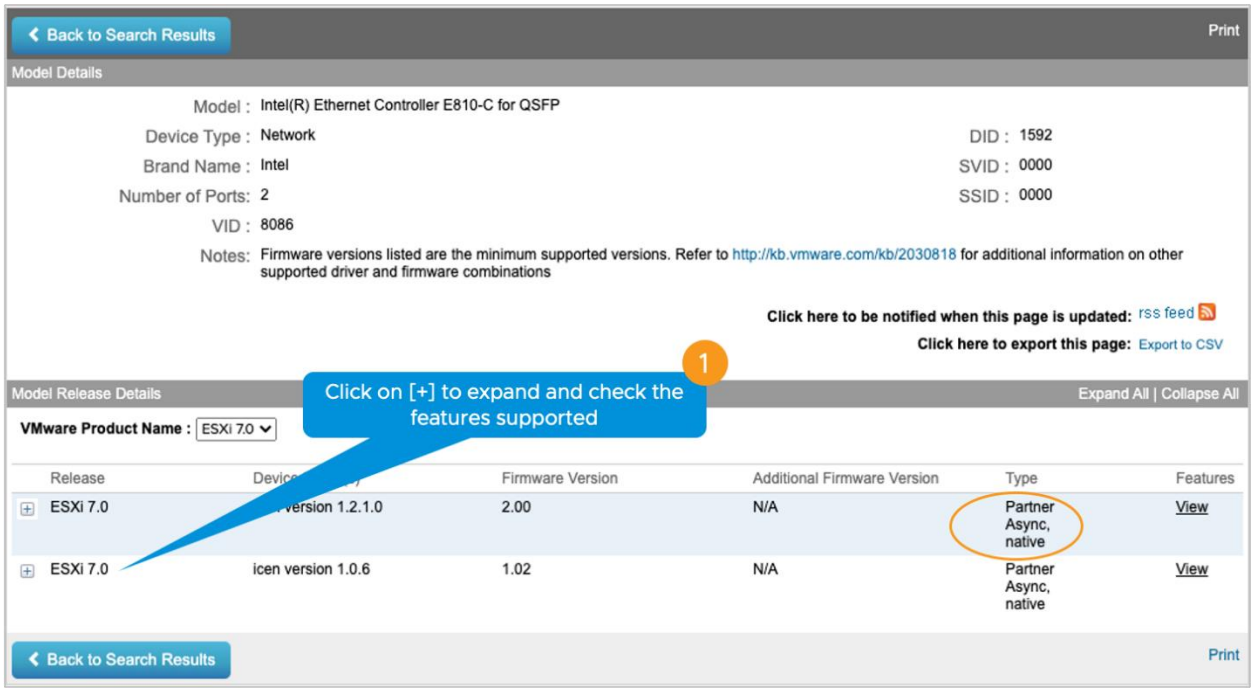


Figure 8-12: Steps to Verify Compatibility - Part 3

5. Make sure the concerned driver actually has the "Geneve-Offload and Geneve-Rx Filters" as listed features.

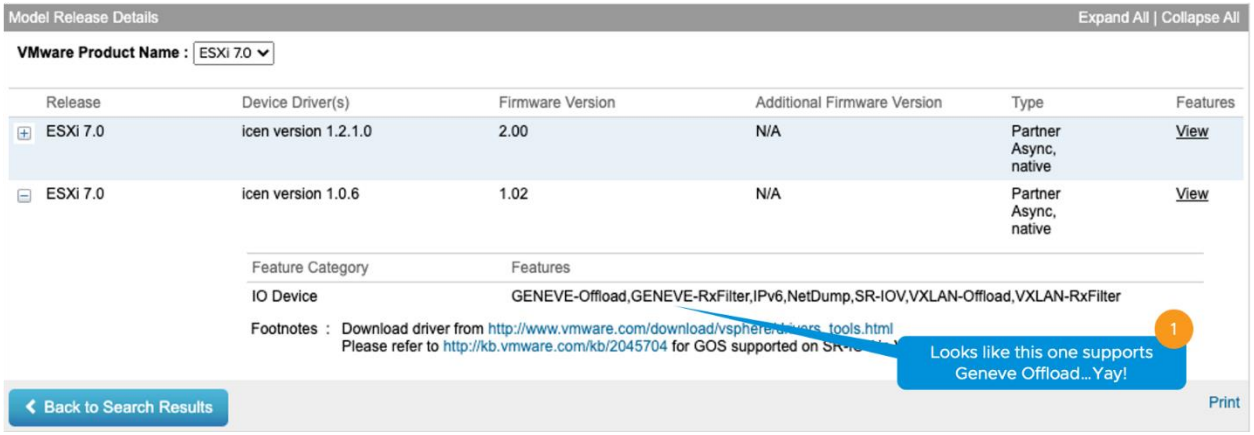


Figure 8-13: Steps to Verify Compatibility - Part 4

Follow the above procedure to ensure Geneve offload and Geneve Rx Filters are available on any NIC card you are planning to deploy for use with NSX. As mentioned earlier, not having Geneve offload will impact performance with higher CPU cycles spent to make up for the lack of software based Geneve offload capabilities.

8.3.1.5 NIC Port Speed

The port speed of the NIC, in combination with all the above features, also has a direct impact on the achieved throughput. The following graph shows throughput achieved on an Intel® E810C 100Gbps NIC using 8800 MTU. Thanks to its x16 lane design, this NIC's performance is close to line rate at 100Gbps @ 8800 MTU.

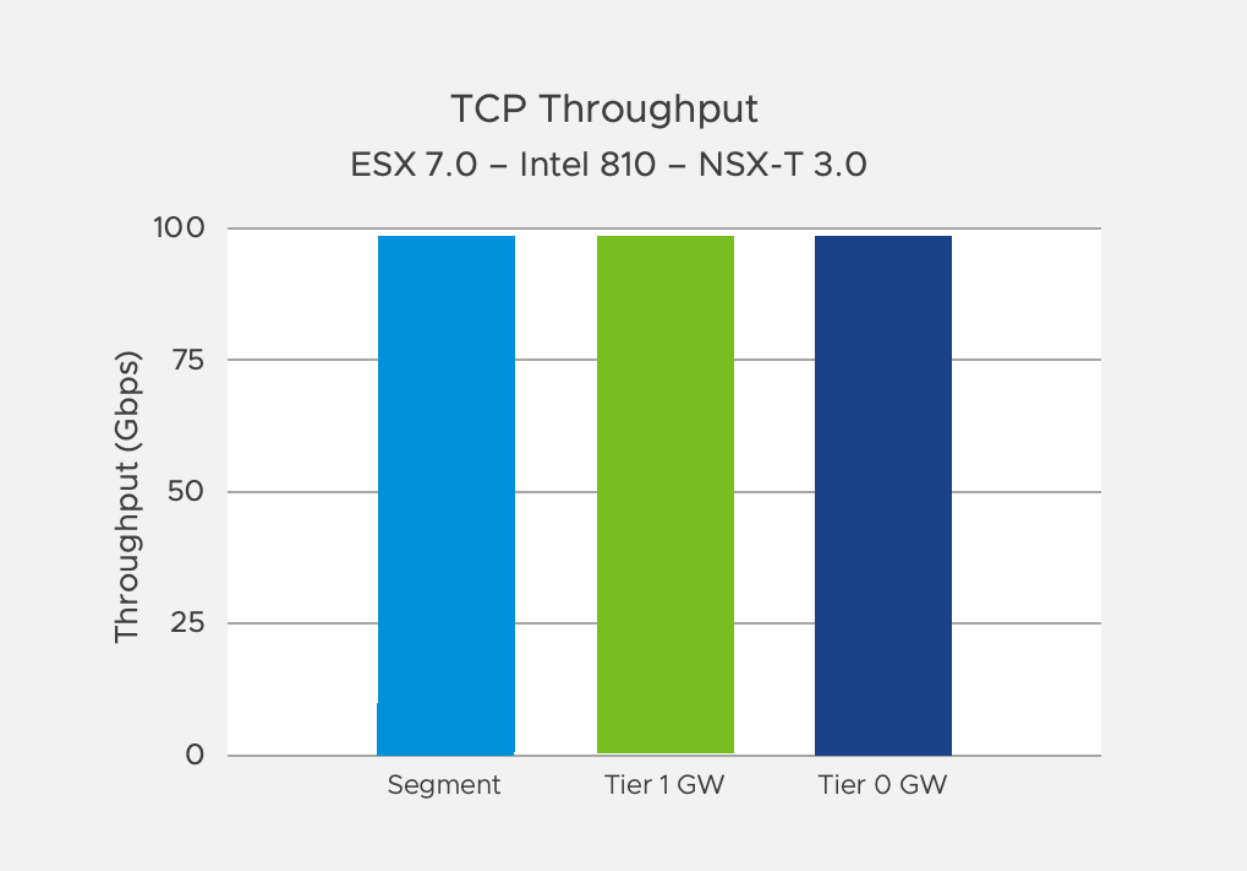


Figure 8-14: Throughput with Intel® E810-C 100Gbps NIC

8.3.1.6 PCIe Gen 4

PCIe Gen 4 doubles the bandwidth when compared to PCIe Gen3. Which translates into higher throughput even when using standard 1500 MTU. Following graph shows the throughput on PCIe Gen 4 Platform based on AMD EPYCTM 7F72 @ 3.2 GHz – Single-Socket using PCIe Gen 4 NIC Mellanox ConnectX-6 single port using a standard 1500 MTU.

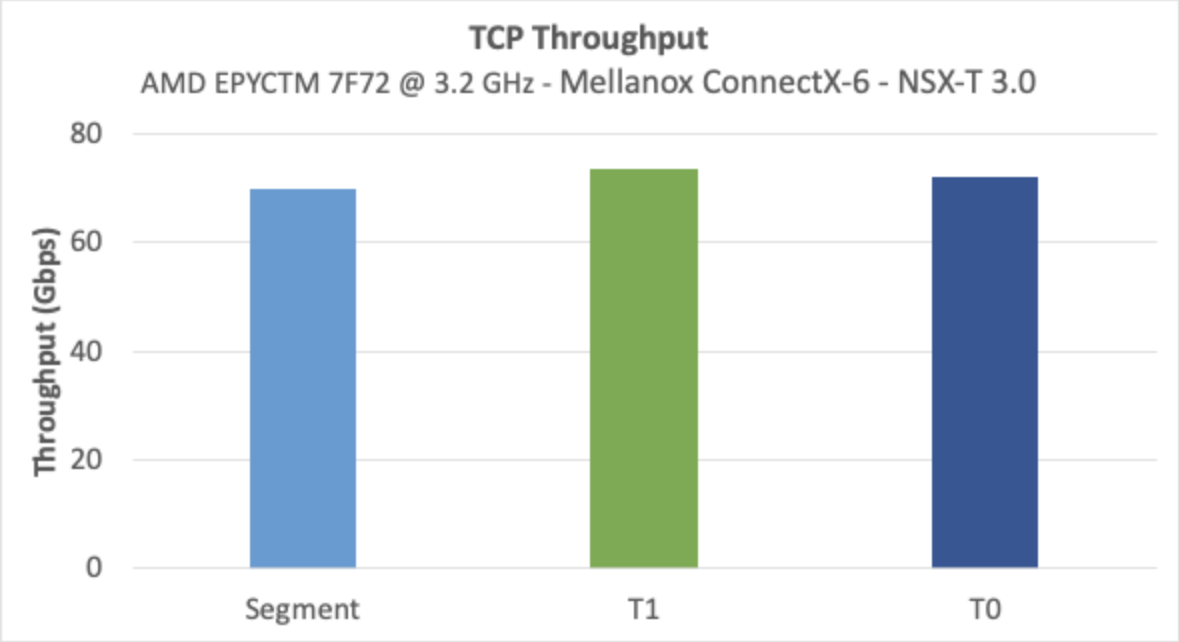


Figure 8-15: Throughput with AMD® EPYCTM 7F72 – MTU 1500 Bytes

8.3.1.7 Oversubscription

Oversubscription is a key consideration from a performance perspective. This should be taken into account from both host hardware design and also from the underlying physical.

8.3.1.7.1 Host hardware level Oversubscription

Hardware level details such as any oversubscription built into the hardware should be considered. For example, a PCIe Gen 3 dual port 40G NIC may not provide full 80G bidirectional throughput if it only has 8 lanes. The theoretical max throughput achieved on such a NIC would be limited by the number of PCIe lanes times the max throughput of each lane. Similarly, a chassis may have additional components such as IOModules that could limit the max theoretical throughput based on the type and number of units installed.

Such hardware level details must be considered, as they would restrict max achievable throughput, at a hardware level, even though the host may be presented with NIC that has faster port speeds.

8.3.1.7.2 Network Oversubscription

Network oversubscription, going from Top-of-rack (leaf) to the spine layer and above must also be considered. While the hosts may be presented with high speed ports, the actual throughput could potentially be bottlenecked at the physical network layer based on the oversubscription ratio.

This is especially true in cases where the traffic flows go beyond the top-of-rack layer to the next rack or going north out of the NSX domain through an NSX Edge. In these cases, over subscription ratio should be considered to ensure throughput is not throttled.

8.3.2 Number of pNICs

Adding multiple NICs in a ESXi host will help scale up the network performance of that host. Multiple NICs help by overcoming the limitations of a single NIC both from the port speed perspective and also from the number of queues available across all the pNICs. Considering most NICs have roughly 4 queues available, a 100G pNIC may have only 4 available queues vs a 4 x 25G pNICs could have 16 queues, 4 per pNIC. Thus, multiple pNICs help overcome any NIC hardware queue related limitations. Total number of available queues is an important factor to consider when deploying workloads that are primarily focused on packet processing, such as NSX VM Edge. In the case of normal data center workloads that can leverage offloads, multiple pNICs help overcome pNIC hardware level speed limitation. As noted in the RSS and section, each queue is serviced by a core or a thread. While these cores are not shared, meaning they are available for other activities based on need, at peak network utilization consider having enough cores to service all the used queues.

The following example shows the benefit of having two pNICs compared to a single pNIC using a single TEP vs Dual TEP. Dual TEP will help achieve twice the throughput of single TEP. The following image compares throughput with Single TEP vs throughput with Dual TEP using Intel® XL710 NICs on servers running on Intel® Xeon® Gold 6252 CPU @ 2.10GHz

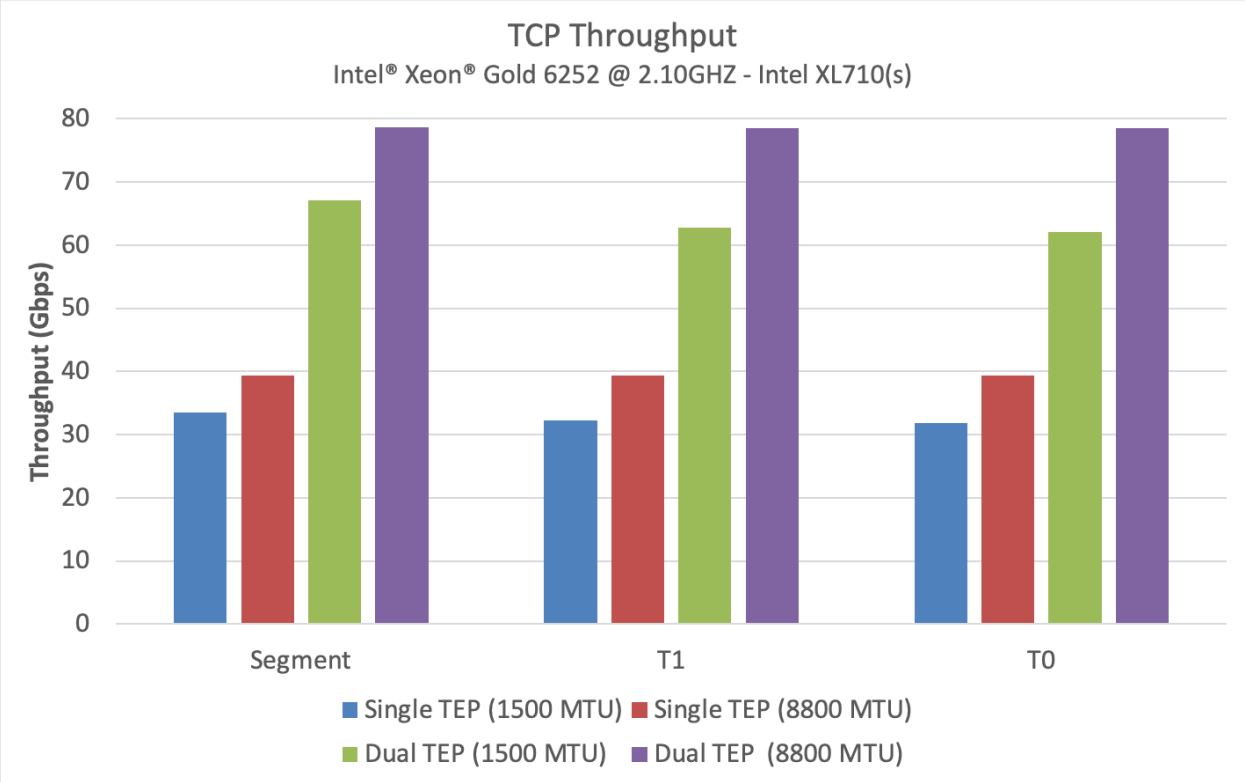


Figure 8-16: Throughput with Single TEP vs Dual TEP with Different MTUs

Note: Intel® XL710s are PCIe Gen 3 x8 lane NICs. On x8 lane NICs, the max throughput is limited to 64Gbps. To achieve the above near-line rate of 80Gbps, 2 x Intel® XL710s must be used.

8.3.3 vNIC Queues

Once the lower level details are tuned for optimal performance, the next consideration moves closer to the workload. At this point the question is, whether the workload is able to consume packets at the rate which ESXi is tuned to send? Note that this is generally not a concern when a single ESXi host is hosting many VMs, optimally, this should be twice the number of total queues provided by the pNIC. That is, if the total queues across all pNICs is 4, that ESXi host should have at least have 8 VMs. In this typical data center use case, while each VM may not be able to consume at the ESXi's packet processing rate, many VMs together may be able to reach that balance.

Queuing at the vNIC only becomes relevant for instances where the ESXi may be running 1 or 2 VMs and the expectation for the workload is pure packet processing, as in the case of Edge VM. In fact, enabling multiple queues is the recommended and default option for the Edge VMs. Similar to the RSS and queuing configuration at the pNIC level, vNIC also allows configuration for multiple queues instead of the default single queue. Edge VM section has more details on how to set this up.

While vNIC queues can be set for any VM. However, the workload running on the VM, should be able to leverage the multiple queues, in order to see a benefit. NSX VM Edge is designed to take advantage of multiple queues and should be leveraged for optimal performance.

8.3.4 vCPU Count

As discussed in the pNIC queues and vNIC queues topics, each queue needs a core/thread to service it. However, these cores are not dedicated to just servicing the queues but are engaged on demand when needed. That is, they may be used for workloads when there is no network traffic. Hence, a system should not only be tuned for max performance via multiple queues, but care should be taken that the system actually has access to the required number of cores to service those queues.

While the pNIC queues depend on the system cores, that is physical cores for packet processing, vNIC queues depend on the vCPU cores. This vCPU count should be considered when allocating vCPUs for the workload. However, as called out in the vNIC queuing section, this is only true for pure packet processing focused workloads such as NSX Edge VM. For most common data center TCP workloads, leveraging offloads such as Geneve offload, this is not a concern.

8.4 Jumbo MTU for Higher Throughput

Maximum Transmission Unit (MTU) denotes the maximum packet size that can be sent on the physical fabric. When setting this configuration on the ESX hosts and the physical fabric, Geneve header size has to be taken into consideration. Our general recommendation is to allow for at least 200 bytes buffer for Geneve headers in order to accommodate the option field for use cases such as service-insertion. As an example, if the VM's MTU is set to 1500 bytes, pNIC and the physical fabric should be set to 1700 or more. See Figure 8-17: MTU Configuration on MTU Configuration.

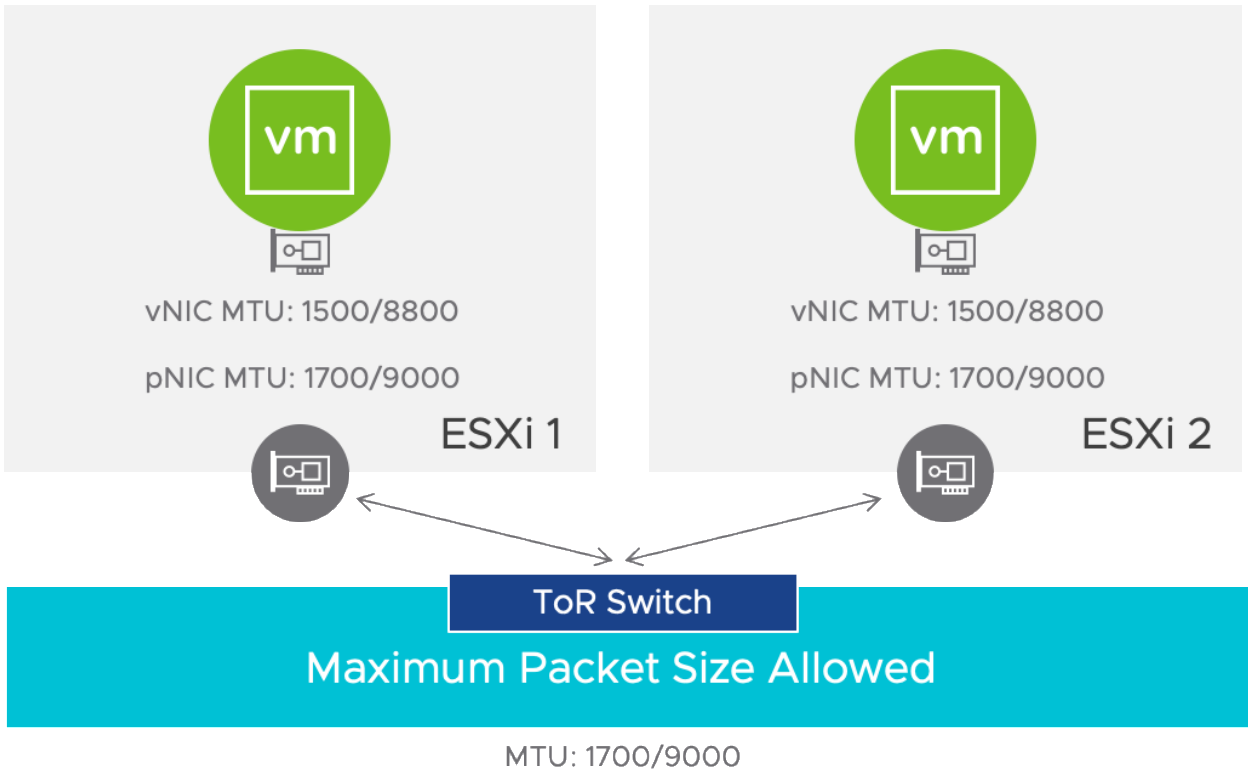


Figure 8-17: MTU Configuration

Why should you care about MTU values? MTU is a key factor for driving high throughput, and this is true for any NIC which supports 9K MTU also known as jumbo frame. The following graph FIGURE 8-18: MTU AND THROUGHPUT shows throughput achieved with MTU set to 1500 and 8800:

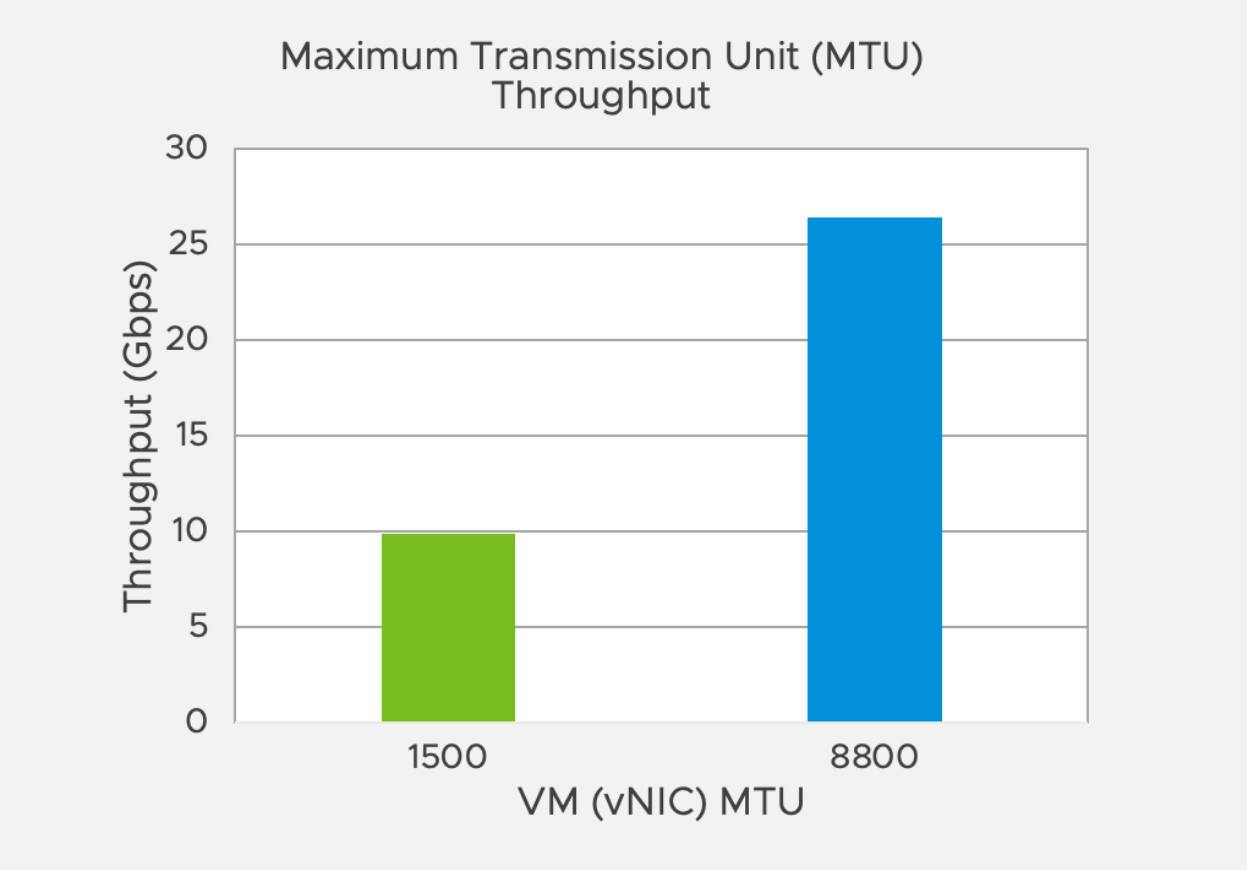


Figure 8-18: MTU and Throughput

Our recommendation for optimal throughput is to set the underlying fabric and ESX host's pNICs to 9000 and the VM vNIC MTU to 8800.

Notes for FIGURE 8-18: MTU AND THROUGHPUT:

- The above graph represents a single pair of VMs running iPerf with 4 sessions.
- For both VM MTU cases of 1500 and 8800, the MTU on the host was 9000 with demonstrated performance improvements.

8.4.1 Checking MTU on an ESXi host

Use the *esxcfg-nics* command to check the MTU:

```
[Host-1] esxcfg-nics -l | grep vmnic5
vmnic5  0000:84:00.0 i40en       Up   40000Mbps  Full
 3c:fd:fe:9d:2b:d0 9000   Intel Corporation Ethernet
Controller XL710 for 40GbE QSFP+
```

CLI 4 Check MTU on ESXi Host

For the VM, use the commands specific to the operating system for checking MTU. For example, "ifconfig" is one of the commands in Linux to check MTU.

8.5 Performance Factors for NSX Edges

8.5.1 Data Plane Development Kit (DPDK)

Data Plane Development Kit (DPDK) is a set of libraries and drivers to enable fast packet processing on a wide variety of CPU architectures including x86 platforms. DPDK is applicable for both the VM and bare metal Edge form factors, with the VM edge capable of delivering over 20Gbps throughput for standard TCP based DC workloads. Bare metal edge delivers over 35Gbps throughput for the same TCP-based DC workloads. The bare metal Edge form factor also excels at processing small packet sizes ~78 Bytes at near line on a 40Gbps port such as Intel® XL710, which is useful in NFV-style workloads. The following graph FIGURE 8-19: BARE METAL EDGE PERFORMANCE TEST (RFC2544) WITH IXIA shows the performance of bare metal Edge with a standard RFC 2544 test with IXIA.

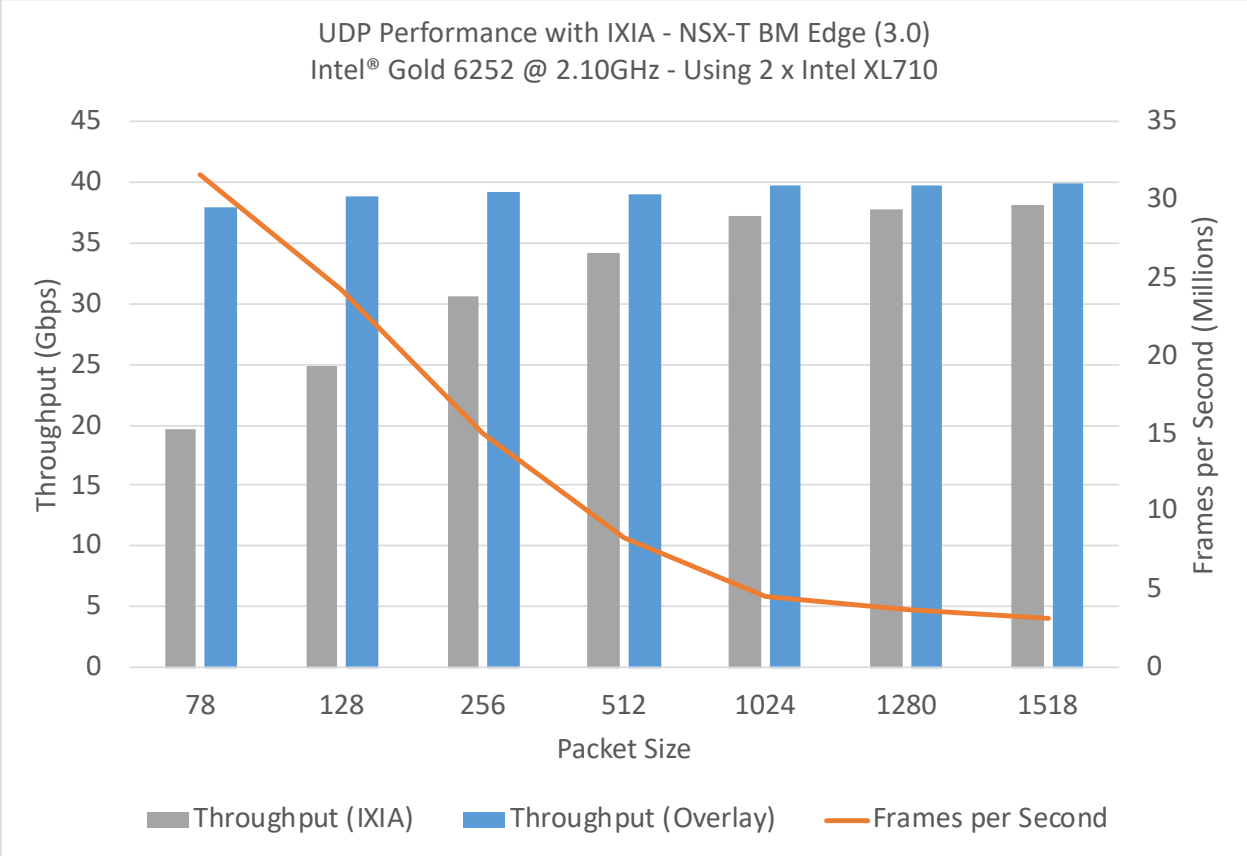


Figure 8-19: Bare Metal Edge Performance Test (RFC2544) with IXIA

Note: For the above test, the overlay lines in blue are calculated by adding throughput reported by IXIA and the Geneve Overlay header size.

8.5.2 SSL Offload

VMware NSX bare metal Edges also support SSL Offload. This configuration helps in reducing the CPU cycles spent on SSL Offload and also has a direct impact on

the throughput achieved. In the following image, Intel® QAT 8960s are used to show case the throughput achieved with SSL Offload.

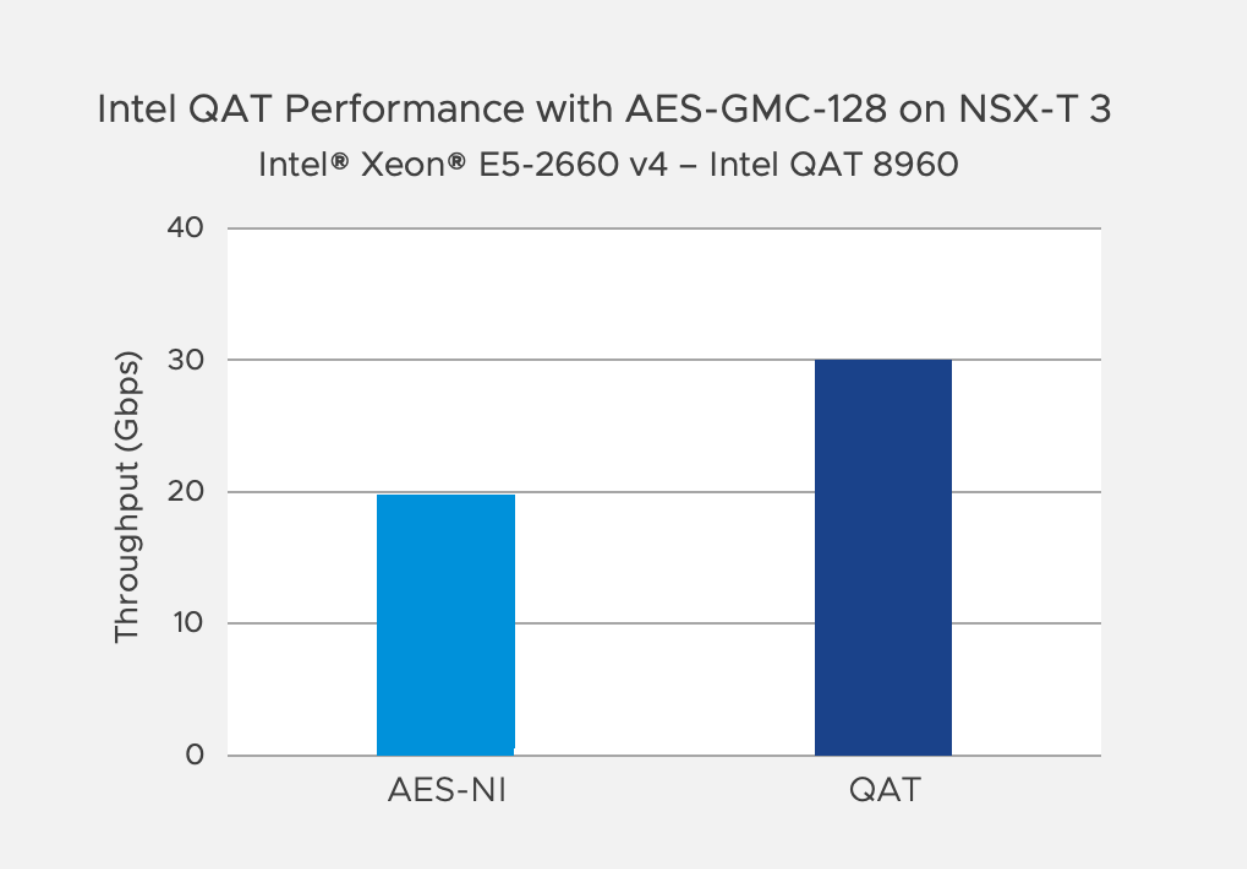


Figure 8-20: Throughput with SSL Offload

8.6 Key Performance Factors

One key to achieve better performance for compute, Edge VM, and bare metal Edge is to ensure having the right set of tools. While DPDK plays a key role, it's not the only factor affecting performance. The combination of DPDK and other capabilities of NIC drivers and hypervisor enhancements will help achieve optimal performance.

8.6.1 Compute Node Performance Factors

For compute clusters, our recommendation is to ensure the following two features are available on your NIC of choice:

- Geneve Offload
- Geneve Rx Filters

Geneve Offload helps decrease the CPU usage by offloading the task of dividing TSO segments into MTU-determined packets to the NIC card while also helping to increase throughput. Geneve Rx Filters help increase the number of cores used

to process incoming traffic, which is in turn increases performance by a factor of 4x times based on the number of hardware queues available on the NIC. For older cards which do not support Geneve Rx Filters, check whether they at minimum have RSS capability. The following graph shows the throughput achieved with and without using Geneve Rx Filters, 10Gbps vs near line rate:

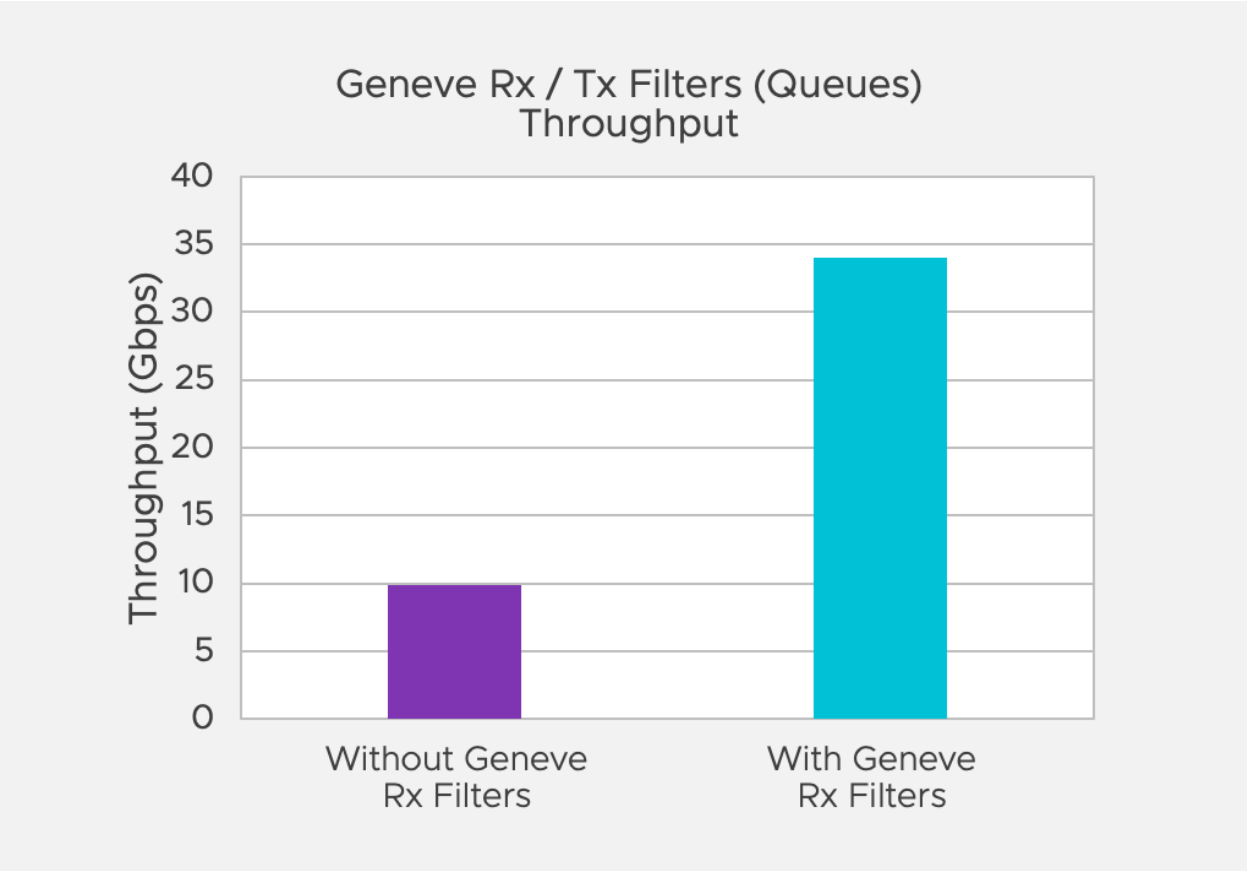


Figure 8-21: Throughput Improvement with Rx Filters

==========

FIGURE 8-21 Note: This test was run with LRO enabled, which is software-supported starting with ESX version 6.5 and higher on the latest NICs which support the Geneve Rx Filter. Thus, along with Rx Filters, LRO contributes to the higher throughput depicted here.

==========

8.6.2 VM Edge Node Performance Factors

In the case of edge clusters, DPDK is the key factor for performance. In the case of VM Edge, RSS-enabled NICs are best for optimal throughput.

RSS at pNIC

To achieve the best throughput performance, use an RSS-enabled NIC on the host running Edge VM, and ensure an appropriate driver which supports RSS is also being used. Use the VMware Compatibility Guide for I/O to confirm driver support.

RSS on VM Edge

For best results, enable RSS for Edge VMs. Following is the process to enable RSS on Edge VMs:

1. Shutdown the Edge VM
2. Find the ".vmx" associated with the Edge VM (https://kb.vmware.com/s/article/1714)
3. Change the following two parameters, for the Ethernet devices in use, inside the .vmx file
    a. ethernet3.ctxPerDev = "3"
    b. ethernet3.pnicFeatures = "4"
4. Start the Edge VM

Alternatively, use the vSphere Client to make the changes:

1. Right click on the appropriate Edge VM and click "Edit Settings":

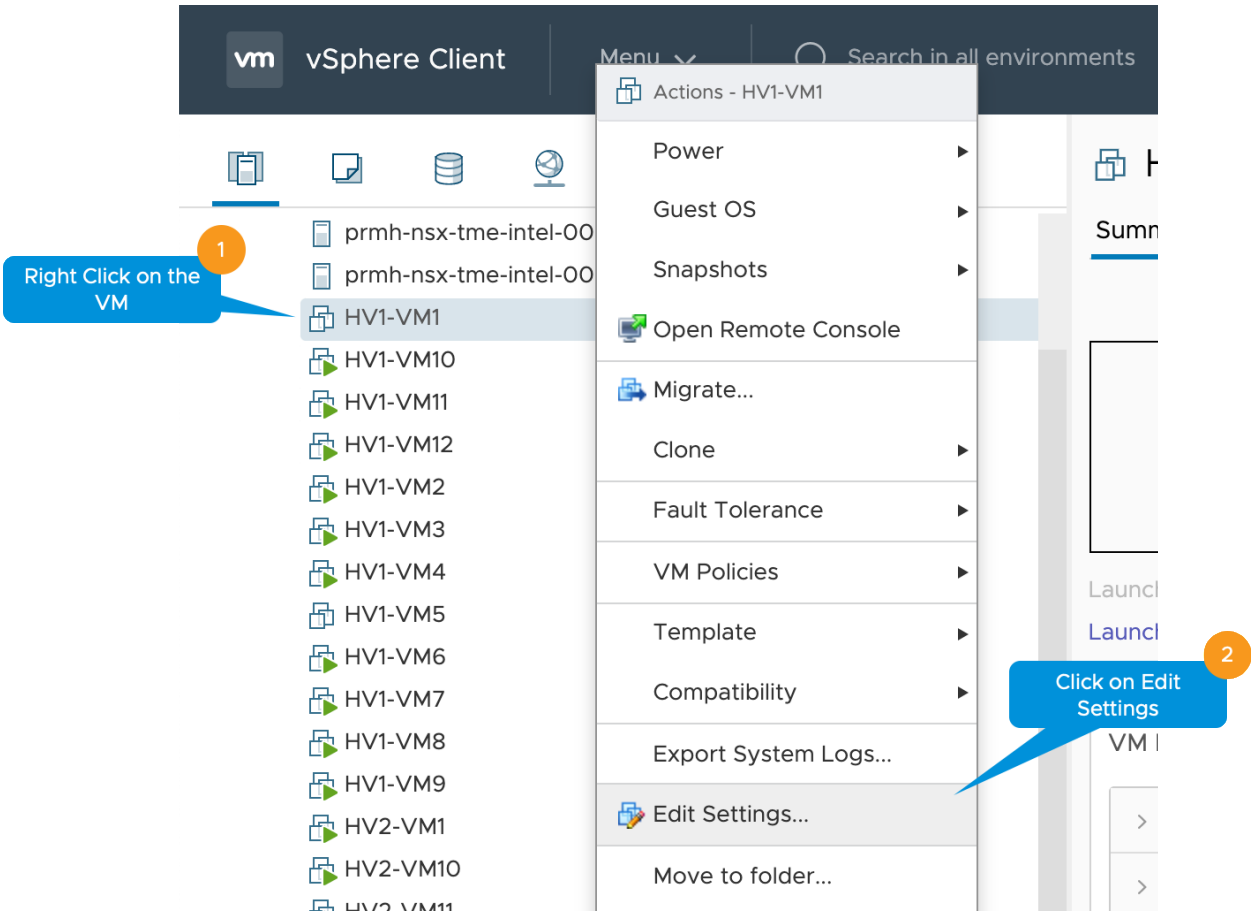


Figure 8-22: Change VM RSS Settings via vSphere Client – Part 1

2. Under “VM Options”, expand “Advanced” and click on “Edit Configuration”:

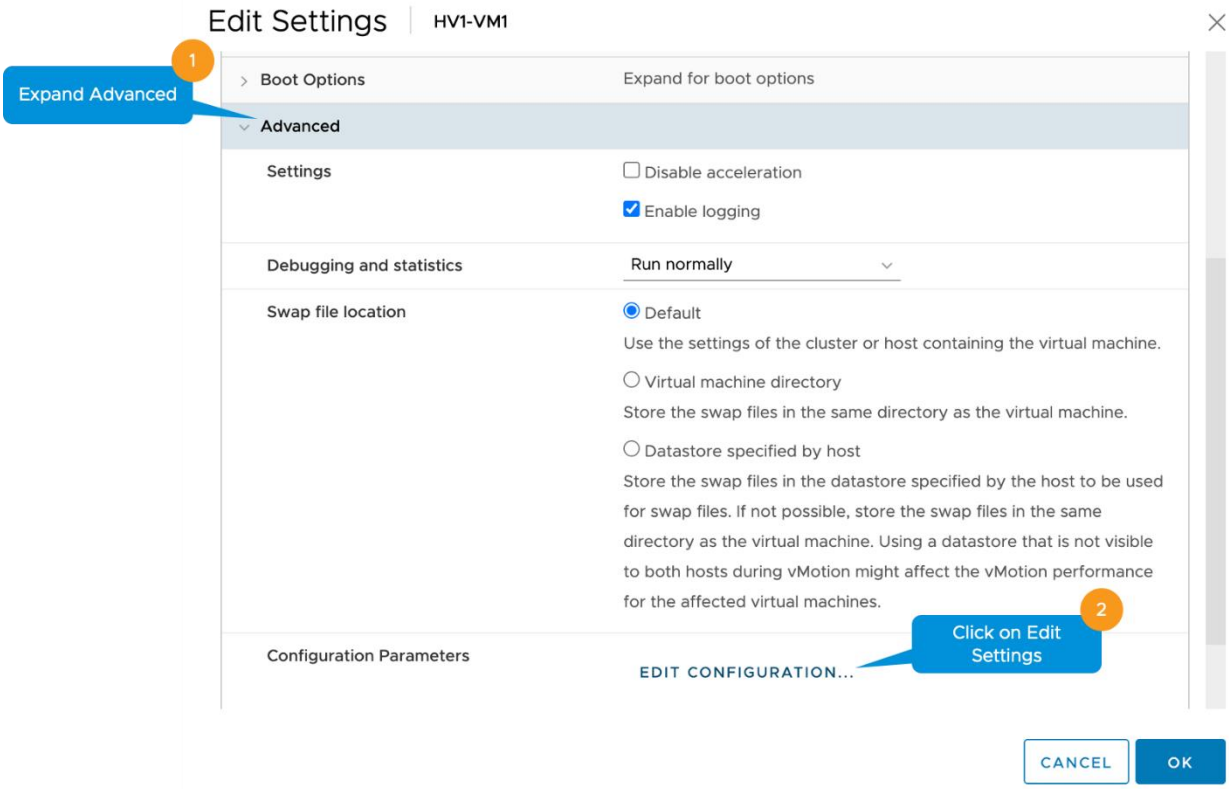


Figure 8-23: Change VM RSS Settings via vSphere Client – Part 2

3. Add the two configuration parameters by clicking on “Add Configuration” for each item to add:

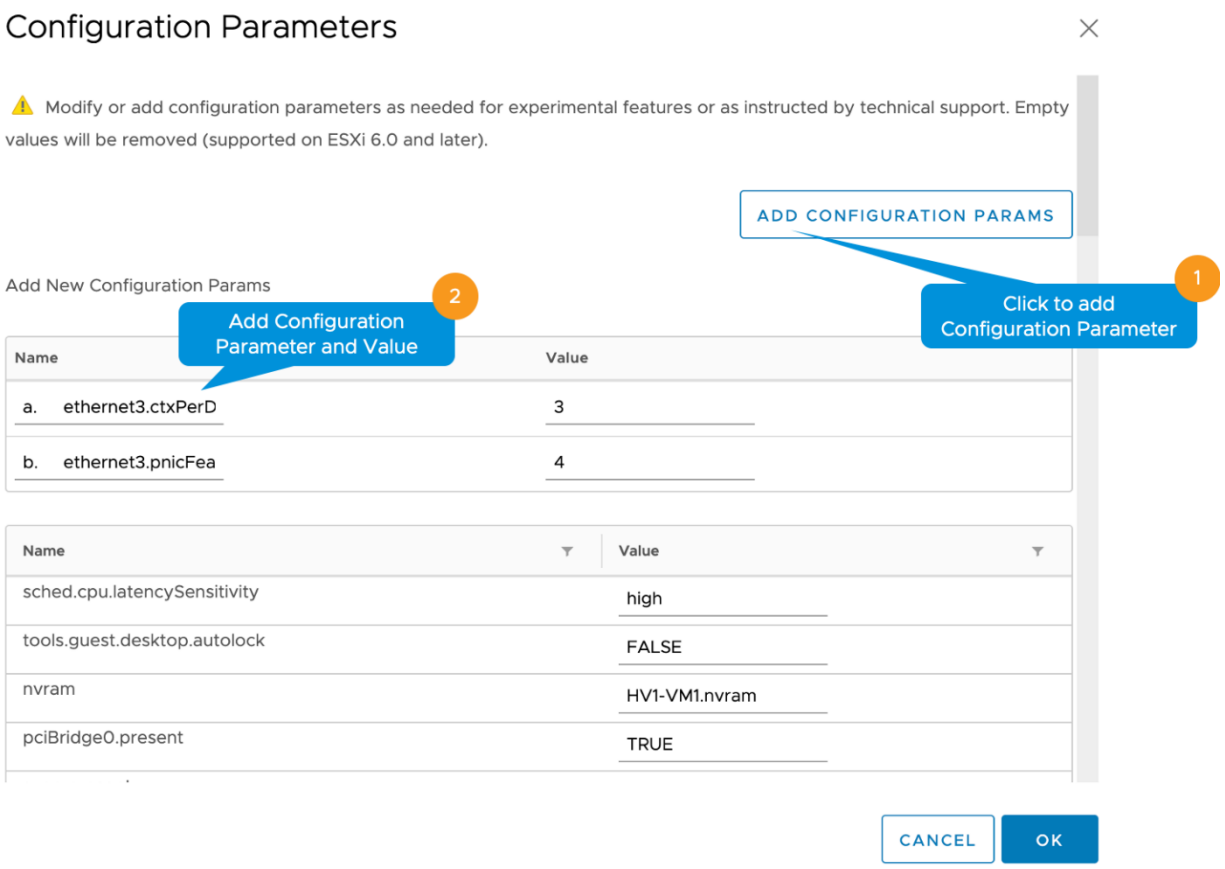


Figure 8-24: Change VM RSS Settings via vSphere Client – Part 3

The following graph FIGURE 8-25: RSS FOR VM EDGE shows the comparison of throughput between a NIC which supports RSS and a NIC which does not. Note: In both cases, even where the pNIC doesn't support RSS, RSS was enabled on the VM Edge:

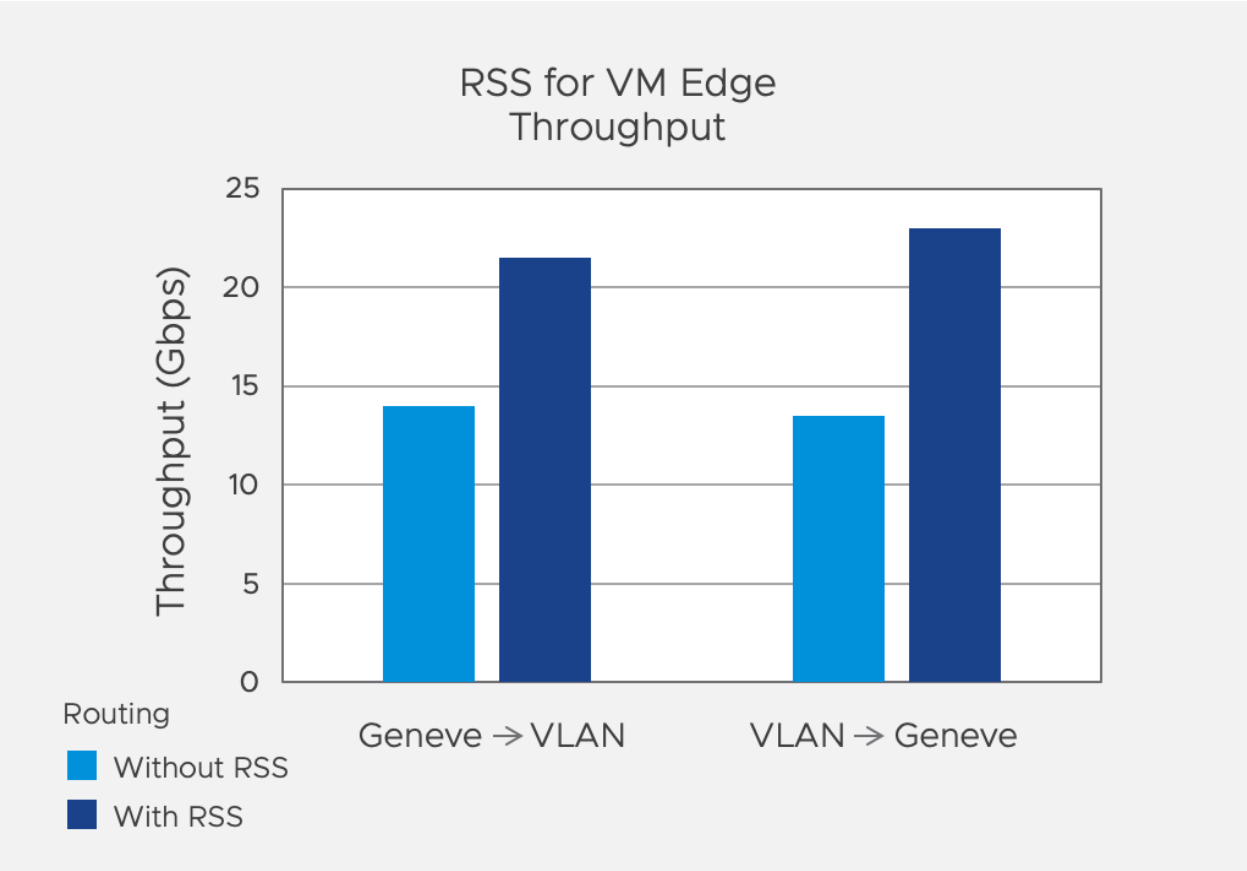


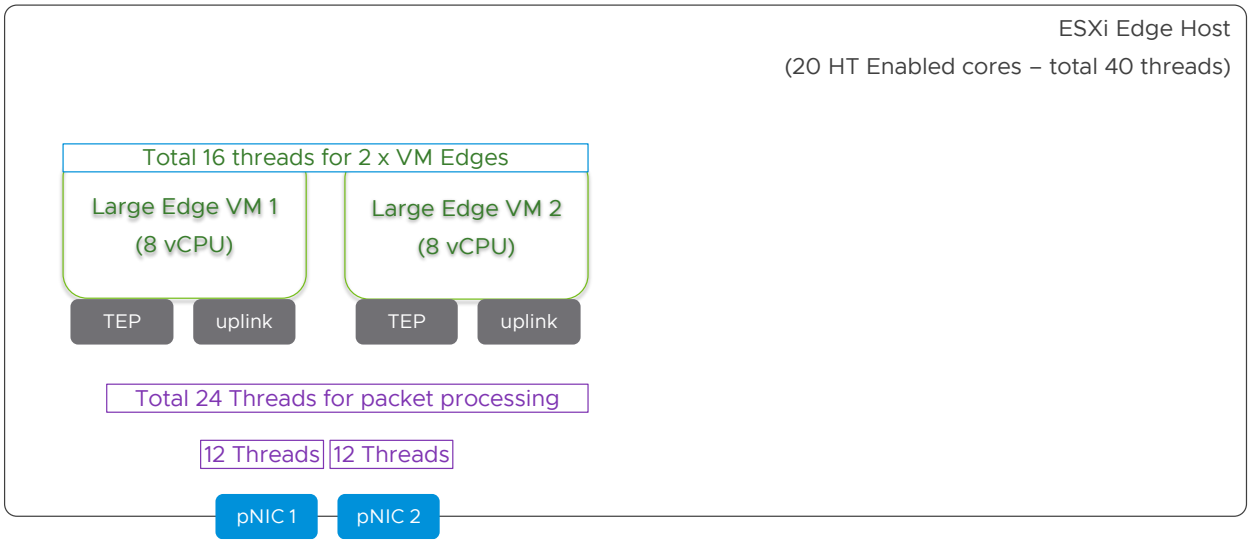
Figure 8-25: RSS for VM Edge

With an RSS-enabled NIC, a single Edge VM may be tuned to drive over 20Gbps throughput. As the above graph shows, RSS may not be required for 10Gbps NICs as they can achieve close to ~15 Gbps throughput even without enabling RSS.

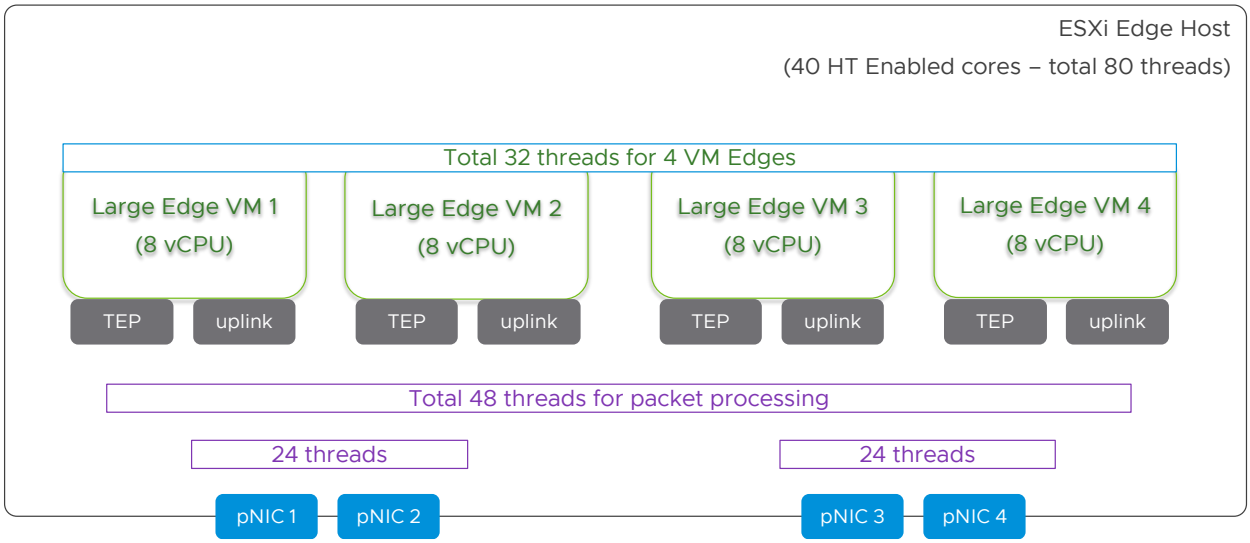
8.6.2.1 pNIC Count

When deploying VM Edges, if performance is a concern, recommendation is to use a dedicated edge node cluster. Given that most modern systems are built by default with dual sockets and multiple core processors, running a single VM edge on such a server may leave a lot of CPU power unutilized. Which begs the question of running multiple edge VMs on each hypervisor.

To run multiple Edge VMs on the same hypervisor, we must consider the available CPU count and also the count of dedicated pNICs available for the VM Edge. For each VM Edge, the recommendation is to dedicate at least 2 pNICs. This is based on the packet processing capacity of RSS enabled pNIC, is roughly around a million+ packets per second. Packet processing capacity of Edge VM is also around 1 million+ packets per second. Given that, a pair of pNICs may be sufficient to run a pair of Edge VMs for most common DC workloads. With that configuration, considering most modern systems have many cores to spare, the following image illustrates that the entire may not be utilized optimally, based on the number of available cores.



In the above image, if the Edge host has a total of 20 HT Enabled cores (40 threads), then a pair of Edge VMs may be optimal. However, if the system more cores to spare, the recommended way to leverage those cores is to add more pNICs dedicated to an additional pair of Edge VMs. In the following image, 4 Edge VMs are spread across 4 pNICs, on a host that has 40 HT enabled cores (80 threads)



Note: For workloads that are heavy on packet processing, it would be better to dedicate a pair of pNICs to a single edge VM.

In summary, increasing the VM Edge count along with increasing the dedicated pNIC count could help leverage all the cores on a dedicated Edge host.

8.6.3 Bare Metal Edge Node Performance Factors

Bare metal Edge has specific requirements dependent on the type of NIC used. Please refer to the NSX INSTALLATION GUIDE for details on currently supported cards.

While VM and bare metal Edges leverage Data Path Development Kit (DPDK), they differ in deployment. While VM Edge does not have restrictions on the physical NIC which can be used because VMXNET3 provides DPDK functionality for the VM Edge, bare metal Edge does have strict restrictions on the NICs which may be used. (Note however, bare metal Edge nodes do support Intel® QATs for SSL Offload.) Please refer to the relevant hardware requirements section of the NSX installation guide for specific details on compatibility. The features required on the NIC for a Bare metal Edge are enabled by default, before the data path gets enabled. For example, RSS is enabled by default on bare metal edges and does not require any configuration from the user. To check RSS status on a bare metal edge, following CLI may be used:

```
get dataplane | find rss
```

The following LINK shows the general system requirements for NSX 3.2 components.

From the above page, for requirements specific to the NSX bare metal Edge, click on NSX Edge Bare Metal Requirements.

8.6.4 Summary of Performance – NSX Components to NIC Features

The following table (TABLE 8-1 SUMMARY OF PERFORMANCE IMPROVEMENTS WITH VARIOUS CAPABILITIES) provides an overall view and guidance on NSX components and NIC features per given use case. For common datacenter applications and deployments, Standard N-VDS is the recommendation. Enhanced Data Path is only meant for NFV style workloads with fast packet processing requirements.

Table 8-1 Summary of Performance Improvements with Various Capabilities

	Compute Transport Nodes (N-VDS Standard)	Compute Transport Nodes ("Enhanced Data Path")	ESXi nodes with VM Edges	Bare Metal Edge
Features that Matter	**Geneve-Offload:** To save on CPU cycles **Geneve-RxFilters**: To increase throughput by using more cores and using software based LRO **RSS** (if Geneve-RxFilters does not exist): To increase	**N-VDS Enhanced Data Path**: For DPDK-like capabilities	**RSS**: To leverage multiple cores	**DPDK**: Poll mode driver with memory-related enhancements to help maximize packet processing speed **QATs**: For high encrypt/decrypt

	throughput by using more cores			performance with SSL-offload
Benefits	High Throughput for typical TCP-based DC Workloads	Maximum PPS for NFV style workloads	Maximize Throughput for typical TCP based Workloads with Edge VM VM Tuning + NIC with RSS Support Add/Edit two parameters to the Edge VM's vmx file and restart	Maximum PPS Maximum Throughput even for small packet sizes Maximum encrypt/decrypt performance with SSL Offload Low latency Maximum Scale Fast Failover

Table 8-2 Summary of Performance Improvements with Various Capabilities

8.7 Results

The following section takes a look at the results achievable under various scenarios with hardware designed for performance. First, here are the test bed specs and methodology:

Compute	Virtual Machine	Edge Bare Metal	Test Tools
• CPU: Intel(R) Xeon(R) Gold 6252 CPU @ 2.10GHz • RAM: 192 GB • Hyper Threading: Enabled • MTU: 1700 • NIC: XL710 • NIC Driver: i40e - 1.3.1-18vmw.670.0.0.8169922 • ESXi 6.7	vCPU: 2 RAM: 2 GB Network: VMXNET 3 MTU: 1500	CPU: Intel ® Xeon ® E5-2637 v4 3.5Ghz • RAM: 256 GB • Hyper Threading : Enabled • MTU: 1700 • NIC: XL710 • NIC Driver: In-Box	iPerf 2 (2.0.5) with • 4 – 12 VM Pairs • 4 Threads per VM Pair • 30 seconds per test • Average over three iterations

Table 8-3 Specific Configuration of Performance Results

NSX Components

- Segments
- T1 Gateways
- T0 Gateways
- Distributed Firewall: Enabled with default rules
- NAT: 12 rules – one per VM
- Bridging
    a. Six Bridge Backed Segments
    b. Six Bridge Backed Segments + Routing

The following graph shows in every scenario above, NSX throughput performance stays consistently close to line rate on an Intel® XL710 40Gbps NIC.

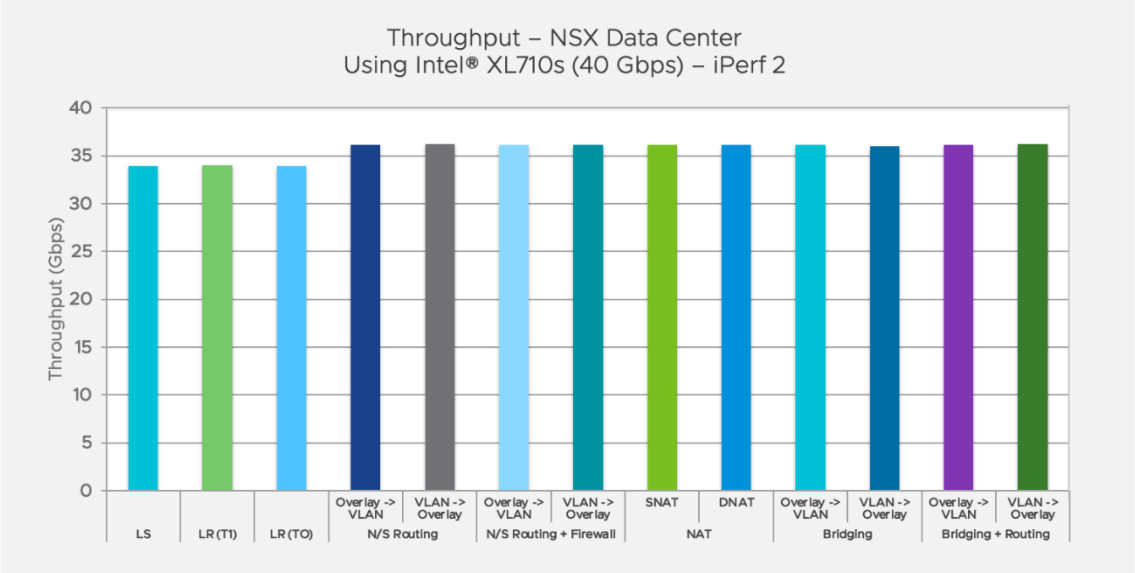


Figure 8-26: Throughput Summary for NSX Based Datacenter

8.8 NFV: Raw Packet Processing Performance

TCP-based workloads are generally optimized for throughput and are not sensitive to raw packet processing speed. NFV-style workloads are on the opposite end where the raw packet processing is key. For these specific workloads, NSX provides an enhanced version of N-VDS called N-VDS enhanced.

8.8.1 N-VDS Enhanced Data Path

Based on the DPDK-like features such as Poll Mode Driver (PMD), CPU affinity, and optimization and buffer management, N-VDS Enhanced caters to applications requiring high speed raw packet processing, for details on the N-VDS enhanced switch and its application, please refer to the resource below:

8.9 Acceleration with the N-VDS in Enhanced Datapath Mode

https://docs.vmware.com/en/VMware-vCloud-NFV-OpenStack-Edition/3.1/vmware-vcloud-nfv-openstack-edition-ra31/GUID-0695F86B-20AD-4BFB-94E0-5EAF94087758.html?hWord=N4IghgNiBcIHIFoBqARAygAgKIDsAWYOAxgKYAmlAvkA

8.9.1 Poll Mode Driver

One of the recent key changes with DPDK is the addition of Poll Mode Driver (PMD). With the Poll Mode Driver, instead of the NIC sending an interrupt to the CPU once a packet arrives, a core is assigned to poll the NIC to check for packets. This polling procedures eliminates CPU context switching, unavoidable in the traditional interrupt mode of packet processing, resulting in higher packet processing performance.

8.9.2 CPU Affinity and Optimization

With DPDK, dedicated cores are assigned to process packets. This assignment procedure ensures consistent latency in packet processing and enables instruction sets such SSE, which helps with floating point calculations, to be available where needed.

8.9.3 Buffer Management

Buffer management is optimized to represent the packets being processed in simpler fashion with low footprint, assisting with faster memory allocation and processing. Buffer allocation is also Non-uniform memory access (NUMA) aware. NUMA awareness reduces traffic flows between the NUMA nodes, thus improving overall throughput.

Instead of requiring regular packet handlers for packets, Enhanced Datapath uses mbuf, a library to allocate and free buffers resulting in packet-related info with low overhead. As traditional packet handlers have heavy overhead for initialization, mbuf simplifies packet descriptors by decreasing the CPU overhead for packet initialization. To further support the mbuf-based packet, VMXNET3 has also been enhanced. In addition to the above DPDK enhancements, ESX TCP Stack has also been optimized with features such as Flow Cache.

8.9.4 Flow Cache

Flow Cache is an optimization enhancement which helps reduce CPU cycles spent on known flows. With the start of a new flow, Flow Cache tables are immediately populated. This procedure enables follow-up decisions for the rest of packets within a flow to be skipped if the flow already exists in the flow table. Flow Cache uses two mechanisms to figure out fast path decisions for packets in a flow:

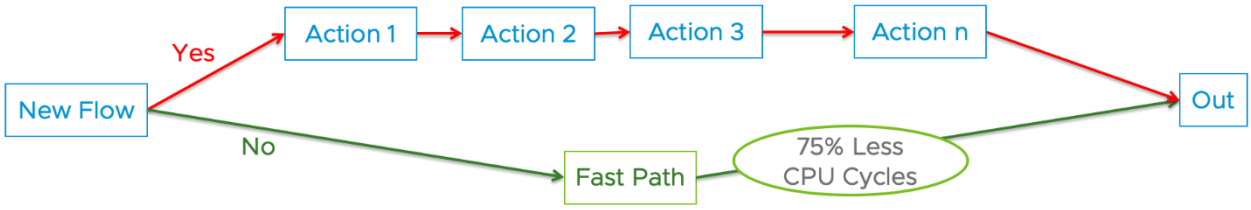


Figure 8-27: Flow Cache Pipeline

If the packets from the same flow arrive consecutively, the fast path decision for that packet is stored in memory and applied directly for the rest of the packets in that cluster of packets.

If packets are from different flows, the decision per flow is saved to a hash table and used to decide the next hop for packets in each of the flows. Flow Cache helps reduce CPU cycles by as much as 75%, a substantial improvement.

8.9.5 Checking whether a NIC is N-VDS Enhanced Data Path Capable

Use the VMware IO Compatibility Guide (VCG I/O) described in the previous sections to find out which cards are N-VDS (E) capable. The feature to look for is "N-VDS Enhanced Data Path" highlighted in blue in the following image:

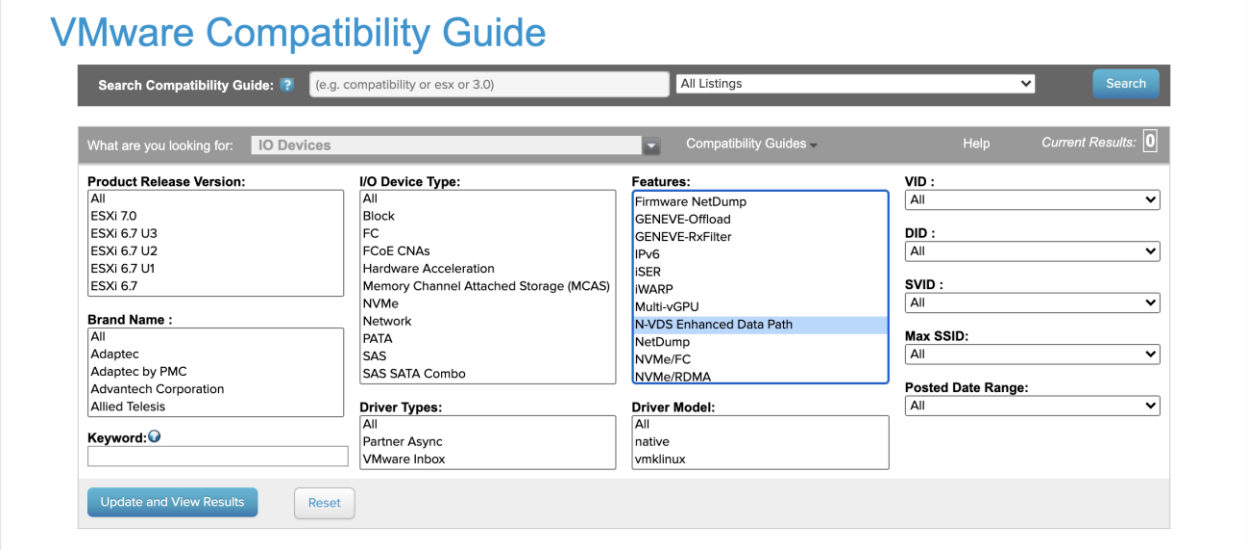


Figure 8-28: VMware Compatibility Guide for I/O - Selection Step for N-VDS Enhanced Data Path

==========

Note: N-VDS Enhanced Data Path cannot share the pNIC with N-VDS - they both need a dedicated pNIC.

==========

8.10 Benchmarking Tools

8.10.1 Compute

On the compute side, our recommendation for testing the software components is to use a benchmarking tool close to the application layer. Application layer benchmarking tools will help take advantage of many features and show the true performance characteristics of the system. While application benchmarking tools are ideal, they may not be very easy to setup and run. In such cases, iPerf is a great tool to quickly setup and check throughput. Netperf is another tool to help check both throughput and latency.

Here is a github resource for an example script to run iPerf on multiple VMs simultaneously and summarize results: HTTPS://GITHUB.COM/VMWARE-SAMPLES/NSX-PERFORMANCE-TESTING-SCRIPTS

8.11 Edges

8.11.1 VM Edge

As VM Edges are designed for typical DC workloads, application layer tools are best for testing VM Edge performance.

8.11.2 Bare Metal Edge

With the Bare Metal Edges, either application layer benchmarking tools or typical network benchmarking and packet generation tools of choice, such as Keysight PathWave (formerly IXIA) or Spirent Network Emulator may be used. One of the challenges with using a hardware benchmarking tool is to find one which includes provision for Geneve encap/decap. Following is a topology with two bare metal edges, each within its own cluster. A segment, aptly called a crosslink segment, connects them both over overlay. PathWave sends and receives only non-overlay packets. However, the segment between the two Edge clusters forces the packets to use overlay. Check the image below for details.

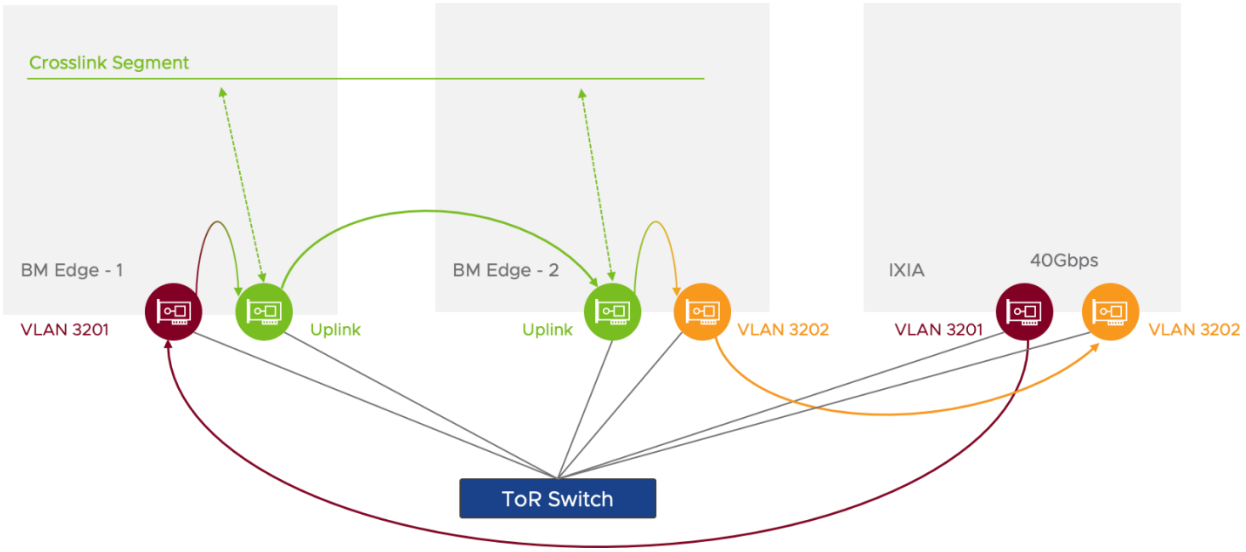


Figure 8-29: Example Topology to Use Geneve Overlay with Hardware IXIA or Spirent

An alternative approach is to use a software tool such as Pktgen (HTTPS://GITHUB.COM/PKTGEN/PKTGEN-DPDK).

8.12 Conclusion

To drive enhanced performance, NSX uses a number of features supported in hardware.

On the compute side, these are:

1. Geneve Offload for CPU cycle reduction and marginal performance benefits
2. Geneve Rx Filters, an intelligent queuing mechanism to multiply throughput
3. RSS an older hardware-based queuing mechanism – alternative if Rx Filters are missing
4. Jumbo MTU an age-old trick to enable high throughput on NICs lacking above features
5. NIC port speed
6. Number of NICs – single vs dual TEP
7. PCIe Gen 3 with x16 lane NICs or multiple PCIe Gen 3 x8 NICs
8. PCIe Gen 4

For compute workloads focused on high packet processing rate for primarily small packet sizes, Enhanced Data Path Enabled NICs provide a performance boost.

For the Edges, if its VM Edges, then:

1. NIC cards which support RSS and
2. Enabling RSS on both the pNIC and the Edge VM NIC (vNIC)

For the Bare Metal Edges, leveraging optimal SSL offload performance such as Intel® QAT 8960s and deploying supported hardware from the VMware NSX install guide will result in performance gains.