Network Virtualization in Data Centers

André Silva up202107419 **Leonor Guedes** up202107691 **Rafael Parody** up202502656

Carried out within the scope of the
"Telecommunications Networks" course
of the Master's in Electrical and Computer Engineering

7 of November of 2025

Abstract

Contents

List of Figures

List of Abbreviations

DC	Data Center
ECMP	Equal-Cost Multi-Path
eBGP	External Border Gateway Protocol
Geneve	Generic Network Virtualization Encapsulation
GRE	Generic Routing Encapsulation
GTEP	Geneve Tunnel Endpoint
HNV	Hyper-V Network Virtualization
IP	Internet Protocol
LAN	Local Area Network
MTU	Maximum Transmission Unit
NVGRE	Network Virtualization using Generic Routing Encapsulation
OAM	Operations, Administration and Maintenance
OSPF	Open Shortest Path First
OSI	Open Systems Interconnection
OVS	Open vSwitch
TLV	Type-Length-Value
UDP	User Datagram Protocol
VDC	Virtualized Data Center
VLAN	Virtual Local Area Network
VNI	VXLAN Network Identifier
VN	Virtual Network
VTEP	VXLAN Tunnel Endpoint
VXLAN	Virtual Extensible LAN
VM	Virtual Machine

1 Introduction

Nowadays, modern organizations rely on rapidly growing data centers that support high workloads. Therefore, the data center network must be able to isolate and move data loads, even scaling, without complicating the physical core of the network—that is, by modifying only the logical layer.

Traditional VLAN-based techniques do not provide sufficient capacity for today's service needs. Therefore, network overlays allow us to create logical virtual networks on an underlay (link layer), encapsulating traffic. Among the technologies considered are VXLAN and NVGRE, which may include Geneve.

Work in progress

2 Data Center

This section aims to provides an overview of the main architectural principles that underpin modern data centers. It is explored the separation between the physical and virtual network layers as well as how these components interact to deliver scalability and reliability in large-scale environments.

2.1 Overview

Data centers are physical facilities designed to house information systems and telecommunications equipment, as well as to process and distribute large volumes of data securely and efficiently. They integrate cooling systems, redundant power supplies and both physical and logical controls to ensure the continuous and reliable operation of organizations.

A data centers is generally divided into five main blocks. The first is the physical infrastructure, which encompasses racks, structured cabling, cooling systems, and fire detection and suppression systems. The second is data processing and storage, including both physical and virtualized servers, storage arrays, and associated systems. The third component is the network infrastructure, the key to operations, which interconnects the data center with external environments, such as cloud services or remote sites, comprising routers, firewalls and switches. The fourth component is the management system, responsible for event logging, continuous monitoring, and maintaining environmental stability. Finally, the fifth component corresponds to the security mechanisms, which protects data through physical and logical controls, including network segmentation (VLAN, VNI), access policies and encryption in transit and at rest.

The predominant network architecture in today's data centers is the leaf-spine IP mesh, which operates at Layer 3 of the OSI model. The underlay typically employs a high Maximum Transmission Unit (MTU) to support encapsulation overhead and relies on stable routing protocols, such as eBGP and OSPF. The overlay network is built atop this physical fabric, providing logical segmentation and workload mobility without IP addresses modification. Technologies such as VXLAN, NVGRE and Geneve are central to this virtualized architecture and will be examined in greater depth in subsequent sections.

2.2 Data Center Network Architecture

The foundation of modern data centers lies in the separation between the underlay, which represents the physical network infrastructure, and the overlay, which comprises the virtual networks.

The underlay corresponds to the physical IP fabric of the network, typically organized in a Layer 3 leaf-spine topology. This topology, as illustrated in Figure 1, consists of two layers: the spine, formed by high-speed switches that constitute the network backbone; and the leaf, consisting of access switches that connect servers, storage devices and other endpoints. The primary objective of the underlay is to transport packets quickly and efficiently. For this reason, a high MTU, usually around 9000 bytes, is used to to accommodate encapsulation overhead and minimize fragmentation.

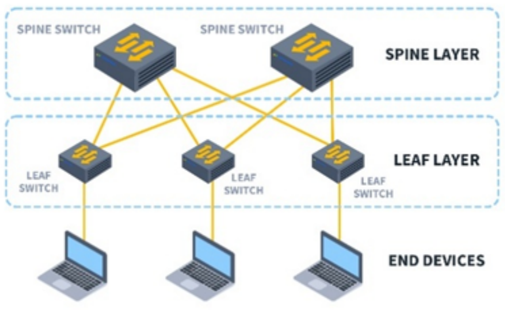

Figure 1: Leaf-spine structure.

The overlay is built on top of this fabric and consists of a set of virtual networks that encapsulate each user's traffic, providing logical isolation and mobility, without changing IP addresses. In practice, encapsulation mechanisms such as VXLAN, NVGRE or Geneve are employed. Each virtual network is identified by a unique segment ID, enabling scalable segmentation within large infrastructures. The central component of the overlay is the VXLAN Tunnel Endpoint (VTEP), typically residing on a leaf switch, responsible for encapsulating and decapsulating traffic between the overlay and the IP underlay. These virtual networks are often integrated with orchestration and automation platforms, which facilitate dynamic provisioning, policy enforcement and scalability.

Correct MTU configuration within the underlay is critical, as an insufficient MTU will result in packet packet fragmentation and drops. Therefore, it is essential to maintain a adequately large MTUs and continuously monitor fragmentation and transmission metrics. The use of UDP-based encapsulation enhances multiple available paths through Equal-Cost Multi-Path (ECMP) routing, improving flow distribution, maximizing the inherent parallelism of the leaf-spine architecture and improving overall network efficiency.

2.3 Data Centers Virtualization

A Virtualized Data Center (VDC) is a data center where some or all the hardware components, such as servers, routers, switches and network links, are virtualized. Typically, a physical hardware is virtualized using software or firmware known as hypervisor, which divides the equipment into multiple isolated and independent virtual instances.

A VDC is defined as a collection of virtual resources, including VMs, virtual switches, and virtual routers, connected via virtual links and therefore constitutes a segment of a Virtual Data Center. While a Virtualized Data Center is a physical data center with deployed resource virtualization techniques, a Virtual Data Center is a logical instance of a Virtualized Data Center consisting of a subset of the physical data center resources.

A Virtual Network (VN) is a set of virtual networking resources, such as virtual nodes (end-hosts, switches, routers) and virtual links and therefore, a VN is a part of a VDC. A network virtualization level is one of the layers of the network stack (application to physical) in which the virtualization is introduced. Figure 2, shows how several VDCs can be deployed over a virtualized data center.

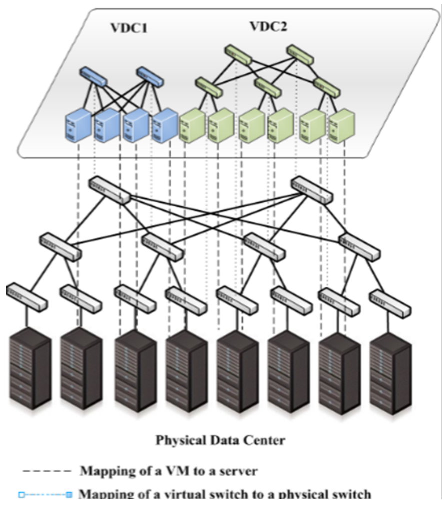
VDC1
VDC2
Physical Data Center
----- Mapping of a VM to a server
Mapping of a virtual switch to a physical switch

Figure 2: Virtualization of a Data Center.

3 Overlay Network Technologies

After defining the fundamental architecture of modern data centers, the following section examines the core technologies that enable network virtualization.

3.1 Virtual Extensible LAN

Traditional Virtual Local Area Networks (VLANs) have long been used to logically segment Layer 2 networks within a limited broadcast domain, identified by a 12-bit VLAN ID that supports up to 4096 tenants. In conventional Layer 2 switches, communication between servers connected to the same device is natively supported. When a server is migrated from one port of the Layer 2 switch to another port, its IP address can remain unchanged, satisfying the requirements for dynamic VM migration. However, as data centers evolved, VLANs began to show limitations, particularly in large-scale environments that require workload mobility and multi-tenant isolation. To overcome these constraints, the Virtual Extensible LAN (VXLAN) technology was introduced.

VXLAN is a network virtualization technology designed to extend Layer 2 connectivity across Layer 3 networks. It establishes a logical tunnel between network devices, through which it employs a MAC-in-UDP encapsulation mechanism for packet transport. In this model, original Ethernet frames generated by a Virtual Machine (VM) are encapsulated into UDP packets. These packets are then wrapped with IP and Ethernet headers of the physical network as outer headers, allowing them to be routed as standard IP traffic. This design removes the structural limitations of traditional Ethernet domains and enables greater scalability and isolation within modern data centers.

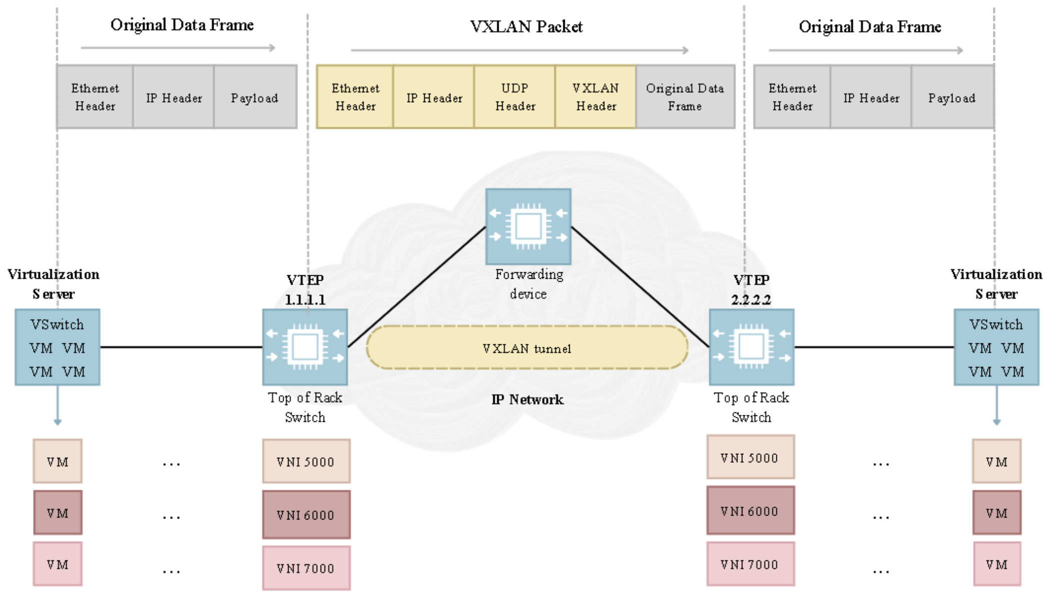

Figure 3: VXLAN network model.

As shown in Figure 3, VXLAN extends Layer 2 communication over an IP underlay, creating a virtualized network that behaves as a single logical switch interconnecting all endpoints. Any two nodes can communicate transparently through VXLAN tunnels, regardless of the underlying physical topology. From the server's perspective, VXLAN virtualizes the entire infrastructure network into a large "Layer 2 virtual switch", interconnecting all servers as if they resided on the same Layer 2 segment. VXLAN therefore plays an important role in modern data centers as it leads to a significant increase in the number of tenants that the network can effectively isolate and manage.

In terms of scalability, VXLAN uses a 24-bit VXLAN Network Identifier (VNI), allowing the identification of up to 16 million tenants, far higher than the supported by VLANs. Furthermore, in terms of migration flexibility, VXLAN establishes virtual tunnels between switches across the underlying IP network, effectively virtualizing the infrastructure into a large logical Layer 2 domain that supports large-scale dynamic VM migration.

3.1.1 VXLAN Encapsulation Structure

The VXLAN encapsulation process is illustrated in Figure 4, where the original Ethernet frame is encapsulated within UDP and IPv4 headers and transmitted over the underlay network.

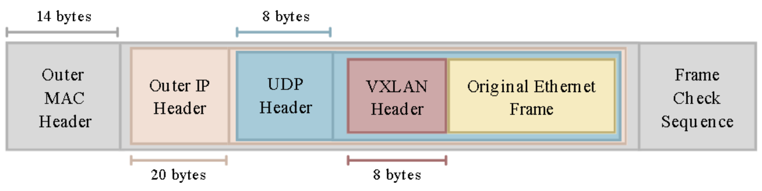

Figure 4: VXLAN packet format. [1]

The VXLAN header, with a total length of 8 bytes, carries a 24-bit VNI, which uniquely identifies each virtual network or tenants within the VXLAN network. It also contains an 8-bit Flags field, typically set to 00001000, indicating the presence of a valid VNI, along with two reserved fields of 24 and 8 bits that are maintained for future use and protocol consistency.

The VXLAN header and the original Ethernet frame are transported as UDP data. Within the UDP header, the destination port number is fixed at 4789, while the source port number is dynamically calculated using a hash algorithm based on the original Ethernet frame.

In the outer IP header, the source IP address corresponds to the VXLAN Tunnel Endpoint (VTEP) connected to the source VM, while the destination IP address identifies the VTEP connected to the destination VM. These addresses enable communication across the IP underlay network.

Finally, the outer MAC header, also known as the outer Ethernet header, contains the source and destination MAC addresses used for physical network forwarding. The source MAC address represents the VTEP initiating the transmission, while the destination MAC address corresponds to the next hop on the path toward the destination VTEP.

3.2 Network Virtualization using Generic Routing Encapsulation

Network Virtualization using Generic Routing Encapsulation (NVGRE) is a network virtualization mechanism that extends Layer 2 domains over Layer 3 IP infrastructures. Its purpose is to preserve transparency for virtual machines by encapsulating Ethernet frames within IP packets using Generic Routing Encapsulation (GRE), a generic Layer 3 tunnelling protocol. Unlike VXLAN, which employs a MAC-in-UDP encapsulation scheme, NVGRE utilizes MAC-in-IP encapsulation. In addition, while VXLAN uses a 24-bit VXLAN VNI, NVGRE introduces the Virtual Subnet Identifier (VSID), which is carried in the GRE Key field.

The encapsulation structure is illustrated in Figure 5 and operates as follows. When a virtual machine transmits a frame, the source NVGRE endpoint encapsulates the inner Ethernet and IP headers, which form the inner frame. It then adds a GRE header, followed by an outer IP header and an outer Ethernet header, which are used for routing and transmission across the underlay network. The outer Ethernet header constitutes the frame that is physically transmitted across the underlay network. It carries the destination MAC address of the next Layer 2 hop and the source MAC address of the transmitting NVGRE endpoint. Following this, the outer IP header is added at the NVGRE endpoint, containing the source and destination IP addresses that the underlay routers use for packet forwarding.

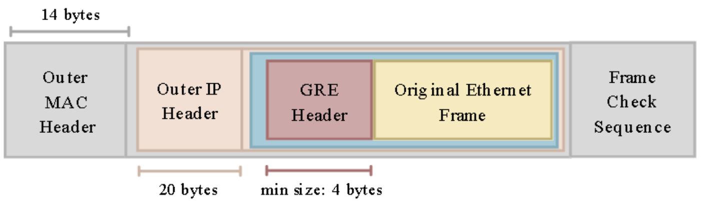

Figure 5: NVGRE packet format. [2]

The GRE header comprises the VSID field, which is 24 bits in length and enables the identification of up to approximately 16 million virtual networks. It also includes the FlowID field, which increases hash entropy for improved load distribution across equal-cost paths. Following this, the encapsulated tenant traffic, corresponding to the overlay network, includes the inner Ethernet header, containing the MAC addresses of the communicating virtual machines within the virtual network.

From a practical standpoint, this encapsulation process adds several bytes to the frame size. Consequently, the MTU of the underlay network must be increased to prevent fragmentation of the outer packets. Since NVGRE operates over an IP-based underlay, it benefits from Equal-Cost Multi-Path (ECMP) routing, enabling efficient distribution of flows across multiple available paths. The GRE Key field is one of the input parameters in the load-balancing hash calculation used for load balancing across these paths.

In terms of development, NVGRE has been primarily used in Microsoft-based environments, as it originated within Microsoft's ecosystem and was standardized as the encapsulation method for Hyper-V Network Virtualization (HNV).

3.3 Geneve

Geneve is a network encapsulation protocol operating over UDP/IP, designed to provide extensibility through Type-Length-Value (TLV) options. Unlike VXLAN, Geneve decouples a minimal base header from an optional set of extension fields. The packet structure (Figure 6) is composed of outer Ethernet, IP and UDP headers, followed by the Geneve header and the encapsulated payload, which may include Ethernet and IP headers from the original frame.

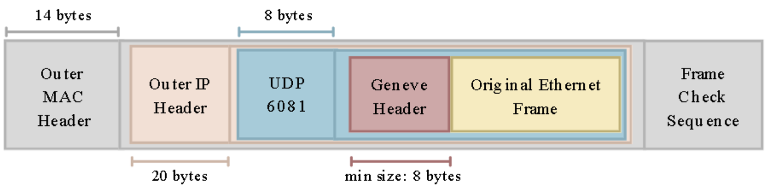

Figure 6: Geneve packet structure. [3]

Analysed from the outside in, we first encounter the Ethernet underlay header, which carriers frames between the GTEP, analogous to the VTEP in VXLAN, and the next hop. This header contains the source MAC of the GTEP and the destination MAC of the Layer 2 next hop. In the next layer, we encounter the IP underlay header, which transports the tunnel between the source and destination GTEPs, using their IP addresses.

Proceeding deeper into the encapsulation, we encounter the UDP header, which Geneve employs to introduce entropy for load balancing. The destination port assigned by IANA is

6081, while the source port is dynamically selected by the GTEP. The final header within the underlay is the Geneve, which is divided into an 8-byte base header and an optional block of Type-Length-Value (TLV) extensions. The base header is composed of several fields, most notably the VNI, which, as with other technologies, is the virtual network identifier. The optional TLV fields are used to attach supplementary information such as telemetry data or operational metadata.

After examining the outer encapsulation, attention shifts to the inner structure of the packet. Immediately following the Geneve header, the payload comprises the overlay frame, representing the original data frame generated by the VM. This frame includes its own set of link-layer, network-layer, and transport-layer headers, corresponding to the standard protocol stack within the virtualized environment.

Geneve is independent of the control plane, and to ensure proper distribution in ECMP, the endpoint varies the source UDP port to increase the entropy of the ECMP hash. Regarding security and Operations, Administration and Maintenance (OAM), the specification defines dedicated fields and behavioural guidelines, including the OAM bit and the handling of critical options, and recommends the use of IPsec when data integrity and confidentiality are required, as the encapsulation mechanism itself does not inherently provide these properties.

In terms of support, Geneve has been widely adopted in software-based network infrastructures. The Linux kernel integrates a Geneve module that can be configured without relying Open vSwitch (OVS), while VMware NSX-T employs Geneve as its primary transport encapsulation protocol.

3.4 Stateless Transport Tunnelling

Stateless Transport Tunnelling (STT) is a comparatively recent addition to network virtualization technologies, developed as an alternative to existing protocols such as VXLAN and NVGRE. It is designed to support overlay networks within multi-tenant environments, providing tenants with control over their logical network domains. The primary objective of STT is to enhance the efficiency of network virtualization by encapsulating and forwarding packets between virtualized environments across a physical underlay network.

STT is one of the principal Layer 2 over Layer 3 tunnelling mechanisms used in network virtualization. Its distinguishing feature lies in the fact that STT packets are processed as standard TCP packets by physical Network Interface Cards (NICs). This allows NICs to leverage the TCP Segmentation Offload (TSO) capability to handle large STT packets efficiently. Under normal circumstances, large data payloads are divided at the TCP layer within the operating system kernel to maintain the size of each TCP segment below the Maximum Segment Size (MSS) threshold. With TSO, this segmentation process is offloaded to the physical NIC, thereby reducing CPU utilization on the host and improving overall performance.

In addition to TSO integration, STT exhibits several important characteristics. It employs MAC-in-IP tunnelling and uses 64-bit context identifiers, which significantly increase the number of possible virtual networks and enable broader service model scalability. The protocol achieves notable performance gains by exploiting hardware-based TSO capabilities, thereby minimizing the overhead associated with transmitting multiple small packets. STT operates in a stateless manner, with its packets supporting unicast communication between tunnel endpoints without relying on TCP windowing, synchronization or flow-control mechanisms. Moreover, STT can be implemented within software-based switches while still benefiting from hardware acceleration through compatible NICs, which substantially reduces the computational load on servers operating in high-bandwidth environments (10 Gbps and above).

Figure 7 shows the sequence of encapsulation and segmentation of a VM's Ethernet frame at the transmitting endpoint. When the TSO feature is enabled on the virtual NIC, the virtual machine may generate large Ethernet frames. Then a tunnel endpoint, such as a virtual switch or a centralized virtual switch (CVSW), encapsulates the frame with STT and pseudo-TCP (P-TCP) headers. Finally, the underlying physical NIC subsequently divides the encapsulated frame into multiple smaller frames, each carrying consecutive sequence numbers within the P-TCP header. On the receiving side, the corresponding tunnel endpoint must reassemble the P-TCP segments prior to decapsulation, as only the first segment contains the STT and inner headers.

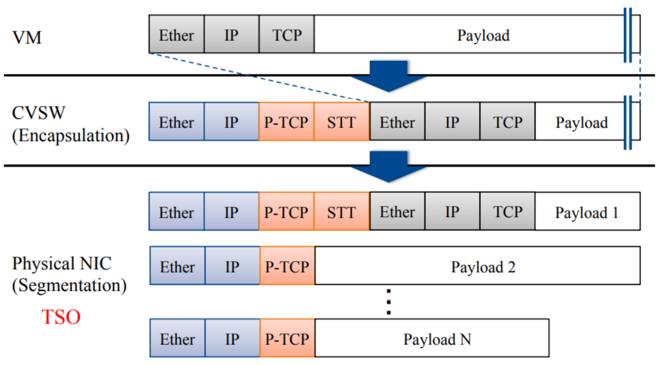

Figure 7: STT's encapsulation and segmentation flows with TSO feature. [4]

4 Control Plane and Orchestration layers in data centers

After examining the main overlay technologies, the following section focuses on the control plane and orchestration layers that enable communication, automation, and coordination across virtualized network infrastructures in modern data centers.

4.1 Control Plane

In traditional networking, the control plane is responsible for maintaining topology information and making forwarding decisions, which are then executed by the data plane. Within virtualized data center environments, the control plane extends this function to manage logical networks that span multiple hypervisors and physical domains. It provides the intelligence necessary to synchronize routing, addressing and policy enforcement across distributed systems.

A key challenge lies in maintaining consistency between logical and physical network states. Software-Defined Networking (SDN) introduces a logically centralized control architecture that decouples the control and data planes, improving programmability and abstraction of network resources. The SDN controller exposes well-defined application programming interfaces (APIs) to upper layers, allowing network resources to be represented as logical entities independent of the underlying transport technology. This abstraction supports the creation of multiple virtual tenant networks (VTNs), each with its own control-plane instance.

In practice, SDN controllers such as OpenDaylight or Floodlight are instantiated within data centers as virtual functions, enabling flexible and isolated control per tenant. The virtualization of control functions, as virtual SDN controllers, reduces configuration time and facilitates high availability through rapid redeployment across data-center servers. The controller's role is central in dynamically programming virtual resources, maintaining topological state, and establishing forwarding paths via protocols such as OpenFlow or NETCONF/YANG.

4.2 Network Orchestration

Network orchestration refers to the coordinated management and automation of network and compute resources to ensure that complex infrastructures operate as a cohesive system. It provides a unified control layer that translates high-level service requirements into automated configuration and provisioning actions across physical and virtual domains. Unlike traditional network management, which often focuses on device-level operations, orchestration operates at a service level, ensuring that the entire network adapts dynamically to application and user demands.

In modern data centers, orchestration integrates the deployment, configuration, scaling and monitoring of network services into a single, policy-driven workflow. It enables consistency and automation by defining service intent through abstract models rather than manual commands. These models describe what the network should achieve, such as bandwidth, latency or security constrains, while the orchestration system determines how to implement them using available resources. Through well-defined interfaces and APIs, orchestration platforms interact with underlying control planes, such as SDN controllers or virtualization managers, to enforce the desired network behaviour automatically.

A key characteristic of network orchestration is automation through a closed loop (CLO), illustrated in Figure 8,where monitoring and analytics continuously evaluate the operational state of the infrastructure and trigger corrective actions when deviations occur. This allows the network to maintain compliance with predefined service-level objectives without human intervention, contributing to the development of self-optimizing infrastructures.

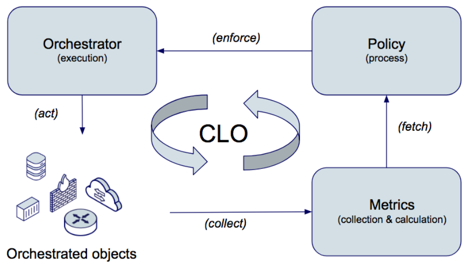

Figure 8: Closed Loop Orchestration. [5]

In virtualized data centers, orchestration systems coordinate with control-plane mechanisms, such as SDN controllers and virtual infrastructure managers, to deploy and adapt network services. The orchestration layer operates at a higher level of abstraction, managing service intent and lifecycle, while the control plane translates these policies into real-time configurations. This separation improves scalability, consistency and agility, allowing data centers to deliver automated, policy-driven network services across diverse physical and virtual environments.

4.3 Integration of Control and Orchestration layers

The integration between orchestration and control layers is essential to achieve end-to-end automation and operational agility in virtualized data centers. As explored in the previous subsections, the control plane is responsible for real-time network configuration, path computation and enforcement of forwarding policies, while the orchestration layer operates at a higher level of abstraction, coordinating multiple control entities to deliver coherent and policy-driven service management. This hierarchical relationship, illustrated in Figure 9, allows service intent, defined at the orchestration layer, to be systematically enforced by the control plane across heterogeneous infrastructures.

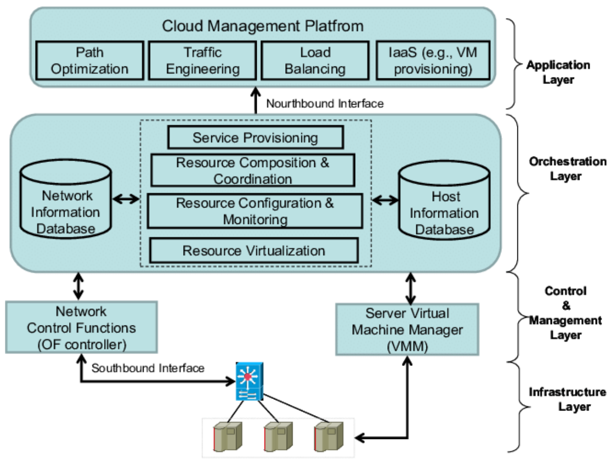
Cloud Management Platfrom
Path Optimization
Traffic Engineering
Load Balancing
IaaS (e.g., VM provisioning)
Application Layer
Nourthbound Interface
Service Provisioning
Resource Composition & Coordination
Resource Configuration & Monitoring
Resource Virtualization
Network Information Database
Host Information Database
Orchestration Layer
Network Control Functions (OF controller)
Server Virtual Machine Manager (VMM)
Control & Management Layer
Southbound Interface
Infrastructure Layer

Figure 9: Hierarchical integration between orchestration, control and infrastructure layers. [6]

The communication between the two layers is typically implemented through standardized APIs and data models, such as REST, NETCONF/YANG or TOSCA, which ensure interoperability across multiple vendors and technologies. Through this layered coordination, orchestration systems gain visibility into network state and can trigger adjustments via the control plane, ensuring alignment between service intent and operational reality.

The result is a scalable and adaptive control architecture in which orchestration defines what the network should achieve, and the control plane determines how to realize it. This separation of concerns enables data centers to operate as automated, policy-driven environments capable of dynamic optimization and rapid service deployment.

References

[1] QSFPTEK. “What is VXLAN and Why Do We Need It.” Accessed: 2025-10-23. [Online]. Available: https://www.qsfptek.com/qt-news/what-is-vxlan-and-why-do-we-need-it.html?srsltid=AfmBOooZgI8JFfVREVzwf-wKfBAgSVeilsCf-6dmk-iQZJ3c67OyrnWO

[2] Microsoft Corporation, “Network Virtualization using Generic Routing Encapsulation (NVGRE),” 2025, Accessed: 2025-10-22. [Online]. Available: https://learn.microsoft.com/en-us/windows-hardware/drivers/network/network-virtualization-using-generic-routing-encapsulation--nvgre--task-offload

[3] L. Krattiger. “All Tunnels Lead to GENEVE.” Accessed: 2025-10-26, Cisco. [Online]. Available: https://blogs.cisco.com/datacenter/all-tunnels-lead-to-geneve

[4] R. Kawashima and H. Matsuo, “Implementation and Performance Analysis of STT Tunneling Using vNIC Offloading Framework (CVSW),” in *Proceedings of the IEEE International Conference on Information Networking (ICOIN)*, Accessed: 2025-10-26, 2020. [Online]. Available: https://scispace.com/pdf/implementation-and-performance-analysis-of-stt-tunneling-25giklerht.pdf

[5] J. Axelsson. “Closed Loop Orchestration (CLO).” Accessed: 27-10-2025, Orchestrated Things Blog. [Online]. Available: https://orchestratedthings.wordpress.com/2018/06/26/closed-loop-orchestration-clo/

[6] D. Adami, B. Martini, A. Sgambelluri, and S. Giordano, “Cloud and network service orchestration in Software-Defined Data Centers,” Figure 1: “Architecture and functional layering of the SDN Orchestrator – The application layer”, ResearchGate. Accessed: 27-Oct-2025, 2015. [Online]. Available: https://www.researchgate.net/figure/Architecture-and-functional-layering-of-the-SDN-Orchestrator-The-application-layer_fig1_308810669

[7] Lumina Networks, *Orchestration White Paper*, Accessed: 2025-10-27, 2019. [Online]. Available: https://www.onap.org/wp-content/uploads/sites/20/2019/07/35752846-0-Lumina-Orchestration-2.pdf

[8] R. Muñoz et al., “Integrated SDN/NFV Management and Orchestration Architecture for Dynamic Deployment of Virtual SDN Control Instances for Virtual Tenant Networks,” *Zenodo Technical Paper*, 2016, Accessed: 2025-10-27. DOI: 10.5281/zenodo.46115 [Online]. Available: https://zenodo.org/record/46115

[9] C. Chappell, *Service Orchestration and Network Virtualization: A Lifecycle View*, Accessed: 2025-10-27, 2015. [Online]. Available: https://community.cisco.com/kxiwq67737/attachments/kxiwq67737/5672j-docs-dev-nso/132/1/Service%20orchestration%20and%20network%20virtualization.pdf

[10] R. Editor, “A Network Virtualization Overlay Solution Using Ethernet VPN (EVPN),” Internet Engineering Task Force (IETF), Tech. Rep. RFC 8365, 2018, Accessed: 27-10-2025. [Online]. Available: https://datatracker.ietf.org/doc/html/rfc8365

[11] R. Editor, “Integrated Routing and Bridging in EVPN (IRB),” Internet Engineering Task Force (IETF), Tech. Rep. RFC 9135, 2021, Accessed: 27-10-2025. [Online]. Available: https://datatracker.ietf.org/doc/html/rfc9135

[12] Open vSwitch Project, *ovs-vtep(8) — Open vSwitch VTEP emulator manual*, Accessed: 27-10-2025, n.d. [Online]. Available: https://www.openvswitch.org/support/dist-docs/ovs-vtep.8.html