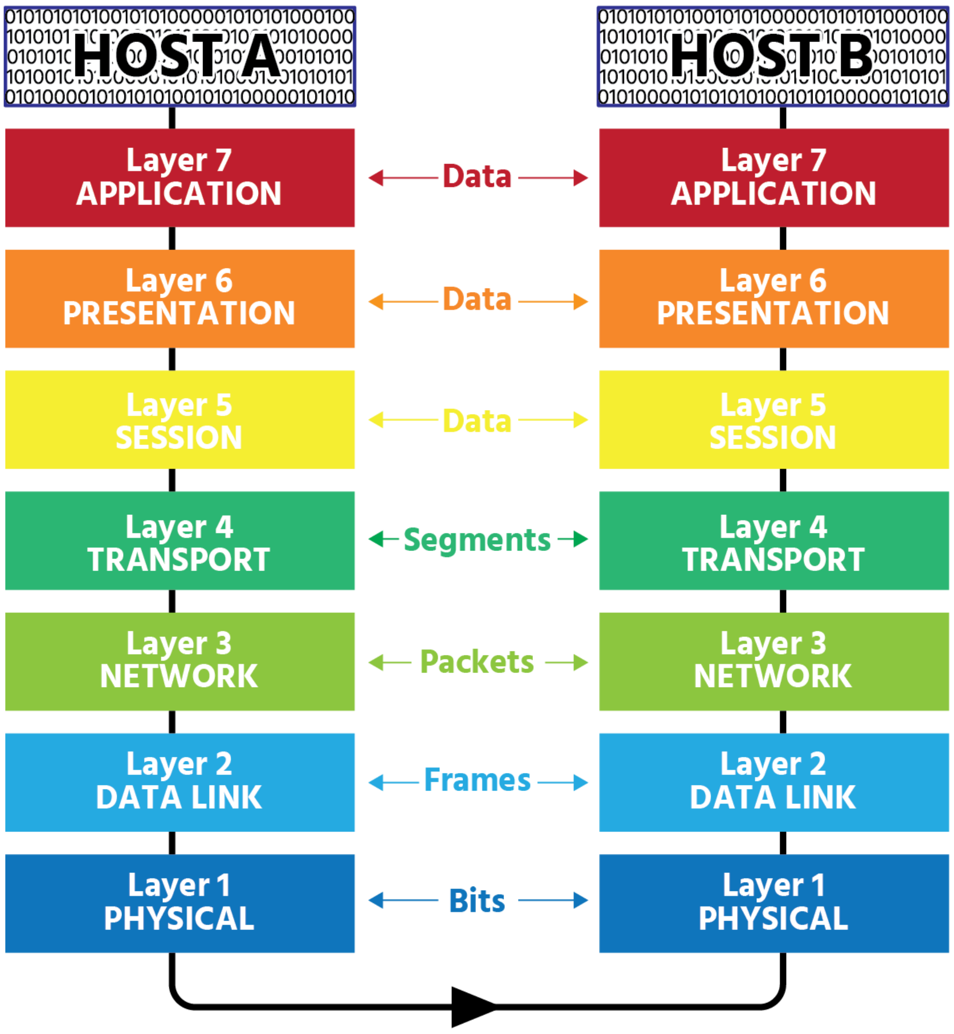
Module 4: Network Virtualization Service

Network Virtualization Services

Before you look at network virtualization services, we think that a quick overview of the **Open Systems Interconnection** (*OSI*) model might be helpful.

In this 7-Layer network model, data is transferred from *Host A* to *Host B* in segments, packets, frames, and bits.



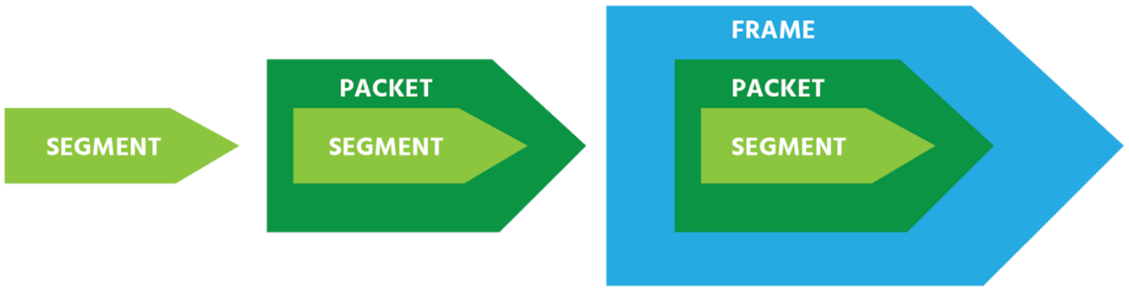
In Layers 7, 6, and 5, data is referred to simply as **data**.

In *Layer 4* (the **transport layer**) data is divided into smaller chunks. A process called **encapsulation** takes place in which information is added to the data as it moves from Layer to Layer. This information (the precise contents of which will vary from layer to Layer) is a called a **header**. In layer 4, once a header is added to the data, the data is referred to as a **segment**. Its header will contain port numbers for the applications that will use the data is Host B. Layer 4 then hands each segment down to Layer 3.

In *Layer 3* (the **network layer**) data is transferred from one **Internet Protocol** (*IP*) address to another via a router. Each segment that's been handed down from Layer 4 will be encapsulated with source information (Host A's IP address), destination information (Host B's IP address), an identification number, another number that identifies how many pieces the information that's being sent (the **payload**) has been broken up into, and part of that information. Once the header is attached, the segment is referred to as a **packet**. Layer 3 hands the packet down to Layer 2.

Layer 2 (the **data link layer**) is responsible for error-free data transfer from one device to another. The packets are encapsulated with Host A and Host B's Media Access Control (MAC- i.e., hardware) addresses, as well as with **Logical Link Control** (*LLC*) which maintains the link between devices as they transmit data across the physical network connection. Each packet is also encapsulated with a **footer** (or *trailer*), containing error-detecting information. With its header and footer/trailer attached, the packet is referred to as a *frame*. Layer 2 hands the frame down to Layer 1.

Layer 1 is the **physical layer** that consists of physical cables and connections. It converts frames into binary *bits* in the form of voltage for physical cables and radio signals for wireless connections.



Host A's Layer 1 forwards these bits on to Host B's Layer 1, which hands them up to Host B's Layer 2. Host B's Layer 2 **de-encapsulates** the frame (removing its header and footer/trailer) before handing it up to Layer 3, where de-encapsulation again occurs. This process is repeated until the fully de-encapsulated data is delivered.

In the overlay networking used by network virtualization, endpoints connect the physical network (the underlay) to the overlay network. Encapsulated traffic is transferred between hosts via a stateless tunnel (i.e., no TCP connections are made with the tunnel) that is created between the source endpoint and the destination endpoint.

Virtual Networking

As we've previously discussed, in virtual networking, some or all of the hardware components are replaced with virtual network components are replaced with virtual network components comprised of software. This eliminates the need to configure physical hardware components physically.

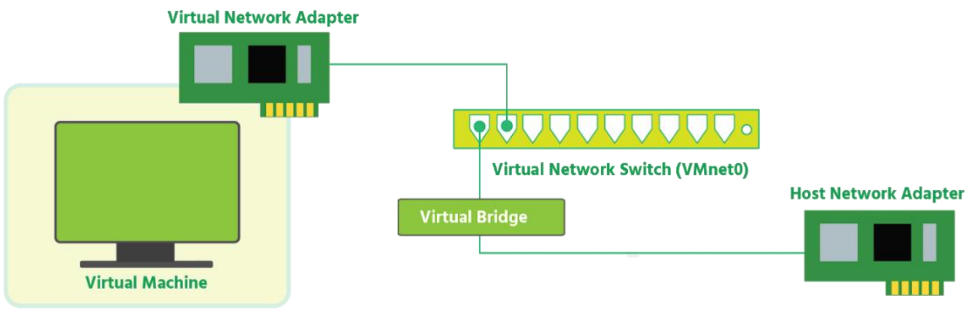
Virtual network components can be configured more easily than physical components, allowing the networking of virtual machines to be managed with greater ease and flexibility.

Virtual machines must be connected to a virtual network component, and that virtual component must have access to the physical network. There are several ways the virtualized components can be connected to a physical network in order to allow VMs to communicate with each other and with other devices. This section explores the three main network types used to set up VMs with a connection in a Type 2 Hypervisor scenario (which is a hypervisor running on a host operating system). Think VMware Workstation and/or Fusion.

Bridged Network

A **bridged network** is a network type where both a virtual machine and the host that it is running on are connected to the same network. Bridged networking connects a VM to the network using the host computer's Ethernet adapter (also known as a *network interface card* or *NIC*. This is possible because the host shares its IP address with the VM.

With bridged networking, the virtual network adapter (vNIC) for the virtual machine connects to a physical NIC on the physical host system. The host network adapter enables the VM to connect to the Local Area Network (LAN) that the host system uses. Bridged networking works with both wired and wireless host network adapters.



Note: The terms network interface card (NIC) and network adapter will be used interchangeably in this section.

Bridged networking treats the virtual machine as a unique identity on the network, separate from and unrelated to the host system. The VM is a full participant in the network. It has access to other machines on the network, and other machines on the network can contact is as if it were a physical computer on the network.

NAT Network

Network Address Translation(NAT) takes an IP address and translates it into another IP address. On a NAT network, a virtual machine does not have its own IP address on the external network. Instead, a separate private network is set up locally on the host computer.

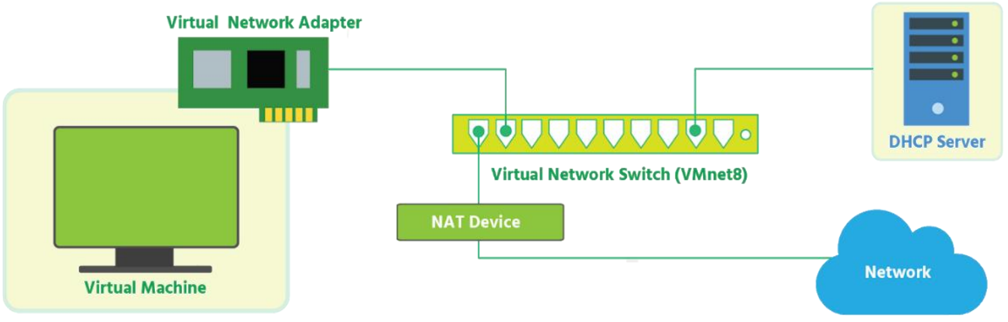
NAT is useful when you have a limited supply of IP addresses. NAT works by translating addresses of virtual machines in a private network called a **VMnet** to that of the host machine. When a VM sends a request to access a network resource, **to the network resource it appears as if the request came from the host machine.**

The NAT device on the network translates the information going to the host's public IP address and forwards it to the private IP address for the VMs.

The VMnet is able to connect to the public external network using the translated IP addresses enabled by a feature called **port forwarding**. Port forwarding allows incoming web traffic to pass through a specific port, chosen by the administrator, to the internal network.

The NAT device is able to sort data packets intended for each virtual machine and sends them to the correct destination. When a packet does not reach its destination, this is called *packet loss*.

The topology (i.e., the physical and logical layout) of a NAT network generally involves a VM connected to a vNIC which allows it to connect to the virtual switch (vSwitch). The vSwitch is also connected to a NAT device that translates the IP addresses and allows port forwarding to connect to the external network.

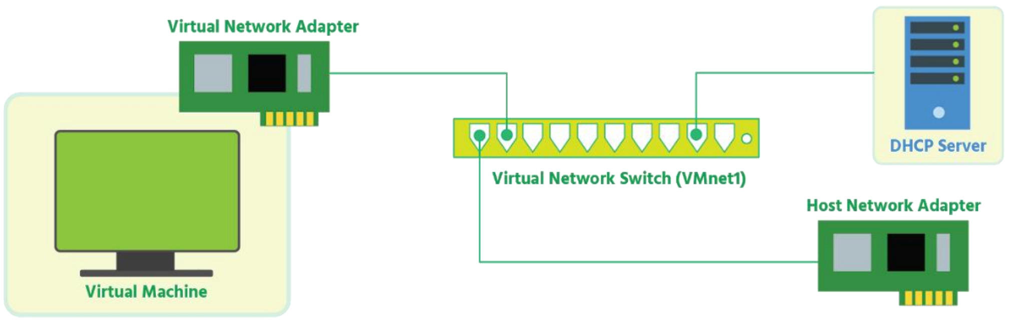


Consider This: In the default configuration, virtual machines get an address on this private network from the virtual DHCP server. DHCP is an acronym for **Dynamic Host Control Protocol**. A DHCP server is a system that uses the DHCP protocol to assign IP addresses to the devices on the network.

Host-only Network

Host-only networking creates a private internal network for the VMs to connect to, similar to a NAT network. However, **without IP address translation, the VMs can only stay in the private network and do not have direct access to the public external network.**

Host-only networking provides a network connection between the virtual machine and other VMs on the same host-only network, using a virtual Ethernet adapter (vNIC) that is visible to the host operating system. This approach can be useful if you need to set up an **isolated virtual network.**



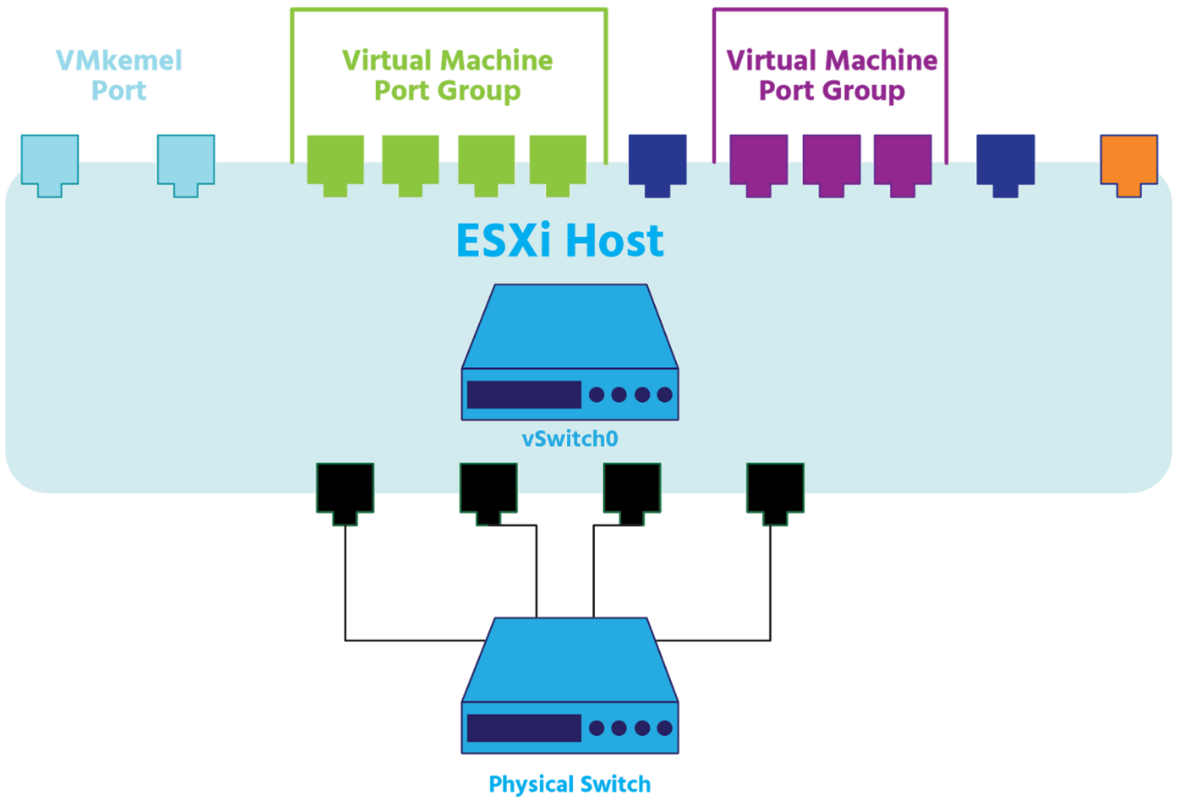
If you use host-only networking, your virtual machine and the host virtual adapter are connected to a private Ethernet network. Addresses on this type of network are also provided by a DHCP service.

Virtual Switches

At the core of vSphere networking are virtual switches (vSwitches). They allow virtual machines to connect to each other and to connect to the outside world. By default, each ESXi host has a single virtual switch called *vSwitch0*.

The connection between a virtual machine and a virtual switch is similar to the connection between a computer's physical network adapter (NIC) and a physical switch. But instead of using a wired Ethernet cable, the virtual machine is connected to the port on the virtual switch by a virtual wire.

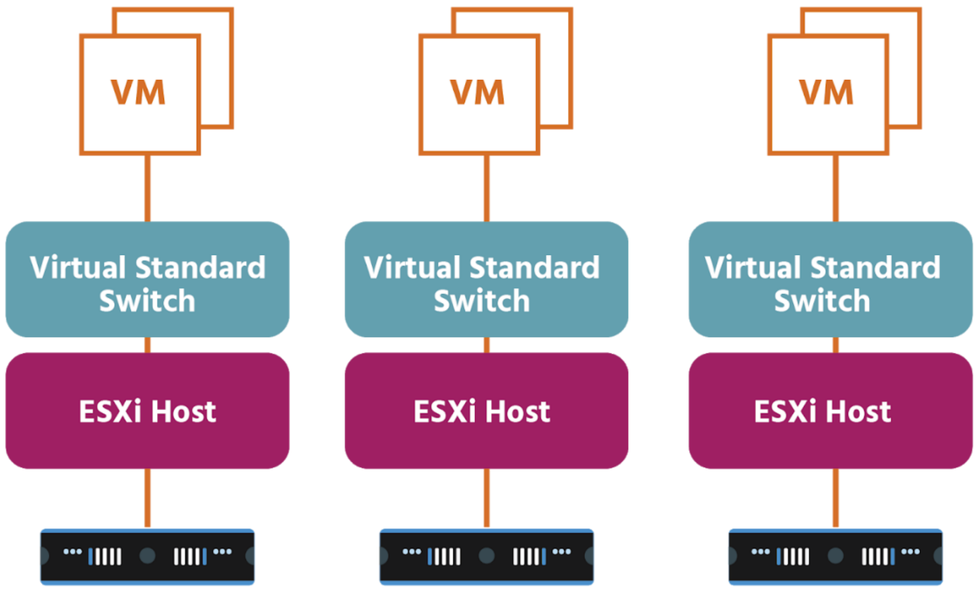
As with a physical switch, Layer 2 frames enter and exit a vSwitch. As with a physical switch, a vSwitch has ports organized into port groups. As with a physical switch, a vSwitch has uplink ports. These are physical network adapter ports found within the ESXi host, and connect the virtual switch within the ESXi host to a physical switch:



Uplinks connect the virtual switch to the physical world: they move physical 0s and 1s off the host and out into the world. A virtual switch can have one or more uplinks. Just as you can connect the uplink ports between the two physical switches in the virtual world, you can connect or uplink a virtual switch to a physical switch.

vSwitches allow you to make adjustments to your *Virtual Local Area Networks (VLANs)*, to some of your security settings, to your load balancing, and to your *Maximum Transmission Units (MTUs)*, which relate to the size of data frames, as well as to other settings which are beyond the scope of this course.

Standard Switches



vSphere supports two types of virtual switches: the **standard virtual switch** (the vSwitch or *VSS*) and the **distributed virtual switch** (or *VDS*).

A standard switch works much like a physical Ethernet switch. It detects which virtual machines are logically connected to each of its virtual ports and uses that information to forward traffic to the correct virtual machines. A standard switch can forward traffic internally between VMs within the same ESXi host,

between VMs on different ESXi hos ts, and between VMs and physical machines, and can link to external networks.

A vSphere standard switch consists of port groups, VMkernel adapters, and uplink ports. To provide network connectivity to hosts and virtual machines, you connect the physical NICs of the hosts to uplink ports on the standard switch. Virtual machines have network adapters (or vNICs) that you connect to port groups on the standard switch. Every port group can use one or more physical NICs to handle its network traffic. If a port group does not have a physical NIC connected to it, VMs on the same port group can only communicate with each other and not with the external network.

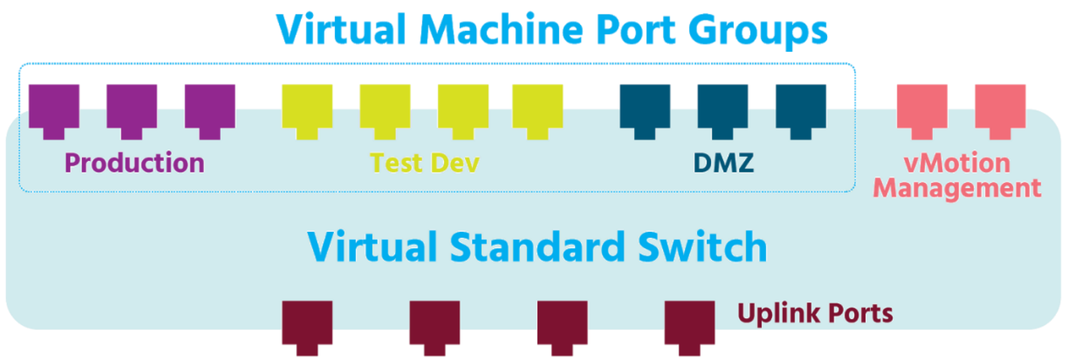
To ensure efficient use of host resources on ESXi hosts, the number of ports of standard switches are dynamically scaled up and down. A standard switch on such a host can expand up to the maximum number of ports supported on the host.

A VMkernel adapter is a port that is used by the hypervisor to attach a service to the network. Every VMkernel adapter has an IP address by which this service is accessible. The uses of this VMkernel adapter include:

- VMware vMotion (which enables you to move VMs from one host to another while they're powered on with no downtime.
- Management port (which is used for ESXi management traffic and in most cases-except vSAN implementations- HA (or high availability) traffic)
- IP storage (which is any form of storage that uses TCP/IP network communication as its foundation)
- vSphere replication
- vSAN data replication

vSphere standard switches are created and configured on a per-host basis. So, if you have three hosts, you'll need three virtual networks, three virtual switches, and three supporting port groups. Each host can have up to 4096 ports across both standard and distributed switches; a maximum of 1016 of these ports can be active at one time. Each standard switch can have up to 512 port groups.

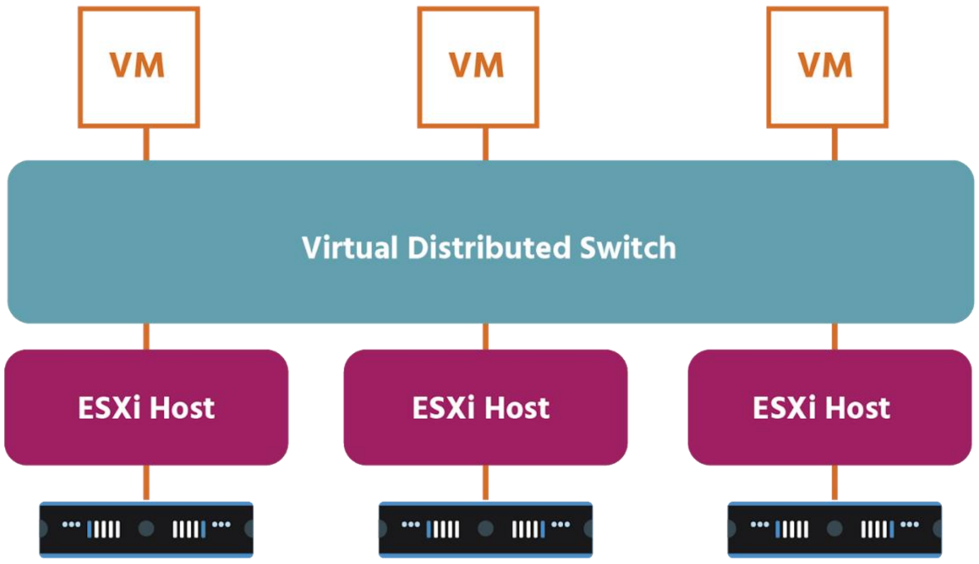
Each logical port on the standard switch is a member of a single port group. Each port group on a standard switch is identified by a network label, which must be unique amongst other port groups on a host but consistent across hosts in order to ensure network connectivity.



Although by default vSwitch policies (security policies, for example) are automatically assigned to the vSwitch's port groups, port group policies can be configured manually.

vSphere network switches can be divided into two logical sections. The *data plane* carries out tasks such as packet switching, filtering (where a switch discards a frame that has the same source and destination MAC addresses), and tagging (where frames are tagged to indicate which VLAN they belong to). The *management plane* is where an administrator configures the functions of the data plane. Each vSphere standard switch contains both data and management planes, and the administrator configures and maintains each switch individually vSphere standard switches are supported by NSX-V, but not by NSX-T. Instead, VMware has developed a new virtual switch for the changing demands of modern networking. The switch is called an N-VDS, and we will look at it in the next section.

Distributed Switches



A **vSphere Distributed Switch** (or *vDS*) acts as a single switch across all associated hosts in a data center and provides centralized provisioning, administration, and monitoring of virtual networks. You configure a vDS on the *vCenter Server Appliance* (*vCSA*),

and the same settings are then added to all ESXi hosts that are associated with the switch. This lets virtual machines maintain consistent network configuration as they move (or migrate) from one host to another. Each vCenter Server system can support up to 128 vDSs, and each vDS can manage up to 2000 hosts. Each vDS can also support up to 10,000 port groups.

A vDS uses the physical NIC's of the ESXi host on which the VMs are running to connect them to the external network.

With a vDS, policies can be set for each individual port, not just for whole port groups. You can also block all ports in a distributed port group (though this might disrupt the normal network operations of the hosts or virtual machines using the ports).

Whereas a standard switch contains both data and management planes, and an administrator configures and maintains each switch individually, a vDS eases this management burden by treating the network as a pooled resource. Individual vSwitches are abstracted into one large vDS spanning multiple hosts across the data center. The data plane remains local to each vDS, but the management plane (the vCenter Server in this case) is centralized, which streamlines and simplifies VM network configuration. The vDS also provides enhanced network monitoring and troubleshooting capabilities, including templates to enable backup and restore for virtual networking configuration.

The data plane section of the vDS is called a ***host proxy switch***. The networking configuration that you create on a vCenter Server Appliance (the management plane) is automatically pushed down to all proxy switches (the data plane). Proxy switches support:

- network traffic between virtual machines on any hosts that are members of the distributed virtual switch
- network traffic between a virtual machine that uses a distributed virtual switch and a virtual machine that uses a VMware standard virtual switch
- network traffic between a virtual machine and a remote system on a physical network connected to the ESXi host.

NSX-V requires the use of vDS. NSX licensing entitles customers to the vDS. NSX-T comes with its own vDS type: the **N-VDS**, or **NSX Managed Virtual Distributed Switch**. As we'll see later in the course, NSX-T can be deployed without a vCenter server. This means that an N-VDS switch is similarly not dependent on vCenter (as a vDS is) and can be used in a variety of cloud environments. It can provide network services to VMs running on both ESXi hypervisors and KVM hypervisors (which are built into Linux kernels).

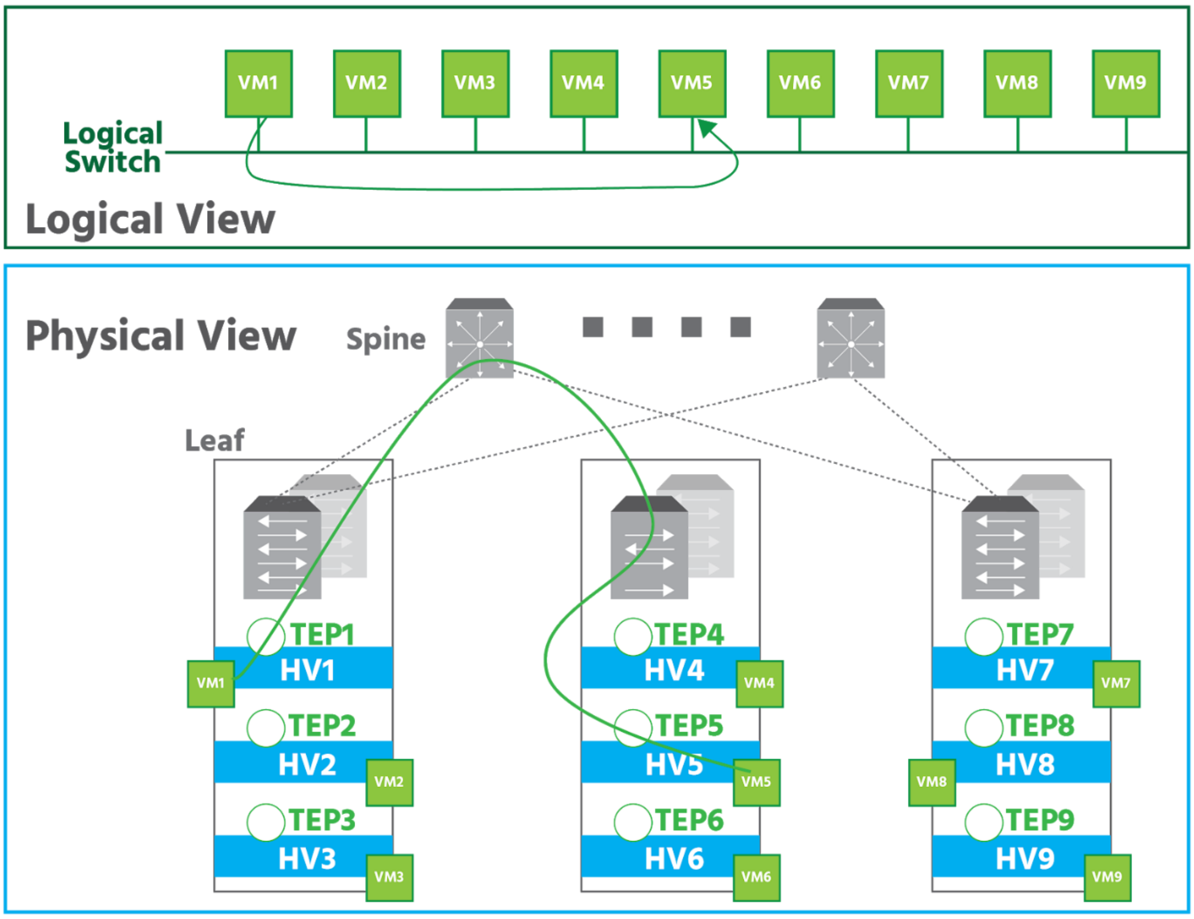
On an ESXi hypervisor, *N-DVS* is implemented via the *NSX-vSwitch module* that is loaded into the hypervisor's kernel. On a KVM hypervisor, N-DVS is implemented by the *Open-vSwitch* (*OVS*) module for the Linux kernel.

The primary purpose of an N-VDS is to forward the traffic that runs on transport nodes. Transport nodes are hypervisor hosts and NSX Edges that will participate in an NSX-T overlay. When you add a transport node to a transport zone (which defines the span of an NSX-V or NSX-T virtual network), the N-VDS associated with the transport zone is installed on the transport node. Each transport zone supports a single N-VDS which must have the same name as the transport zone. There are two types of transport zone: an **overlay transport zone** and a **VLAN transport zone.**

Remember that an N-VDS:

- can only attach to a single overlay transport zone
- can only attach to a single VLAN transport zone
- can attach to both an overlay transport zone and a VLAN transport zone at the same time; in that case, both transport zones and the N-VDS will have the same name.

Multiple N-VDSs and vDSs can coexist on a transport node; however, a physical NIC can only be associated with a single N-VDS or vDS.

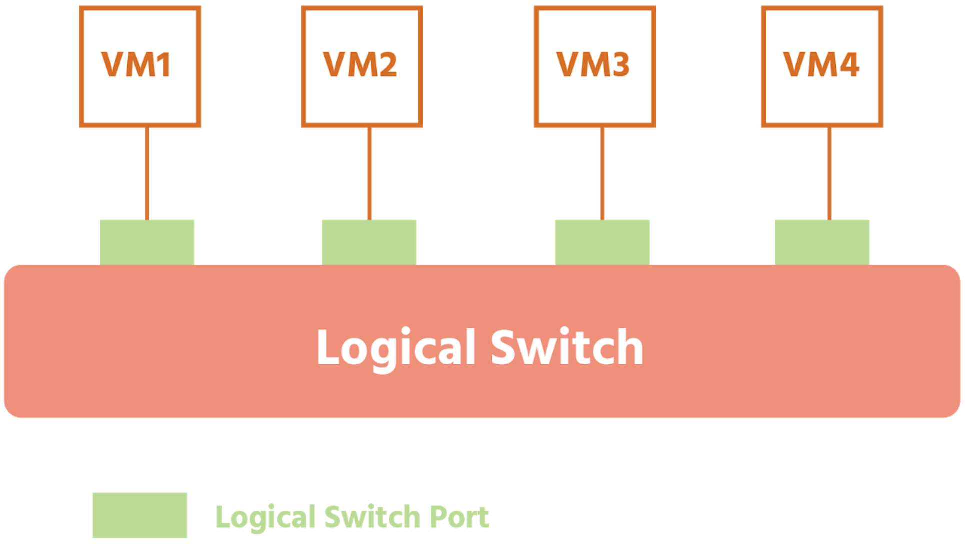


In the upper part of the diagram above, the logical view consists of eight virtual machines that are attached to the same logical switch, forming a virtual broadcast domain. The physical representation, at the bottom, shows that the five virtual machines are running on hypervisors spread across three racks in a data center. Each hypervisor is an NSX-T transport node equipped with a **tunnel endpoint** (TEP). The TEPs are configured with IP addresses, and the physical network infrastructure provides IP connectivity between them. An *NSX Controller* (which we'll discuss later in the course) distributes the IP addresses of the TEPs 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 a tunnel between transport node *HV1* to transport node *HV5*.

The GENEVE overlay encapsulation protocol used by NSX-T provides better throughput (i.e., data transfer in a set amount of time) and uses fewer CPU resources.

NSX Logical Switching

As mentioned in section 2.1, logical switching in NSX-V is based on the VXLAN protocol while NSX-T is based on the GENEVE protocol. A logical switch is mapped to a unique VXLAN or GENEVE, which encapsulates the virtual machine traffic and carries it over the physical IP network. The NSX logical switch creates logical **broadcast domains** (devices connected to the same switch) or *segments* to which an application or virtual machine can be logically wired. This allows for flexibility and speed of deployment while still providing all the characteristics of a physical network's broadcast domains (VLANs).



Both VXLAN and GENEVE protocols help you move to a software-defined data center model. It allows an administrator to provision a virtual machine that can communicate with another virtual machine on a different network without having to configure the physical switches and routers.

You may remember from section . 2.4 that VXLAN has several advantages over VLAN:

- VLAN networks can't be saved, snapshotted, cloned, deleted, or moved, which could negatively impact business continuity in the event of a system failure;
- every time a VLAN is extended, a time-consuming physical configuration is needed; by contrast, because VXLAN uses overlay technology, with virtual Layer 2 network is abstracted from the underlying physical network and can be configured and reconfigured very quickly.

What’s more, you can use VXLAN logical switches (which are Layer 2 Ethernet broadcast domains) to `cross Layer 3 network boundaries. This allows for virtual machine mobility within the data center (with vMotion) without limitations of the physical Layer 2 (VLAN) boundary.

As discussed in section 2.4, the original Ethernet frame generated by a workload is encapsulated with external VXLAN, UDP, IP and Ethernet headers to ensure it can be transported across the network infrastructure interconnecting the ESXi hosts.

Each logical switch is assigned a unique VXLAN numerical identifier. Each logical switch is created as a port group on the distributed switch. Logical switches can extend across multiple distributed switches.

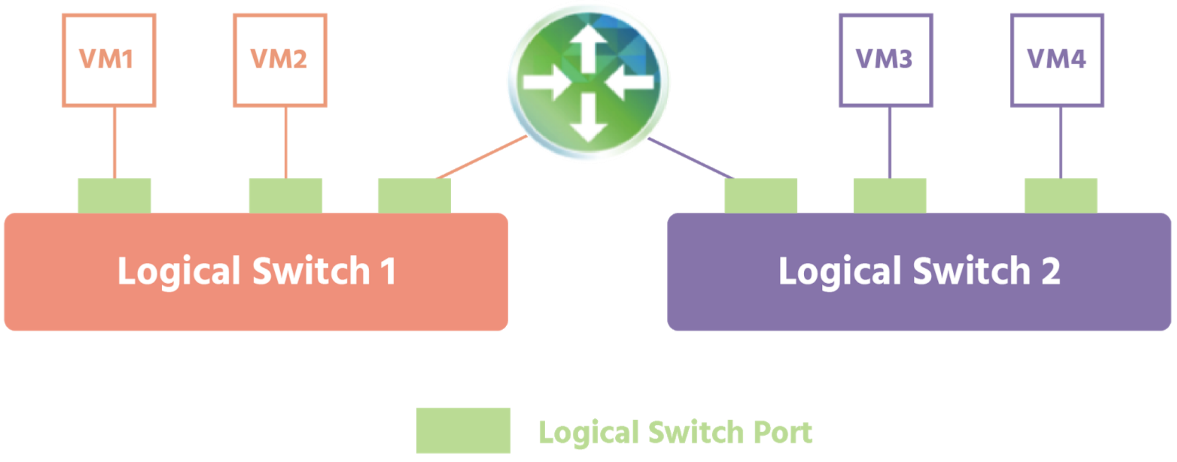
VXLAN runs over standard switching hardware and has been embraced by more vendors.

The logical switching capability in the NSX-T platform provides the ability to create isolated logical L2 networks with the same flexibility and agility that exists for virtual machines.

NSX Logical Routing

Today we have routing that is built directly into the host's hypervisor. With NSX logical routing, we can now connect both virtual and physical endpoints that are located in different logical Layer 2 networks. This is made possible by the separation of physical network infrastructure from logical networks that network virtualization provides.

Network edge security and gateway services (such as DHCP, NAT, Virtual Private Networks - VPN - and load balancing) are provided in NSX-V by what's known as an NSX Edge. An **NSX Edge** can be installed as a distributed logical router (DLR), which is a virtual router that can use both the fixed, manually configured network routes of static routing and dynamic routing, where routers communicate with each other updating routes in real-time. An NSX Edge can also be installed as an *Edge Services Gateway* or ESG (more about *ESG* will be discussed in the next section).



NSX-V’s DLR provides **East-West** distributed routing. (*East-West* refers to traffic within the same data center while in the same NSX environment.) This means that two VMs can be on the same host but on different subnets, and still communicate without their traffic having to leave the hypervisor.

By providing the gateway services mentioned above, NSX-V's ESG connects isolated networks to shared uplinks.

NSX-T introduces a two-tiered routing architecture that enables the management of networks at the provider tier (**tier-0**) and user tier (**tier-1**). The tier-0 logical router is attached to the physical network for **North-South traffic** (that is, traffic coming into the data center from the outside world); it handles traffic between the logical and physical networks. The tier-1 router can connect to the tier-0 router via uplinks, that can connect to logical switches and manage east-west communications.

It’s not necessary to use both tiers. A tier-0 logical router can be connected by itself to the physical infrastructure for traffic heading outwards exiting the perimeter network (**northbound traffic**) and then connect directly to logical switches in the NSX environment for traffic coming into the data center (**southbound traffic**).

Tier-0 and tier-1 logical routers are both created on transport nodes.

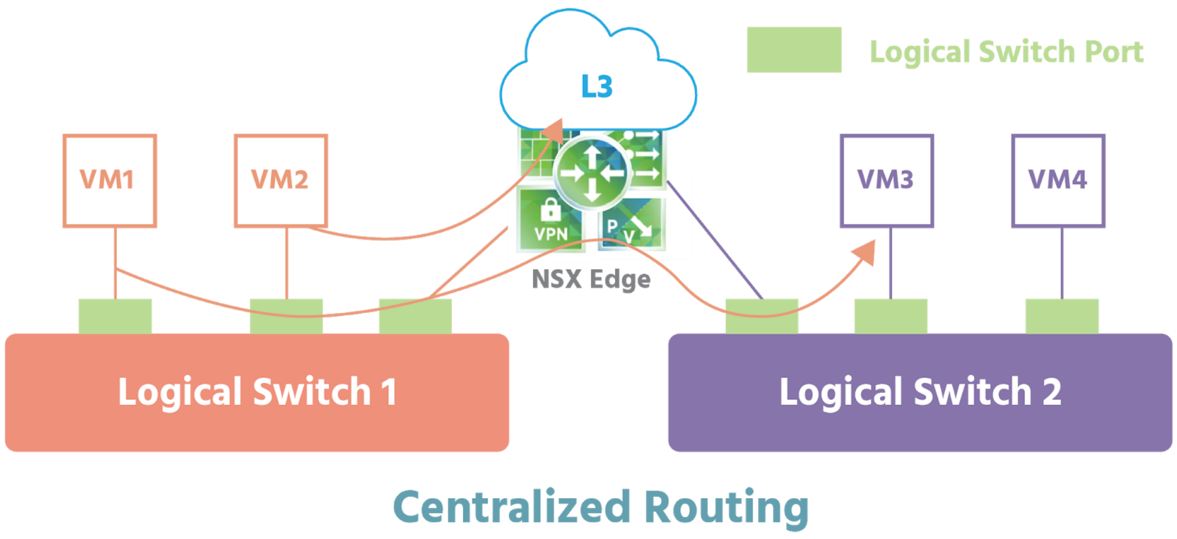
NSX-T supports static routing and the dynamic routing protocol eBGP on tier-0 logical routers. (*eBGP* stands for **External Border Gateway Protocol**, and it helps connect the networks of different organizations.) Tier-1 logical routers support static routes but do not support any dynamic routing protocols.

If NSX-T requires services such as NAT or an edge firewall (see section 4.7.1), these can be enabled on Edge nodes. To improve availability, edge nodes can be combined into a cluster.

Edge Routing and NAT

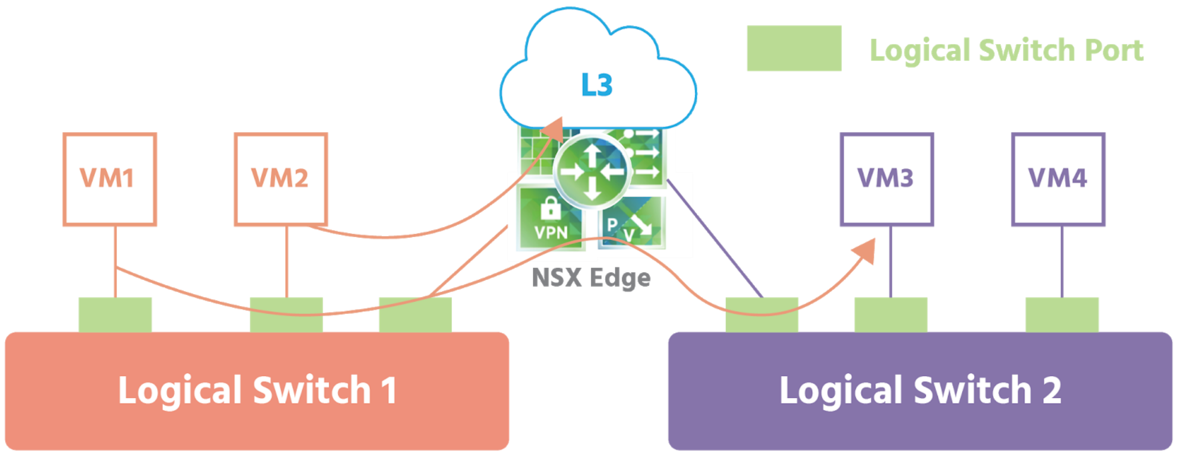
In a network, the *edge* is typically the point where every customer and device connection come into and depart from a data center. Specialized **edge routers** are used here. These are designed to be able to deal with the many types of data packets (and the different routing protocols) coming in and out of the data center. *High availability* (a high percentage of uptime) and redundancy (duplicate devices kept in case of system failure) are crucial. And as a data center's initial contact-point with the outside world, edge routers have a key role to play in a network's security.

NSX-V Edge Services Gateway (*ESG*) is a multi-function, multi-use virtual machine appliance for network virtualization. It gives you access to the logical services that we'll be looking at in the coming sections. Multiple ESG virtual appliances can be installed in a data center. NSX-T provides the same services through an **NSX Edge** appliance - not to be confused with an Edge Services Gateway!



ECMP (*equal cost multi-path*) can be used to increase bandwidth between physical and virtual networks. (Bandwidth is the amount of data that can be sent from one point to another in a set amount of time.) ECMP also provides faster convergence (the merging of data, telephone, and video networks into a single network).

If centralized services (such as NAT) need to run on the Edge appliance, the appliance will need to be in what's known as active-standby mode. In this mode, all stateful services (i.e., services that keep track of the network's state - load balancing, for example) are available.

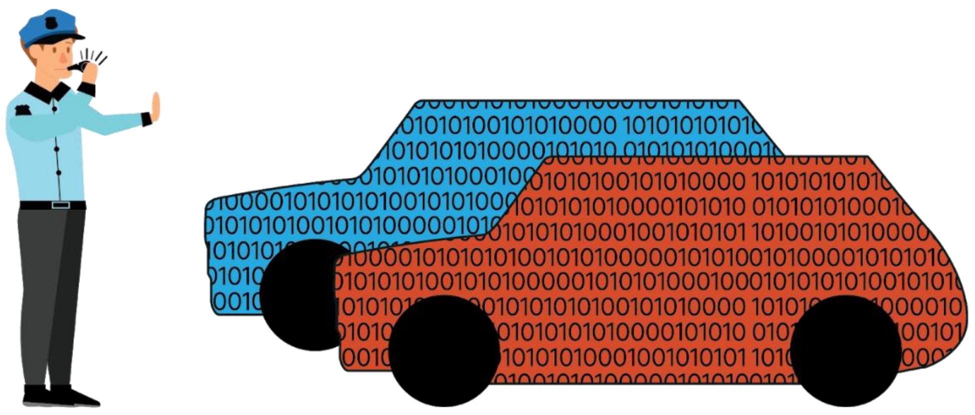


Centralized Routing

Dynamic routing uses protocols such as **Open Shortest Path First** (*OSPF* – an intra-domain protocol that prioritizes the shortest path based on the cost of available paths) in the case of NSX-V, and **Border Gateway Protocol** (*BGP* – an inter-domain protocol that prioritizes the best path as defined by a list of attributes) in the case of NSX-T Data Center.

As discussed in section 4.1, *Network Address Translation* (NAT), takes an IP address and translates it into another IP address. *ESG* supports both source NAT where a private IP address is translated into a public IP address, and destination NAT, a public IP address to private IP address translation. NAT is also an integral part of load balancing, which we'll discuss next.

Load Balancing



A **load balancer** evens out workloads to prevent servers from being overwhelmed. Its other main use is to provide high availability. Imagine, for example, a service being available on several hosts but all traffic being sent to just one node. In this situation, a load balancer will redirect ALL traffic - just in case there's a failure.

Rather like traffic police directing traffic to keep it flowing and free from congestion, it distributes incoming network and application traffic across multiple servers (together known as a **server pool**). It only routes traffic to servers that are able to fulfill the client request and do so in a way that prevents any single server being over-burdened while maximizing overall network speed and use of resources. How does it know that a server is able to handle requests? By carrying out **health checks** in which it periodically attempts to connect with the server. If a response is received, the load balancer knows that the server is available. If a server fails to respond (i.e., is offline), the load balancer diverts traffic to the servers that are still online. When a new server is added to the **server pool**, the load balancer immediately starts sending traffic its way.

The NSX load balancing service is specially designed for IT automation and uses the same central point of management and monitoring as other NSX network services. It's rich with features and functionalities. Here are just a few:

- the Edge VM active-standby mode provides high availability for the load balancer
- support for any TCP (Transmission Control Protocol) application
- support for UDP (User Datagram Protocol) applications
- health checks for multiple connection types (TCP, HTTP, HTTPS), including content inspection

- the NSX platform can also integrate load-balancing services offered by 3rd party vendors
- NSX Edge offers support for two deployment models: **proxy mode** (a.k.a. one-arm mode), and **inline mode** (a.k.a. transparent mode).

In *proxy mode*, an *NSX Edge* is connected directly to the logical network where load-balancing services are required. The external client sends traffic to the **Virtual IP Address** (*VIP*) provided by the load balancer. (VIPs are multiple virtual IP addresses assigned to servers that share a regular IP address based on a single NIC.) The load balancer performs two address translations on the original packets received from the client: **destination NAT** (*DNAT*) to replace the VIP with the IP address of one of the servers in the server pool, and **source NAT** (*SNAT*) to replace the client IP address with the IP address identifying the load balancer itself. *SNAT* forces through the load balancer traffic on its way back from the server pool to the client. The server in the server pool replies by sending the traffic to the load balancer. The load balancer again performs a source and destination NAT service to send traffic to the external client, using its VIP as the source IP address.

Proxy mode is simpler to deploy and provides greater flexibility than traditional load balancers. It allows the deployment of load balancer services (e.g., *NSX Edge* appliances) directly on the logical segments without requiring any modification on the centralized *NSX Edge* that is providing routing communication to the physical network.

One limitation of proxy mode is that it requires provisioning more *NSX Edges* and requires the deployment of source NAT, which means that the servers in the data center do not have the original client IP address. The load balancer can insert the original IP address of the client into the HTTP header before SNAT is performed – a function named **Insert X-Forwarded-For HTTP** header. The servers, therefore, have the client IP address. This is, however, limited to HTTP traffic.

With *inline mode*, the *NSX Edge* is inline to the traffic destined for the server pool. The external client sends traffic to the VIP provided by the load balancer. The load balancer – a centralized *NSX Edge* – performs only destination NAT (DNAT) to replace the VIP with the IP address of one of the servers deployed in the server pool. The server in the server pool replies to the original client IP address. The traffic is received again by the load balancer since it is deployed inline, typically as the default gateway for the server pool. The load balancer performs source NAT to send traffic to the external client, using its VIP as the source IP address.

Inline mode is also quite simple, and additionally, the servers have the original client IP address.

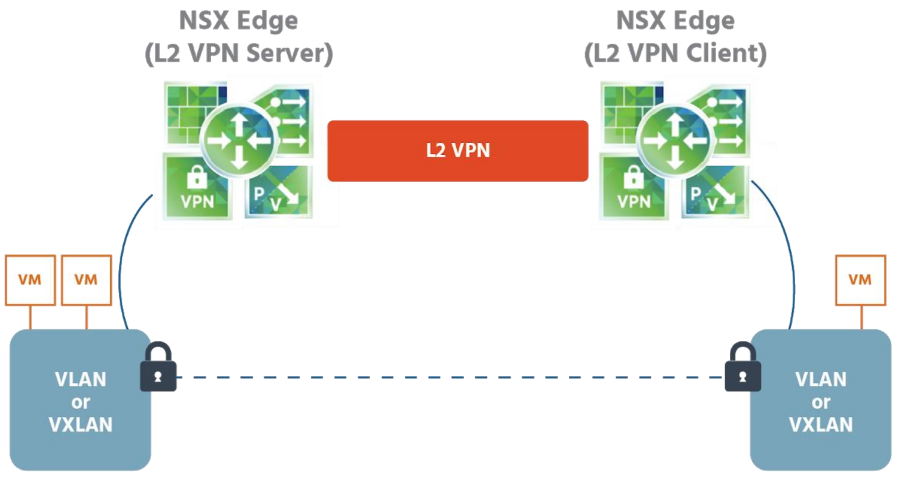
L2/L3 VPN

A **Virtual Private Network** (or *VPN*) helps extend a private network over a public network privately, as the name suggests, and securely. It does this by creating a tunnel or private line from a local network to an external network (and vice versa) for the secure transmission of data. It encrypts the data before it leaves a device, and then masks the sender/recipient's IP (Internet Protocol) address by connecting to the server of a VPN service provider. The VPN service provider assigns the sender/recipient a temporary IP address that cannot be traced back to them.

The *tunnel* is a virtual connection established between two endpoints using a tunneling protocol. Tunneling protocols enable the secure transmission of every type of data between two points on a network or from one network to another. Two of the most widely-used tunneling protocols are:

- **Internet Protocol Security** (*IPsec*), which authenticates senders, checks the integrity of data being transmitted and encrypts it; users need to install software on their machines in order to be able to establish a connection
- **Secure Sockets Layer** (*SSL*), or its successor **Transport Layer Security** (*TLS*), both of which enable secure communication across public networks from a web browser.

The VPN service provider's server decrypts the data and sends it to its destination.

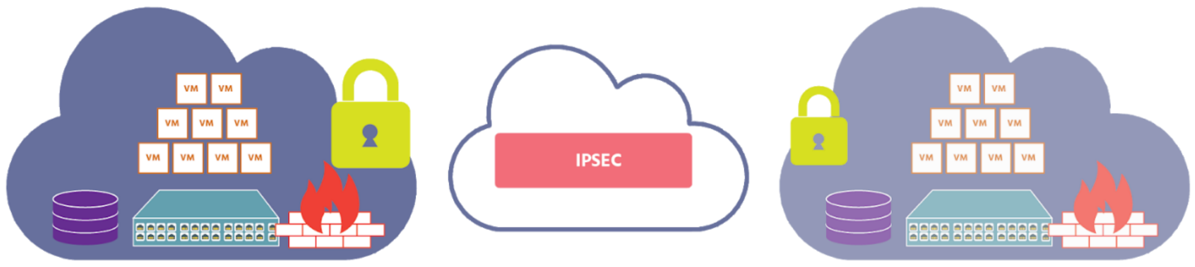


With **Layer 2 VPN** (*L2 VPN*), you can extend Layer 2 networks (*VLANs* or *VXLANs*) across multiple sites that are on the same broadcast domain. Data is forwarded to one of the Layer 2 formats (*Medium Access Control (MAC)* and *Logical Link Control (LLC)*, for example) on a service provider's Layer 3 network running over the public internet infrastructure and is then converted back to Layer 2 format at the receiving end. VMs in Layer 2 can seamlessly communicate with each other over an *L2 VPN* even if they are located in different data centers. The extended network is a single subnet with a single broadcast domain, so VMs remain on the same subnet when they are moved between network sites, and their IP addresses remain the same.

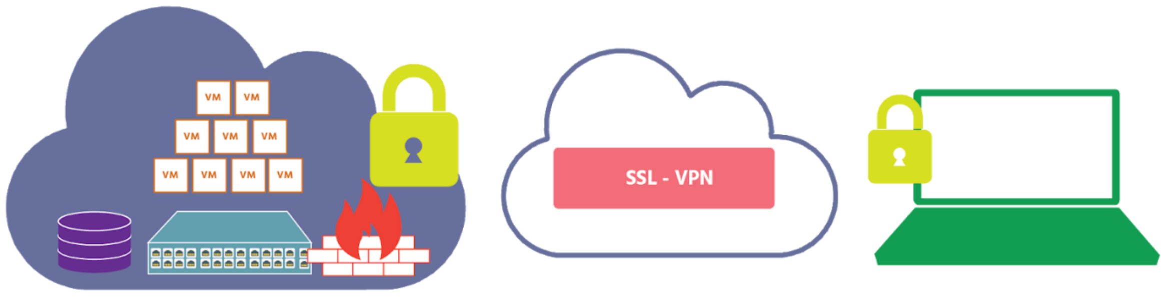
A *L2 VPN* might be used for connections in the following scenarios:

- between an *NSX-T Data Center L2 VPN* server and an *L2 VPN* client hosted on an *NSX Edge* managed in an *NSX Data Center for vSphere*; a managed *L2 VPN* client is limited to supporting VXLANs
- between an *NSX-T Data Center L2 VPN* server and an *L2 VPN* client hosted on a standalone or unmanaged *NSX Edge*; an unmanaged *L2 VPN* client supports VLANs
- between an *NSX-T Data Center L2 VPN* server and *NSX-T Data Center L2 VPN* client; in this scenario, you can extend the L2 network between two software-defined data centers (SDDCs) deployed on the cloud, such as *VMware Cloud on Amazon*.

Layer 3 VPN (*L3 VPN*) services are used to provide secure Layer 3 connectivity into the data center network from remote locations.

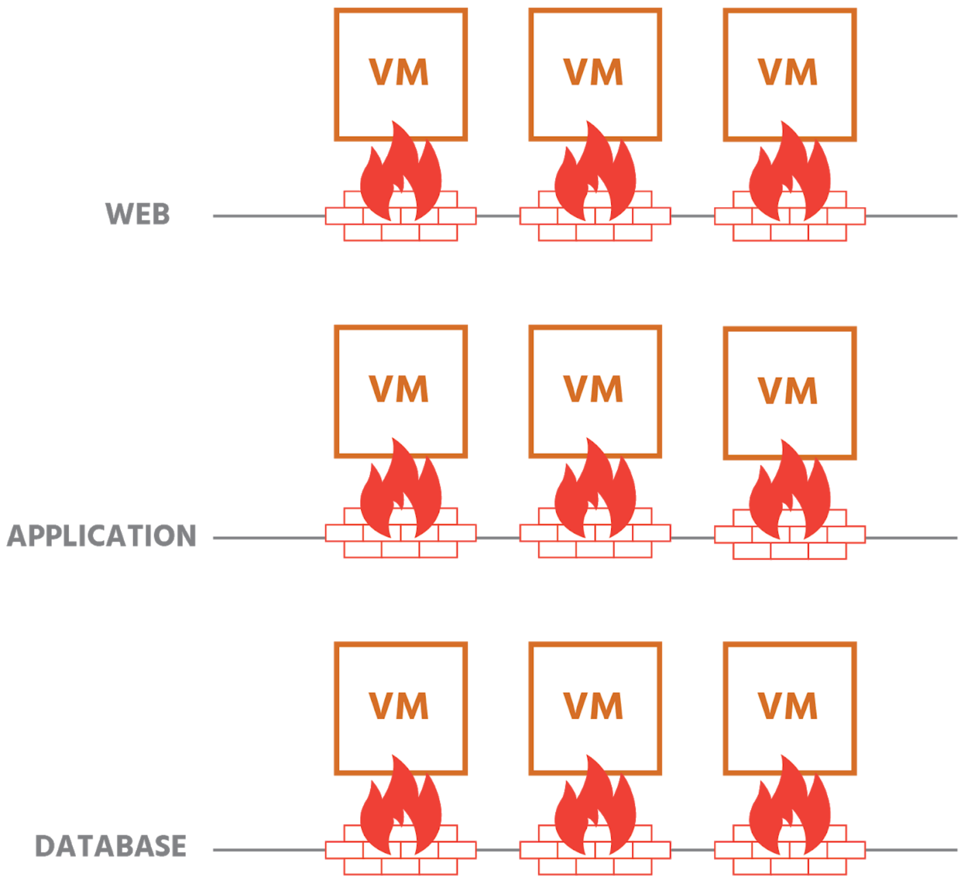


As shown in the illustration above, the *L3 VPN* services can be used by remote clients using SSL tunnels to securely connect to private networks behind an *NSX Edge* gateway which is acting as an *L3 VPN* server in the data center. This service is usually referred to as **SSL VPN-Plus**. (A *gateway* is a device that enables data to be transmitted from one network to another. Unlike switches and routers, they can use more than one protocol at the same time and can function at all seven layers of the OSI model.)



Alternatively, the *NSX Edge* can be deployed to use standard IPSec protocol settings to operate with all major physical VPN vendors' equipment and establish site-so-site secure L3 connections. It is possible to connect multiple remote IP subnets (broadcast domains) to the internal network behind the *NSX Edge*. Connectivity, in this case, is routed since the remote and local subnets are part of different address spaces (broadcast domains).

NSX Logical Firewall



NSX logical firewalls provide security mechanisms for dynamic virtual data centers and consist of two components to address different uses. The centralized Edge firewall offered by **NSX Edge Services Gateway** (*ESG*) focuses on the north-south traffic enforcement at the data center perimeter. And the **Distributed Firewall** (*DFW*) is enabled in the kernel on the ESXi host and focuses on east-west traffic controls. Together, these two components address the firewall needs of virtual data centers.

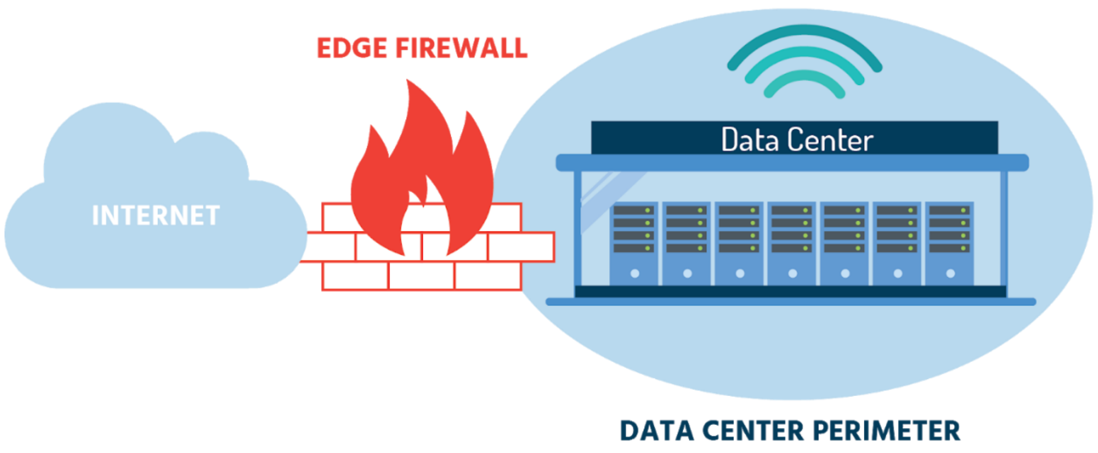
These security technologies can be deployed independently or together. Ideally, they'll be used together as part of a broader security strategy.

The NSX distributed firewall is a stateful firewall, meaning that it monitors the state of active connections and uses this information to determine which network packets to allow through the firewall. Data packets flowing through the network are identified by the following:

- source address
- source port
- destination address
- destination port
- protocol

A distributed firewall on an ESXi host (one instance per virtual machine vNIC) contains two tables: a rule table to store all policy rules, and a connection tracker table to temporarily store (or *cache*) traffic flow entries for rules with a permit action. DFW rules are enforced in a top-to-bottom order. Traffic that needs to go through a firewall is first matched against a firewall rules list. 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 last rule in the table is the DFW default policy rule: packets not matching any rule above the default rule will be enforced by the default rule.

Edge Firewall



An edge/perimeter firewall is a network security system, either hardware or software-based, that controls incoming and outgoing traffic based on a set of predetermined security rules. The **NSX Edge firewall** provides stateful perimeter defense for north-south traffic flows between the virtual and physical networks. It's used on the logical router and provides network address translation (NAT) as well as site-to-site IPsec and SSL VPN functionality. The NSX Edge firewall is available for virtual machines and has a high availability mode.

The Edge firewall can be managed with the same management tools as for the distributed firewall. ESG also provides a **multiple management model** whereby, for example, individual teams within an organization can configure their own firewalls without needing access to the entire network.