Hardware Layer 2 Gateways Integration with NSX

TECHNICAL WHITE PAPER

vmware®

Table of Contents

Executive Summary

This document is targeted at networking and virtualization architects interested in deploying VMware NSX Network virtualization solution.

VMware's Software Defined Data Center (SDDC) vision leverages core data center virtualization technologies to transform data center economics and business agility through automation and non-disruptive deployment that embraces and extends existing compute, network and storage infrastructure investments. NSX is the component providing the networking virtualization pillar of this vision. As a platform, NSX provides partners the capability of integrating their solution and build on the top of the existing functionalities. NSX enables an agile overlay infrastructure for Public and Private Cloud environments leveraging a resilient underlay infrastructure.

In many data centers, some workloads have not been virtualized, or cannot be virtualized. In order to integrate them into the SDDC architecture, NSX provides the capability of extending virtual networking to the physical one by the way of Layer 2 or Layer 3 gateways. This document will focus on the Layer 2 gateway feature, and how it can be achieved natively on a host running NSX, but also through a third party hardware device that can still be controlled by NSX.

This first part of this document presents a summary of the benefits of NSX and the use cases for a Layer 2 gateway service. The second part will focus on the technical overview of the solution and the configuration required on the NSX side for the integration of a third party hardware-based layer 2 gateway.

Network Virtualization with NSX

Server virtualization has dramatically changed the way compute resources are consumed in a data center. With the introduction of the hypervisor, a thin layer of software abstracting the server hardware, virtualization brought to the market straightforward benefits: several virtual machines could now be consolidated on fewer, cheaper generic devices. But a second wave of innovation followed, directly resulting from the flexibility of a software model. A compute admin can now expect to instantiate a virtual machine on-demand, move it from one physical location to another with no service interruption, get high availability, snapshot capabilities and many other high value features that were just not imaginable in a purely physical environment.

Today, an application is more than a software running on a single server. It typically requires communication between several tiers of resources through some network components, and the agility in the compute space must directly map to the same flexibility on the networking space. Indeed, as networking is all about forwarding traffic to a determined location, if compute virtualization allows the location of compute resources to move freely, it is necessary to update the networking components of those moves. The possible solutions considered before NSX were:

- Manual reconfiguration of the network. The complexity of the interaction between networking, security, storage and compute teams makes this solution very slow and only suitable to small, static environments.
- Complete automation of the network devices. Ideally, all the network devices would have similar characteristics and could have their configuration entirely automated. This model was never possible to achieve across vendors, even with OpenFlow.
- Layer 2 based solutions. Most networking vendors have worked on enhancing those solutions, but Layer 2 still provides flexibility at the expense of scale. Stable implementations require segmenting the data center in one way or another, re-introducing the silos virtualization is trying to fight.

Network reachability is not the only challenge those solutions are trying to address. They show the same limitations when it's a matter of implementing end-to-end security policies, or insert services like load balancing for example.

NSX is taking an approach very similar to compute virtualization. With server virtualization, a software abstraction layer (server hypervisor) reproduces the familiar attributes of a physical server (e.g., CPU, RAM, Disk, NIC) in software, allowing them to be programmatically assembled in any arbitrary combination to produce a unique virtual machine (VM) in a matter of seconds. With network virtualization, the functional equivalent of a "network hypervisor" reproduces the complete set of Layer 2 to Layer 7 networking services (e.g., switching, routing, access control, load-balancing, firewalling or QoS) in software. As a result, they too can be programmatically assembled in any arbitrary combination, this time to produce a unique virtual network in a matter of seconds.

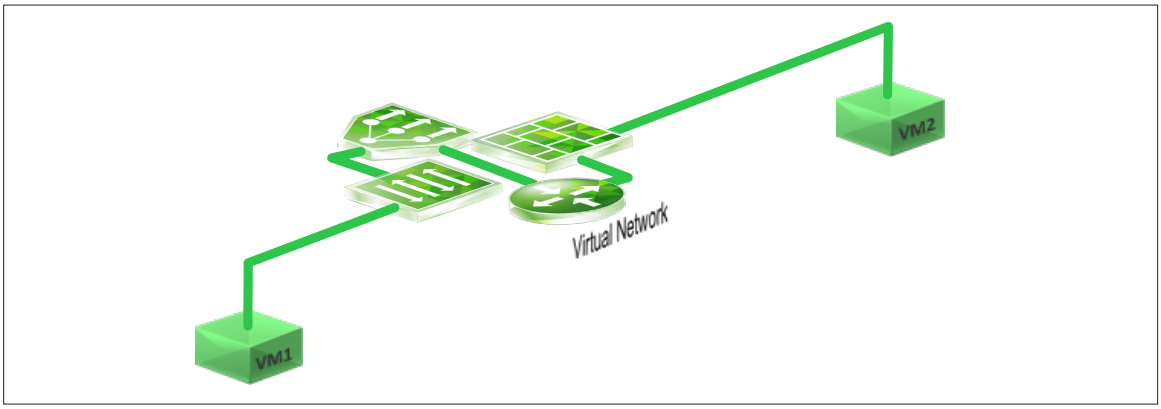


Figure 1: virtual network view

In the Figure 1 above, NSX presents to the virtual machines a virtualized version of all the traditional networking functions. Those virtual functions are achieved in a distributed fashion, across the different hosts in the data center. So, taking the example of the traffic going between VM1 and VM2 above, from a logical standpoint, everything looks like this traffic is going through some network devices: routers, switches, firewalls, load-balancers. The real traffic flow is however following the path represented in Figure 2 below:

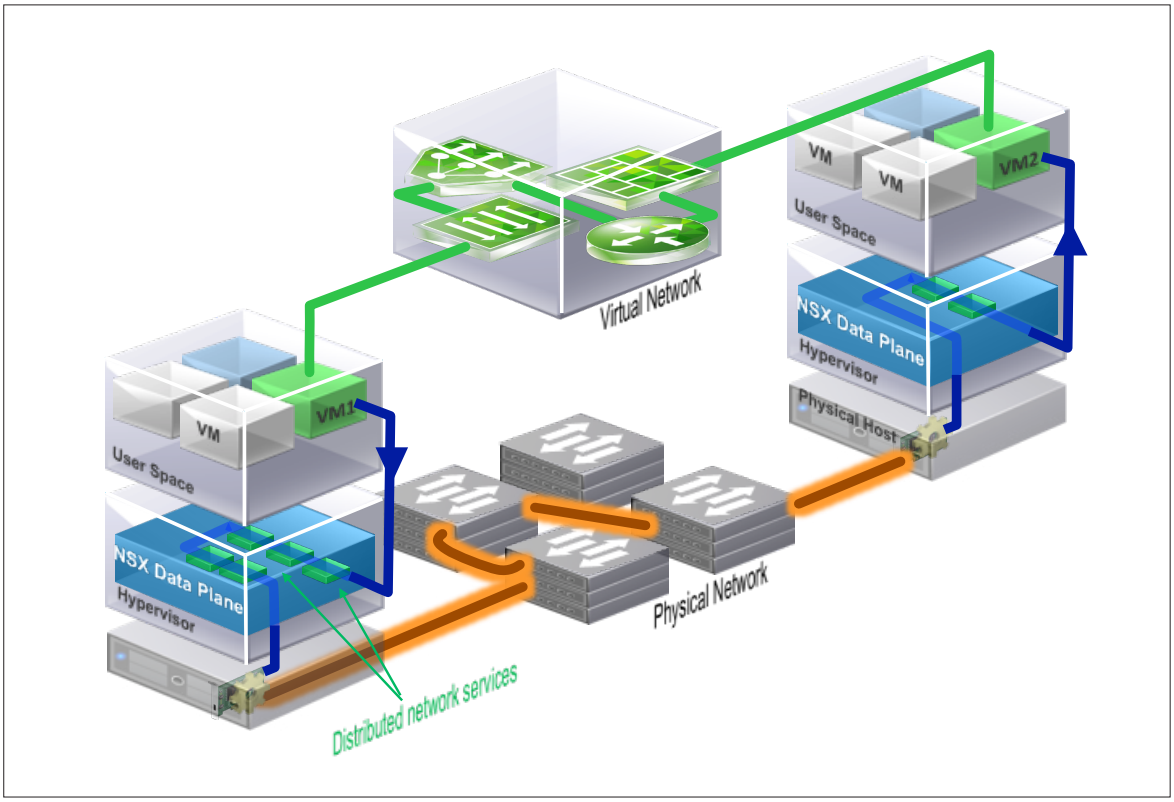


Figure 2: transport view

Traffic from VM1 is in fact fed to local instances of the switches, router, firewall and other services implemented by NSX on the host. Those local instances determine that the destination of the traffic is on a remote host, where VM2 is located. They encapsulate the traffic and forward it to the remote host, where after decapsulation, it is finally presented to the target VM2, as if it had gone through a physical instantiation of those services. The tunnel (represented in orange in the diagram), is seen as plain IP traffic from the point of view of networking infrastructure, and does not require any particular functionality from the physical network.

Control plane

The control plane disseminates the network state for the distributed components like the NSX vSwitches such that they can create the required tunnels and switch the data packets appropriately. In NSX, the control plane is controlled and managed by the NSX Controller Cluster. This is a high available distributed clustered application that runs on x86 servers. The one key aspect about the NSX Controller Cluster is that it does not sit in the data path. It is designed to manage and control thousands of switching devices.

Data plane

The distributed components responsible for providing the network functions for tunneling, queuing management, security and packet scheduling are managed by the controller cluster. However, once they have been configured, they are able to perform their task independently, even in the case the controller cluster should fail completely. NSX creates an overlay by establishing IP tunnels between hosts. This document will focus on VXLAN (Virtual Extensible LAN) as a tunnel encapsulation, as it is the industry standard used for the integration of hardware Layer 2 gateways. NSX can thus exchange data and perform advanced functions without introducing any dependency to the underlying network. That latter only needs to be able to switch efficiently IP traffic between hosts while providing high availability.

In summary:

- NSX does not introduce any requirement on the physical network and provides its advanced features over a multi-vendor or legacy network.
- NSX is fast, flexible and simplifies networking by providing automation. A virtual network can be provisioned in minutes and there is no need to go through an error prone configuration of all the physical devices so that they have a consistent view of the VLANs, the ACLs, or the Firewall rules for example.
- NSX is scalable and efficient: virtual networking functions run in kernel space and as a result, NSX introduces minimal overhead at the edge. Traffic is also forwarded to its final destination using an optimal path. For example, there is never a need for “hair-pinning” traffic through a firewall when the firewall functionality is directly implemented in virtual switch running on the local hypervisor.
- NSX has been able to satisfy the requirements of the largest providers in the world thanks to its distributed and scale-out model.

Extending virtual networks to physical space with Layer 2 gateways

NSX operates efficiently using a “network hypervisor” layer, distributed across all the hosts. However, in some cases, certain hosts in the network are not virtualized and cannot implement natively the NSX components. NSX provides thus the capability to bridge or route toward external, non-virtualized networks. This document is more specifically focusing on the bridging solution, where a Layer 2 gateway extends a logical Layer 2 network to a physical Layer 2 network. This part will go over some use cases and introduce the different form factors for the Layer 2 gateway.

The main functionality that a Layer 2 gateway achieves is:

- Map an NSX logical switch to a VLAN. The configuration and management of the Layer 2 gateway is embedded in NSX.
- Traffic received on the NSX logical switch via a tunnel is decapsulated and forwarded to the appropriate port/VLAN on the physical network. Similarly, VLAN traffic in the other direction is encapsulated and forwarded appropriately on the NSX logical switch.

Use cases

Because virtualization introduces significant benefits in term of flexibility, automation and management in the data center, companies typically try to virtualize as much of their infrastructure as they can. However, there are some cases where total virtualization is not possible and where a gateway from logical to physical world is necessary. Following is a list of use cases for this service:

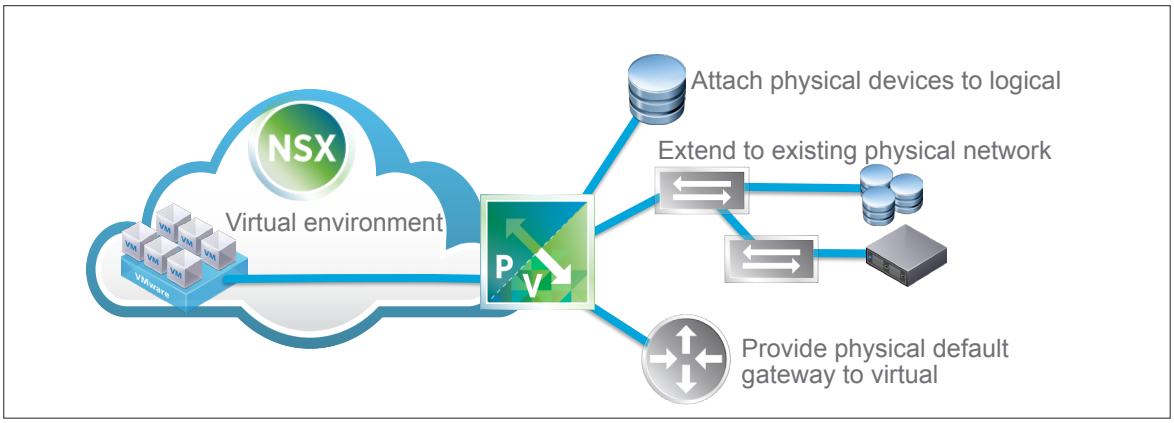


Figure 3: Layer 2 gateway use cases

Physical tier

Because of licensing or performance issues, some database servers cannot be run as virtual machines. The result is that a single application tier must reside in physical space, but the vast majority of the workloads can still be virtualized. The extension of a logical switch to the physical world using a layer 2 gateway still allows getting as many of the virtualization benefits as possible in that case.

Physical to Virtual migration

Another use case of a gateway functionality is the migration of workload from Physical to Virtual that typically requires preservation of the IP address (same VLAN) during the migration process. Extending a logical switch to a physical layer 2 network allows virtualizing servers with minimal impact on their configuration. As a result a newly virtualized server can keep connectivity to its peers, whether they are virtual or physical, allowing for a safe, incremental virtualization of the data center.

Attach physical services

Some services, like firewalls, load balancers or routers providing a default gateway, might be already existing under a physical form factor in the data center and it might not be practical or even possible to virtualize them (in some cases, those devices might be servicing both virtual and physical parts of the network for example.) Extending the NSX logical switches into the physical world will allow virtual machines to easily leverage those physical services. These physical resources can be within or across datacenters.

Software gateway

NSX includes natively a software version of the Layer 2 gateway functionality called Layer 2 Bridge in NSX-v.

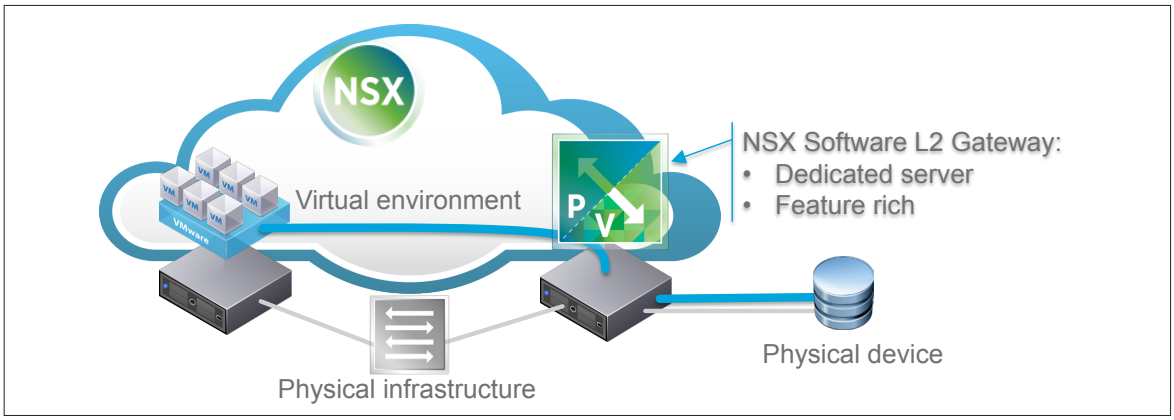


Figure 4: Software Layer 2 gateway

Benefits of a software Layer 2 gateway.

- The functionality can be provided leveraging a generic server, as qualified by the customer's IT department.
- The L2 Bridge is leveraging performance optimized code in the Hypervisor kernel of the host running it. Modern servers can typically achieve wire rate performance on 10Gbps NICs (Network Interface Cards). In most cases, this is enough bandwidth for physical/virtual communication.
- The Layer 2 gateway being implemented in software, it is benefiting from advanced features and capabilities of the NSX release it belongs to.

Hardware gateway

NSX as a platform allows the integration of third party components, and the Layer 2 gateway functionality can be achieved in hardware.

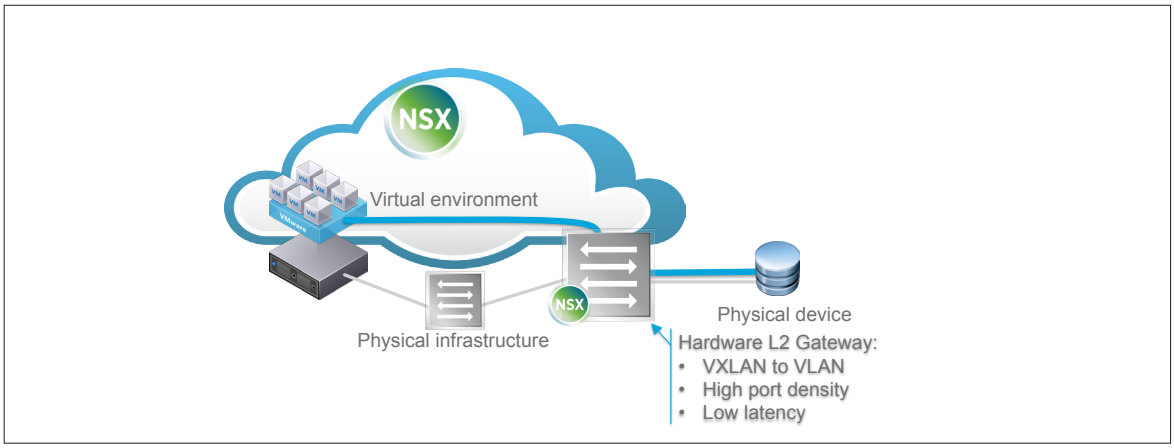


Figure 5: Hardware Layer 2 gateway

This form factor is beneficial to deployments where bandwidth and port density cannot be conveniently achieved by the software gateway.

Technical overview of the solution

This part is representing an overview of the technical aspects of the solution by illustrating the simple case of a virtual machine VM1 connected at layer 2 to a physical server S. The virtual machine is connected to a logical switch extended to the VLAN to which the server is attached to. The following Figure 6 is showing the logical view of the example.

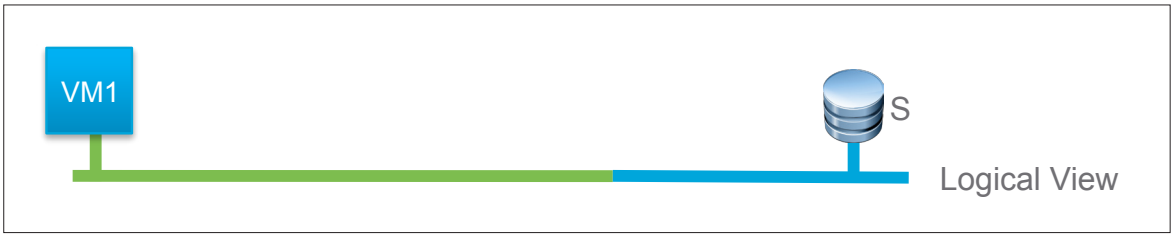


Figure 6: logical representation of an L2 extension

The rest of the chapter will be dedicated to introducing the different NSX components necessary to implement this solution.

NSX components

The following Figure 7 shows a simple possible implementation of the logical view from Figure 6. The components are detailed in the next parts.

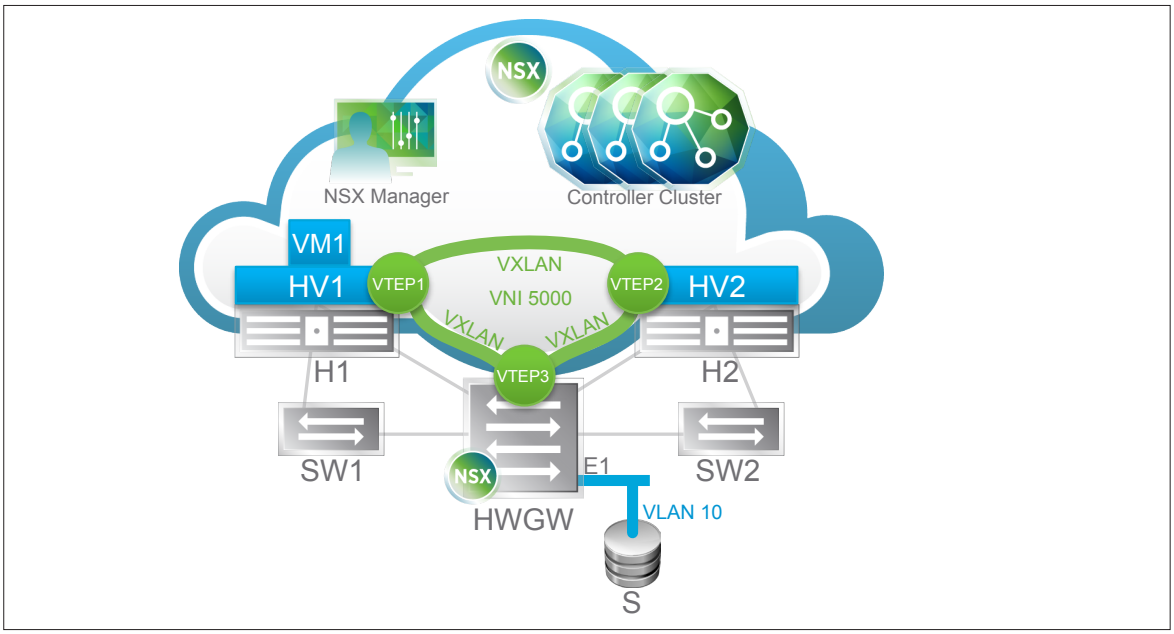


Figure 7: NSX components overview

Physical components

The lower part of Figure 7 shows the physical components in grey. They include:

- Two hosts H1 and H2,
- Three network switches SW1, SW2 and the hardware gateway HWGW
- A server S, attached to HWGW on port E1, VLAN 10.

Note that because the goal is to provide a minimalistic example, the network depicted here is by no means a design recommendation.

NSX only requires IP connectivity between its different components. As a result, the network can be Layer 2 end-to-end just as well as an IP fabric, where hosts might be in different subnets and reachable via routers. NSX does not require any Layer 2 adjacency, does not require any particular routing protocol and does not require IP multicast.

Hypervisors

Figure 7 is showing two vSphere hypervisors HV1 and HV2, on the two different hosts. The hypervisors have been prepared for NSX and are running the distributed components capable of achieving virtual routing, switching, firewalling etc... Our virtual machine VM1 is running on host1.

Controller cluster

The controller cluster is a distributed system that runs on a set of x86 servers and that is responsible for running the control plane of the NSX domain. The responsibilities of the controller cluster also include the instantiation of the components of a distributed logical switch. For example, when a virtual machine is instantiated on a hypervisor, the controller cluster configures the components that implement the logical switch to which this virtual machine is attached. It is also responsible for programming the mac address table that will be used to reach other virtual machine attached to the logical switch.

NSX Manager

Tunnels

The different components of NSX are connected via IP tunnels, represented as green bold lines in the Figure 7 above. As mentioned earlier, tunnels are creating the overlay that is decoupling the virtual network from the hardware.

Virtual Machine, logical switch

The virtual machine VM1, residing on the hypervisor of host1, is connected to a logical switch LS1. The logical switch is not directly represented in Figure 7. However, the traffic forwarded on this L2 broadcast domain is encapsulated in the VXLAN tunnels using VNI 5000 (VNI: VXLAN Network Identifier).

Hardware Layer 2 gateway

The hardware L2 gateway is a physical device provided by a VMware partner and that can be controlled directly by NSX. This gateway will encapsulate/decapsulate the VXLAN traffic from NSX in order to allow the virtual workload to interact with non-virtualized devices. Because we're talking of a layer 2 extension, the gateway in this example will be bridging between a logical switch on the NSX side and a VLAN on the physical side.

Several components are involved in the connection of the hardware gateway to NSX. They are represented in Figure 8.

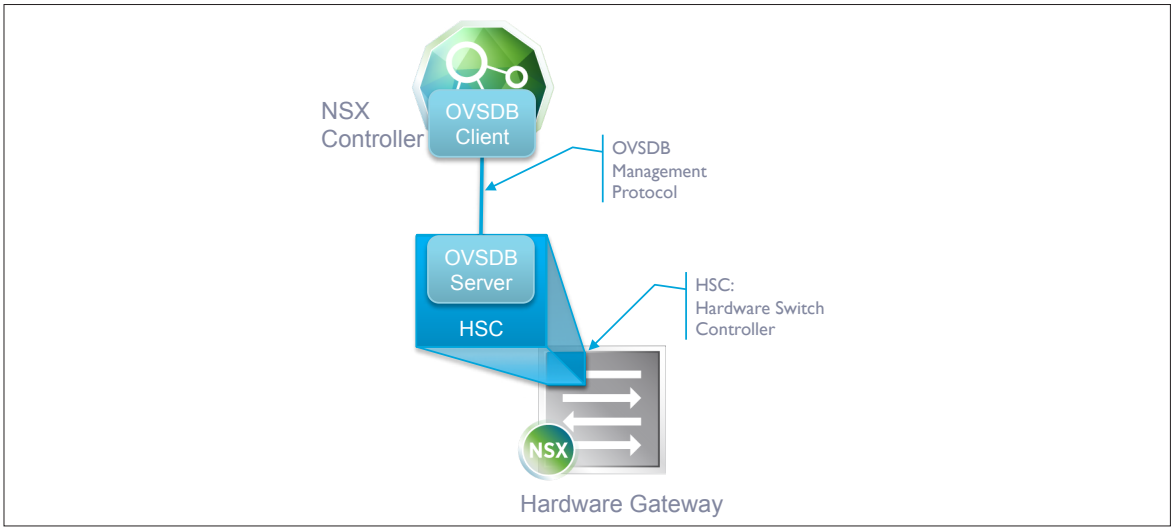


Figure 8: Connection Hardware Gateway/NSX Controller

The NSX controller is responsible for handling the interaction with the hardware gateway. For this purpose, a connection is established between the NSX controller and a dedicated piece of software called the Hardware Switch Controller (HSC). The HSC can be embedded in the hardware gateway or can be running as a separate standalone appliance. The HSC can control one or several hardware gateways. In any case, the communication between the HSC and the hardware gateway(s) is the responsibility of the hardware vendor.
The HSC runs an OVSDB server, and the NSX controller connects as an OVSDB client. OVSDB is the Open vSwitch Database Management Protocol detailed in RFC 7047. It is an open source project that basically provides the capability of managing a database remotely.

This communication channel between the NSX Controller and the Hardware gateway will be used mainly to propagate two kinds of information:

Configuration information.

- The NSX controller will push the administrator-configured association between a Logical Switch and Physical Switch/Port/VLAN to the Hardware Gateway via the HSC.
- The NSX Controller will also advertise a list of Replication Service Nodes (SRNs) that the Hardware Gateway will leverage to forward Broadcast, Unknown unicast or Multicast (BUM) traffic. The SRNs are detailed in the section showing examples of packet flows, further down.

Run-time information.

- The NSX Controller will advertise to the HSC the list of Hypervisor VTEPs relevant to the Logical Switches configured on the Hardware Gateway.
- The NSX Controller will also advertise to the HSC the association between the mac address of the VMs in the virtual network and the VTEP through which they can be reached.

Note that there are several NSX controllers in an NSX deployment, providing redundancy and scale-out. The tasks mentioned as being performed by the NSX Controller are in fact shared across all the NSX Controllers in the network. The HSC will in fact open a connection to all the Controllers.

Packet flow examples

This section illustrates some few examples of traffic flow between Virtual and Physical environment through a Layer 2 Hardware Gateway.

Unicast traffic

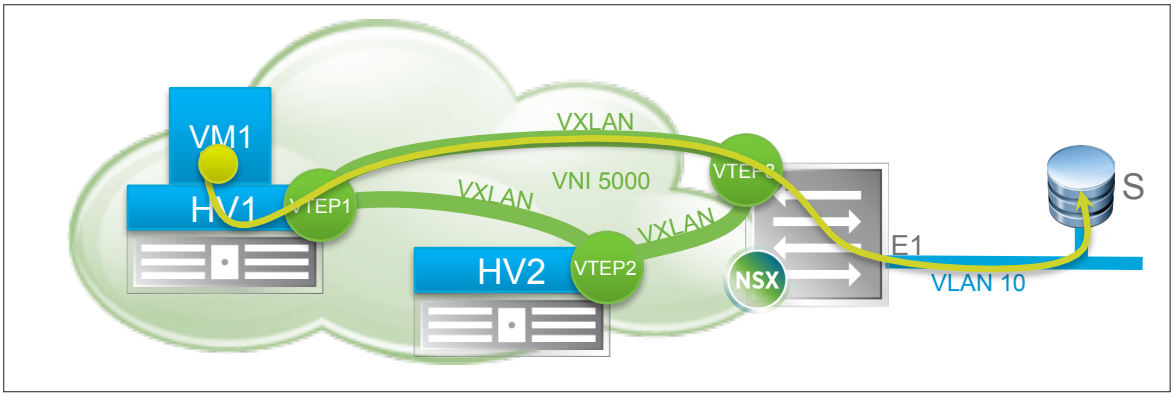


Figure 9: Unicast traffic from VM1 to server S

In the example represented in Figure 9, a virtual machine VM1 sends a unicast frame to a physical server S reachable through a Hardware Gateway. It is assumed that the mac address of S is known, the unknown unicast case being described in the next part.

The traffic from VM1 is following the following path:

1. VM1 sends a frame to server S on its vnic. It is received by the virtual switch in hypervisor HV1. It is assumed that the mac address table already has an entry for S. This entry might have been populated directly by the NSX controller for example, based on information that the Hardware Gateway has advertised to it. So when the virtual switch on HV1 does a lookup on destination mac address S, there is a hit for an entry pointing to VTEP3.
2. Hypervisor HV1 encapsulates the frame with a VXLAN header including the VXLAN Network Identifier (VNI) 5000. This value is associated with the Logical Switch to which the vnic of VM1 is attached. The destination IP address of the VXLAN packet is the one of VTEP3, and it is routed directly to its destination
3. The Hardware Gateway receives the VXLAN encapsulated frame on its VTEP and decapsulates it.
4. The Hardware Gateway does a lookup on the destination mac address of S. The lookup is done in the context of port E1, VLAN10 because the Logical Switch with VNI 5000 is mapped to this port (that association has been pre-configured by the administrator.)
5. There is a hit and the frame is forwarded on that port and reaches its final destination S.

Multidestination traffic

The following Figure 10 represents the case where a Hardware Gateway injects some traffic that needs to be flooded in the virtual network. Broadcast, Unknown unicast and Multicast traffic is handled the exact same way.

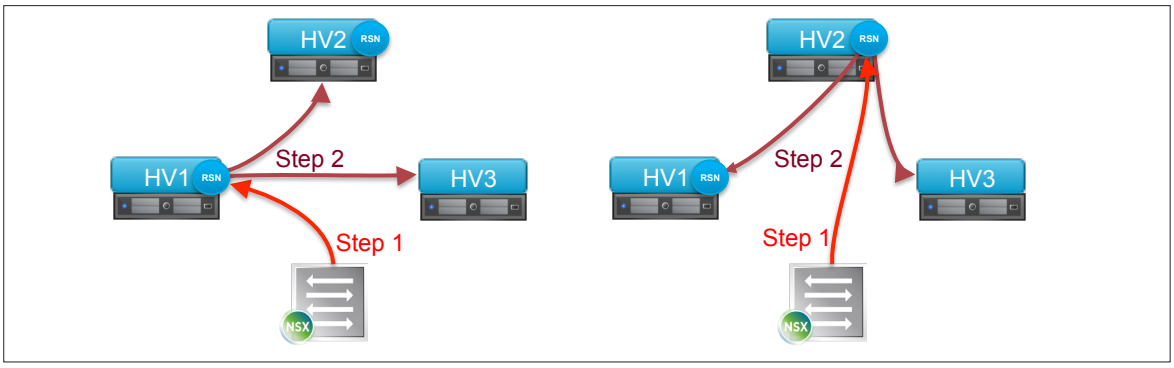


Figure 10: Multidestination traffic injected from a Hardware Gateway

In the above example, the virtual network comprises three hypervisors to which the traffic must be flooded. The Hardware Gateway has established a VXLAN point-to-point tunnel to each of those hypervisors. However, in the current model, the Hardware Gateway will not replicate the frame it needs to flood to each of those hypervisors. Instead, the NSX Controller has provided a list of Service Replication Nodes (RSNs) that the Hardware Gateway will leverage for the replication.

The RSNs are statically defined by the administrator. For each frame that needs to be flooded, the Hardware Gateway picks one RSN, based on a hash value computed from some fields in the frame (the details of the hash is up to the Hardware Gateway vendor) and forwards the frame to this RSN. The RSN then takes care of replicating in software this frame to the appropriate hypervisors that need to receive it. Load balancing is thus achieved by hashing flooded traffic to arbitrary RSNs.

So, in the example of Figure 10, the Hardware Gateway on the left diagram determines that it needs to send this particular multidestination frame to HV1 in step 1. The, the RSN on HV1 will take care of replicating this frame to HV2 and HV3. Note that the hash computed for different frame might have been selected the RSN on HV2. In that case, represented in the diagram on the right, HV2 would have been tasked with the replication to HV1 and HV3.

The Hardware Gateway also maintains a BFD session to each of the RSNs. This way, should an RSN fail, the Hardware Gateway is capable of removing that failed node from the list and avoid black-holing BUM traffic.

Configuration

This section provides an overview of the configuration required on the NSX side for integrating and operating a Hardware Gateway.

Registering a Hardware Gateway to NSX

First, a Hardware Gateway must be registered to NSX. The steps detailed in this section only have to be performed once for this purpose.

The user must configure the HSC of their Hardware Gateway with the NSX controller IP address. Note that there are typically several redundant NSX Controllers, only one of them needs to be specified (the others will be automatically discovered.) This configuration is, of course, dependent on the Hardware Gateway vendor and is not shown in this document.

Once the administrator has pointed the HSC to an NSX Controller, they also need to collect the certificate that will be used by the OVSDB Client on the Controller to connect to the server on the HSC. This step is also hardware vendor specific.

From there, the registration of a new Hardware Gateway in the NSX UI is relatively straightforward in vCenter: Navigate to the Networking & Security/Service Definition tab. Then select the Hardware Devices menu.

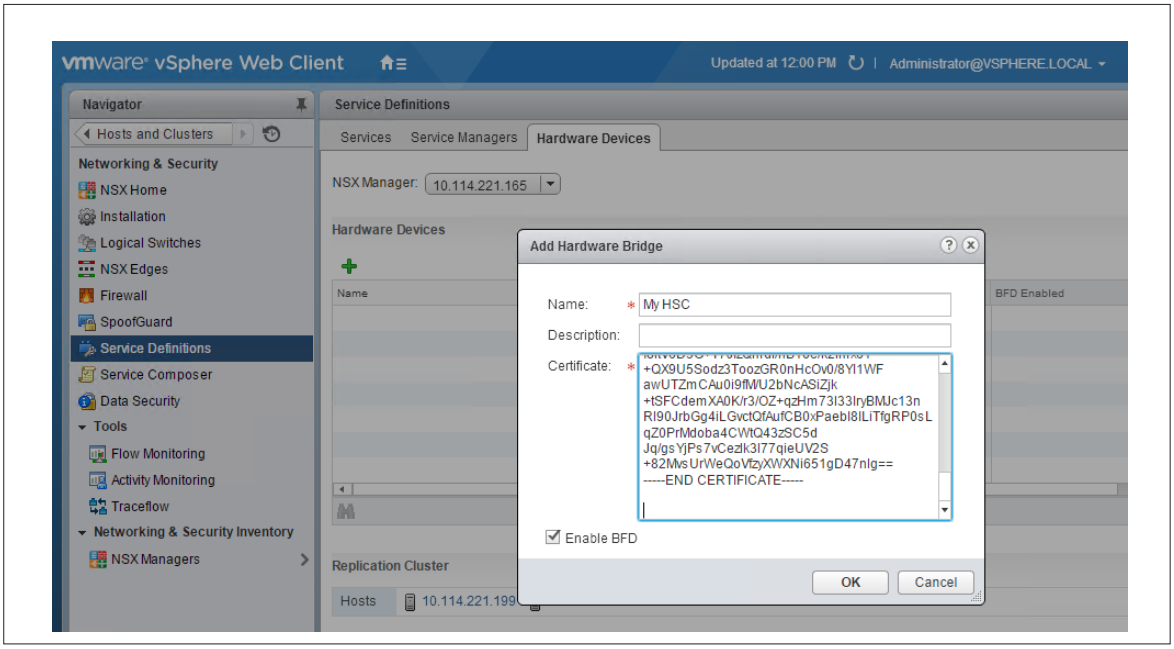


Figure 11: Registration of a Hardware Gateway

Here, you will add a new profile by clicking on the “plus” sign in the Hardware Devices section. Figure 11 above is showing the resulting pop-up window that has been populated with a profile name and the certificate retrieved from the HSC. Note that BFD is enabled by default, meaning the different hardware switches will establish BFD sessions to the RSNs. This is critical for protecting against the silent failure of an RSN and VMware will only support configurations running BFD.

Once this configuration is done, the Hardware Gateway should show up as available, as pictured in Figure 12 below:

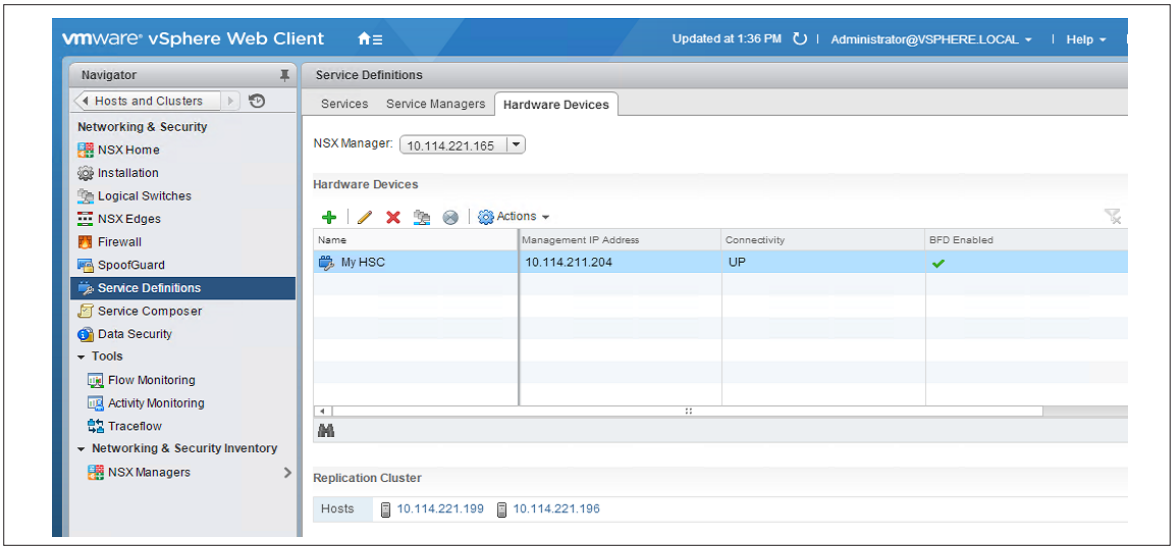


Figure 12: Successful registration

Note also that the Replication Cluster must also be configured on the same screen. The Replication Cluster is the set of Hypervisors that will act as RSNs. The administrator will select a subset of the Hypervisors in the network.

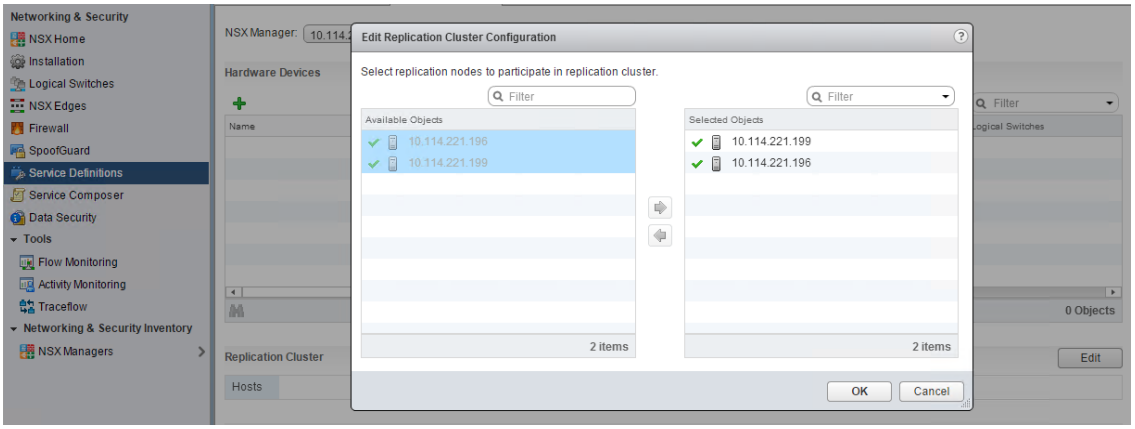


Figure 13: Replication Service Node Selection

In theory, an arbitrary number of RSNs can be configured. The more, the better in term of redundancy and load balancing. However, because each RSN is protected by a BFD session from the Hardware Gateway, it is recommended to limit the number of RSNs configured according to the capabilities of the hardware devices (the CPU of some physical switches might not be capable of sustaining a large number of BFD session, please check the scale information provided by the hardware vendor.)

Binding a Logical Switch to a Physical Switch/Physical port/VLAN

Once a Hardware Gateway is added to NSX, an arbitrary Logical Switch can programmatically be mapped to any physical port/VLAN advertised by this gateway. This section will illustrate the mapping of a logical switch to a particular port, leveraging vCenter UI.

First, the administrator selects a Logical Switch in the Network & Security/Logical Switches tab.

Figure 14: Logical Switch Selection

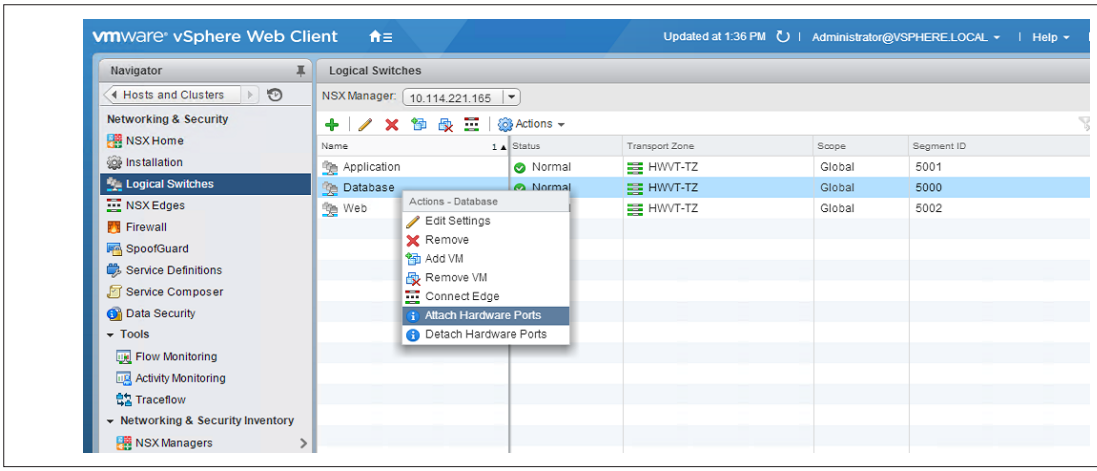
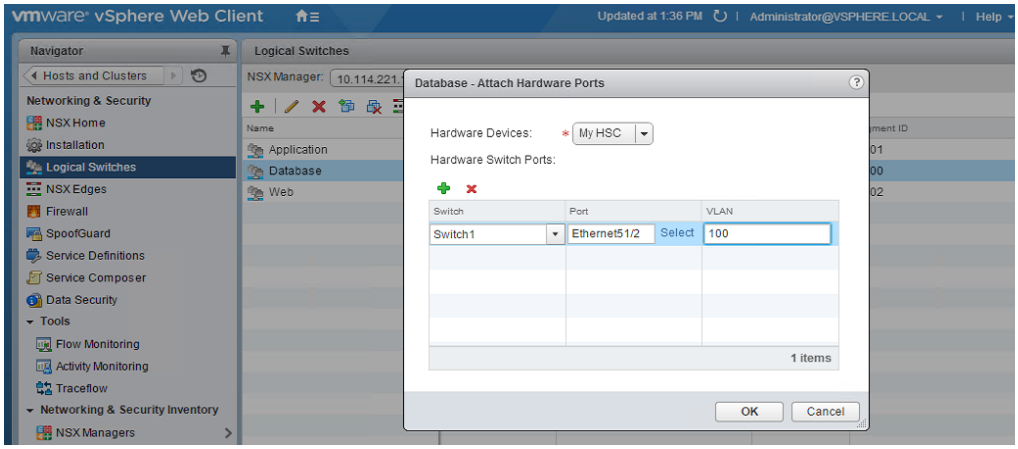


Figure 14 above shows a menu that is dropped down by right clicking on the "Database" Logical Switch entry in the table. From there, selecting "Attach Hardware Ports" will open a pop-up (represented in Figure 15 below) allowing to specify a port binding.

Figure 15: Binding a Logical Switch to a physical switch, physical port and VLAN



Three columns are available in Figure 15:

- Physical switch: remember that the HSC might control several hardware switches, so this selection is necessary to identify which one is concerned by this configuration
- Port: the HSC provides a list of physical ports available for binding on the physical switch.
- VLAN: specify which VLAN tag will be used on the particular port selected. A VLAN value of 0 represent an access port, where the extended Logical Switch traffic will be sent untagged on the port.

Once this selection is done, the Logical Switch is extended to the physical world at Layer 2 on the physical switch/physical port/VLAN specified. Note that several bindings can be achieved for a particular Logical Switch. The hardware vendor might introduce some constraint on those bindings though.

Design Considerations

There is a significant difference between Hardware and Software Gateways:

- With the current implementation, a Logical Switch can only have one active bridging instance at a time.
- On the other hand, there can be several Hardware Gateways extending the same Logical Switch.

This has an impact on redundancy and on the scope of Layer 2 in the network.

Impact on Redundancy

The reason why the Software Gateway is limited to extending a Logical Switch to a single VLAN is to simplify the redundancy model. A single bridge instance is responsible for extending a Logical Switch to a particular VLAN. This active bridging instance is backed by a standby bridging instance that immediately takes over should the active fail. This way, it is impossible to have two bridging instance forwarding traffic at the same time, thus preventing the possibility of a loop between Virtual Layer 2 domain and Physical Layer 2 domain.

The Hardware Gateway integration mechanism allows several physical switches to forward traffic simultaneously between a Logical Switch and some VLANs. As a result, it is possible to introduce a permanent bridging loop as a result of a configuration error. Figure 16 below is showing an example where two physical switches (HWGW1 & HWGW2) are translating the Logical Switch with VNI 5000 to VLAN 10. Initially, the two hardware switches were just connecting directly to host1 and host2. Then, an operator inserted the switch SW1 between the two Hardware Gateways and their hosts.

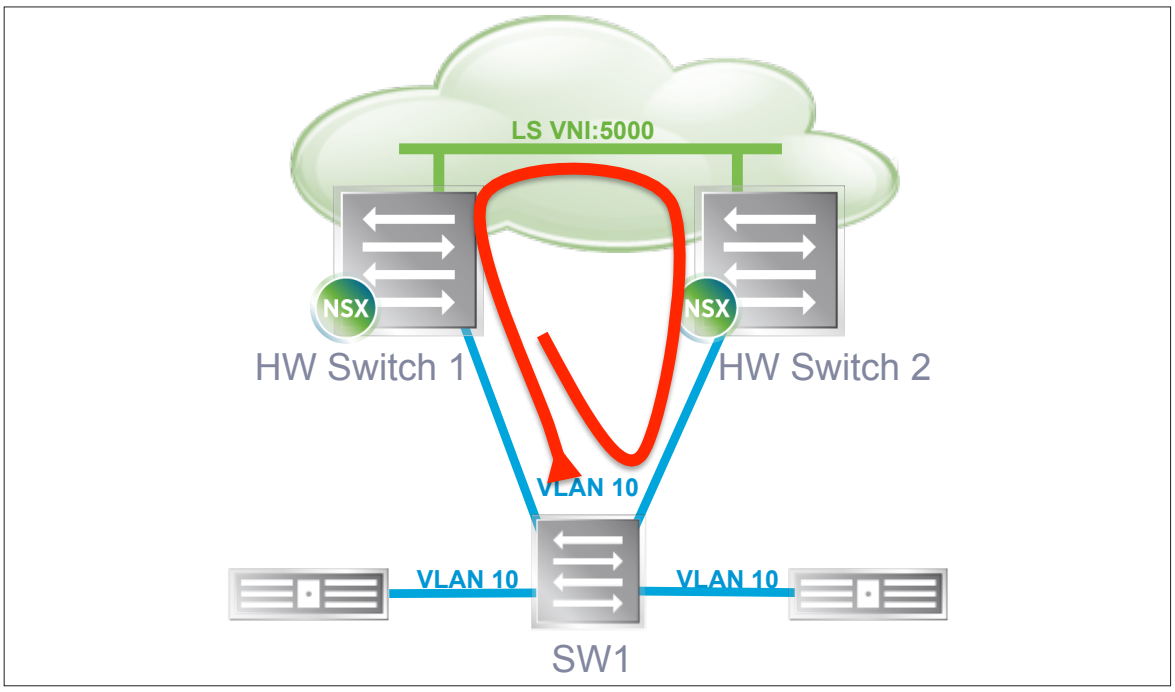


Figure 16: Loop between Hardware Gateways resulting from a cabling error

This operation resulted in a loop between VLAN 10 and the Logical Switch with VNI 5000 (the loop is represented with the red arrow in Figure 16.)

In order to minimize this risk, it is recommended to only connect hosts to physical ports that are extended to a Logical Switch.

The current Hardware Gateway integration model does not have any concept of redundancy built-in. Redundancy has to be provided at a lower layer. Redundancy can be achieved at the host level itself. The following Figure 17 shows how to connect redundantly a host to two active Hardware Gateways.

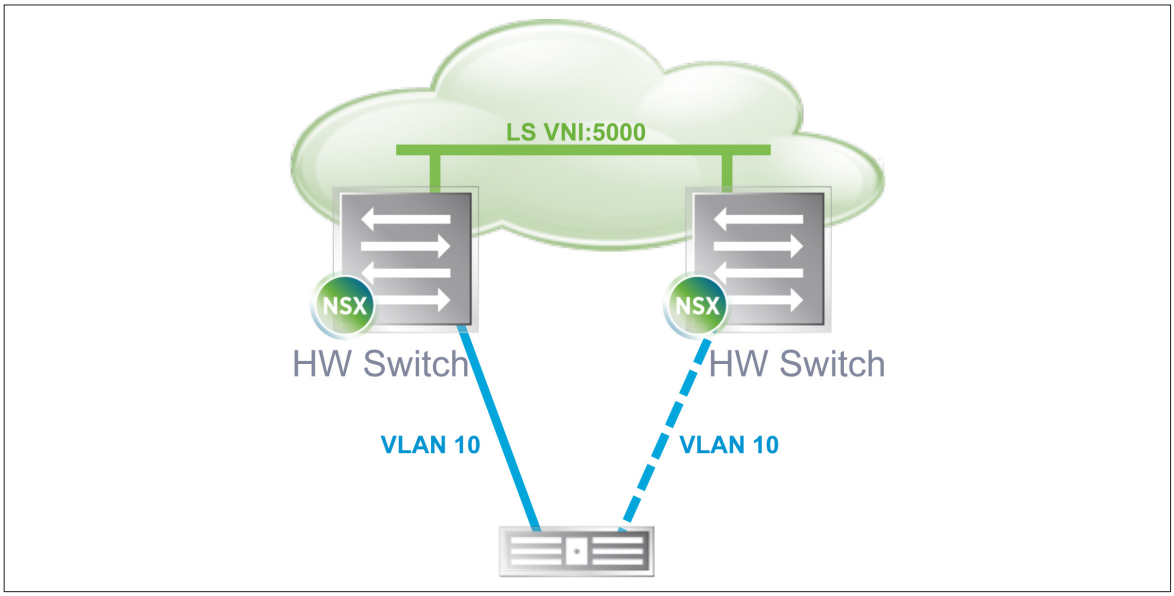


Figure 17: Host driven redundancy

Note that in this scenario, it's up to the host to elect a standby uplink and to decide whether to fail-over. Another model of redundancy is achieved by the networking infrastructure itself. Most partners providing the Hardware Gateway integration to NSX are offering this functionality, generally based on a distributed port channel mechanism. Figure 18 below is illustrating how those redundancy scheme are seen by NSX

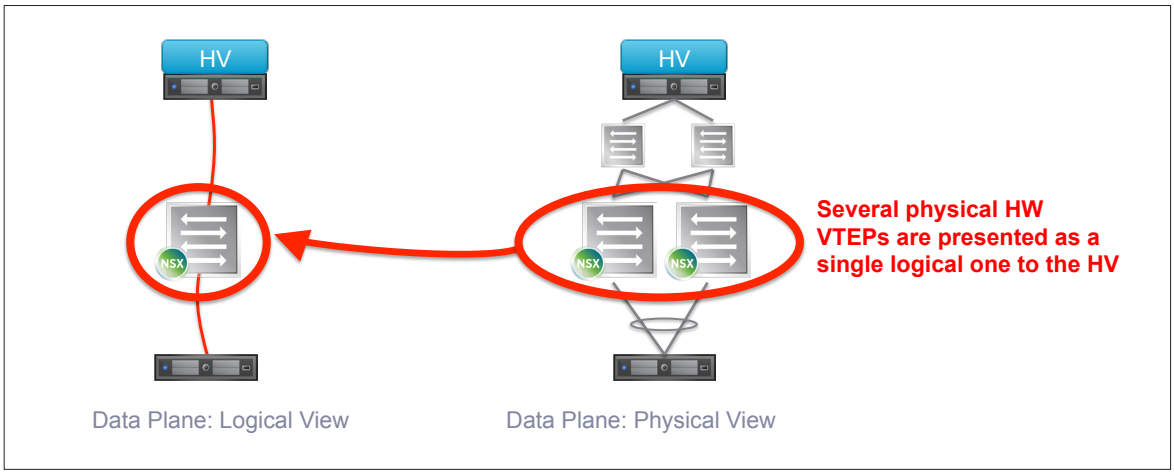


Figure 18: Network infrastructure-based redundancy

The physical redundancy is achieved by the NSX partner boxes that are emulating a single Hardware Gateway out of many physical devices and present it to NSX. The mechanism is often based on some distributed port channel mechanism (VPC, MLAG etc…) in order to attach the host redundantly on the physical side.

Impact on the scope of Layer 2 in the network

The fact that several Hardware Gateways can be active at the same time can also influence the network design. Typically, a Logical Switch is extended to a VLAN in order to provide connectivity to some service that cannot be virtualized. This service is usually redundant, meaning that its physical implementation spans several different racks in the data center. In the left part of Figure 19 below, some virtual machines attached a Logical Switch access physical servers through a Software Gateway. All the traffic from the Logical Switch to the VLAN 10, where the physical servers are located, have to go through a single bridging instance.

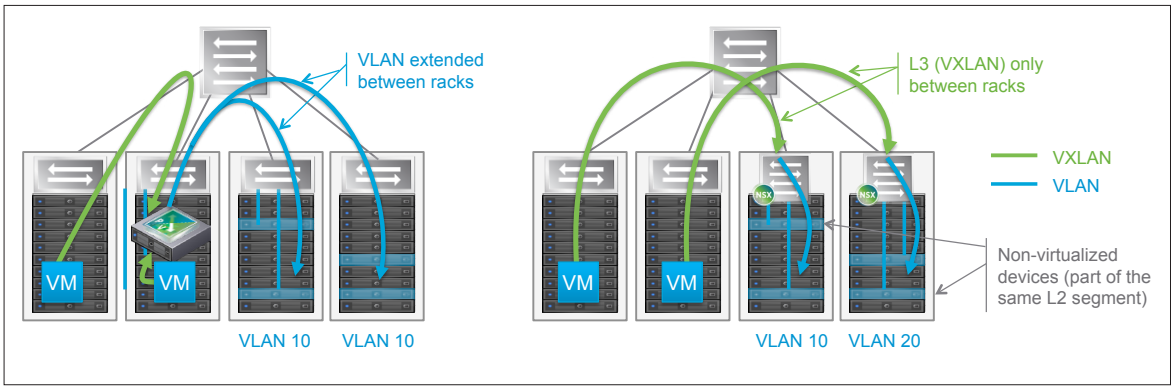


Figure 19: Network

This means that VLAN 10 has to be extended between racks in order to reach all the necessary physical servers. The trend in data center networking in the last few years has been to try to reduce as much as possible the span of Layer 2 connectivity in order to minimize its associated risks and limitations. The right side of Figure 19 shows how this can be achieved leveraging separate active Hardware Gateway. In this alternate design, each rack hosting physical servers is configured with a Hardware Gateway. Thanks to this model, there is no need to extend Layer 2 connectivity between racks, as Logical Switches can extend directly to the relevant Hardware Gateway when reaching physical servers. Note also that the VLANs defined in the racks have local significance (the example is showing that the Logical Switch is extended to VLAN 10 in one rack and VLAN 20 in the other.)

Conclusion

VMware NSX is the platform providing network virtualization in VMware's Software Defined Data Center vision. However, NSX is not limited to virtualized environments and is capable of integrating efficiently physical workloads. This integration can be achieved by the way of Software Gateways or Hardware Gateways. If Software Gateways are full featured and provide high bandwidth, there are still some use cases where high port density is required to access several non-virtualized workloads close to wire-speed. Hardware Gateways developed by third party vendors provide this functionality. As summarized in this paper, those Hardware Gateways can be inserted in an NSX environment with minimal one-time configuration, their day-to-day operation is also completely consistent with the NSX experience (API or vCenter UI driven configuration.)

In conclusion, Hardware Gateways provide new ways of designing networks based entirely on end-to-end Layer 3 fabric, even when extending Virtual to Physical. This model offers high bandwidth and resiliency compared to the one relying on extended Layer 2 connectivity. With this approach, VMware and its partners are delivering the industry's first fully automated and programmatic SDDC solution that bridges the virtual and physical infrastructure.

vmware

VMware, Inc. 3401 Hillview Avenue Palo Alto CA 94304 USA Tel 877-486-9273 Fax 650-427-5001 www.vmware.com