Technical white paper

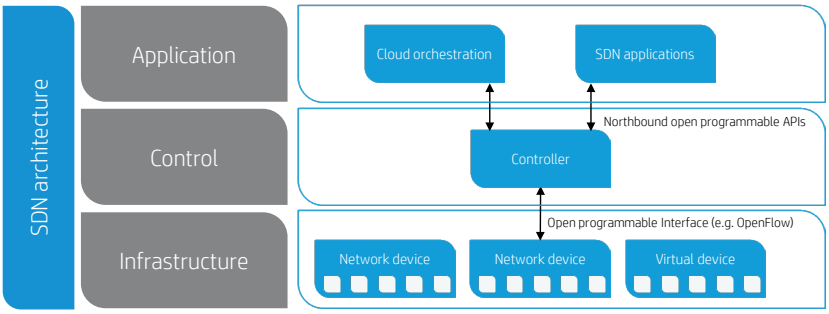
Software-defined networking and network virtualization

Unifying the virtual overlay and physical underlay

Table of contents

Software-defined networking (SDN), as defined by the ONF, is the physical separation of the network control plane from the forwarding plane, and where the control plane controls several devices. When it comes to network virtualization, the SDN approach allows the network provider to integrate physical and virtual environments; and, if done correctly, it also unlocks never-before realized capability, intelligence, and visibility. The network provider has a choice in how this integration is accomplished—a choice which has direct implications on whether they will merely solve their network virtualization problems or whether they'll place themselves on a path to unlock the full potential of an intelligent, SDN-enabled converged infrastructure.

Figure 1. Separation of control and infrastructure



To begin, it is important to understand certain network virtualization and SDN concepts to fully understand the available benefits of using SDN as a technology to realize network virtualization.

Network virtualization

There is no standard that defines network virtualization. One of the better definitions is Gartner's definition:

Network virtualization is the process of combining hardware and software network resources and functionality into a single virtual network. This offers access to routing features and data streams that can provide newer, service-aware, resilient solutions; newer security services that are native within network elements; support for subscriber-aware policy control for peer-to-peer traffic management; and application-aware, real-time session control for converged voice and video applications with guaranteed on-demand bandwidth. (gartner.com/it-glossary/network-virtualization)

Combining hardware and software network resources and functionality into a single virtual network

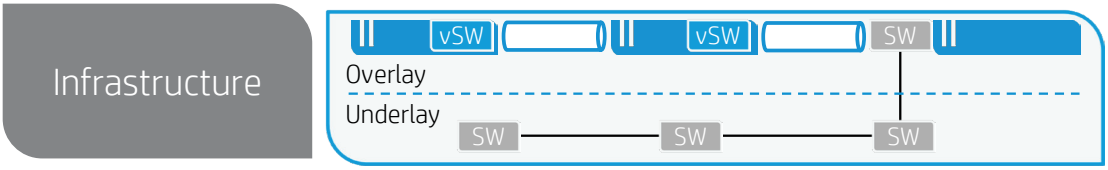
VMware led the industry into a new virtual computing era in the data center. Virtual machines, services, and workloads are now being brought up and down continually, which has led to a new level of automation not seen before. As virtualized data centers grew, virtualized workloads began to span multiple server racks spanning across a large networking domain. These virtualized workloads also need virtual domains to exist to provide logical traffic separation. These virtual domains expand past the server into the network in the form of VLANs. The need for a large number of VLANs began to grow along with the growth in workloads moving to virtualized environments; and this started to expand past the limit allowed on the network. Also, the rise in moving virtual machines (VM) from one server to another, VM mobility, created a requirement on the networking infrastructure to support the movement of IP/MAC addresses across the network infrastructure. Further, leaving everything to hardware slows the rapid pace of innovation allowed in software.

To address the requirements for automation, large number of virtual domains, and VM mobility, there were several decisions to be made. New networking technologies, such as TRILL, created one flat layer-2 domain across the network and alleviated the VM mobility problem. This left the issues of automation and VLAN scale. PBB originally started to address the scale issue with MAC-in-MAC encapsulation. However, it addressed only part of the problem and along with TRILL and didn't address the real need to move to a software-defined strategy to unlock faster innovation that the network has lacked.

Overlay and underlay networks

To address the VLAN scale within the timeframes and flexibility required by rapidly scaling public clouds, Virtual Extensible LAN (VXLAN) was created to provide established tunnels between endpoints with a large scale of virtual domains. VXLAN created a new header to the packets forming tunnels between physical and virtual endpoints. The overlay was managed through a communication and control protocol on the physical network; and it happened largely independently of the actual virtual environment.

Figure 2. Overlay and underlay



History of overlay and underlay

- Looking back at networking history, prime examples of how this problem has been solved came from network service providers. In a sense, these network service providers were providing segmented networks, at large scales, for unique customers while addressing issues of VLAN space and overlapping IP/MAC amongst their customers. This is when the concept over overlay networking was taken to the full scale. Protocols, such as Multi-Protocol Label Switching (MPLS) created tunnels across networks. Then, other protocols, such as Virtual Private LAN Service (VPLS), created unique domains within these tunnels for each customer of the network provider. With this addition of overlays, signaled at the service edge, there remained challenges in providing holistic service insertion and lack of end-to-end support. Further, there was an explosion in the control layer as the solutions scaled.

Also, during the same time, Nicira/VMware was looking at the same issues. In this case, the overlay network began and ended inside the server, more specifically the virtual switch, and was controlled separately from the actual underlay. Nicira created its own tunneling framework based on the Stateless Transparent Tunneling Protocol (STTP) (tools.ietf.org/html/draft-davie-stt-02).

Along with any higher level abstraction, overlays lose some basic fundamental understanding of what is happening in the layers below. This is especially the case with true software overlays, where there is no fundamental interaction between the overlay and the underlay.

History of signaling between overlay and underlay

- Looking back at the MPLS and VPLS examples, many protocols were created to provide intelligence of the lower level protocols to the upper level. Signaling protocols, such as Ethernet OAM or even BFD, are used to monitor anything from link faults to link quality. Even outside of MPLS, you have BFD for BGP or other layer-3 routing protocols to detect and failover when communication across various components of the lower level protocols are impaired. One key differentiator between these scenarios and Nicira/VMware overlay technologies is that the overlay is run completely separately, starting from the server, than the actual lower level protocols.

One other consideration of a software-only overlay solution is that of bridging the servers that are virtualized and can understand the overlay and the servers that are not virtualized and do not understand the overlay. This needs to be done by some gateway device that will bridge the two. Instead of creating a new class of devices in the middle, the actual physical switch, which is part of the underlay, can become a gateway to the overlay. Now we are left with a software overlay connecting virtual machines to other virtual machines that are completely abstracted from the underlay. This is combined with some other virtual machines connected to other non-virtual devices through the gateway functionality running in the underlay. Combining this all together into one architecture sounds fairly confusing; but it does not need to be.

Combining the virtual overlay and physical underlay

The criticality of integration points was mentioned at the beginning of this paper. As migration to virtualized environments continues, it is approaching a saturation point. Customers will generally not virtualize their entire environments due to a number of reasons: first, investment protection of exiting assets to optimize ROI; and secondly, the mere need to support applications that do not perform well in virtual environments, such as databases. This makes it mandatory to bridge the two environments through the gateway functionality on the switch. With the VMware NSX platform, this bridging is done through VXLAN tunneling between the virtual and the physical, where one tunnel endpoint is on the virtualized server's virtual switch and the other is on the physical server's physical switch.

There are, however, some key challenges with a network virtualization approach, which treats the physical transport supporting it as a completely separate entity. It has been pointed out in the past that such an approach assumes configuration of the underlying fabric, requiring the orchestration and management of the physical network. Enterprise network and hosting providers will, however, highlight an additional pitfall: such a separation between the logical network environment supporting the applications and the transport introduces an operational barrier between the applications enabling the business and the networks those applications depend on. For example, when a communication problem occurs, how does the network organization, limited to mere visibility of UDP encapsulated packets, troubleshoot and resolve network issues. This problem is exacerbated by the integration of physical edge ports into the virtual overlay network.

Implementing only the gateway functionality on the physical switch leaves a vacuum in overall orchestration, visibility, and troubleshooting. Take for example, a link failure in an ECMP group that is load balancing traffic across various links in the physical network. The network switch does not know virtual from physical and generally will use a hashing algorithm to choose links based on the input of the hashing algorithm (source IP, destination IP, source MAC, destination MAC, ports, etc.). Thanks to many innovations in data center switching, this link failure is largely missed by the application, as there are many links in that ECMP group still available to send traffic. What may be noticed is the loss in aggregate throughput caused by the link going down. To the application, this just looks like poor performance; and what comes next is the application team scrambling to figure out a root cause.

History of integration of underlay and overlay

Looking back at port channels in general, there has traditionally been a "min-links" command to specify the minimum number of links needed to be active for the port channel to be up, before failing and letting another path take over. This works well in homogeneous environments. But with virtualized environments, the "min-links" will be a completely different value depending on the application. Further, virtualization enables mobility and can open up a whole new set of possibilities on how to handle traffic.

History shows that a tight integration between higher levels of abstractions (the overlay) and the foundation they are built on (the underlay) is an inherent necessity for creating a true end-to-end architecture.

Integrated network virtualization solutions

In order to address the end-to-end requirements and truly realize the agility gained from a SDN virtualization solution, the integration must happen at the control layers. Just as previous hardware enabled overlays had tight integration between each control protocol, enabling tight integration between the virtual overlay control and the physical underlay control mechanisms will provide the combined virtual and physical gateway functionality, and the visibility and control needed to scale and realize true agility.

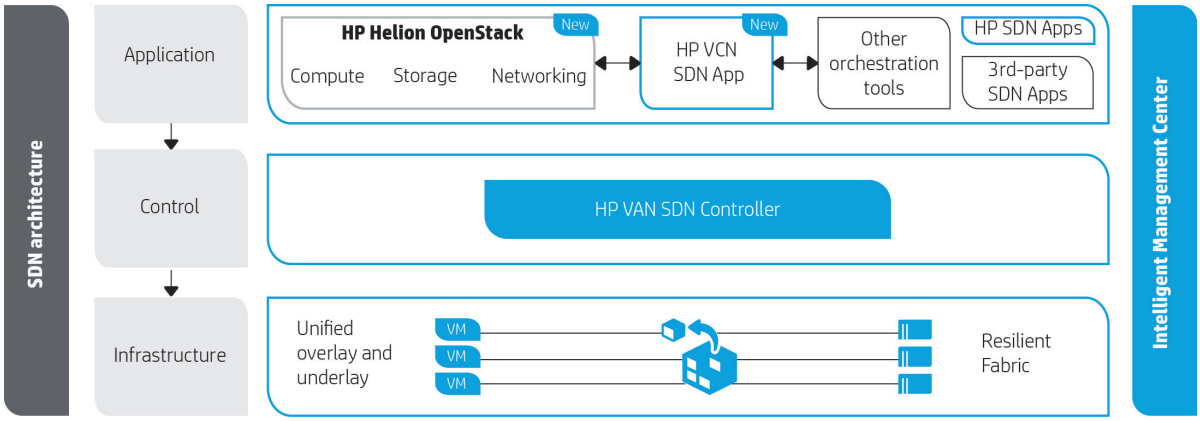
There are two solutions to provide this integration:

1. Provide an integration layer between SDN and the higher level orchestration tools.
2. Provide physical east-west federation with existing network virtualization control.

Integrated SDN Network Virtualization

In virtualized environments, cloud orchestration tools such as OpenStack, offer a high-level mechanism that provides ease of deployment for applications. These tools leverage integration between all the lower level infrastructure, server, storage, and networking. Providing SDN integration with these orchestration tools, allows for not only an integrated orchestration experience, but unifies physical and virtual and innovations with SDN.

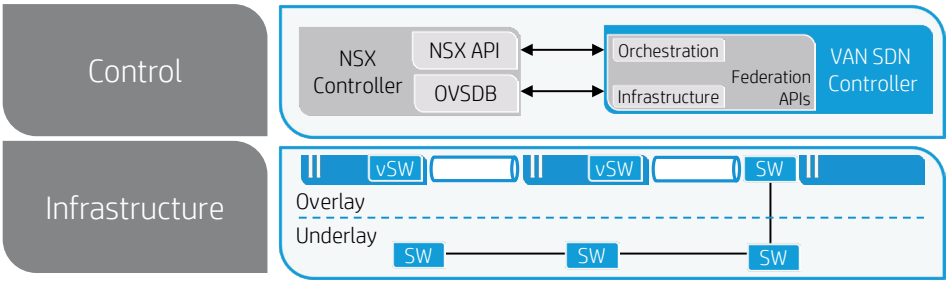
Figure 3. HP Virtual Cloud Networking SDN application



East-west federation APIs

In order to address the end-to-end requirements and truly realize the agility of a software-defined networking virtualization solution, the integration must happen at the control layers. Just as previous hardware-enabled overlays had tight integration between each control protocol, enabling tight integration between the virtual overlay and the physical underlay control mechanisms will provide the combined virtual and physical gateway functionality as well as the visibility and control needed to scale and realize true agility. This is done by federating the control layer through east-west APIs between the two control layer components.

Figure 4. Federation of controllers to provide integration of underlay and overlay



Looking back at the ECMP link failure example, now if the link failure happens, the physical underlay controller can signal the failure to the virtualization overlay controller. Today, there are lots of options on what to do. The virtual controller can move the workload elsewhere or the physical controller can enter prioritization rules based on applications, combined with many such options.

This solution puts us on a path toward a converged infrastructure, empowered by a software-defined and fully integrated network that unleashes the promise of an agile enterprise.

Learn more at
hp.com/networking/sdn

Sign up for updates
hp.com/go/getupdated

Share with colleagues

Rate this document

4AA5-0092ENW, June 2014, Rev. 1