Software-Defined Networking

INFR 4621U – Garrett Hayes

Excerpts and figures from: Data Center Handbook by H. Geng, 2015

SDN: The Big Picture

- The main goal of SDN is to detach the network control plane from the forwarding plane, enabling standardization and network agility
  - VMware NSX and OpenStack's Neutron are examples of SDN software

- Three main components of all SDN systems:

  1. Controller(s) – Provide centralized control over overlay network topologies, distributed routing/switching policies, and other software-defined network resources

SDN: The Big Picture

2. Southbound APIs – Relay routing and switching information/policies to physical and virtual devices downstream (often the hypervisor)
    - OpenFlow is an example of a southbound API system
3. Northbound APIs – Integrate with upstream business logic software or cloud orchestration software (OpenStack, for example)

SDN: A Layered Approach

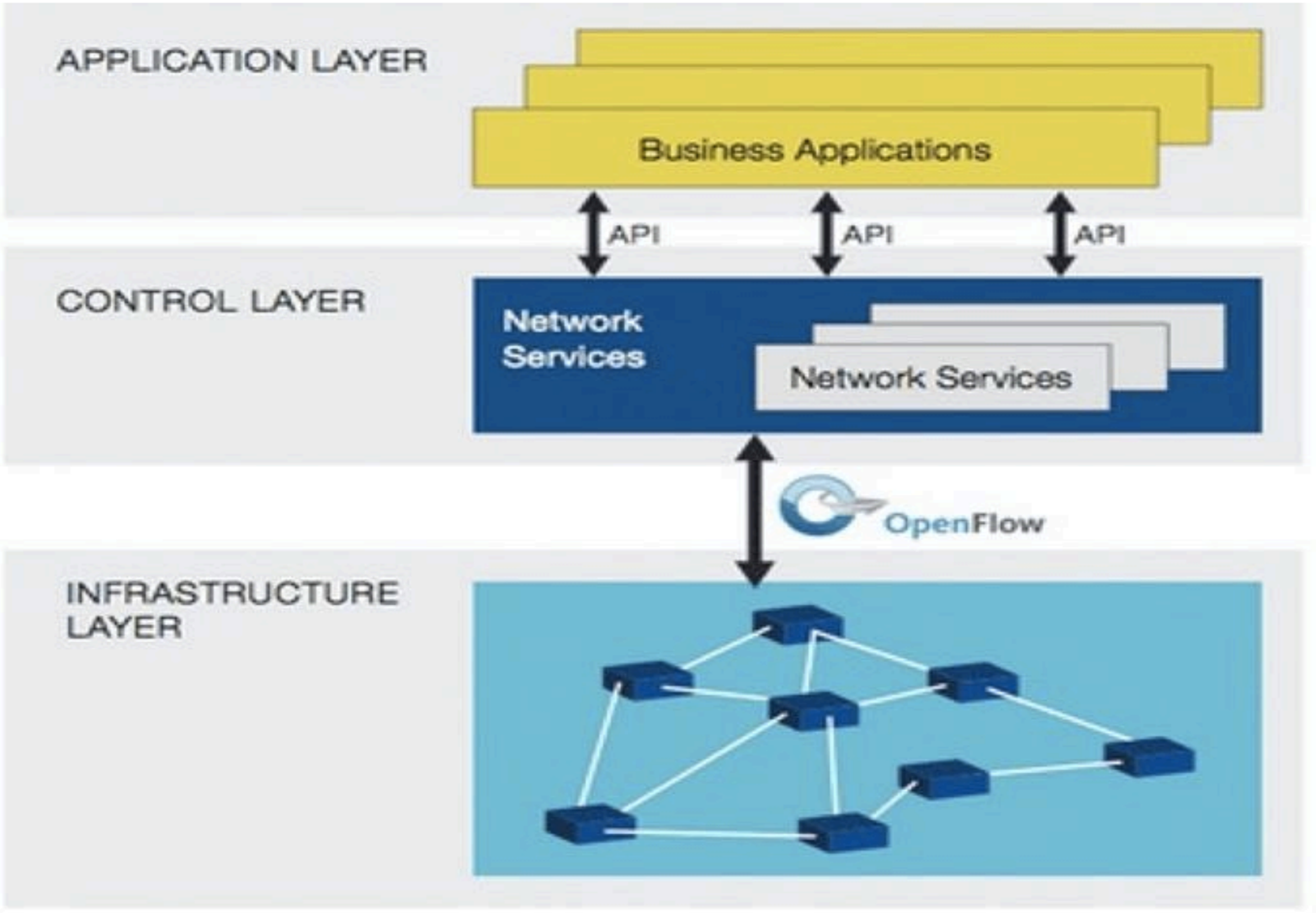


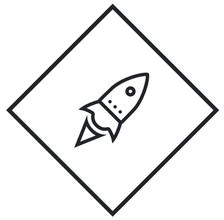
Image Source: SDN Central

SDN Benefits

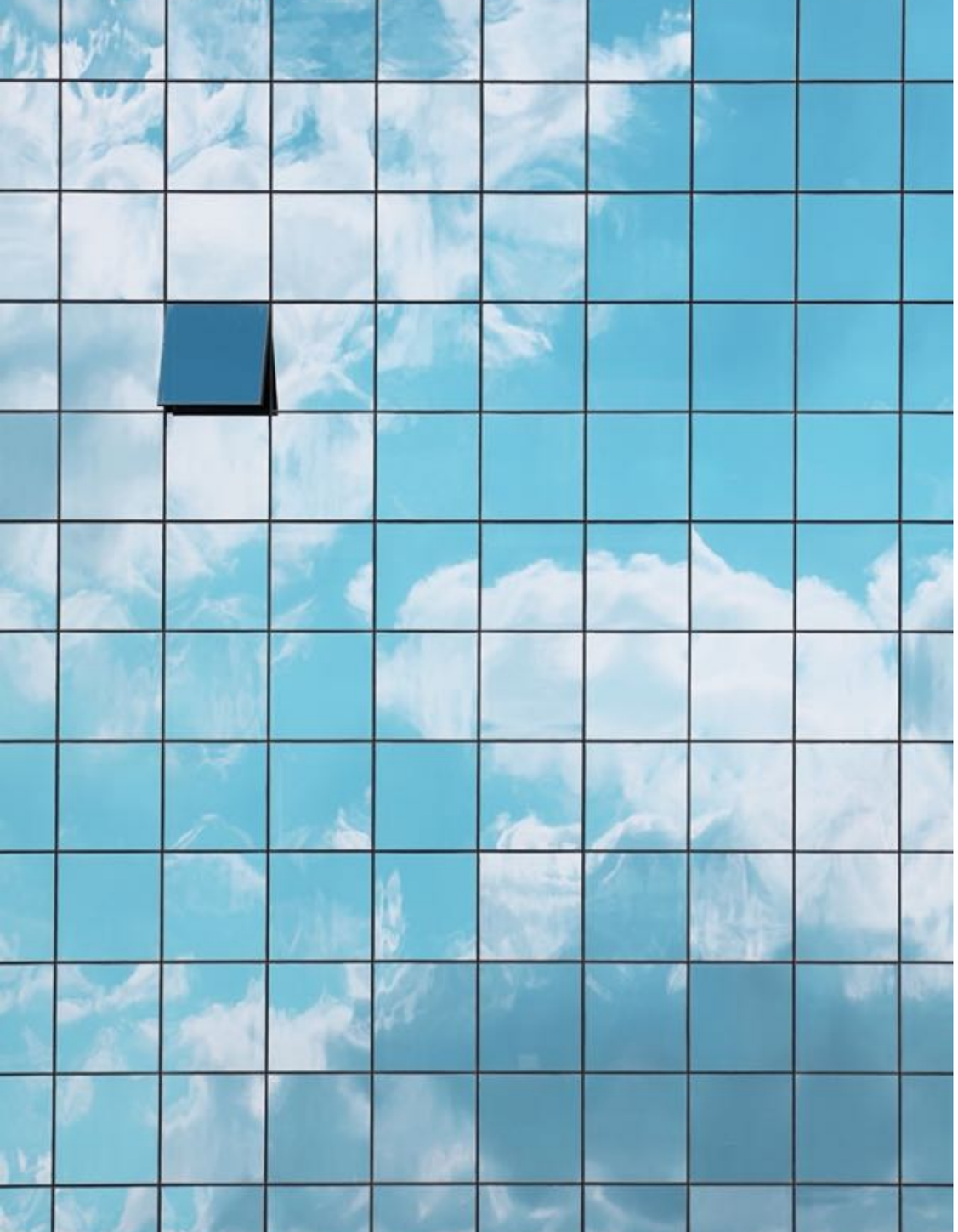
- Standardized and programmable
  - Allows networks to be deployed, destroyed, and modified programmatically using northbound APIs (with OpenStack, for example)
- Reduction of OpEx
  - Complex topology changes can be automated without requiring human intervention (also reducing human error)

SDN Benefits

- Reduction of CapEx
  - No vendor or ASIC-specific devices are required for the physical network
- Increased flexibility and agility
  - Networks are centrally managed, easing troubleshooting and policy management



Architectural Components



Architecture Components

- Most SDN systems contain the following architecture components:

1. Physical Network Plane
2. Data Plane
3. Control Plane
    - This is the southbound API component
4. Management Plane
    - This is the northbound API component
5. Cloud Plane

Physical Network Plane

- The physical network plane provides basic L1/L2 connectivity
- Frames are punted up to a software layer that facilitates the SDN overlay networks
  - Open vSwitch, for example
- Network efficiency is directly impacted by the number of packets sent from the data plane to a software controller
  - Ideally the CPU should be used minimally in order to increase efficiency
  - Some frame-specific offloading can be done using NIC ASICs

The Data Plane

- The following components need to be present on each hypervisor to provide the SDN overlay on the data plane layer:
  - Logical switching software (e.g. Open vSwitch)
  - Logical routing software (if supported by the hypervisor)
  - Distributed routing agent (across hosts)
  - Distributed switching agent (across hosts)
  - Distributed L3+ agents (firewall, VPN, security groups, etc.)

Control Plane

- The control plane consists of clustered and highly-available controller nodes capable of:
  - Tracking routing topologies, distributed VLANS, etc.
  - Providing tunneled L2 connectivity between VMs using VXLAN
  - Suppressing ARP and broadcasts between domains
  - Playing a supportive role for assisting routing and switching (for example, responding to ARP queries for the distributed gateway device)

- This component can also be called the southbound API

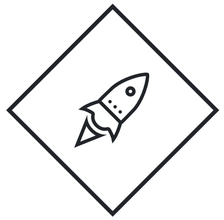
Management Plane

- The management plane is a piece of centralized software with a standardized API that can delegate to the control plane based on user actions
  - Also called the northbound API

- Users or administrators can connect to this API to:
  - Create networks
  - Destroy networks
  - Create additional overlay networks of different types (VXLAN, VLAN, GRE, flat)
  - Modify logical subnets and additional overlay services (DHCP, for example)

Cloud Plane

- The cloud plane is not really part of the SDN architecture, rather it is the layer that sits above the whole SDDC deployment!
- This plane refers to the integration between SDN and cloud management platforms, IaaS automation software, and other related services
    - OpenStack is an example of a cloud plane, whereas Neutron is the actual SDN software



SDN Overlay Components

Overlay Components

- The following network components are provided to overlay networks managed by SDN:
  - Switching & Routing
  - Load Balancing
  - Firewalling
  - VPN
  - Bridging physical and virtual networks

Traditional Switching

- Traditional switching downsides:
  - Large L2 networks tend to have difficulties with STP
  - Hardware capacity may be fully utilized as the L2 network increases in size (MAC, FIB, etc.)
  - L2 segmentation may be difficult to manage or needs may be greater than capacity (e.g. running out of VLANs)

Logical Switching

- Logical switching benefits:
  - L2 over L3 infrastructure!
  - Overlayed and distributed (therefore, scalable) using VXLAN
  - Logical switches can span multiple hypervisors, regardless of vendor
    - This allows distributed VMs to communicate with each other as if they’re on the same switch

Logical Switching: Open vSwitch

- Open vSwitch is a software-based switching technology capable of providing full L2/L3 switching functionality in virtualized environments
  - All normal switching features are supported like: STP, LACP, QoS, VLANs
  - Hardware acceleration is possible using generic NICs with ASIC offloading
  - Open vSwitch is installed on each hypervisor

- Management of virtual switches is programmable (surprise!) using a variety of southbound APIs like NetFlow, SPAN, RSPAN, etc.

- Virtual switches are distributed between hypervisors transparently (even in mixed vendor environments), abstracting underlying L2/L3 technologies

- Open vSwitch is supported on KVM, Xen, XCP, and Hyper-V (unstable port)

Logical Switching: Open vSwitch

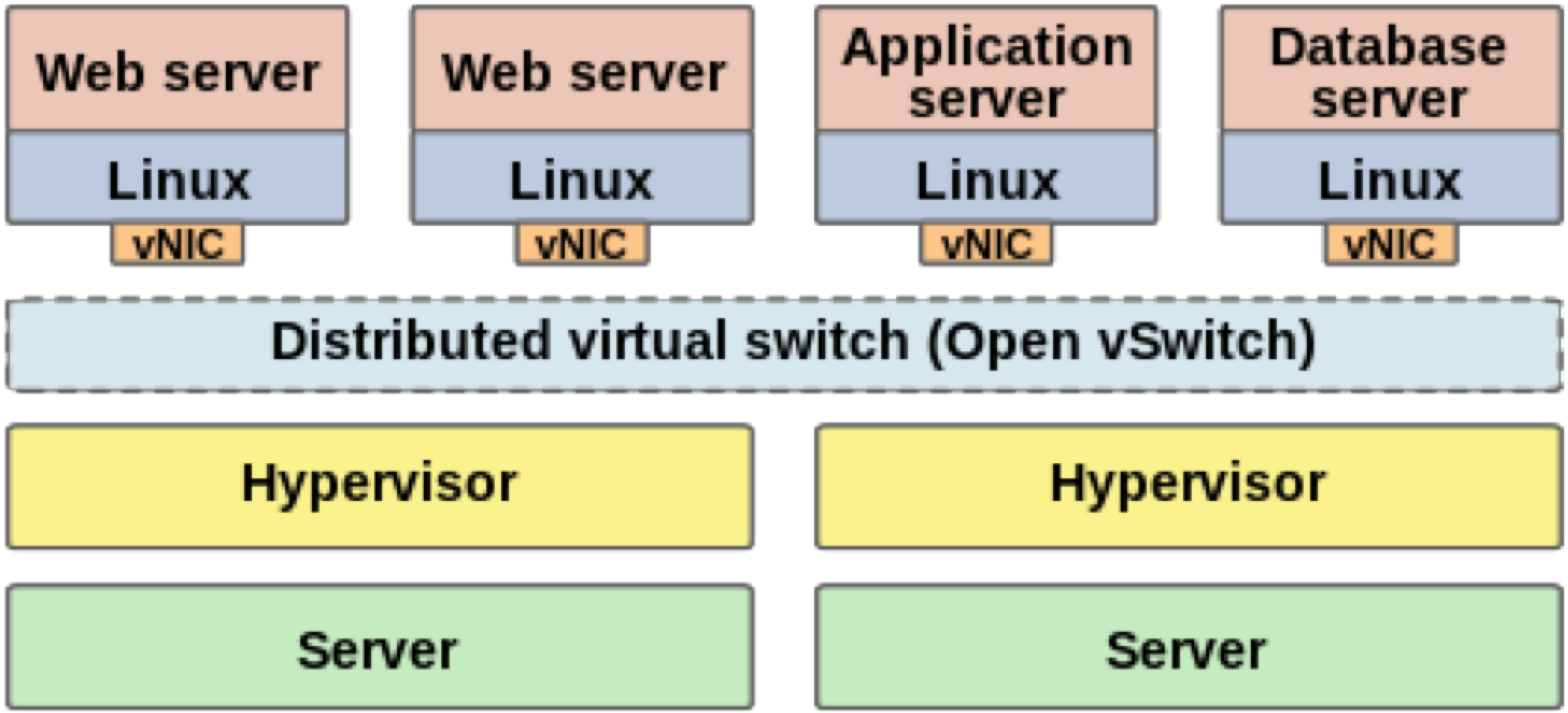


Image Source: IBM

VXLAN & VTEPs

- VXLAN = Virtual Extensible LAN
- VXLAN encapsulates L2 packets (VLAN-style) using UDP, of which are then transported between hypervisors
- Up to 16 million logical networks are supported, unlike traditional VLAN deployments

VXLAN & VTEPs

- VXLAN usually relies on multicast to discover other endpoints part of the same system (hypervisors); however, unicast is also possible when the control plane is present
    - This reduces complexity and prevents the need for creating and managing multicast groups
- A VTEP is the software endpoint installed on the hypervisor that delegates between outbound packets and the VXLAN overlay networks

Logical Switching: VXLAN & VTEPs

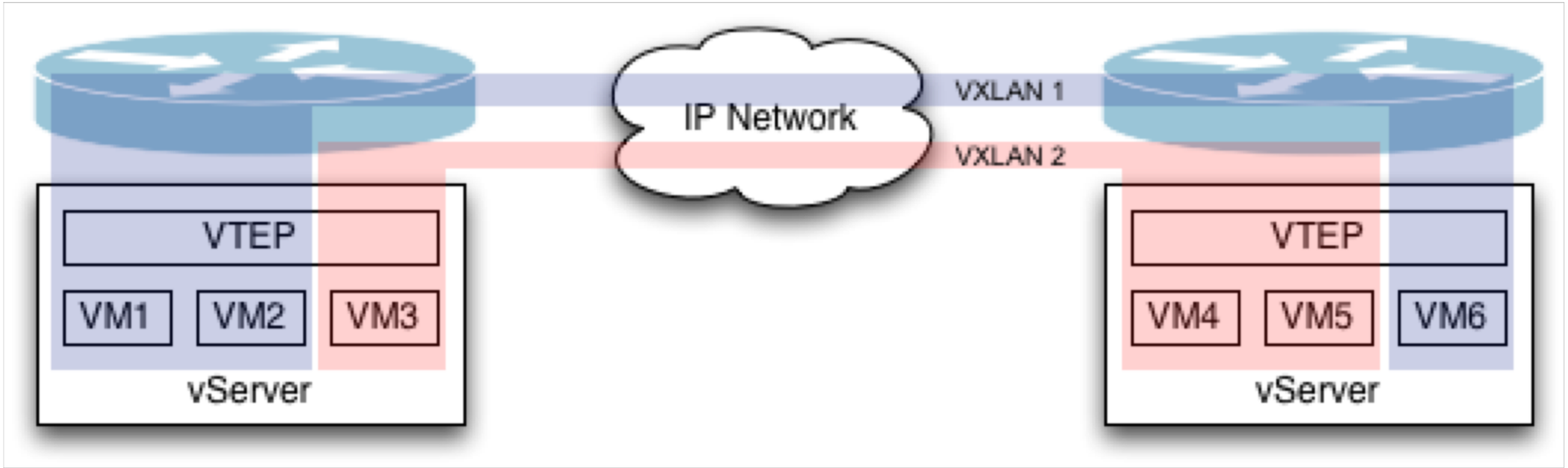


Image Source: Geek Hold

Logical Switching: VXLAN & VTEPs

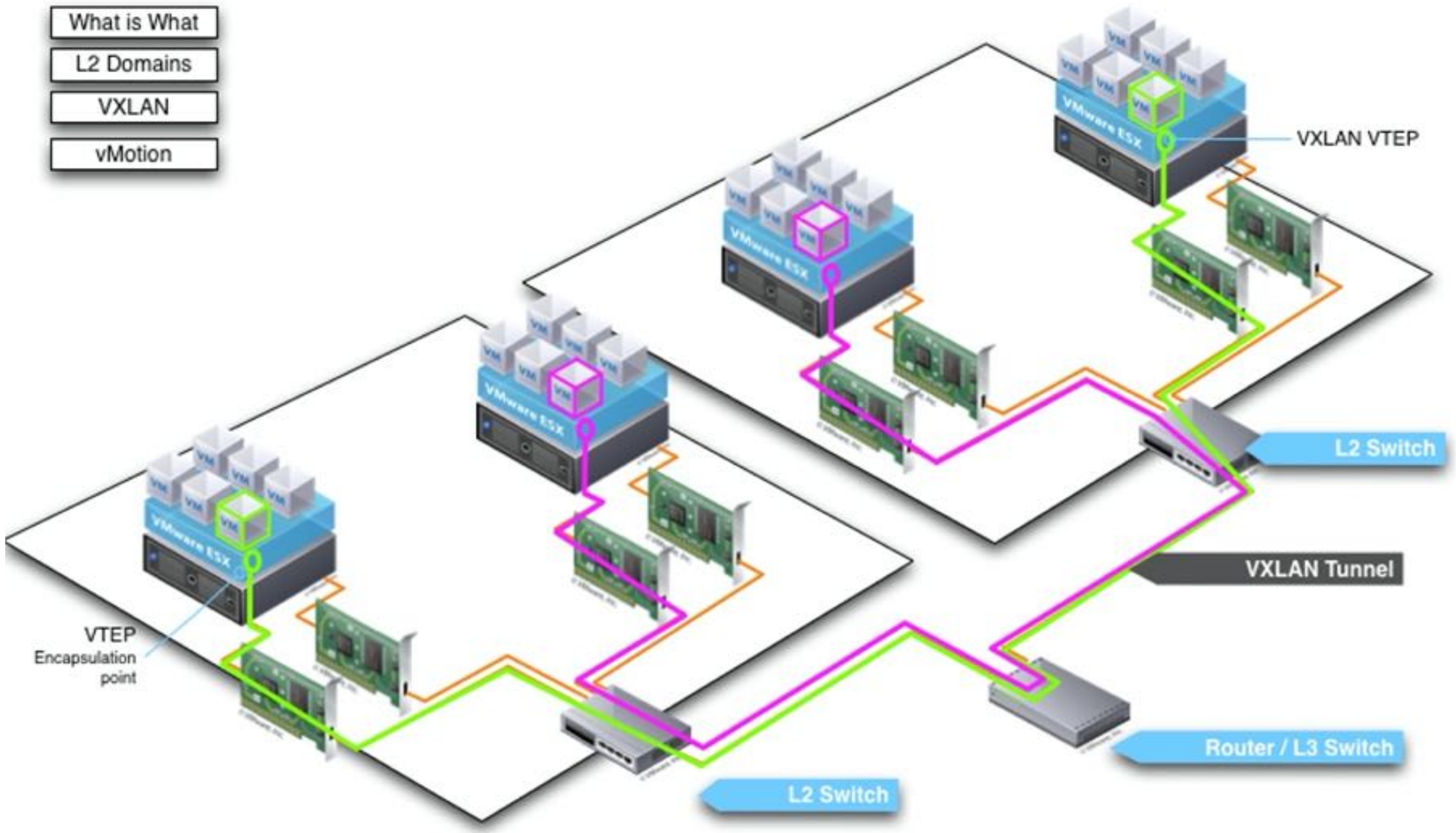


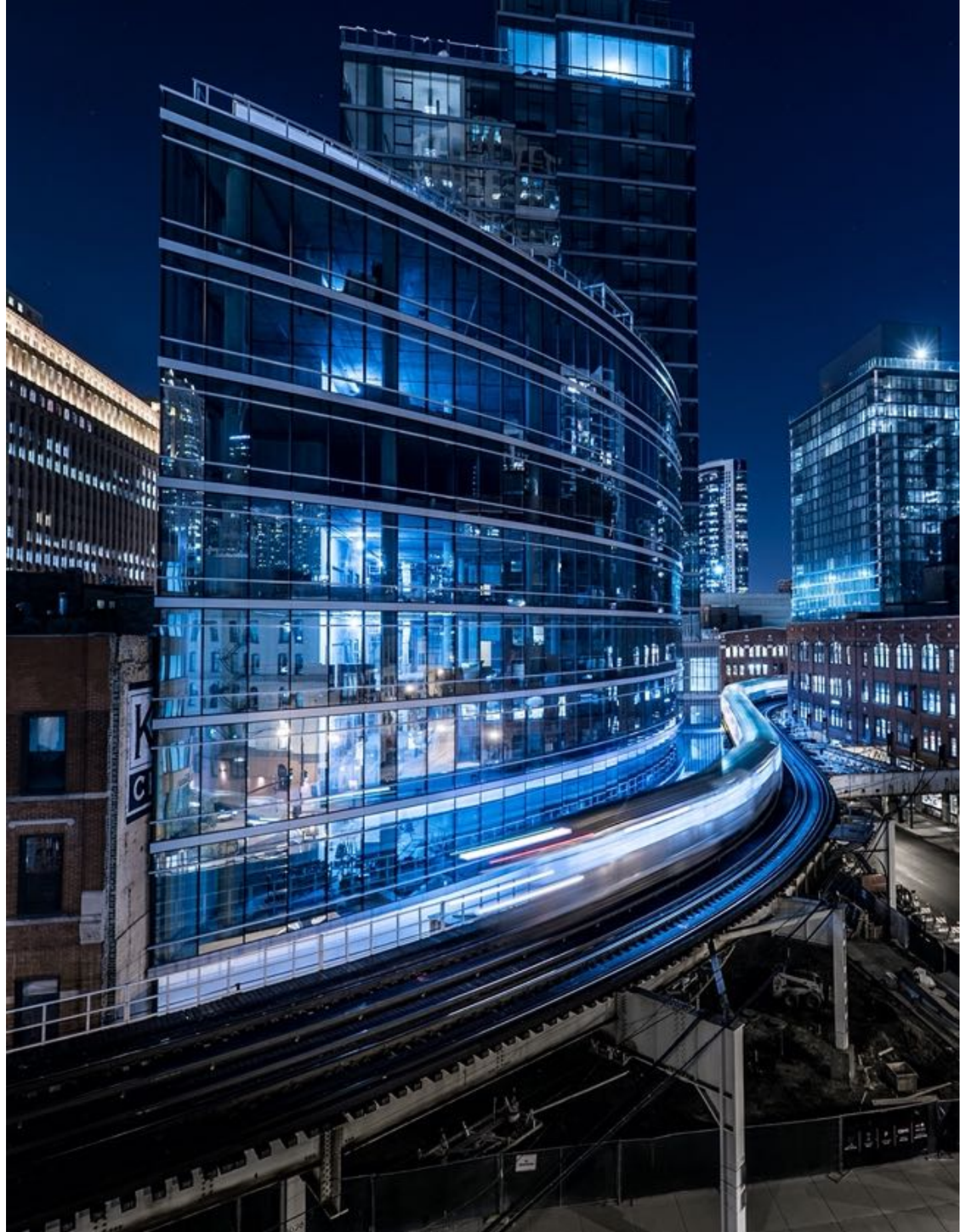
Image Source: Geek Hold

Traditional Routing

- Traditional routing downsides:
  - Scaling and managing routing topologies can be difficult as the number of tenants increases
  - Managing connectivity between VMs in different routing domains can be difficult or inefficient (e.g. two VMs on the same hypervisor on different subnets communicating through an external router)

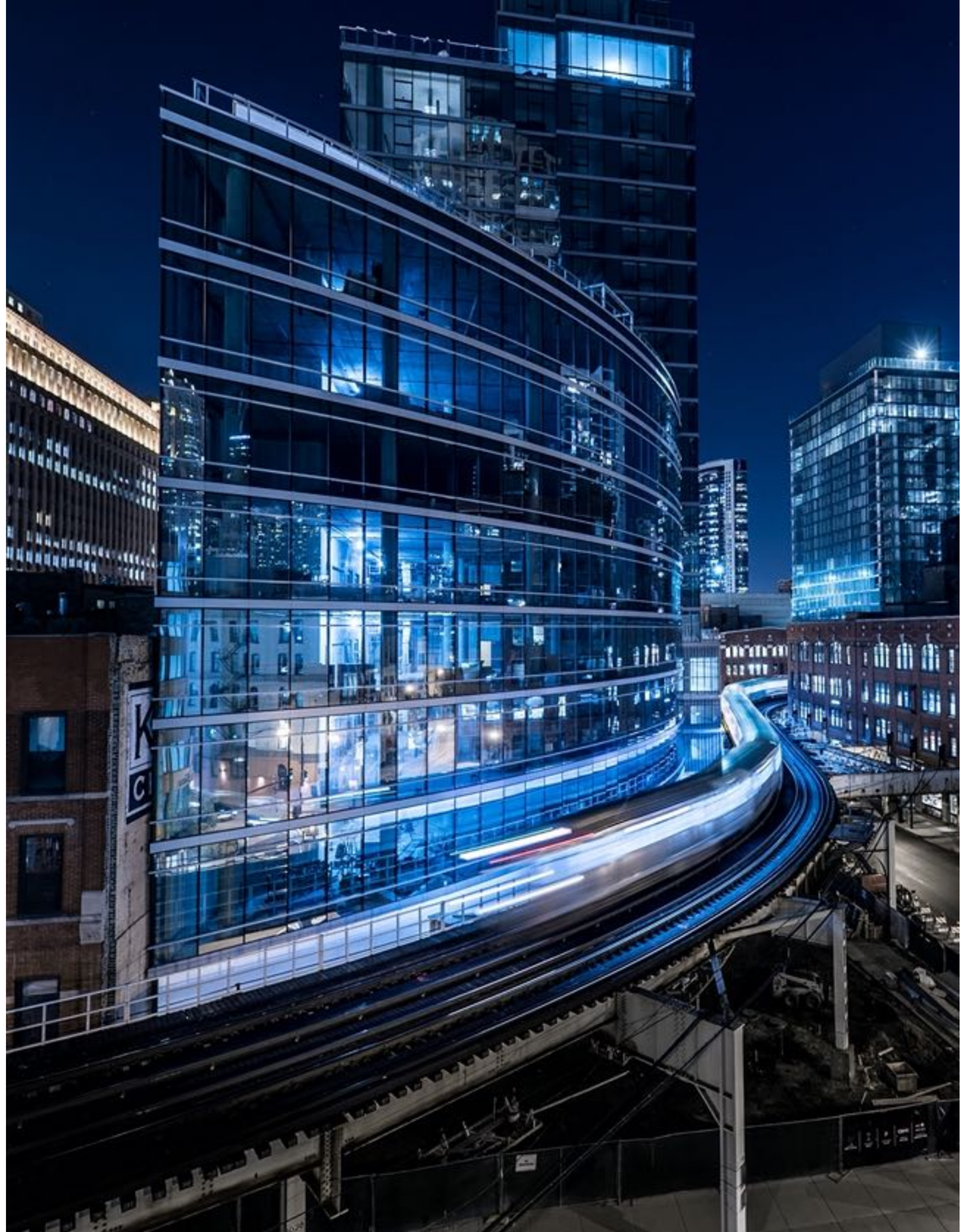
Logical Routing

- Logical routing benefits:
  - Routing overlays are distributed between all hypervisors, allowing localized routing if required (more efficient)
  - Networks are deployed, destroyed, or managed instantly using a standardized API
  - Logical routers (and security controls!) can be provided to each tenant without intervention from network support staff

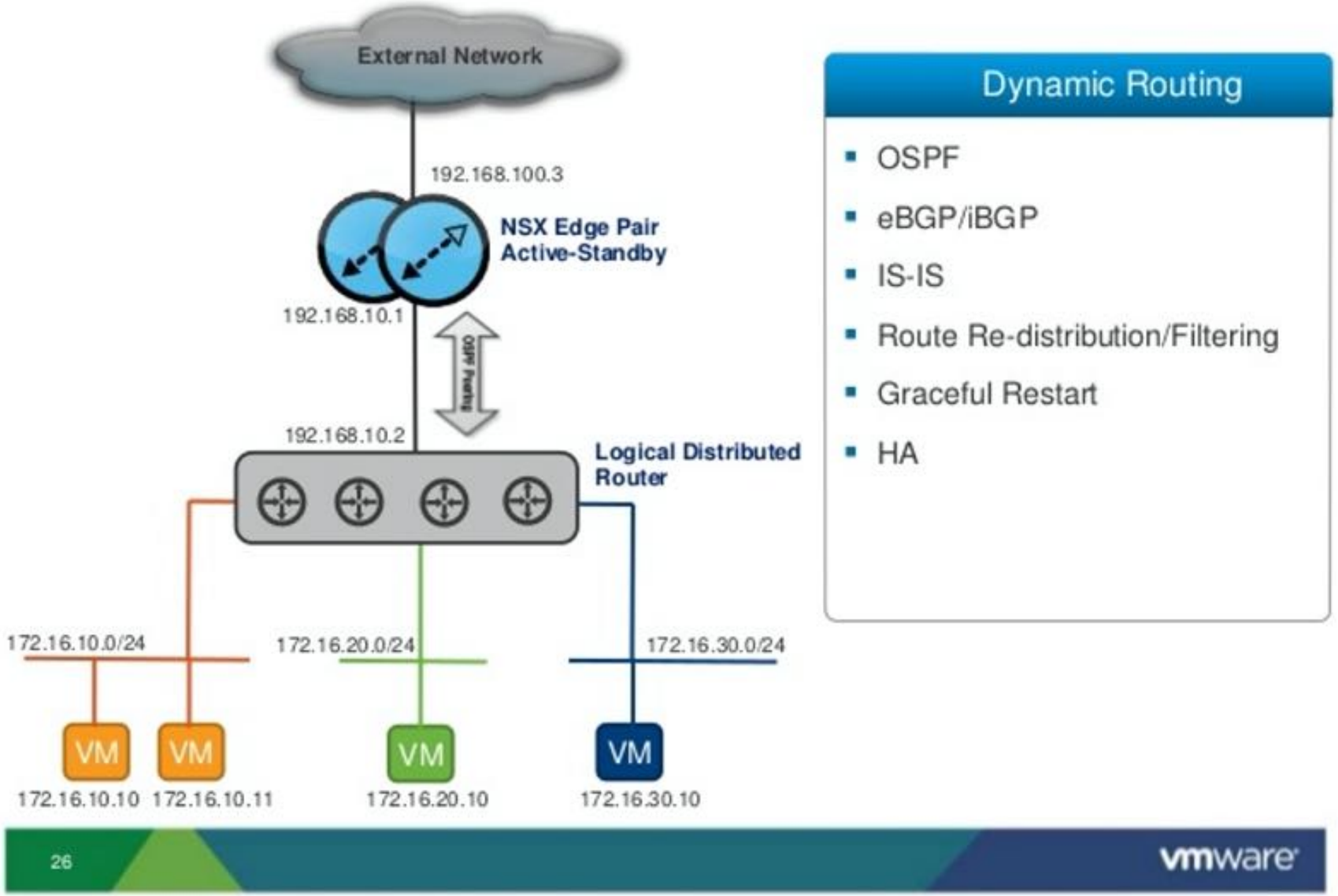


Logical Routing

- Distributed logical routing happens between hypervisors using software from the data plane
  - Connectivity occurs using VXLAN and is managed using the controller
  - The controller provides distributed routing topologies to all hypervisors
- Traffic destined for physical hosts or ingress/egress traffic will be processed using routing VMs or hardware VTEPs
  - More on this shortly



Logical Routing with VMware NSX



Dynamic Routing

- OSPF
- eBGP/iBGP
- IS-IS
- Route Re-distribution/Filtering
- Graceful Restart
- HA

Image Source: VMware

Traditional Firewalling

- Traditional firewall design downsides:
  - Firewall rules and deployment are centralized, both in hardware and software
  - Rules are based on existing known IP subnets/addresses, of which may be unknown in environments with a large number of tenants
  - Appliance traffic requirements are high and expensive to deploy (think 40Gbps!)
  - Visibility with encapsulated traffic is low if the appliance doesn't known an overlay network is present (i.e. it thinks the encapsulated traffic is L7)

Logical Firewalling

- Logical firewall design benefits:
  - Micro-segmentation of security domains occurs at the hypervisor level since security groups are distributed
  - Security groups are configurable by the tenant and administrators via a standardized API, regardless of the underlying infrastructure
  - Security rules can be applied to virtual objects (i.e. specific VMs that can communicate)
  - Encapsulated traffic is fully visible to the software enforcing the policies
  - Network bandwidth requirements are lower since enforcement happens at the hypervisor level (for the most part)
  - Security groups can span multiple DCs!

Logical Firewalling with VMware NSX

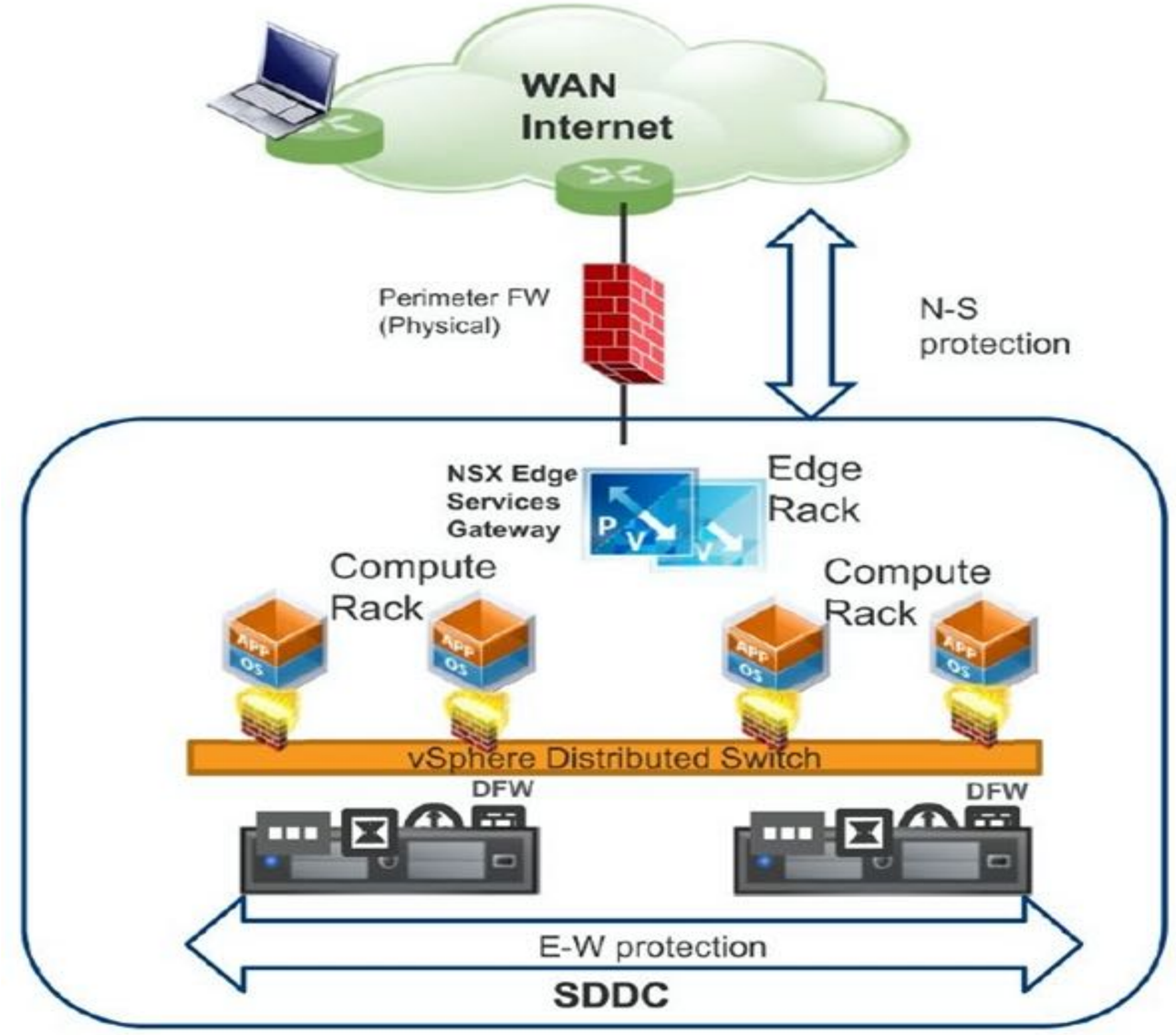


Image Source: VMware

Traditional vs. Logical Load Balancing

- Traditional load balancing downsides:
  - Mobility of virtualized applications is limited due to load balancer placement
  - Configuration and management complexity is high in multi-tenant environments

- Logical load balancing benefits:
  - Deployed instantly and managed using a standardized API across multiple hypervisors
  - Multiple logical load balancers can exists for different applications, easing management
  - Layer 7 load balancing is more effective and efficient when provided to specific applications (e.g. SSL termination)

Bridging the Gap

Question:

Overlay networks and distributed networks do not innately provide connectivity northbound (egress), so how do we provide this connectivity?

Physical & Logical Bridging

- Distributed virtual routers can provide upstream connections through the controller nodes (single nodes could be bottlenecks, clustered nodes use excess resources)

- Hardware VTEP devices can join VXLAN multicast or unicast domains to provide gateway services (i.e. a physical router or switch)

- Virtual routers (VyOS, Vyatta, Cisco CSR, etc.) can be deployed on the hypervisor(s), bridging virtual NICs and physical NICs
- This is often done in smaller environments