Kubernetes Container Networking with NSX-T Data Center Deep Dive

Yasen Simeonov, VMware, Inc.
Disclaimer

This presentation may contain product features or functionality that are currently under development.

This overview of new technology represents no commitment from VMware to deliver these features in any generally available product.

Features are subject to change, and must not be included in contracts, purchase orders, or sales agreements of any kind.

Technical feasibility and market demand will affect final delivery.

Pricing and packaging for any new features/functionality/technology discussed or presented, have not been determined.
Agenda

NSX-T Intro
Quick level set on NSX-T

Kubernetes Overview
Technical overview of Kubernetes, nomenclature & networking details

NSX-T & Kubernetes
Details of the NSX-T integration with Kubernetes

Demo
Seeing is believing
NSX-T Data Center Intro
Quick level set on NSX-T Data Center
The Virtual Cloud Network
Connect and protect your business
Virtual Cloud Networking

Connect & Protect any workload across any environment
VMware NSX Portfolio

The foundation of the Virtual Cloud Network

### Networking and Security Management and Automation

<table>
<thead>
<tr>
<th>Cloud-Based Management</th>
<th>Workflow Automation</th>
<th>Blueprints / Templates</th>
<th>Insights / Discovery</th>
<th>Visibility</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Network Insight</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Network discovery and insights</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>vRealize Automation</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>End-to-end workload automation</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Network and Security Virtualization

<table>
<thead>
<tr>
<th>Security</th>
<th>Integration</th>
<th>Extensibility</th>
<th>Automation</th>
<th>Elasticity</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>NSX Data Center</strong></td>
<td>Networking and security for data center workloads</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>NSX Cloud</strong></td>
<td>Networking and security for Public Cloud workloads</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>AppDefense</strong></td>
<td>Modern application security</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>NSX SD-WAN by VeloCloud</strong></td>
<td>WAN connectivity services</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>NSX Hybrid Connect</strong></td>
<td>Data center and cloud workload migration</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
NSX-T Data Center Architecture and Components

Cloud Consumption

Management Plane

Central Control Cluster (CCP)

Local Control Plane (LCP)

Control Plane

Data Plane

ESXi (+ kernel modules)

KVM (+ kernel modules)

Support for endpoint heterogeneity

Highly available and scalable
Built for consumption by developers

Improved performance and resiliency

OpenStack, k8s or Custom

VMworld 2018 Content: Not for publication or distribution
Data Plane

**Improved performance and resiliency**

Designed for multi-tenancy and scale

New distributed edge architecture with increased performance with DPDK

Geneve Tunnel

TEP: Overlay Tunnel End Point (with its own IP address)

Overlay Transport Zone

Next gen overlay maintaining performance with increased flexibility
Kubernetes Overview

Technical overview of Kubernetes, nomenclature & networking details
What is Kubernetes?

Kubernetes is an open-source platform for automating deployment, scaling, and operations of application containers across clusters of hosts, providing container-centric infrastructure.
Kubernetes Components

K8s Cluster Consists of Master(s) and Nodes

**K8s Master Components**
- API Server
- Scheduler
- Controller Manager
- Dashboard

**K8s Node Components**
- Kubelet
- Kube-Proxy
- Containers Runtime (Docker or Rocket)
A Pod is a group of one or more containers that shares an IP address and a Data Volume.
**Kubernetes Namespace**

Namespaces are a way to divide cluster resources amongst users and groups. They can be thought of as Tenants. They are a way to provide Resources Quotas, RBAC, Networking Multitenancy, and Name uniqueness.

### Namespace: foo
Base URI: /api/v1/namespaces/foo

- **redis-master Pod:**
  /api/v1/namespaces/foo/pods/redis-master

- **redis service:**
  /api/v1/namespaces/foo/services/redis-master

### Namespace: bar
Base URI: /api/v1/namespaces/bar

- **redis-master Pod:**
  /api/v1/namespaces/bar/pods/redis-master

- **redis service:**
  /api/v1/namespaces/bar/services/redis-master
Kubernetes Service

A Kubernetes Service defines a logical set of Pods, selected with matching labels

Serves multiple functions:

- Service Discovery / DNS
- East/West load balancing in the Cluster (Type: ClusterIP)
- External load balancing for L4 TCP/UDP (Type: LoadBalancer)
- External access to the service through the nodes IPs (Type: NodePort)
Kubernetes Ingress

A Kubernetes Ingress Object is a L7 LoadBalancing rule that binds a hostname and url to a Service

The LoadBalancer Datapath can be implemented as an external Load Balancer or as a K8s Pod
Kubernetes Networking Topologies

Non-multitenant routed topology

Every Node is an IP Router and responsible for its Pod Subnet

Subnets are associated with Nodes, not Tenants

Physical Network Configuration is required

```
Node
int eth0 10.240.0.3
int cbr0
net.ipv4.ip_forward=1
10.24.1.1/24
10.24.1.2
10.24.1.3
10.24.1.4

Node
int eth0 10.240.0.4
int cbr0
net.ipv4.ip_forward=1
10.24.2.1/24
10.24.2.2
10.24.2.3
10.24.2.4

ip route 10.24.1.0/24 10.240.0.3
ip route 10.24.2.0/24 10.240.0.4
```
Kubernetes Networking Topologies

Node-to-Node overlay topology

Overlays are typically used to avoid Physical Network Configuration

Subnets are still associated with Nodes, not Tenants

External outbound connectivity needs SNAT using the Nodes IP

External inbound connectivity needs Node Port or Ingress in Host Network Mode
NSX-T & Kubernetes
Details of the NSX-T integration with Kubernetes
Key Design Goals of the NSX-T Data Center

Kubernetes Integration

- Don't stand in the way of the developer!
- Provide solutions to map the Kubernetes constructs to enterprise networking constructs
- Secure Containers, VMs and any other endpoints with overarching Firewall Policies
- Provide visibility & troubleshooting tools to ease the container adoption in the enterprise
Kubernetes NSX Topology

Dynamic per Namespace Topology

Dynamically network topology per K8s namespace

K8s Nodes are not doing IP routing

Every Pod has its own logical port on a NSX logical switch, and is supporting all features a VM interface supports

Every Pod has Dynamic Firewall rules applied on its logical Interface
NSX Container Plugin (NCP)

NCP is a software component provided by VMware in form of a container image, e.g. to be run as a K8s Pod.

NCP is build in a modular way, so that individual adapters can be added for different CaaS and PaaS systems at some point.
With most networking technologies in K8s like Flannel, OpenShift OVS Networking, Calico, etc. the source IP of the traffic can't be mapped to the tenancy. This is the biggest hurdle today to get K8s integrated in enterprise IT environments.

Tenancy / Topology Mapping
The open source way

Did the traffic come from 'foo' or 'bar'? SNAT to Node IP

SNAT to Node IP

Physical or virtual Router

172.16.1.1/24

Tenant: foo

Physical or virtual Router

172.16.11/24

Tenant: bar

Tenant: foo

Tenant: bar

Tenant: foo

Tenant: bar
Tenancy / Topology Mapping

Persistent IPs for K8s Namespaces

With NSX-T each Tenant (Kubernetes Namespace) either gets its own SNAT IP (NAT Mode), or is directly identifiable by its source subnet (No NAT Mode)

In NAT Mode, the external DC Firewall and the DB can distinguish tenant ‘foo’ and tenant ‘bar’ using the source SNAT IP that is allocated to a specific Tenant.

In No-NAT Mode, the external DC Firewall and the DB can distinguish tenant ‘foo’ and tenant ‘bar’ using the source IP Subnet that is allocated to a specific Tenant.
Persistent SNAT IP per K8s Service
Specifying the source IP Kubernetes Workloads using the K8s service

Table: Feature
<table>
<thead>
<tr>
<th>Feature</th>
</tr>
</thead>
<tbody>
<tr>
<td>Persistent SNAT IP per K8s Service</td>
</tr>
</tbody>
</table>

Specifying the source IP Kubernetes Workloads using the K8s service

Feature
With this feature a set of Kubernetes Workloads (Pods) can be assigned to use a specific IP or group of SNAT IPs to source their traffic from

Before this feature we only assigned a SNAT IP to a Kubernetes Namespace

Benefits
Infrastructure Teams can pre-create Firewall rules in existing DC physical Firewalls to allow traffic from specific workloads in K8s

The K8s user / DevOps can deploy applications that are easily identifiable in the physical network
Central Visibility

With most other networking technologies in K8s and PCF like Flannel, OpenShift OVS Networking, PCF Silk, Calico, etc. there is no centralized control plane. So, there’s no counters, troubleshooting tools, 'span ports', Firewall Rules Overview, etc.
Central Visibility

With NSX-T you are gain deep visibility into the container networks, and you can use the same troubleshooting tools we created for VM based workloads.
Kubernetes Metadata / NSX Logical Port Mapping

Metadata within Kubernetes like Namespace, Pod names, **Labels** all get copied to the NSX Logical Port as Port Tags.

```
kubectl get pod nsx-demo-rc-c7x65 -o yaml
```

```
apiVersion: v1
kind: Pod
metadata:
  creationTimestamp: 2018-07-25T12:05:56Z
  generateName: nsx-demo-rc-
  labels:
    app: nsx-demo
    name: nsx-demo-rc-c7x65
  namespace: nsx-ujo
```
Pre-Created Security Groups / Firewall rules (admin rules)

NSX can be configured to collect ports and switches in dynamic security groups based on Tags (Kubernetes Metadata) and apply Firewall rules on them.

### NSX-Demo-Pods

<table>
<thead>
<tr>
<th>Overview</th>
<th>Membership Criteria</th>
<th>Members</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Membership Criteria</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1. Logical Port</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Tag Equals nsx-demo Scope Equals app</strong></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Match on Port Tags**

- Matching Pods are part of the Group
- Groups are used in Firewall sections as src and dst

![NSX-Demo-Pods](image)

![Matching Pods are part of the Group](image)

![Groups are used in Firewall sections as src and dst](image)
Unified Policy for K8s, PCF & VMs

Both K8s and PCF have 'built-in' micro segmentation policy languages (network policy), and there’s a broad set of products and open source projects implementing micro segmentation inside of K8s or PCF. However there is no technology other than NSX-T today that allows you to define policies across K8s, PCF and VM based workloads using Metadata from each system.
Support of Kubernetes Network Policy

Besides supporting admin pre-defined rules, NCP is also translating Kubernetes NetworkPolicy Objects to NSX security groups and Firewall rules.

Admin pre-defined rules can be used concurrently in NSX, admin rules are put in sections before or after K8s network policy rules.

```yaml
code
apiVersion: networking.k8s.io/v1
kind: NetworkPolicy
metadata:
  name: nsx-demo-policy
spec:
  podSelector: {}  
  policyTypes:
    - Ingress
  ingress:
    - from:
      - ipBlock:
        cidr: 100.64.160.11/32
      ports:
        - port: 80
          protocol: TCP
```

```yaml
code
apiVersion: networking.k8s.io/v1
kind: NetworkPolicy
metadata:
  name: default-deny
spec:
  podSelector: {}  
  policyTypes:
    - Ingress
```
Built-in Load Balancing

We have built-in support for Ingress (L7 HTTP/HTTPS) and Svc Type LB (L4 TCP/UDP) in the NSX-K8s integration. Most other K8s networking choices don't support Svc Type LB (L4), and you need an additional technology like NGINX from Ingress (L7).
1. NCP watches for Svc events in Kubernetes

2. User creates a new Svc of Type LoadBalancer

3. The Kubernetes API server notifies NCP of the new Svc

4. NCP creates a new Virtual Server with a unique IP and a Server Pool with the Pods as targets
K8s / NSX Workflows

Ingress

1. NCP watches for Ingress events in Kubernetes
2. User creates a new Ingress rule
3. The Kubernetes API server notifies NCP of the new Ingress rule
4. NCP creates a new forwarding rule sending a specific HTTP/S hostname and path to a specific Server Pool
# NSX-T Data Center Timeline

## Kubernetes, OpenShift and PKS

<table>
<thead>
<tr>
<th>2017</th>
<th>2018</th>
</tr>
</thead>
<tbody>
<tr>
<td>September</td>
<td>October</td>
</tr>
</tbody>
</table>

### NSX-T 2.0:
Support for 'Do It Yourself' K8s & OpenShift

Core value add:
- Mapping of K8s Namespaces to Network Topology & source IP Addresses
- NAT & No-NAT modes per Namespace
- Network Policy (Firewall) across K8s and VM workloads
- Support for K8s Network Policy
- Logical Network Port per K8s workload (Pod) for visibility and troubleshooting

### NSX-T 2.1:
Support for PKS 0.8 and PKS 1.0
Support for K8s Ingress and Svc Type LB with Platform LB

Core value add:
- One of the only SDN solution in the market that includes LB with Ingress and Svc Type LB for K8s
- PKS / OPS MGR Integration
- Gives PKS support for Network Policy
NSX-T Timeline

PCF 2.0

2018

January | February | March | April | May | June | July

NSX-T 2.1
Support for PCF 2.0 -> PAS

Core value add:
- Allows mapping of CF tenancy (Orgs) to Network Topology & source IP Addresses
- Network Policy (Firewall) support across PKS, PCF and VM workloads
- Only solution that allows for direct, no_NAT communication from CF Apps to backend services
- Logical Network Port per CF workload (AI) for visibility and troubleshooting

NSX-T 2.2
Operational Enhancement & Additional LB features

Core value add:
- Persistent SNAT IP for Kubernetes Services and CF Apps
- TLS/SSL Offload support for Kubernetes Ingress
- OpenShift 'router' support for HTTP and HTTPS (feature parity with K8s Ingress)
- URL rewrite support for K8s Ingress
- Various install & operational improvements
NSX-T Data Center Values for Containers

Enterprise-class Networking

Advanced Security

Enhanced Operations

Unified VM-to-Container Networking

Micro-segmentation

Full Network Visibility

Enterprise Support

Features
Where to Get Started

Engage and Learn

Join the NSX VMUG Community
vmug.com/nsx
Connect with your Peers
communities.vmware.com
Embrace the NSX Mindset
nsxmindset.com
Find NSX Resources
vmware.com/go/networking
Read the Network Virtualization Blog
blogs.vmware.com/networkvirtualization

Experience

Attend the Networking and Security Sessions
Showcases, breakouts, quick talks & group discussions.

Visit the VMware Booth
Product overviews, use-case demos

Visit Technical Partner Booths
Integration demos – Infrastructure, security, operations, visibility, and more

Meet the Experts
Join our experts in an intimate roundtable discussion

Try

Free Hands-on Labs
labs.hol.vmware.com
Virtual Cloud Network Guided Demo
vcndemo.com

Take

VMware Education – Training and Certification
vmware.com/go/nsxtraining
Free NSX Training on Coursera
vmware.com/go/coursera
Accelerate your success with VMUG

VMware User Group (VMUG) is an independent, global, customer-led organization, created to maximize members’ use of VMware and partner solutions through knowledge sharing, training, collaboration, and events.

Free Membership Includes:

- **GLOBAL NETWORK**: A network of 125,000+ members worldwide
- **ENGAGED COMMUNITIES**: Connect with 200+ local groups of peers and experts
- **EXCITING TECH EVENTS**: Participate in 400+ local events, 40+ Usercons and Product Roadshows
- **VIRTUAL OFFERINGS**: Learn from experts and industry pros right from your desk

Become a VMUG member today

Ready to have an entire crew of experts available to help you learn, grow, and advance your career?

Join VMUG for FREE today at vmug.com.

Follow us:
- [VMwareUserGroup](https://www.vmware.com)
- [@MyVMUG](https://twitter.com/MyVMUG)
PLEASE FILL OUT YOUR SURVEY.

Take a survey and enter a drawing for a VMware company store gift card.
POSSIBLE BEGINS WITH YOU

THANK YOU!

#vmworld  #NET1677BU