WHITE PAPER – **JUNE 2018**

CONTAINERS ON VIRTUAL MACHINES OR BARE METAL?

Deploying and Securely Managing Containerized Applications at Scale

vmware®

Table of Contents

"Although containers are sometimes thought of as the next phase of virtualization, surpassing hardware virtualization, the reality for most organizations is less about revolution than evolution. Containers and hardware virtualization not only can, but very frequently do, coexist well and actually enhance each other's capabilities. VMs provide many benefits, such as strong isolation, OS automation, and a wide and deep ecosystem of solutions. Organizations do not need to make a choice between containers and VMs. Instead, organizations can continue to use VMs to deploy, partition, and manage their hardware, while using containers to package their apps and utilize each VM more efficiently."

APPLICATION CONTAINER SECURITY GUIDE, NIST SPECIAL PUBLICATION 800-190

Introduction

Containers, some say, render hardware virtualization unnecessary: Why do you need virtual machines (VMs) now that you can run containers on physical hardware?

Running an application in production comes with an established set of operational requirements: security, compliance, performance, resource management, scalability, availability, data persistence, networking, and monitoring. Containerized applications are no different. They do, however, carry an additional requirement: orchestration.

You can, at great risk and expense, build a custom stack on physical hardware to try to fulfill your containerized apps' requirements, or you can use proven, cost-effective, low-risk virtualization solutions as the underlying infrastructure for containers and their orchestration system.

But there's more: Combining containers and VMs taps the benefits of each technology, creating an organized whole that is greater than the sum of its parts—which is one reason why the major cloud providers, such as Google and Amazon, use VMs to run containers. Virtual machines let you securely and efficiently run containerized applications in production on software-defined infrastructure that you can easily manage, monitor, scale, and optimize. Containers, meanwhile, empower you to make developers more agile, applications more portable, and deployments more automatable. The combination of the two streamlines the development, deployment, and management of enterprise applications.

This paper meets objections to running containers on VMs with technical explanations and evidence-based responses. It argues that combining containers and VMs establishes the perfect catalyst for reliably and robustly deploying and operating containerized applications at scale. VMware® Pivotal Container Service, which uses Kubernetes to orchestrate containers on virtual machines in a VMware software-defined data center, stands at the center of this combination.

Security

In September 2017, the National Institute of Standards and Technology (NIST) published its Application Container Security Guide, also known as NIST Special Publication 800-190. It explains the security concerns with containers and recommends how to address them. The guide exposes several fundamental areas of concern with containers:

- Degree of isolation
- Operating system management and configuration
- Orchestration systems without adequate protection

Containers Alone Are Inadequate Security Boundaries

Containers are not miniature VMs, and containers do not establish security boundaries as VMs do. An important implication of the Application Container Security Guide is to run containerized applications on virtual machines: Containers, the guide says, "do not offer as clear and concrete of a security boundary as a VM. Because containers share the same

"Docker containers pair well with virtualization technologies by protecting the virtual machine itself and providing defense in-depth for the host."

DOCKER SECURITY WHITE PAPER

kernel and can be run with varying capabilities and privileges on a host, the degree of segmentation between them is far less than that provided to VMs by a hypervisor." [1]

Deploying containers with VMs encases an application with two layers of isolation, an approach that is well-suited to cloud-style environments with multitenancy and multiple workloads. "Docker containers pair well with virtualization technologies by protecting the virtual machine itself and providing defense in-depth for the host," a Docker security white paper says.[2]

To run containers that properly isolate tenants on physical Linux hosts, you would need to run different tenants on separate physical machines. The likely outcome is either low resource utilization stemming from fragmentation or overly high utilization leading to long wait times for using new hardware. The major cloud providers, such as Google and Amazon Web Services (AWS), isolate the container workloads of tenants by using separate VMs. Because containers are inadequate security boundaries, only highly trusted code should be run in containers on the same VM or physical host.

The same holds true for pods in Kubernetes. "Ultimately, in the case of applications running in both VMs and containers, the VM provides the final security barrier. Just like you wouldn't run programs with mixed security levels on the same VM, you shouldn't run pods with mixed security levels on the same node due to the lack of guaranteed security boundaries between pods," writes Jianing Guo on the Google Cloud Platform Blog.[3]

Risks of Misconfiguration on a Physical Host

Containers or the operating system of a physical host can easily be misconfigured, increasing the attack surface and the level of risk, the NIST Application Container Security Guide says. "Carelessly configured environments can result in containers having the ability to interact with each other and the host far more easily and directly than multiple VMs on the same host."

In contrast, the abstraction, automation, and isolation of an operating system running on a VM in a hypervisor reduces the attack surface and decreases the risk of a security breach.

Securing the Orchestration System

Another concern of the Application Container Security Guide is recommending countermeasures to secure the orchestration system managing containers. The suggested countermeasures in the NIST guide include the following:

- The use of enterprise-grade authentication services using strong credentials and directory services
- Granular access control for administrative actions based on hosts, containers, and images
- Isolating containers to separate hosts based on the sensitivity level of the applications running in them

1 NIST Special Publication 800-190, Application Container Security Guide, by Murugiah Souppaya, Computer Security Division Information Technology Laboratory; John Morello, Twistlock, Baton Rouge, Louisiana; Karen Scarfone, Scarfone Cybersecurity, Clifton, Virginia. September 2017, https://doi.org/10.6028/NIST.SP.800-190

2 Introduction to Container Security, Docker white paper, Docker.com.

3 "Demystifying container vs VM-based security: Security in plaintext," The Google Cloud Platform Blog, by Jianing Guo, August 9, 2017, https://cloudplatform.googleblog.com/2017/08/demystifying-container-vs-VM-based-security-security-in-plaintext.html.

Another NIST document, Security Assurance Requirements for Linux Application Container Deployments, sets forth security requirements and countermeasures to help meet the recommendations of the Application Container Security Guide when containerized applications are deployed in production environments. The orchestration system or its components and tools should have the following capabilities:

- Logging and monitoring of resource consumption of containers to ensure availability of critical resources
- The orchestration system must work with many container hosts, not just one, to be able to provide a global summary of resource usage for all running containers

Running containers on physical hardware and managing the containers with an orchestration system would require you to connect each physical machine to an authentication and access control system.

To isolate containers by sensitivity level, you would have to use an inefficient number of physical machines. As a result, resource utilization would suffer while management overhead increased—a situation made worse by an NIST requirement to use many types of container hosts.[1]

Taking Advantage of Advanced Trends

Running containers on VMs lets you take advantage of security innovations in virtualization technology. AMD SEV-ES provides an example. Secure Encrypted Virtualization (SEV) technology integrates memory encryption with AMD-V virtualization to support encrypted VMs, which are ideal for multitenant environments.

SEV with Encrypted State (SEV-ES) builds upon SEV to provide an even smaller attack surface and additional protection for a guest VM from the hypervisor even if the hypervisor is compromised. SEV-ES blocks attacks by encrypting and protecting all CPU register contents when a VM stops running to prevent the leakage of information in CPU registers to the hypervisor. SEV-ES can detect and prevent malicious modifications to the CPU register state.[4]

Securing Microservices with VMs

Microservices add another dimension to container security. According to a Docker white paper on security, "Deploying Docker containers in conjunction with VMs allows an entire group of services to be isolated from each other and then grouped inside of a virtual machine host."[2]

Availability

Containerized applications that are perfectly architected with microservices can rely on a container orchestration system like Kubernetes to manage availability. But most containerized applications are not perfectly architected: They may replatformed monolithic applications separated into a few macro components, or they may be partially refactored with some microservices and some remaining n-tier patterns. Such applications continue to rely, at least in part, on the underlying infrastructure for availability. Proven technologies from a VMware software-defined data center—such as VMware vSphere® vMotion®, VMware vSphere High Availability, and VMware vSphere Distributed Resource Scheduler™ (DRS)—are essential to maintaining availability.

4 Protecting VM Register State with SEV-ES, David Kaplan, AMD, February 2017, https://support.amd.com/TechDocs/Protecting%20VM%20Register%20State%20with%20SEV-ES.pdf.

"Applications can benefit from the security and performance isolation provided by the VM, and still take advantage of the provisioning and deployment aspects of containers. This approach is quite popular, and is used to run containers in public clouds where isolation and security are important concerns."

CONTAINERS AND VIRTUAL MACHINES AT SCALE: A COMPARATIVE STUDY

Even a containerized app that is perfectly architected with microservices to handle its own availability can still benefit from software-defined infrastructure. For example, Redis, as an in-memory replicated database, tolerates infrastructure failures: A new Redis instance starts up to replace a failed instance. But as the new instance starts, it replicates data from other Redis nodes until its state is fully restored. The data transfer for the replication, however, comes with a cost: It degrades the overall performance of the Redis cluster. A more efficient approach is to use vMotion to move the original Redis node, avoiding the performance hit.

Resource Management

Kubernetes provides powerful quality-of-service (QoS) mechanisms to share a cluster between teams running different workloads. Running Kubernetes clusters on vSphere complements the QoS mechanisms of Kubernetes, especially when you require strong workload isolation. The advanced scheduling and dynamic resource management of vSphere helps reclaim and share unused resources between teams or across Kubernetes clusters.

The resource management capabilities of vSphere include intelligent initial placement, dynamic rebalancing, resource pools, shares, reservations, limits, and safe overcommitment. All these capabilities empower you to run traditional and containerized workloads on common infrastructure while ensuring optimal performance and preventing interference between workloads.

For Kubernetes clusters running on vSphere, such VMware technologies as vMotion and DRS maximize hardware utilization by dynamically rebalancing clusters without disrupting workloads.

Data Persistence

Although many containerized applications are stateless, the drive to port applications to containers is generating requirements to host stateful apps on containers and to furnish them with host-local, shared-nothing storage for data persistence across hosts.

Managing physical storage devices, however, is a painful, manual process often worsened by application-specific workflows. Adding new SSDs to expand capacity is inefficient. Different technologies and processes across varying infrastructure silos can complicate the situation, and dedicating physical hardware to an individual application does not make economic sense.

A single software-driven model that works for applications, whether containerized or not, radically simplifies storage management, operations, troubleshooting, capacity expansion, and storage operations like backup and disaster recovery. By providing a distributed, shared-nothing storage abstraction, VMware vSAN™ simplifies storage operations and consolidates workloads, both traditional and cloud native, on the same storage infrastructure.

Performance

The CPU scheduler of VMware ESXi™ empowers the hypervisor to provide equivalent or better overall workload performance for containers than Linux systems running on physical hardware.

"Containers promise bare metal performance, but as we have shown, they may suffer from performance interference in multi-tenant scenarios. Containers share the underlying OS kernel, and this contributes to the lack of isolation. Unlike VMs, which have strict resource limits, the containers also allow soft limits, which are helpful in overcommitment scenarios, since they may use underutilized resources allocated to other containers. The lack of isolation and more efficient resource sharing due to soft-limits makes running containers inside VMs a viable architecture."

CONTAINERS AND VIRTUAL MACHINES AT SCALE: A COMPARATIVE STUDY

A comparative study by VMware shows that an enterprise web application can run in Docker containers on vSphere 6.5 with better performance than Docker containers on bare metal, largely because of optimizations in the vSphere CPU scheduler for non-uniform memory access (NUMA) architectures, quashing the belief that running containers on VMs comes with a performance tax.[5] vSphere is better at scheduling VMs on NUMA nodes where their memory resides. Linux, on the other hand, tries to maximize processor utilization, meaning processes may be scheduled on different NUMA nodes from their memory, slowing memory access and degrading performance. A performance analysis of big data workloads on vSphere shows the same results.

Virtualization can offer better performance isolation than running containers in Linux, especially in noisy neighbor situations. The results of an academic comparative study of containers and VMs at scale show that "co-located applications can cause performance interference, and the degree of interference is higher in the case of containers for certain types of workloads."[6] Because of how the Linux kernel works, you can also get cross-container interference from containers sharing the same kernel resources or components. It's a mistake to assume that "the kernel is fully isolating every underlying resource at a container granularity."[7]

Kubernetes clusters also benefit from running on vSphere. Kubernetes on bare metal is unlikely to outperform running Kubernetes on vSphere, which uses advanced scheduling algorithms to optimize all workloads, including those using containers. Companies that run Kubernetes on physical hardware can find it difficult to scale and efficiently operate the infrastructure. Running Kubernetes clusters on vSphere, in contrast to physical hardware, benefits from years of fine-tuning to make it adept at optimizing the performance of large clusters and mixed workloads.

Scalability

Hypervisors were originally developed to address the pain of working with physical hardware, pain that ranges from time-consuming management problems and cash-consuming underutilization to the difficulty of scaling hardware for a developer environment or an application's expanding workload. By optimizing utilization, virtualization lets you reduce physical hardware costs while improving scalability.

If IT operations deploys a bare-metal server with a container runtime to which developers can push their containers, scaling the system is difficult and time consuming: You would have to add another bare-metal server, install a container runtime on it, manually hook the server up to the network, and connect the runtime to a container orchestration engine.

In contrast, with a hypervisor, you can connect a new bare-metal server to the container domain in minutes. The ease of scalability that comes with virtualization is one of the reasons that major public cloud providers use hypervisors to run their container services. For example, when you create a Kubernetes cluster on Google Container Engine or

5 Performance of Enterprise Web Applications in Docker Containers on VMware vSphere 6.5, VMware, September 2017, https://www.vmware.com/content/dam/digitalmarketing/vmware/en/pdf/techpaper/performance/docker-vsphere65-weathervane-perf.pdf.

6"Containers and Virtual Machines at Scale: A Comparative Study," Prateek Sharma, Lucas Chaufournier, Prashant Shenoy, Y.C. Tay; Middleware'16, December 12–16, 2016, http://dx.doi.org/10.1145/2988336.2988337

7"Container isolation gone wrong," by Gianluca Borello, Sysdig, May 22, 2017, https://sysdig.com/blog/container-isolation-gone-wrong/

MICRO-SEGMENTATION FOR CONTAINERS

Micro-segmentation uses network virtualization to divide a data center and its workloads into logical segments, each of which contain a single workload. You can then apply security controls to each segment, restricting an attacker's ability to move to another segment or workload.

Amazon Elastic Container Service, the respective cloud provider fires up one or more VMs. The turn-around time is much faster with VMs than bare metal.

The same formula holds true on a global scale. If you were to build Kubernetes on bare metal in a data center, how would you scale it to 1,000 sites around the world while maintaining security, networking, monitoring, and centralized management?

For large clusters, using Kubernetes on bare metal can also have repercussions for utilization in relation to scale. As of Kubernetes version 1.10, the published supported configurations for clusters in the Kubernetes documentation are as follows:[8]

- No more than 5,000 nodes
- No more than 150,000 total pods
- No more than 300,000 total containers
- No more than 100 pods per node

If each pod has only one container, which is not uncommon, there will be untapped resources on bare metal if any of the containers are low functioning.

With vSphere, however, you can run more than 100 pods per physical host, improving utilization of the underlying hardware.

Networking

Containers rely on three levels of abstraction within networking:

- Underlay network
- Overlay network
- Service mesh

The underlay network connects machines, whether virtual or physical, by using either a traditional hardware-based approach or a combination of hardware and software. The overlay rides on top of the underlay to provide networking, such as IP addresses and ports, for the lifecycle of containers and hosts. The service mesh moves above IP addresses and ports to focus on connecting services for containerized applications.

In this context, how would you securely and efficiently connect a large number of bare-metal container hosts running Linux? If you use a flat L2 network that pushes everything to the overlay, you would have to group containerized applications by sets of physical hosts (because containers alone are inadequate security boundaries).

If the networking for the container is not isolated and not tied to the container's lifecycle, however, an attacking application could gain connectivity to all the other physical hosts in the environment. To ensure network security, the environment would require hardware-level network isolation with VLANs, east-west firewalls, or other techniques—all of which call for manual, time-consuming management that is not tied to the containerized application's lifecycle. And when the lifecycle changes, more manual, error-prone network changes would be required.

VMware NSX® Data Center also implements a single underlay network on VMs to provide end-to-end connectivity and management for both containers and traditional applications. A single underlay network comes with several advantages:

8 "Building Large Clusters," Kubernetes, https://kubernetes.io/docs/admin/cluster-large/

- Connect containerized applications to traditional, non-containerized components like databases with ease.
- Radically simplify network management with centralized policies and advanced security, such as micro-segmentation.[9]
- Select the overlay network and the service mesh that works best for your containerized application.

NSX Data Center integrates with the container network interface to furnish an overlay network. When a new containerized app is deployed, NSX Data Center can automatically create a new virtual network that fully isolates the app from all other applications in the environment.

Infrastructure Management and IT Operations

Running containers on physical hardware would resurrect difficult infrastructure management and operational problems. Kubernetes manages containerized applications, not the underlying infrastructure on which they are running. If you were to choose to use physical hardware as the underlying infrastructure, you must address a number of requirements while avoiding the creation of difficult-to-manage silos:

- Infrastructure deployment and configuration
- Patching, updating, and upgrading
- Backup and disaster recovery
- Logging and monitoring

Multiple infrastructure silos can duplicate teams, tooling, and processes. Rather than driving focused innovation on a single platform, IT ends up repeatedly performing the same tasks.

By running containers on physical hosts, many of the old problems that virtualization solved would come back to plague IT at the same time that IT is under pressure to increase agility, help accelerate time to market for applications, rapidly adopt new services, and manage costs—all without increasing complexity and risk. These are now core IT requirements, or rapidly becoming new requirements. But as heterogeneous cloud services enter the enterprise, IT is finding it more and more difficult to fulfill these requirements. A VMware software-defined data center (SDDC) with Pivotal Container Service (PKS) solves these problems with a comprehensive, flexible solution that uses the power of BOSH to deploy and manage multiple Kubernetes clusters as well as to patch and upgrade the container host OS.

Running Containers on VMs in a VMware SDDC

vSphere combines with several other VMware technologies to deliver a full-stack SDDC to securely and efficiently run containers on virtual machines while managing the underlying infrastructure with ease. vSAN provides distributed, scalable software-defined storage. NSX Data Center delivers software-defined network virtualization for containers. And PKS integrates Kubernetes and BOSH with this SDDC to orchestrate containerized applications, extend them to the cloud, and manage their underlying infrastructure.

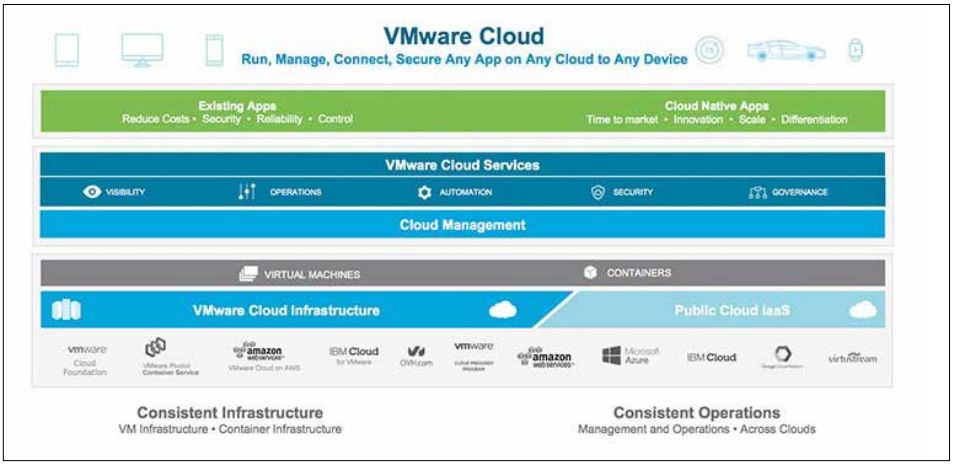
9 Micro-segmentation for Dummies, by Lawrence Miller and Joshua Soto, John Wiley & Sons, Inc. 2015.

VMWARE PKS AT A GLANCE

VMware PKS provides a highly available, production-grade Kubernetes-based container service equipped with advanced networking from VMware NSX Data Center, a secure image registry called Harbor, and lifecycle management with BOSH. The solution radically simplifies the deployment and operation of Kubernetes clusters so that you can run, manage, secure, and maintain containers at scale on VMware vSphere.

KEY BENEFITS OF VMWARE PKS

- Quickly provision Kubernetes clusters on demand
- Deliver high availability for Kubernetes components with rolling upgrades, health checks, and auto-healing
- Use advanced container networking with micro-segmentation, load balancing, and security policies
- Secure container images with vulnerability scanning and image signing
- Improve operational efficiency with monitoring, logging, and analytics



Running containers on VMs in a VMware SDDC with VMware Pivotal Container Services gives you consistent operations on consistent infrastructure.

The result is a common platform that delivers interoperability with existing workloads, where canonical datasets often still reside, and the public cloud, where new applications are taking root and critical services, like machine learning and analytics, can be tapped.

For running containers on VMs instead of physical hardware, this common platform minimizes operational complexity and simplifies management while simultaneously driving maximum hardware utilization and economies of scale.

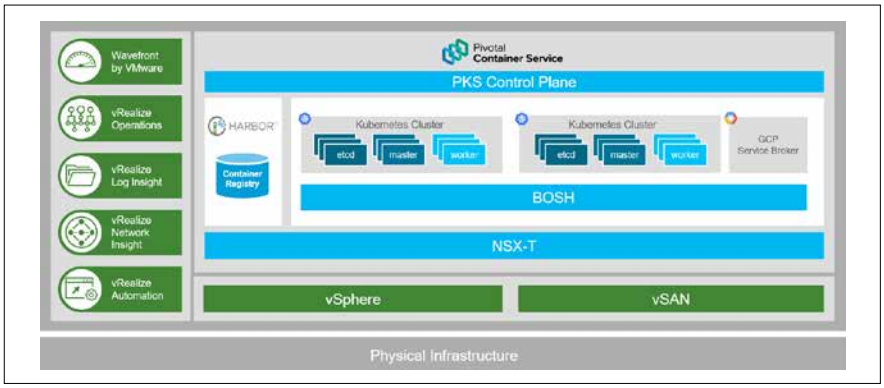
VMware Pivotal Container Service

Although containers themselves are not new, barriers have hindered their use for building and deploying enterprise applications. Until fairly recently, containers lacked the tooling and ecosystem for enterprise-grade deployment, management, operations, security, and scalability. In addition, the requirements of IT administrators often went unfulfilled: Infrastructure for running containers has neglected networking, storage, monitoring, logging, backup, disaster recovery, maintenance, and high availability.

PKS provides a production-grade Kubernetes-based container service equipped with advanced networking, a private container registry, and full lifecycle management. The solution radically simplifies the deployment and operation of Kubernetes clusters so you can run and manage containers at scale on vSphere and in public clouds.

Architecture

PKS combines Kubernetes, BOSH, VMware NSX-T™ Data Center, and Harbor to form a highly available container service. PKS ties these open-source and commercial modules together to efficiently deploy and manage Kubernetes and the containers running on it.



The architecture of VMware Pivotal Container Service.

Minimizing Overhead and Simplifying Management with BOSH

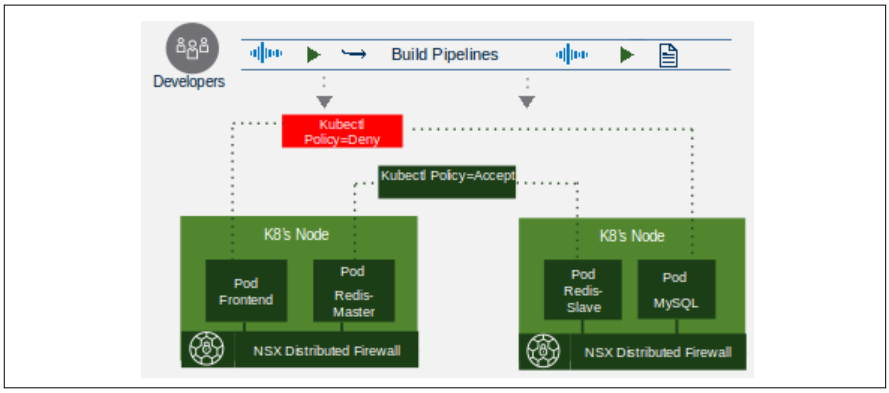
BOSH is an open-source tool for release engineering that simplifies the deployment and lifecycle management of large distributed systems. PKS uses BOSH to version, package, and deploy Kubernetes in a consistent and reproducible manner. By using BOSH, PKS supports deployments across different infrastructure-as-a-service (IaaS) providers, such as vSphere and Google Compute Platform.

Securing Container Images with Harbor

Harbor is an open-source, enterprise-class registry from VMware that stores and distributes Docker images in a private registry behind your firewall. Harbor includes role-based access control, vulnerability scanning for container images, policy-based image replication, integration with LDAP or Microsoft Active Directory, and notary and auditing services.

Networking and Securing Container Workloads with NSX-T Data Center

NSX-T Data Center supplies Kubernetes clusters with advanced container networking, security policies, and micro-segmentation. It furnishes the complete set of Layer 2 through Layer 7 networking services needed for pod-level networking in Kubernetes. You can quickly deploy networks with micro-segmentation and on-demand network virtualization, including load balancing and ingress services, for containers and pods.



NSX furnishes pod-level networking and security policies on Kubernetes.

LEARN MORE ABOUT VMWARE PIVOTAL CONTAINER SERVICE

To find out more about how VMware can help you build, run, and manage cloud-native applications, see https://cloud.vmware.com/pivotal-container-service.

The integration of NSX-T Data Center with PKS delivers an immediate, far-reaching impact on network operations for containers on VMs:

- The native support for NSX-T Data Center load balancers provides highly reliable, high-performance distribution of traffic to applications running on a Kubernetes cluster.
- Policies for micro-segmentation go beyond the standard security policies of Kubernetes.
- Network polices help secure traffic across Kubernetes namespaces and between pods in the same namespace.
- Operational tools and troubleshooting utilities can debug inter-pod communication.

Conclusion: Multicloud Flexibility, Management, and Security

Using a VMware software-defined data center with PKS to run and orchestrate containers on virtual machines instead of physical hardware satisfies the complete set of operational, management, and security requirements for containerized applications while extending their portability to the cloud. The multicloud flexibility of using PKS to run and manage containers on vSphere yields a range of benefits:

- Secure containers and the orchestration system with isolation, strong security boundaries, authentication, access control, image vulnerability scanning, micro-segmentation, and other measures.
- Maintain high availability with proven VMware technology, such as vMotion and DRS.
- Use the resource management capabilities of vSphere to maximize hardware utilization for Kubernetes clusters.
- Furnish containerized applications with data persistence by using vSAN to simplify storage operations and consolidate workloads with distributed, shared-nothing storage.
- Optimize the performance of large clusters and mixed workloads.
- Scale containerized applications without the pain of adding and configuring physical hardware.
- Streamline network management and improve network security by using NSX Data Center.
- Minimize operational complexity and simplify management while simultaneously driving maximum hardware utilization and economies of scale.

Using PKS to run containers on virtual machines in a VMware SDDC fuses the benefits of proven virtualization technology with the emerging benefits of containers and Kubernetes orchestration. The combination produces a sustainable multicloud solution with the power and flexibility to drive your cloud-native strategy to fruition.

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