WHITE PAPER–NOVEMBER 2019

Best Practices for VMware Enterprise PKS on the VMware SDDC

vmware

Table of Contents

Executive Summary

In this paper, we outline the best practices gained from helping our customers to run VMware Enterprise PKS on a VMware Software-Defined Data Center (SDDC). To aid in the readability of the document, we have created an example company called Example.com as a use case for applying these best practices.

In the past three years, Example.com had some debates between virtual machines or bare metal servers to power their ambitious kubernetes project. This debate was primarily triggered by the fact Example.com had two different groups, one of the application side advocating kubernetes on bare metal servers, and the other of the cloud infrastructure side advocating the use of enterprise kubernetes on the VMware SDDC.

Once the VMware technical specialist team was engaged with Example.com teams, many of the concerns that the application teams had were alleviated as they learnt about such features as live migration (vMotion), high-availability (vSphere HA), smart placement (DRS), and a refined ability to manage physical layer through vSphere virtualization.

Just like containers can create an encapsulation construct to abstract the lifecycle of an application process to make it easier to manage, in much the same way, vSphere virtualization can create a software construct to abstract physical compute resources to be able to manipulate them easily. This made the debate tip in favor of the VMware SDDC, and then VMware technical specialists moved to sizing, performance, and scalability, all of which drove Example.com to success. Just the ability to quickly reinstate kubernetes master node with vSphere HA, after a hardware failure, was a significant factor for the resiliency of the environment.

Example.com quickly realized that VMware bigger value proposition besides around operational capabilities was enabling them to run a vast diversity of workloads (cloud-native workloads and traditional workloads) in production, on a single platform such as the VMware SDDC. The VMware SDDC acts as a lightweight infrastructure software fabric for the datacenter across multiple areas: physical servers (vSphere), networking (NSX-T, NSX-SM, SD-WAN), storage (vSAN), multi-cloud (VCF and all six biggest public cloud providers run VMware), and containers (VMware Enterprise PKS, VMware Tanzu and Project Pacific).

The biggest lesson was that many of the assumptions against the VMware SDDC by the Example.com's application teams were due to the knowledge gap between them and the Example.com's cloud infrastructure group. Consequently, they worked together to form the Application Platforms Architecture group to foster collaboration and knowledge sharing across multiple disciplines, resulting in a new persona called Application Platform Architect (read more on Application Platform Architecture here).

After deploying many production VMware Enterprise PKS-deployed kubernetes clusters, with thousands of users, Example.com found that **relying on a VMware SDDC was one of the best choices that they have made for enterprise containers** for several business, functional, technical, and operational reasons as outlined below:

Business and Functional Benefits

1-) The adoption of containers alone is a significant disruption to both application developers and operations teams as a new paradigm in how to develop, deploy, and operate mission-critical workloads. As such, it was critical that a trusted virtualization and automation layer would be adopted to minimize operational risks and complexities.

2-) A mature virtual infrastructure layer for container-based application platforms brings efficiencies such as reuse of compute resources and skills. Mostly, Example.com was able to leverage the existing vSphere and NSX-T team to deliver fully configured PKS platforms. This allowed for the development to entirely focus on application business logic and not be concerned with details of the infrastructure associated with PKS.

Technical and Operational Benefits

1-) Kubernetes is still unfamiliar territory to many. It is complex to install and to operate on a large scale with hundreds of users running thousands of kubernetes pods. In a non-virtualized environment (kubernetes on bare metal) Example.com had lost two kubernetes control plane nodes on the same rack, and the process of rebuilding a kubernetes control node from scratch was daunting, causing damage of credibility with the user base. They found that virtualization provided a protection layer from hardware malfunction with much faster recovery compared to kubernetes alone (*kubernetes gives automatic recovery, not high availability*. See vSphere High Availability and DRS and Availability Zones with PKS).

2-) vSphere virtualization acts as another layer of resource management, opening many more possibilities for business to extract the most benefit by optimizing the hardware resources. This paper explains Example.com's journey on leveraging virtualization to fine-tune the application node size and its performance (see section Worker Node Sizes) and multiple reservation grades of PKS cluster sizes (see VMware SDDC Logical View with Reservation Grade Scenarios).

3-) It is wishful thinking to assume that all VMware Enterprise PKS users can master the knobs of kubernetes pod resource scheduling and prioritization to achieve the most optimized utilization of the compute resources on the kubernetes level. Virtualization is key to reduce the total cost of ownership for the kubernetes cluster, which allows for the onboarding more users on larger PKS-deployed kubernetes clusters (See vSphere Utilization Metrics and Overallocation).

4-) Unfortunately, kubernetes clusters are not "set-it-and-forget-it" and they require Day 2 tasks such as monitoring, backup, patching, and migration. VMware Enterprise PKS on vSphere streamlines these tasks by reducing error-prone procedures and avoids extra hardware costs (See Other Practices and Operational Procedures).

5-) PKS clusters running on the VMware SDDC tend to integrate further strategically with enterprise ecosystem when compared to bare metal deployments. Two examples of these integrations are NSX-T for load balancing and micro-segmentation and VMware volumes and vSAN for container data persistency.

6-) Kubernetes runs faster on a VMware SDDC if compared to bare metal. While it may sound a bit counter-intuitive, this is because of the optimized vSphere virtualization scheduler. See Worker Node Sizes section.

Overview of this Paper

VMware Enterprise PKS is an enterprise kubernetes distribution targeted at corporations of multiple segments and sizes available as part of Pivotal Cloud Foundry or as a stand-alone product. Kubernetes is a critical workload platform, and without a virtualization layer, users and workloads can suffer every time there is a hardware event that will reduce kubernetes compute capacity and possibly lead to kubernetes cluster outages.

Over the last 20 years, VMware has mastered running some of the most critical workloads in large enterprises [1], ensuring that a hardware failure does not impact operations. VMware has firmly established itself in the enterprise datacenter, first with its vSphere compute virtualization technology and then extending to vSAN storage virtualization and NSX network virtualization, in addition to robust management with vRealize. Today, VMware helps customers build modern applications, ensures customers run and manage kubernetes.

This paper outlines the rationale of why corporations such as Example.com can benefit from integrating the VMware SDDC and VMware Enterprise PKS to support an exponential adoption of container workloads with clusters scaled to run 10,000+ kubernetes pods with real-world production environments.

Components such as VMware HA and DRS are detailed in this paper for the optimal configuration of PKS and its resiliency.

VMware Enterprise PKS integrates with VMware NSX-T for advanced container networking, including micro-segmentation, ingress controller, load balancing, and security policy.

This paper only documents the architecture of Enterprise PKS with NSX-T and it is a complement of Design for VMware Enterprise PKS with VMware NSX-T Workload Domains and Deployment of VMware Enterprise PKS with VMware NSX-T Workload Domains.

Other sections of this paper detail the role of VMware on Day 2 operational procedures on PKS.

The adoption of VMware Enterprise PKS has impacted multiple groups in Example.com in the way they develop, deploy, and manage applications. Multiple personas have been part of this journey, but for simplicity, this paper focuses on the following actors:

- **Application Platform Architect**: A multi-disciplinary professional across infrastructure (VMware), PaaS (Platform as a Service), cloud, software development, DevOps, and SRE practices. It is a technical lead responsible for designing and architecting kubernetes clusters and practices. Bridges the teams of Application Developers and Operations.
- **Application Developers**: Individuals (and groups) responsible for developing and managing the lifecycle of Example.com's applications. Usually in communication with business units to code their business logic under business priority and to provide time-to-market. They are the Enterprise PKS end-users.
- **Operations**: The team responsible for managing and supporting the IaaS (VMware) and PaaS (PKS). This paper refers to them as a single group, but in reality, Operations can be multiple teams (one for vSphere, one for PKS clusters, one per business unit, and so on).

The examples and considerations in this document provide guidance only and do not represent strict design requirements, as varying application requirements might result in many valid configuration possibilities.

[1] https://blogs.vmware.com/vsphere/tag/business-critical-apps

High-Level Architecture and Components of VMware Enterprise PKS

Before explaining the best practices of vSphere with VMware Enterprise PKS, this section reviews at high-level the components and architecture of the PKS cluster (Figure 1).

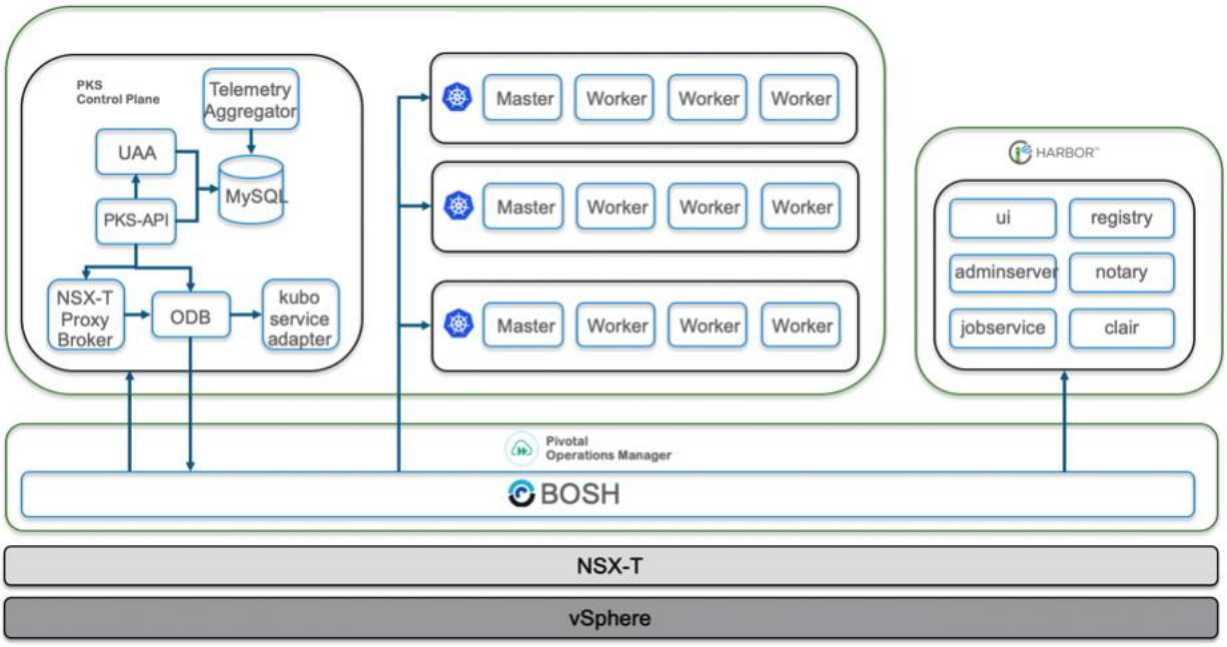

Figure 1: Enterprise PKS High-Level Architecture and Components.

VMware Enterprise PKS Management

VMware Enterprise PKS management components are VMware PKS Control Plane, PCF Operations Manager, BOSH Director, and Harbor:

VMware Enterprise PKS Control Plane: The PKS control plane enables users to deploy and manage kubernetes clusters. This host contains sub-components such as kubo (container interface), PKS-API, NSX-T Proxy Broker (interfaces with NSX-T for load balancers, firewalls, switches), Telemetry Aggregator and UAA (User Account and Authentication to LDAP, SAML, and OpenID connects). Refer here for more information.

PCF Operations Manager: PCF Ops Manager is a host that contains a set of APIs and a graphical interface to deploy platform components such as PKS control plane. Refer here for more information.

BOSH Director: Host that contains BOSH, the automation engine behind PKS that includes a mechanism to monitor and to repairs PKS-deployed kubernetes nodes. Refer here for more information.

Harbor registry: Host that contains the Harbor registry with functions of container registry, image signing (Notary), and container scanning for vulnerabilities (Clair). Harbor has its administration and User Interface (UI). Refer here for more information.

NOTE: VMware Enterprise PKS management components do not have a built-in high availability mechanism. Each component is a single VM. vSphere HA must be configured to protect the environment against physical server outages.

Kubernetes

Main components of a kubernetes cluster are:

Kubernetes Master Nodes: Hosts that contain the kubernetes control plane components such as API/Authentication, Controller, and Scheduler. Additionally, the same nodes run the etcd (DataStore). The masters are only accessible over API, and they are also known as "kubernetes control plane nodes." It is recommended to have at least *three master nodes* for large and mission-critical clusters. This setup is referred to as a multi-master node cluster. The etcd's Raft distributed consensus algorithm requires that the number of kubernetes master nodes is in odd numbers (1,3,5, and so on.) [2].

Kubernetes Worker Nodes: Hosts that run the actual application containers. The size and number of worker nodes will depend on the application footprint (see section Worker Node Sizes).

It is outside of the scope of this paper explaining the internals of a kubernetes cluster, refer here for more information.

VMware NSX-T

The integration of VMware Enterprise PKS with VMware NSX-T provides multiple advantages to PKS-deployed kubernetes clusters, such as automatic micro-segmentation and load balancing and other resources on demand such as logical switches, firewalls, and so on. NSX-T is a nondisruptive solution. You can deploy NSX-T on any IP network, including existing traditional networking models and next-generation fabric architectures, regardless of the vendor. VMware NSX-T components are NSX-T Manager and NSX-T Edge nodes. The configuration of NSX-T is out of the scope of this document, and you can refer here for NSX-T designing best practices and here for NSX-T integration with PKS.

VMware vSphere

vSphere is the fundamental abstract layer on top of the physical layer to power the VMware Enterprise PKS deployment. At a minimum, one dedicated vSphere cluster is required for PKS functionality. Multiple vSphere clusters are recommended.

NOTE: VMware Validated Design (VVD) practices of a Standard SDDC recommends separating VMs on different vSphere clusters according to the type of processing nature: computing, management, and networking. Following the VVD, kubernetes VMs should be on vSphere ***"Compute Pod"*** clusters, vSphere, and PKS management nodes (vCenter, PKS Control Plane, and so on.) should be on vSphere ***"Management Pod"*** cluster, and NSX-T Edge Node components should be on vSphere ***"Edge Pod"*** cluster.

[2] https://raft.github.io/

VMware SDDC Logical View with Reservation Grade Scenarios

There are multiple possible configurations on how to run VMware Enterprise PKS clusters in vSphere with NSX-T. Some of the pertinent factors are PKS cluster size, vSphere HA configuration, reservation grade of PKS cluster, and so on.

The reservation grade of PKS-deployed kubernetes clusters is based on the amount of resources (Memory and vCPUs) guaranteed using Resource Allocation Reservation to the kubernetes worker VMs powering each PKS-deployed kubernetes cluster.

Reservation Grade	Kubernetes Master and Worker VMs Resource Allocation Reservation
Platinum	100% Reservation of the size of the VM.
Gold	50% Reservation of the size of the VM.
Silver	Reservation set to 0

Table 1: Reservation Grade of PKS Clusters based on Resource Allocation Reservation

The initial adoption of PKS (or any kubernetes) usually comes with a certain level of uncertainty with little requirements and a limited budget from the business. It is best to leverage the existing internal cloud for initial PKS deployments to reduce initial investments on the PKS platform. On "Day 1", Example.com did not need a large amount of computing for their kubernetes environments. However, the adoption of kubernetes can be remarkably rapid, requiring scalability of the PKS clusters to absorb the demand. This scenario is a testament to why a VMware SDDC is strategic and relevant in any environment: with multiple businesses competing for compute resources to accommodate the growing user base and number of PKS clusters, resource prioritization comes with various reservation grades of PKS cluster and respective chargebacks.

With this in mind, this paper documents (for illustration purposes) two possible scenarios of PKS-deployed kubernetes clusters running on vSphere clusters:

- Scenario #1: Collapsed vSphere cluster to Support Multiple PKS-deployed kubernetes clusters with different reservation grades ("Platinum", "Gold", "Silver").
- Scenario #2: Multiple vSphere clusters to Support Multiple "Platinum" PKS-deployed kubernetes Clusters.

The collapsed vSphere cluster in Scenario #1 is a single vSphere cluster acting as a Compute, Management, and Edge vSphere cluster. In Scenario 2, some types of vSphere clusters are separated: Compute, Management, and Edge vSphere clusters. For more information on different types of vSphere clusters, see "Virtual Infrastructure Layer in Standard SDDC" section in VMware Validated Design Reference Architecture 5.1

ATTENTION: The VMware Enterprise PKS management components VMs must be always set to 100% Reservation regardless the type of reservation grade of PKS-deployed kubernetes cluster.

For illustration purposes, we assume the following on both scenarios:

- Specs of vSphere host with 1 TB RAM and dual-socket, 28-core per socket. A total of 56 physical CPUs with hyperthreading enabled yielding 112 logical CPUs per host. Four NICs per vSphere host (one NIC must be dedicated to NSX-T).
- vSphere HA configured with Cluster Resource Percentage as Admission Control leveraging the automatic calculation of "Host failures cluster tolerates" to "1" . More about Admission Control on section
- Admission Control Algorithms.

Application Platform Architects must coordinate with Operations (vSphere administrators) on the correct configuration of kubernetes VMs to honor the reservation grades. The PKS management VMs must always have full reservations with vSphere HA (see the section vSphere High Availability and DRS). Preferably, PKS-deployed kubernetes clusters can have their Resource Pool especially the Platinum and Gold reservation grade clusters. Resource Pools can be specified during the deployment of the kubernetes cluster (See PKS Kubernetes Plans section). In this case, the vSphere administrators must pre-create the Resource Pools manually before kubernetes clusters deployment.

Scenario #1: Collapsed vSphere Cluster for varied Reservation Grades of Kubernetes Clusters

Figure 2 depicts the logical view of a collapsed vSphere cluster with four vSphere hosts to power multiple PKS-deployed kubernetes clusters. The collapsed vSphere cluster is a single vSphere cluster acting as a Compute, Management, and Edge vSphere cluster. The HA Admission control is configured to support one vSphere host failure (based on automatic percentage-based failover capacity) [3] .

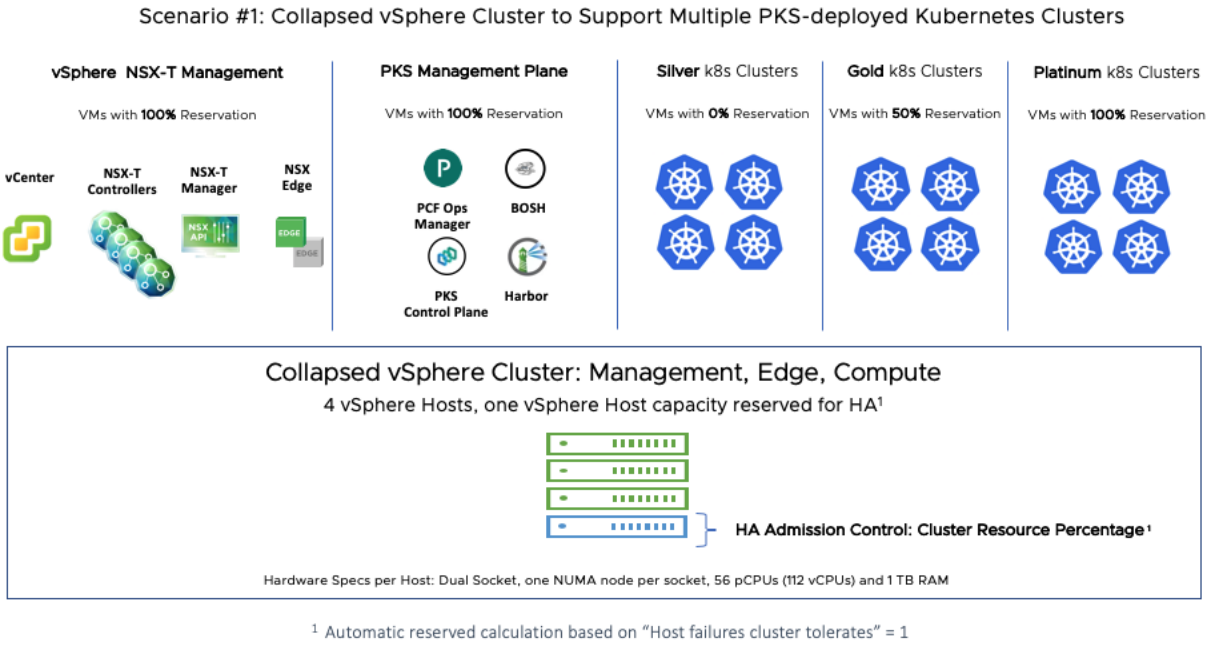

Figure 2: Example of a collapsed vSphere Cluster supporting multiple PKS-deployed kubernetes clusters with Different Reservation Grades

The combined capacity of the three vSphere hosts (one vSphere host capacity is reserved for HA) can accommodate multiple kubernetes clusters with mixed resource reservation grades with or without Memory Overcommitment and CPU overcommitment. The Resource Allocation Reservation guarantees resources for "Platinum" and "Gold" PKS-deployed kubernetes clusters. The net capacity on the Collapsed vSphere cluster is 3 x (112 logical CPUs and 1 TB RAM) = 336 logical CPUs and 3 TB. From this capacity, it is necessary to discount the vSphere, NSX-T, and PKS management nodes. Refer to the section Virtual Machine Sizes for the footprint required for these nodes.

ATTENTION: The minimum footprint of any deployment of VMware Enterprise PKS is three vSphere hosts. This scenario has four vSphere hosts so it can provide resiliency of one physical server failure through HA admission control of Host failures cluster tolerates = 1.

The collapsed vSphere cluster configuration is very common in the initial phases of the adoption of PKS to transition to the highest critical workloads. The same vSphere cluster can be expanded up to 64 vSphere hosts.

[3] See VMware Validated Design of Enterprise PK with NSX-T, line PKS-VI-VC-004

Scenario #2: Multiple vSphere clusters for "Platinum" Kubernetes Clusters

Figure 3 depicts the logical view of 4 vSphere clusters to power multiple Platinum PKS-deployed kubernetes clusters. This scenario is a compound of a vSphere cluster for Management and Edge (4 vSphere hosts) and three Compute vSphere clusters (8 vSphere hosts each vSphere cluster). Each vSphere cluster is configured with HA Admission control to support one vSphere host failure equals to 1 (based on automatic percentage-based failover capacity) [4].

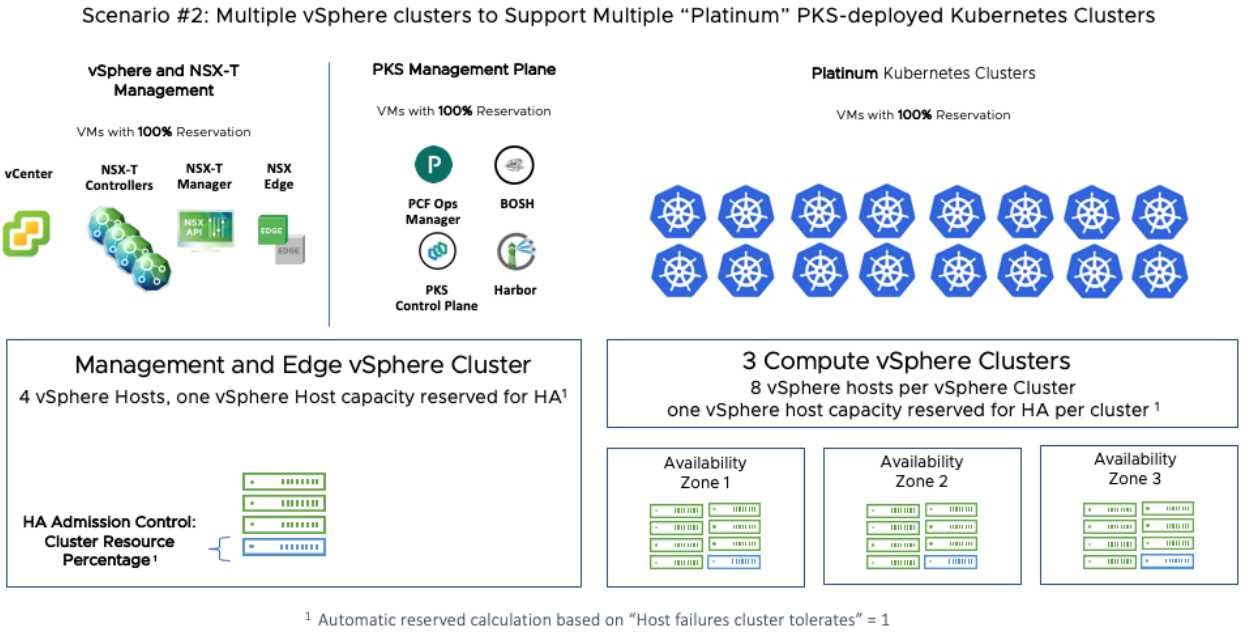

Figure 3: Example of Multiple vSphere clusters to Support Multiple "Platinum" PKS-deployed kubernetes Clusters

In this example, each vSphere host on Compute vSphere clusters has 1 TB RAM and 56 physical CPUs with hyperthreading enabled yielding 112 logical CPUs per host. The net capacity (discounted HA reserved capacity) per Compute cluster is 7 x (112 logical CPUs and 1 TB RAM) = 784 logical CPUs and 7 TB.

The total net capacity available for kubernetes clusters in this scenario among all three Compute vSphere clusters is 3 x (784 logical CPUs and 7 TB RAM) = 2,382 logical CPUs and 21 TB RAM.

Each Compute vSphere cluster is on a different Availability Zone (1, 2, or 3) for the total redundancy of kubernetes clusters. This paper reviews the configuration of Availability Zones for kubernetes clusters on the DRS section.

All master and worker VMs must be configured **with full reservation** with enough capacity to be powered up all the time to assure kubernetes Platinum reservation grade.

[4] See VMware Validated Design of Enterprise PK with NSX-T , PKS-VI-VC-004

NOTE: Always take into consideration some capacity headroom for the vSphere cluster to avoid 100% resource allocation. Later in this paper, there is a section on Scenario #2 as an example to calculate sizes of worker nodes VMs, and reserve 15% of the compute resources. The actual vSphere overhead is substantially less, but for stringent production level measures, Example.com used 15% as ample headroom to absorb workload spikes in normal daily operations.

Scenario Comparison and Recommendation

Scenario #2 (Dedicated vSphere cluster) can be seen as a natural evolution from Scenario #1 (Shared vSphere cluster), and Scenario #2 works best to support mission-critical workloads on PKS-deployed kubernetes clusters. Example.com used scenario #2 for production and UAT kubernetes clusters and scenario #1 for development kubernetes clusters.

VMware NSX-T and Network Considerations

VMware NSX-T is a nondisruptive solution, and you can deploy NSX-T on any IP network, including existing traditional networking models and next-generation fabric architectures, regardless of the vendor.

For best results, set up the VMware Enterprise PKS infrastructure on NSX-T logical switches as documented best practices by VMware Validated Design of Enterprise PKS with NSX-T.

These external (corporate) CIDRs are required for VMware Enterprise PKS integration so traffic can be routed between PKS infrastructure and corporate network:

- PKS floating IP pool: IPs in this pool are assigned for kubernetes Services (type LoadBalancer), kubernetes Ingress, SNAT rules for any kubernetes namespace in the kubernetes clusters (allowing kubernetes pods to reach the external network) and any SNAT rule for kubernetes nodes (allowing kubernetes nodes to reach the external network). Larger CIDRs (/22 and /23 are common) are recommended to support a more significant number of kubernetes objects. Refer to PKS LB CIDR on PKS official documentation.
- PKS management subnet: PCF Ops Manager, Harbor, and other PKS management components to be accessible from the corporate network. A CIDR /28 is usually sufficient for these components. This subnet can be routed or optionally accessed externally through DNAT rules on the NSX-T T0 router.
- vCenter and NSX-T manager subnet: A CIDR /28 is usually sufficient for vSphere and NSX-T components to be accessible from a corporate network.

Multiple non-routable internal CIDRs for vSphere/NSX-T functionality (vMotion, vSAN, and so on) are required as well.

The addresses 172.17.0.0/16 - 172.22.0.0/16 and 10.100.200.0/24 are internal CIDRs for kubernetes clusters and Harbor. These addresses must not be used anywhere.

Each vSphere host must have two dedicated NICs for NSX-T (total four physical NICs per server is a plus). Refer to the documentation for the installation of VMware Enterprise PKS with VMware vSphere and VMware NSX-T.

Virtual Machine Sizes

vSphere and NSX-T Management Nodes

All sizing recommendations of this section target both Scenario #1 and Scenario #2.

The vCenter sizing guidelines are found here with storage guidelines here. The recommended vCenter VM size is 4 vCPUs, 16 GB RAM, and 340 GB disk.

The NSX Manager sizing guidelines are found here. The recommended size is medium NSX Manager with 6 vCPUs, 24 GB RAM, and 200 GB disk.

The NSX Edge sizing guidelines are found here. PKS demands a large NSX Edge size due to the number of load balancers: 8 vCPUs, 32 GB, and 200 GB disk.

VMware Enterprise PKS Management Nodes

Figure 4 shows the sizing recommendations of VMware PKS Management Nodes based on VMware Validated Design, and they are recommended for both Scenarios #1 and Scenarios #2.

Virtual Machine	vCPU	Memory (GB)	Storage (GB)	Number per Deployment
PCF Ops Manager	1	8	160	1
BOSH Director	2	8	103	1
VMware Enterprise PKS control plane	2	8	29	1
BOSH Compilation VM	4	4	32	4
CLI/API Client	1	2	8	1
VMware Harbor Registry	2	8	167	1
Total	12	38	499	9

Figure 4: PKS Management Nodes Sizes

The BOSH Compilation VMs are ephemeral VMs, and they do not need to be provisioned or configured, The CLI/API client is covered on section CLI/API Jump Box.

PKS Kubernetes Plans

One of the main features of PKS is the ability to create multiple kubernetes clusters seamlessly using PKS API or PKS CLI. PKS has the construct of plans with pre-defined templates for sizes of kubernetes clusters. The plans are instrumental in allowing developers to self-serve and create their clusters under pre-defined constructs.

The Application Platform Architects can customize the kubernetes clusters sizes based on business purpose (Development, UAT (User and Acceptance Testing), Production, and so on) and business demand (small, medium, large, and so on) and create new plans using these steps.

See (Figure 5) for the recommended VM sizes for master nodes.

Number of Workers	CPU	RAM (GB)
1-5	1	3.75
6-10	2	7.5
11-100	4	15
101-250	8	30
251-500	16	60
500+	32	120

Figure 5: Kubernetes Master Node VM Sizes

For built-in kubernetes high availability, this paper recommends Platinum kubernetes clusters be created with three master nodes with full reservation.

SCALABILITY: Application Platform Architects can always leverage vSphere on Day 2 and further scale the sizes of the master and workers VMs to support more pods on growing kubernetes clusters.

Worker Node Sizes

Worker node sizing calculation – *both horizontally (number of nodes) and vertically (specs of nodes)* - requires more attention since their sizing is done case-by-case for each enterprise.

It is essential to understand the type of application footprints on the kubernetes cluster. Applications with homogenous requirements, that is, with the same kind of middleware, consumption, and nature (jobs, microservice, and so on), can drive the sizing of worker nodes tailored to a desirable number of kubernetes pods per application node. If the footprint of the application is very heterogeneous, for example, from small microservices to large memory cache databases, it is better to have large worker nodes and have the kubernetes scheduler to manage the different size of kubernetes pods. There is a maximum of 100 kubernetes pods per worker node.

For the maximum the size of worker nodes, take into consideration the CPU architecture – such as Non-Uniform Memory Architecture (NUMA).

PERFORMANCE: When compared to generic Linux running on bare metal, vSphere does a better job at scheduling the worker node VMs and their pods to run on the right CPUs. vSphere provides better localization, and it dramatically reduces the number of remote memory accesses. kubernetes pods running memory-bound workloads show superior performance when deployed on a VMware SDDC, as seen on the performance testing with Project Pacific.

There are multiple ways to calculate the optimal sizing of worker node VMs. PKS offers a formula to calculate worker nodes sizes based on the number of desirable pods and application footprint.

A simple method of calculation is opting for the maximum vertical size of the worker VM to get the memory size and vCPUs of the NUMA node with the headroom for vSphere.

You can arrive at values for the memory and vCPU by applying the following formula [5]:

Memory Size	(0.85 * Total RAM on Host) / Number of Sockets
vCPU (*)	(0.85 * Total Logical Processors on Host) / Number of Sockets

(*) Hyperthreaded vCPUs (please see Appendix section on Hyperthreading)

HEADROOM: The factor of 0.85 is a practical approximation for the reserved capacity for ample vSphere headroom of 15%, tested for mission-critical applications with deterministic performance such as trading systems. [5] (*) If users of Platinum kubernetes clusters report vCPU slowness (which will depend on application footprint), check utilization vSphere metrics, and if necessary, revisit the formula to increase the vSphere overhead.

The maximum horizontal size of worker nodes is the total number of NUMA nodes of the vSphere cluster discounting capacity of the reserved failover.

For illustration purposes, Example.com's calculated the size of worker nodes based on

The collapsed vSphere cluster configuration is very common in the initial phases of the adoption of PKS to transition to the highest critical workloads. The same vSphere cluster can be expanded up to 64 vSphere hosts.

Scenario #2: Multiple vSphere clusters for "Platinum" Kubernetes Clusters would be:

- **Vertical Size.** Example.com's bare metal server is an Intel dual-socket with one NUMA node per socket. Each socket had 28 physical CPUs (or 56 logical CPUs with hyper-threading) and 512 GB. Calculation is (112 logical processors | 1 TB RAM) *0.85 / 2 = **48 vCPUs** and **436 GB RAM.**

- **Horizontal Size.** Example.com has 8 vSphere Hosts per vSphere cluster. Discounting one vSphere host for HA ("Host failures cluster tolerates" to "1"), there is the net capacity of 7 vSphere hosts. Calculation is 7 * 2 NUMA nodes per Host = 30 NUMA nodes = **14 workers VMs.**

NOTE: Application Platform Architects still need to take into consideration the capacity required by kubernetes master VMs for each cluster and subtract from resources of the vSphere cluster.

This simple method naturally drives to fairly large worker VMs in terms of vertical size. Application Platform Architects can make initial assumptions on Day 1 and check USE metrics (Utilization, Saturation, Errors) metrics of Operating System, vSphere, and kubernetes layers to fine-tune the sizing of worker nodes on Day 2 and reduce the vertical size while increasing the horizontal size.

[5] "How to Design and Tune Virtualized Trading Platforms", Minute 26 - Emad Benjamin, VMworld 2016

Over-allocation on Kubernetes for Non-Production Workloads

Some businesses opt for over-committing compute capacity on non-production PKS clusters (such as development clusters) to schedule more pods and onboard users using the same compute capacity.

Commonly, users ask which is the best way of computing capacity over-allocation for kubernetes clusters: vSphere over-committing (VM level, that is, larger application node VMs on the same physical infrastructure) versus kubernetes scheduling using pod QoS and pod priorities and preemption.

vSphere can give the options to corporations to perform over-allocation at a hypervisor layer on worker VMs, transparently and in conjunction with kubernetes scheduler. However, teams need to exercise caution on this combination, mainly when PKS admins and VMware admins belong to different groups and are unaware of each layer's utilization. Use both layers only under close supervision and monitoring. For more information about such monitoring, see the section vSphere Utilization Metrics and Overallocation.

Start with kubernetes as the first layer of over-allocation and leverage PKS metrics (executing the command "**kubelet describe nodes**") to see specific metrics of kubernetes allocation (that is, the requests and limits metrics). Only after analysis, determine whether the second level of over-allocation through hypervisor is required for the fine-tuning of the utilization efficiency.

OVER-ALLOCATION: Example.com chooses to over-commit compute resources (without over subscribing available physical compute resources) primarily on the kubernetes layer because the business users (application developers and managers) have control and permissions on kubernetes scheduler (such as default requests and limits) rather than on vSphere cluster.

See the section vSphere Utilization Metrics for an example of how Example.com used both kubernetes and vSphere metrics to decide for over-allocation on both layers.

vSphere High Availability and DRS

vSphere HA and DRS are paramount to give resource optimization and availability to PKS-deployed kubernetes clusters.

vSphere High Availability (HA)

Kubernetes alone cannot completely handle the resiliency and recovery needs of all mission-critical cloud-native applications. **Kubernetes gives automatic recovery and not high availability** [6].

vSphere HA offers these additional benefits for kubernetes clusters and cloud-native applications:

- Protection and quick recovery for the master VMs, especially to etcd health.
- In case of a worker node failure, vSphere HA acts faster (VMs are restarted before kubernetes pods are evacuated) and more transparent (kubernetes pods continue to run on the same worker node after restart, avoid re-mounting and connectivity on a different node) when compared with the default kubernetes pod timeout mechanism (*5 minutes*). Stateful applications with persistent mounted volumes (VMFS, NFS) can take even longer (up *to 10 minutes*) [6] to recover while awaiting the re-attachment of volumes.

AVOID OUTAGES: It is paramount to configure the restart behavior priority of PKS nodes via Restart Priority to avoid inconsistent cluster state. In the event of complete (planned or unplanned) maintenance of vSphere clusters, instrument the restart priority of PKS management nodes first as the priority sequence documented by article kb67657. The PKS-deployed kubernetes nodes must startup after PKS management nodes (with Platinum cluster with higher priority followed by Gold and Silver clusters) with the sequence documented by article kb67656. For the shutdown sequence, it is recommended that kubernetes worker nodes shut down first, followed by the kubernetes master nodes, and finally, the PKS management nodes.

Additionally, vSphere HA utilizes heavily tested heart beating and partitioning detection mechanisms to monitor hardware, network, VM, and even forecasting failures with vSphere 6.7 proactive HA (such as one out of two power supplies had failed on the host and then proactively vMotion PKS nodes to a healthy metal server).

Lastly, in case there are Platinum, Gold, and Silver PKS-deployed kubernetes clusters on the same vSphere cluster, VMs of Platinum PKS-deployed kubernetes clusters should have higher restart priority than VMs of Gold and Silver PKS-deployed kubernetes clusters. This way, the Platinum worker nodes will be powered up first to guarantee more uptime and resource priority.

Admission Control Algorithms

Quoting from vSphere Availability Guide, "vSphere HA uses admission control to ensure that sufficient resources are reserved for the virtual machine."

There are three Admission Control algorithms: Cluster Resource Percentage, Slot Size, and Failover Hosts.

Regardless of which algorithm used, HA Admission Control provides flexibility and resiliency for PKS-deployed kubernetes clusters when compared to any kubernetes deployments on pure bare metal.

[6] "The Magic of Kubernetes Self-Healing Capabilities" (minute 26), Saad Ali, Kubecon 2019

The most used HA Admission Control algorithm is per percentage basis. In this case, on vSphere 6.5 and later versions, it is recommended to leverage the automatic calculation of percentage on using "Host failures cluster tolerates" to a number of hosts (our scenarios set it to "1") as documented on VMware Validated Design.

We recommend the book "VMware vSphere 6.7 Clustering Deep Dive" for more details on Admission Control. Please note there is no Admission Control for storage: all storage capacity tasks must be done manually as of this writing.

vSphere DRS

vSphere DRS is crucial for a balanced distribution of computing resources and initial placement of PKS-deployed kubernetes nodes VMs.

Example.com has enabled DRS with full automation and migration threshold set to "3" (across all reservation grade clusters). For the kubernetes master nodes, this automation level should be overridden (set as "Partially automated") to avoid unnecessary migrations [7].

VM-VM anti-affinity rule must exist on the NSX-T Manager to ensure that its VMs are never on the same vSphere host.

Availability Zones with PKS

In case the PKS deployment runs across multiple availability zones, it must be configured through Ops Manager (see Figure 6). In this case, PKS can automatically deploy kubernetes clusters across these availability zones.

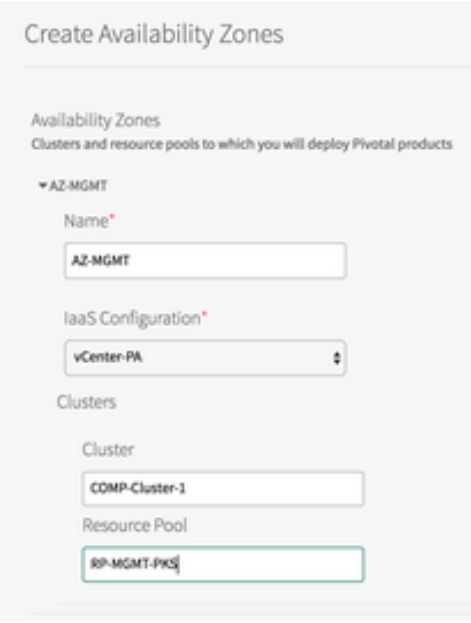

Figure 6: Screenshot of PCF Ops Manager configuring multiple Availability Zones .

As documented previously at Scenario #2: Multiple vSphere clusters for "Platinum" Kubernetes Clusters, this scenario has three Compute vSphere clusters with each Compute vSphere cluster in an individual availability zone (1, 2 or 3). Each availability zone (AZ) corresponds to a vSphere Host group (also known as DRS host group) in each vSphere cluster. The vSphere admin must provision the vSphere Host groups beforehand.

[7] More details on Understanding vSphere DRS Performance Guide, DRS Aggression Levels section

When PKS-deployed kubernetes cluster is created over PKS API, the users can specify which availability zones (AZ) the kubernetes cluster is targeted to be deployed. If three availability zones are selected during the creation of a multi-master kubernetes cluster, each master VM will be deployed on each availability zone (See Figure 7). The worker VMs spread evenly across the AZs.

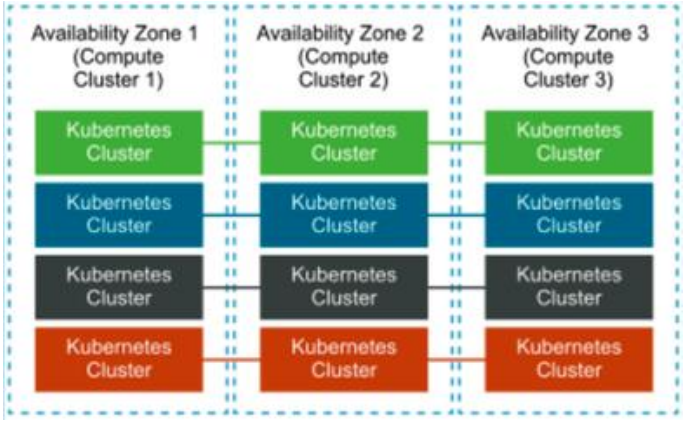

Figure 7: Kubernetes Clusters across Three Available Zones

On single and smaller vSphere cluster (such as in Scenario #1: Collapsed vSphere Cluster for varied Reservation Grades of Kubernetes Clusters), the same approach can follow: configure availability zones via vSphere hosts groups, so the deployed kubernetes masters are running in different vSphere hosts.

USER EDUCATION: Application Platform Architects must publish documentation to educate the application developers to avoid singletons and leverage kubernetes scheduling awareness of worker nodes labels using pod anti-affinity preferred rule.

Another critical feature of PKS kubernetes Plans is the ability to specify the Resource Pool, where the kubernetes will be created. The vSphere administrators can create Resource Plans according to each reservation grade, such as Platinum or Gold Resource Plans, and have PKS deploy the clusters automatically there.

Other Practices and Operational Procedures

Example.com, like any large corporation, has understood that there are multiple operational challenges in productizing and deploying kubernetes on a large scale. VMware vSphere and PKS alleviate some of the operational challenges of day 2, reducing the day-by-day kubernetes complexities and resulting in a more resilient enterprise deployment.

CLI/API Jump Box and Backup Procedures

The CLI/API Client (or jump box) is optional but recommended. This is a linux "jump box" for administrative reasons as invoking PKS CLI commands to the PKS and PKS-deployed kubernetes cluster for administrative tasks such as BOSH Backup and Recovery through BBR. This machine must be on the same NSX-T logical switch as the VMware Enterprise PKS if in NAT setup. The same VM can perform kubernetes clusters' etcd backup through BBR.

CA-certs on all SDDC components

Replace self-signed certificates with CA-certs on all critical SDDC components such as vCenter and NSX-T Manager.

VMDK for Container Data Persistency

PKS supports VMFS datastores (over NFS/iSCSI and FC) and vSAN datastores as providers of persistent storage for containers. For such, the datastore must be accessible by all vSphere hosts hosting the kubernetes worker node. The VMDKs can be pre-created (static provisioning) or dynamically created via the VCP kubernetes plugin (dynamic provisioning).

VMware vSAN allows storage resources attached directly to vSphere hosts to be used for distributed storage and accessed by multiple vSphere hosts with configured vSAN fault tolerance.

As of this writing, there are some limitations on vSAN and PKS:

- The combination of Proactive HA and vSAN is not supported.
- vSAN stretched cluster is not supported by PKS.
- Multiple Compute vSphere clusters are supported with vSAN (however, a complementary shared datastore must be connected to those clusters to support kubernetes persistent volume).

Upgrade of Kubernetes Clusters

Managing and upgrading kubernetes clusters have been a challenge operationally in light of multiple sub-components: kubernetes itself, docker, etcd, etc.

PKS can upgrade one or multiple PKS-deployed kubernetes clusters in one unique operation via Ops Manager or BOSH CLI. An errand exists on the PKS tile allowing the upgrade of all kubernetes clusters once a new PKS tile is installed from Ops Mgr. Please refer to here for more information. In recent versions of PKS, the PKS CLI command pks upgrade-cluster has been introduced, and it is the preferred method of upgrading individual kubernetes clusters.

vSphere Utilization Metrics and Overallocation

Example.com has also realized that high utilization on kubernetes schedule resources does not translate into high utilization on a vSphere cluster. Kubernetes overcommitment is the difference between Request and Limits, and it does not necessarily translate in CPU/Memory physical utilization of the vSphere cluster.

vSphere cluster utilization metrics are critical in validating the effectiveness of PKS-deployed kubernetes cluster utilization and driving higher utilization of hardware resources. With DRS and vSphere metrics, Example.com has decided for extra overallocation of vCPUs on non-production clusters (Silver reservation grade cluster). vSphere overcommitment is assigning more CPUs dedicated to worker nodes VMs.

The success criteria of this journey are the ability of onboarding more PKS users (and consequently creating more kubernetes clusters and scheduling more kubernetes pods) on the PKS DEV cluster (Silver reservation grade) without expenditure (that is, expanding the cluster with more hardware).

etcd Backup

The backup of etcd on Platinum kubernetes clusters is highly recommended. For more information, see the procedure as documented by PKS. Also, you can take the backup of kubernetes objects using VMware Project Velero.

Master Node Recovery and Protection

There is always the chance of permanent loss of a master node. vSphere reduces the chances of such failures dramatically by giving HA protection. However, this paper strongly recommends multi-master nodes for all mission-critical kubernetes clusters.

ATTENTION: In case of a loss of two out of three master nodes, the kubernetes cluster becomes completely unavailable. Again, vSphere HA is another critical protection of the PKS cluster uptime.

Conclusion and Call to Action

This paper has documented the journey of Example.com why and how any corporation can leverage vSphere to successfully deploy VMware Enterprise PKS to power thousands of users running kubernetes pods in production.

We encourage the audience who has not yet been familiar with VMware Enterprise PKS to learn it through VMware Hands-On Lab and download and install VMware Enterprise PKS 1.5.

Relying on virtualization and cloud-native platforms is necessary for any company of any size that is pursuing to run containers to support mission-critical applications.

At VMware, we believe that a fine-tuned kubernetes deployment is a critical building block of what matters: How well you have abstracted the application platform nature of your enterprise workloads and run them successfully. We encourage the audience to read the VMware Office of CTO Application Position Paper for more details.

Finally, read how VMware helps customers build modern applications and ensures customers run and manage kubernetes through strategic initiatives of VMware Tanzu and Project Pacific.

Other Appendices

Audience

This paper is intended to PKS and kubernetes engineers and architects, SREs, VMware practitioners, Engineer Managers, and decision-makers looking for guidance in reducing TCO when integrating vSphere and PKS.

Ultimately, this paper is targeted on what VMware understands as the rise of a new persona in cloud-native applications: Application Platform Architect [8], which is an intersection of three disciplines of development, infrastructure, and production deployments.

In the Scope and Out of the Scope of This Document

In the scope of this document is the rationale of rationale best practices on Enterprise PKS on vSphere 6.5/6.7.

Out of the scope of this document is the rationale (or documentation) of:

- Containers versus non-container deployments.
- vSphere versus KVM.
- PKS capabilities versus other kubernetes distros.
- Differences between versions of Enterprise PKS.

Hyperthreading

Too often, hyperthreading is disabled, which would potentially cause additional CPU cycles not to be utilized. Turning hyperthreading on does not mean you are getting the double of performance of the physical core. However, by turning hyperthreading on, you allow multiple vCPUs to be available to various workloads whenever they need the CPU cycles. By doing this, you are allowing the maximum ability for various kubernetes pods to be scheduled by multiple physical cores by requesting multiple vCPUs.

vSphere makes conscious CPU management decisions regarding mapping vCPUs to physical cores. An example is a VM with four virtual CPUs. Each vCPU will be mapped to a different physical core and not to two logical threads that are part of the same physical core.[9]

Example.com received tremendous benefits from turning on hyperthreading and the ability to allocate multiple vCPUs to various kubernetes pods as opposed to limiting it to a mapping of a single vCPU (or 1.3 vCPU) to one physical CPU.

Always enable hyperthreading and distribute and consume all the vCPUs for the PKS cluster, even if the entire PKS cluster is expected to be dedicated to single-threaded High-Performance workloads.

It is worth mentioning that public cloud services commonly use hyperthreaded (vCPU) charged as a CPU on almost all types of instances. The user needs to choose particular instance types to access the real physical CPU instead of the vCPU.

Additionally, kubernetes understands that one CPU is a hyper-threaded CPU on a bare-metal server.

Example.com took a similar approach of on vCPU of vSphere clusters adopting the following convention:

[8] https://octo.vmware.com/vmware-octo-application-platforms-position-paper/

[9] Reference: Hyper-Threading Section on SQL Server on VMware Best Practices Guide

1 physical CPU (bare metal) = 2 Logical Processors (vSphere Host) = 2 vCPUs (VM) = 2,000 kubernetes millicores

For further discussion on for hyperthreading behavior, refer to vSphere 6.7 performance best practices - "ESXi systems manage processor time intelligently to guarantee that load is spread smoothly across all physical cores in the system." (page 22).

Another source of information is The Value of Running Kubernetes on vSphere, a VMworld 2018 presentation.

Acknowledgements

This document was authored by the following people:

- Rafael Brito

The author would like to thank the following people for their feedback:

- Eric Railine
- Francis Guillier
- Simone Morellato
- Tom Schwaller
- Steven Hoenisch

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

vmware®