



VMware VCAP Cloud Foundation vSphere Kubernetes Service 3V0-24.25 Practice Questions

Shared by Collier on 17-08-2026

For More Free Questions and Preparation Resources

Check the Links on Last Page



Question 1

Question Type: MultipleChoice

An administrator is upgrading to VKS 3.4 and encounters the following error during cluster creation using workload, yami:

```
Error from server (Forbidden): error when creating "workload.yaml": admission webhook
"capimutating.tanzukubernetescluster.run.tanzu.vmware.com" denied the request: Cluster and variable validation failed:
[spec.topology.variables[defaultStorageClass]: Invalid value: "\"vks-storage-policy\"": variable is not defined,
spec.topology.variables[nodePoolVolumes]: Invalid value: "{\capacity\":
{\storage\": \"100Gi\", \"mountPath\": \"/var/lib/containerd\", \"name\": \"containerd\", \"storageClass\": \"vks-storage-policy\"},
{\capacity\": {\storage\": \"100Gi\", \"mountPath\": \"/var/lib/kubelet\", \"name\": \"kubelet\", \"storageClass\": \"vks-storage-
policy\"}}: variable is not defined, spec.topology.variables[trust]: Invalid value: \"{additionalTrustedCAs\":
[{\name\": \"additional-ca-1\"}]\": variable is not defined]
```

How should the administrator resolve this issue to successfully complete the upgrade?"

Options:

- A- Verify workload cluster versions to ensure compatibility
- B- Remove the deprecated variables and apply the new workload, yaml.
- C- Rename the vSphere storage policy and apply the new workload.yaml.
- D- Restart the Kubernetes services and restart the upgrade

Answer:

B

Explanation:

The error shows an admission webhook denial where variable validation failed and multiple entries under `spec.topology.variables` are reported as "variable is not defined". That message indicates the manifest is supplying variables that are not part of the current Cluster API / topology schema enforced by the Supervisor during cluster creation. In VKS, cluster provisioning is declarative: you invoke the VKS API with `kubectl` + a YAML file, and "after the cluster is created, you update the YAML to update the cluster." When the API/schema changes between releases, older manifests can contain fields/variables that are no longer recognized, and the admission webhook blocks them to prevent creating an invalid cluster spec.

This aligns with VMware's broader direction that the older Tanzu Kubernetes Cluster (TKC) API was deprecated and customers are encouraged to use Cluster API for bootstrap/config/lifecycle management. In practice, to complete the upgrade/creation successfully, you must update the cluster manifest to match the supported schema: remove the deprecated/unknown topology variables shown in the error (for example, the undefined storage-policy and trust variables) and re-apply the corrected workload.yaml.

Question 2

Question Type: MultipleChoice

An architect is working on the data protection design for a VMware Cloud Foundation (VCF) solution. The solution consists of a single Workload Domain that has vSphere Supervisor activated. During a customer workshop, the customer requested that vSphere Pods must be used for a number of third-party applications that have to be protected via backup.

Which backup method or tool should be proposed by the architect to satisfy this requirement?

Options:

- A- Standalone Velero with Restic.
- B- vCenter file-based backup.
- C- Velero Plugin for vSphere.
- D- vSAN Snapshots.

Answer:

C

Explanation:

VCF 9.0 distinguishes between backing up the Supervisor control plane and backing up workload that run on the Supervisor, including vSphere Pods. In the "Considerations for Backing Up and Restoring Workload Management" table, the scenario "Backup and restore vSphere Pods" explicitly lists the required tool as "Velero Plugin for vSphere", with the guidance to "Install and configure the plug-in on the Supervisor."

The same document is explicit that standalone Velero with Restic is not valid for vSphere Pods, stating: "You cannot use Velero standalone with Restic to backup and restore vSphere Pods. You must use the Velero Plugin for vSphere installed on the Supervisor."

vCenter file-based backup is documented for restoring the Supervisor control plane state, not for backing up and restoring vSphere Pod workloads themselves. Therefore, to meet the requirement to protect third-party applications running as vSphere Pods, the architect should propose the Velero Plugin for vSphere.

Question 3

Question Type: MultipleChoice

What Kubernetes component is responsible for workload creation?

Options:

- A- etcd
- B- API Server
- C- Scheduler
- D- Kubelet

Answer:

D

Explanation:

In Kubernetes, the component that actually creates and runs workloads on a node is the kubelet. The kubelet is the node agent that ensures the containers described by PodSpecs are running on that node. VCF 9.0 maps this concept directly into vSphere Supervisor by describing Spherelet as "a kubelet that is ported natively to ESXi and allows the ESXi host to become part of the Kubernetes cluster," showing that kubelet functionality is responsible for running workloads on worker nodes (ESXi hosts in the Supervisor case).

The other options have different roles: etcd is the control plane data store, API Server is the front-end for Kubernetes API operations, and the Scheduler decides placement (which node should run a pod). VCF 9.0 even calls out that "the Kubernetes scheduler... cannot place pods intelligently" without visibility into vCenter inventory---reinforcing that scheduling is about placement decisions, not the act of creating/running the workload on the node.

So, while the scheduler selects where a pod should run, the kubelet is the component responsible for actually instantiating and maintaining the workload on the target node.

Question 4

Question Type: MultipleChoice

Which two types of Kubernetes member objects can be used when creating groups to collect and manage objects for service-level networking/security policies (for example, in a service mesh or Kubernetes-aware policy model)? (Choose two.)

Options:

- A- Node
- B- Cluster
- C- Service
- D- Security
- E- API

Answer:

A, C

Explanation:

When you build Kubernetes-aware policy constructs, "groups" are commonly used to collect objects so you can apply consistent controls (security posture, traffic rules, observability scope, etc.) to a set of endpoints. In VCF 9.0 documentation, the Kubernetes member types that can be used for group-based collection include Kubernetes Node and Kubernetes Service as supported member object categories. Nodes represent the worker compute endpoints that run workloads, while Services represent stable networking front-ends for sets of pods (and are often the anchoring object for policy and routing decisions at the Kubernetes layer). Using Node-based grouping helps apply policies to the infrastructure execution points where workloads run, and Service-based grouping helps apply policies consistently to application entry points and east-west communication targets, regardless of pod churn. This combination is especially useful in Kubernetes-centric operational models because it aligns policy scope with both (1) where workloads execute (nodes) and (2) how workloads are exposed and discovered (services).

Question 5

Question Type: MultipleChoice

An administrator is tasked with installing Istio Service Mesh for VMware vSphere Kubernetes Service (VKS). Which command installs Istio?

Options:

- A- Use curl to download and install Istio.
- B- Use vcf package install to install the Istio package.
- C- Use kubectl istioctl install.
- D- Use docker build istioctl.

Answer:

B

Explanation:

VCF 9.0 describes a standardized approach to deploying optional capabilities on VKS clusters via standard packages. In the VCF Automation cloud services overview, VMware lists Istio among the packages that "can be optionally installed" for VKS. For installing those packages, the VCF CLI provides a dedicated package plugin that "works on a workload cluster" and exposes explicit install workflows.

The VCF CLI command reference shows the syntax for installing a package:

`vcf package install INSTALLED_PACKAGE_NAME --package PACKAGE_NAME --version VERSION` (with optional values `files` and `overlays`). This is the supported, VCF-aligned way to install standard packages such as Istio onto a VKS workload cluster in a controlled, declarative manner, including tracking reconciliation status and providing configuration via a values file.

Options that rely on ad-hoc downloads (`curl`) or building tooling (`docker build`) do not match the documented VCF package-management workflow for VKS standard packages. Therefore, the correct command family is `vcf package install`.

Question 6

Question Type: MultipleChoice

An administrator has been tasked with enabling backup for a vSphere Kubernetes cluster. How does the administrator configure a backup?

Options:

- A- Set the `BackupStorageLocation` in Velero to an S3-compatible object store.
- B- Set the `BackupStorageLocation` in Velero to use the storage allocated to the namespace for the cluster.
- C- Use vSphere Replication to back up the virtual machines the cluster is deployed on.
- D- Take a vSAN snapshot of the virtual machines the cluster is deployed on.

Answer:

A

Explanation:

In VCF 9.0 Workload Management, workload backup for VKS clusters (and vSphere Pods) is performed using Velero, specifically the Velero Plugin for vSphere for Supervisor/VKS scenarios. The documentation describes that you must first provide an S3-compatible object store as part of installing and configuring the Velero Plugin for vSphere, because backup data (Kubernetes metadata and volume snapshot data movement) relies on object storage rather than "namespace storage." The backup workflow further indicates that when you create a backup, the system uploads Kubernetes metadata to the object store, and persistent volume snapshot handling is coordinated through the plugin components (Snapshot/Upload custom resources).

Therefore, the correct configuration pattern is to point Velero's backup target (BackupStorageLocation) at an S3-compatible object store endpoint so metadata and snapshot payloads have a durable destination. This aligns with the documented prerequisite that object storage is required to enable backup/restore operations for persistent workloads, and it is the mechanism Velero uses to store backup artifacts for later restore operations in VKS environments.

Question 7

Question Type: MultipleChoice

Which two types of groups can be created to collect and manage objects in an Istio-based service mesh environment? (Choose two.)

Options:

- A- Security
- B- API
- C- Node
- D- Service
- E- Cluster

Answer:

C, D

Explanation:

Comprehensive and Detailed 150 to 250 words of Explanation From Exact Extract of VMware Cloud Foundation (VCF) 9.0 + vSphere Supervisor + vSphere Kubernetes Service documents :

In VMware Cloud Foundation 9.0, the documented "group" construct for collecting and managing Kubernetes objects is implemented as generic groups with Kubernetes member types that can be used for policy-driven operations (for example, securing traffic between infrastructure workloads and Kubernetes workloads). The documentation explicitly states that you can "create generic groups with Kubernetes member types in dynamic membership criteria" and then use these groups in firewall rules to secure traffic involving Kubernetes clusters.

The same section provides a table of Kubernetes member types available for group membership criteria, and it explicitly lists Kubernetes Node (cluster scope) and Kubernetes Service (namespace scope) as supported member types. This maps directly to the answer choices Node and Service as the two valid "types" that can be used to build logical collections of Kubernetes objects for consistent management and policy enforcement.

The other answer options do not match the documented Kubernetes member types. "Security" and "API" are not Kubernetes member types in the group criteria model, and while "Kubernetes Cluster" is also a listed member type, the question asks for two, and Node and Service are the most direct object types for grouping runtime endpoints and service front-ends.

Question 8

Question Type: MultipleChoice

What component is used to provision VMware vSphere Kubernetes Service (VKS) service mesh workload clusters?

Options:

- A- Harbor
- B- Cert-manager
- C- Carvel
- D- Cluster API

Answer:

D

Explanation:

In VCF 9.0, VKS cluster provisioning and lifecycle management is built around Kubernetes-native, declarative APIs. The VMware documentation describes that Cluster API provides declarative, Kubernetes-style APIs for cluster creation, configuration, and management, and that its inputs include the cluster definition plus the resources describing the virtual machines

and "cluster add-ons." This is why Cluster API is the component associated with provisioning (creating and managing) workload clusters in the VKS model.

By contrast, cert-manager is typically used to automate certificate issuance/renewal for in-cluster components, and Carvel is a set of tools often used to package, configure, and install Kubernetes software, but neither is the core provisioning controller for creating the clusters themselves. The VCF documentation is explicit that Cluster API is the API layer responsible for cluster creation/configuration/management, making it the correct answer for provisioning VKS workload clusters (including those intended to run service mesh capabilities).

Question 9

Question Type: MultipleChoice

An administrator is deploying vSphere Kubernetes Service (VKS) to support containerized workloads across multiple regions. Each region hosts a dedicated Workload Domain with Supervisor instances deployed on vSphere Distributed Switch (VDS) networking. The organization's security policy requires that pod-to-pod and pod-to-service communications be fully observable and controllable at the Kubernetes layer, without introducing additional licensing or overlay complexity.

When deploying a Supervisor, which CNI should the administrator select as the default supported option?

Options:

- A- Antrea
- B- Calico
- C- Flannel
- D- Cilium

Answer:

A

Explanation:

VCF 9.0 explicitly documents that VKS supports two CNI options: Antrea and Calico, and that the system-defined default CNI is Antrea. This directly eliminates Flannel and Cilium as default supported options for VKS clusters on Supervisor in this context. VCF 9.0 also describes how a vSphere administrator can view or change this setting in the vSphere Client under Supervisor Management Configure Kubernetes Service Default CNI, further reinforcing that Antrea is the baseline/default choice.

From a policy perspective in the question, the requirement is Kubernetes-layer observability and control of pod communications "without additional licensing or overlay complexity." Antrea is presented in VCF 9.0 as the default CNI and is implemented using Open vSwitch, with networking and network policy capabilities provided at the Kubernetes layer for pods and services. Because it is the documented default (and supported) option for new VKS clusters, selecting Antrea best aligns with the "default supported option" requirement.

Question 10

Question Type: MultipleChoice

What is the purpose of a network policy in a Kubernetes cluster?

Options:

- A- Creates and operates workload clusters natively in vSphere Supervisor.
- B- Defines how pods communicate with each other and other network endpoints.
- C- Provides routing capability for inbound pod traffic.
- D- Allows external access to a port opened on each worker node through the Kubernetes network proxy.

Answer:

B

Explanation:

In VCF 9.0 VKS clusters, network policy is a core Kubernetes networking control implemented by the cluster's CNI (Antrea or Calico). The VCF documentation's "VKS Cluster Networking" table describes Network Policy as the feature that "controls what traffic is allowed to and from selected pods and network endpoints," and identifies Antrea or Calico as the providers for this capability. That definition precisely matches option B: it governs pod-to-pod and pod-to-external endpoint communication rules. This is different from ingress routing (which the same table describes separately as "Cluster ingress ... routing for inbound pod traffic"), so option C is not correct for "network policy." It is also different from NodePort behavior (external access via a port on each worker node through the Kubernetes network proxy), which is explicitly listed as "Service type: NodePort." Finally, creating/operating clusters natively in Supervisor is a broader lifecycle function (Cluster API/VKS API), not the definition of network policy. Therefore, Network Policy is the Kubernetes-layer mechanism to define and enforce allowed traffic flows.

Question 11

Question Type: MultipleChoice

What is the purpose of the VMware vSphere Kubernetes Service (VKS) Service Mesh?

Options:

- A- Provides service discovery across multiple clusters.
- B- Provides an infrastructure layer that makes communication between applications possible, structured, and observable.
- C- Provides dynamic application load balancing and autoscaling across multiple clusters and multiple sites.
- D- Provides a centralized, global routing table to simplify and optimize traffic management.

Answer:

B

Explanation:

A service mesh is an application communication layer that standardizes service-to-service traffic inside Kubernetes. Instead of each development team building custom logic for retries, timeouts, encryption, and telemetry, the mesh provides these capabilities consistently across workloads. This is typically done by inserting a data plane (often sidecar proxies or node-level proxies) that intercepts inbound and outbound traffic for each microservice, plus a control plane that distributes configuration and identity material.

The key outcomes align directly to option B: communication becomes possible (reliable connectivity patterns), structured (consistent routing rules, policies, and identity), and observable (metrics, logs, and distributed tracing for east-west traffic). A service mesh commonly adds controls such as mTLS encryption, fine-grained traffic policy (allow/deny, rate limits, circuit breaking), and progressive delivery patterns (canary/blue-green) without changing application code.

By contrast, service discovery (A) is usually a built-in Kubernetes function, load balancing/autoscaling across sites (C) is not the primary definition of a service mesh, and a single centralized global routing table (D) is not how meshes are typically described or implemented.

To Get Premium Files for 3V0-24.25 Visit

<https://www.p2pexams.com/products/3v0-24.25>

For More Free Questions Visit

<https://www.p2pexams.com/vmware/pdf/3v0-24.25>

20%
DISCOUNT

P2P
exams