TECHNICAL WHITE PAPER: **October 2026**



Splunk on vSphere Kubernetes Service on VMware Cloud Foundation

Reference Architecture

Table of Contents

Executive Summary *3*
vSphere Kubernetes Service 3
Splunk 4
Solution Architecture *4*
VKS Cluster Configuration 6
Kubernetes Packages 7
Solution Validation *8*
Install Splunk OpenTelemetry Collector Helm Chart on Kubernetes 8
1. Add the chart repository 8
2. Create the namespace 8
3. Set the cluster values 8
4. Create the access-token Secret 9
5. Check the rendered resources 9
6. Install 9
Validate Splunk OpenTelemetry for Kubernetes 9
Conclusion *10*

Executive Summary

Running the Splunk OpenTelemetry Collector on vSphere Kubernetes Service gives organizations a consistent, repeatable way to extend Splunk Observability Cloud to Kubernetes workloads on VMware Cloud Foundation. VKS provides the CNCF-writecertified Kubernetes runtime, automated cluster lifecycle management, and integrated vSAN storage, while the Splunk OpenTelemetry Collector delivers real-time visibility into the nodes and workloads running in each cluster.

The deployment validated in this paper follows practices suited to production: the Helm chart version is pinned, Pod Security Admission labels are applied before workloads are submitted, the access token is stored in a Kubernetes Secret rather than in values files, and rendered manifests are checked against the cluster with a server-side dry run before installation. Teams can use the example manifests and Helm values in the accompanying repository as a starting point and adapt them to their own clusters and GitOps workflows.

vSphere Kubernetes Service

vSphere Kubernetes Service (VKS) is the Kubernetes runtime built directly into VMware Cloud Foundation (VCF). With CNCF certified Kubernetes, VKS enables platform engineers to deploy and manage Kubernetes clusters while leveraging a comprehensive set of cloud services in VCF. Cloud admins benefit from the support for N-2 Kubernetes versions, enterprise grade security, and simplified lifecycle management for modern apps adoption.

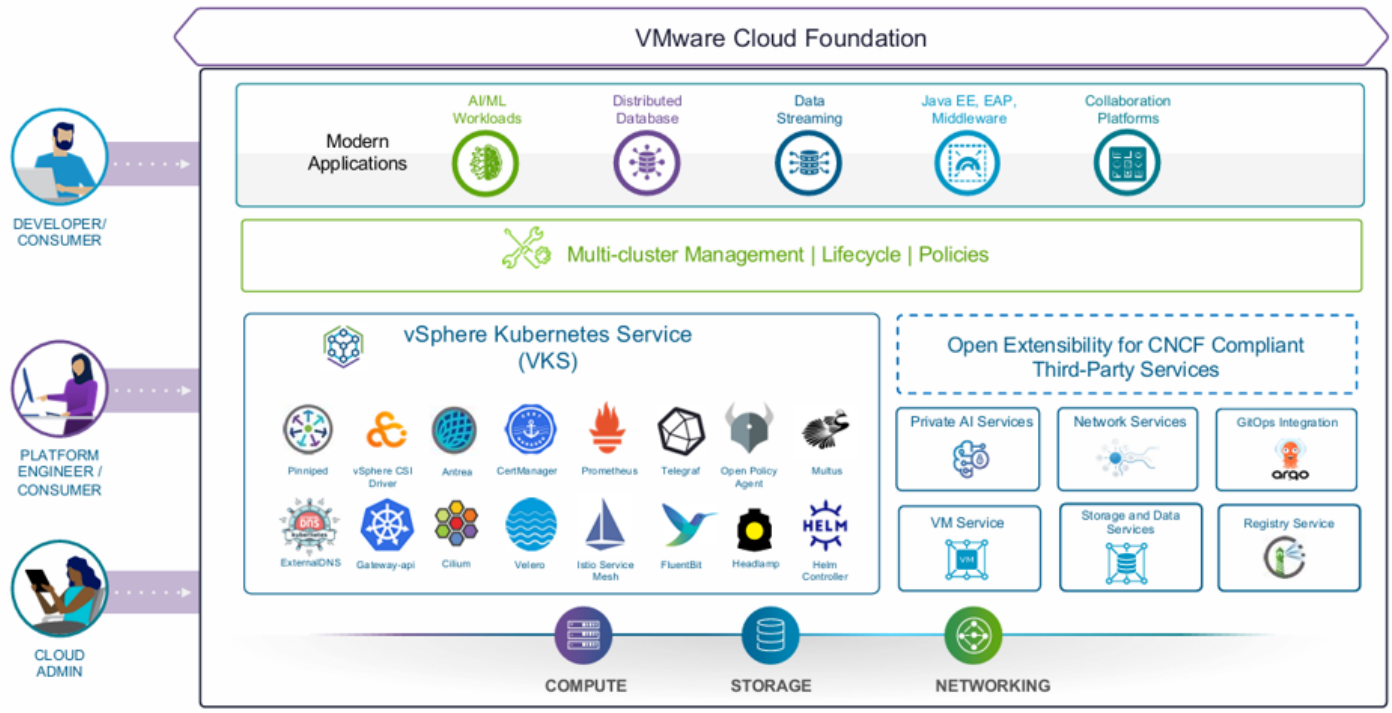


Figure 1: VKS on VCF Ecosystem

Benefits of using VKS with modern applications:

Lower TCO: With VKS organizations can reduce silos, leverage existing tools and skill sets without having to retrain staff and/or change existing processes. Utilizing unified lifecycle management across infrastructure components to stay up to date with the most recent patches and minimizing security risks.

Operational Simplicity: VKS is engineered for unparalleled operational simplicity, leveraging the familiarity of existing vSphere tools, skills, and workflows. This design philosophy significantly reduces the learning curve for IT teams and streamlines management processes. With VKS, organizations benefit from automated cluster provisioning, which accelerates deployment times and minimizes manual configuration errors. Furthermore, its robust capabilities extend to automated upgrades and comprehensive lifecycle management. This integrated

approach ensures consistent operations, reduces overhead, and frees up valuable resources to focus on innovation rather than infrastructure maintenance.

Run and Manage Kubernetes at Scale: Effortlessly deploy and manage Kubernetes clusters at scale, leveraging a built-in, Cloud Native Computing Foundation (CNCF) certified Kubernetes distribution. VKS provides fully automated lifecycle management, streamlining operations from initial setup to ongoing maintenance and upgrades. This comprehensive approach ensures that organizations can harness the power of Kubernetes for their containerized applications with unparalleled efficiency and reliability, without the complexities typically associated with large-scale Kubernetes deployments.

Splunk

Splunk Observability Cloud is a full-fidelity monitoring and troubleshooting solution across infrastructure, applications, and user interfaces, in real time and at any scale.

This white paper presents a validated approach for deploying the Splunk OpenTelemetry Collector on vSphere Kubernetes Service (VKS) clusters running on VMware Cloud Foundation (VCF). It describes the reference cluster architecture and Kubernetes packages, then walks through installing and validating the Splunk OpenTelemetry Collector Helm chart to send cluster and workload telemetry to Splunk Observability Cloud. Using this guidance and the example manifests and Helm values in the accompanying GitHub repository, platform teams can bring full-stack observability to modern applications on VCF while keeping the vSphere tools, skills, and workflows they already rely on.

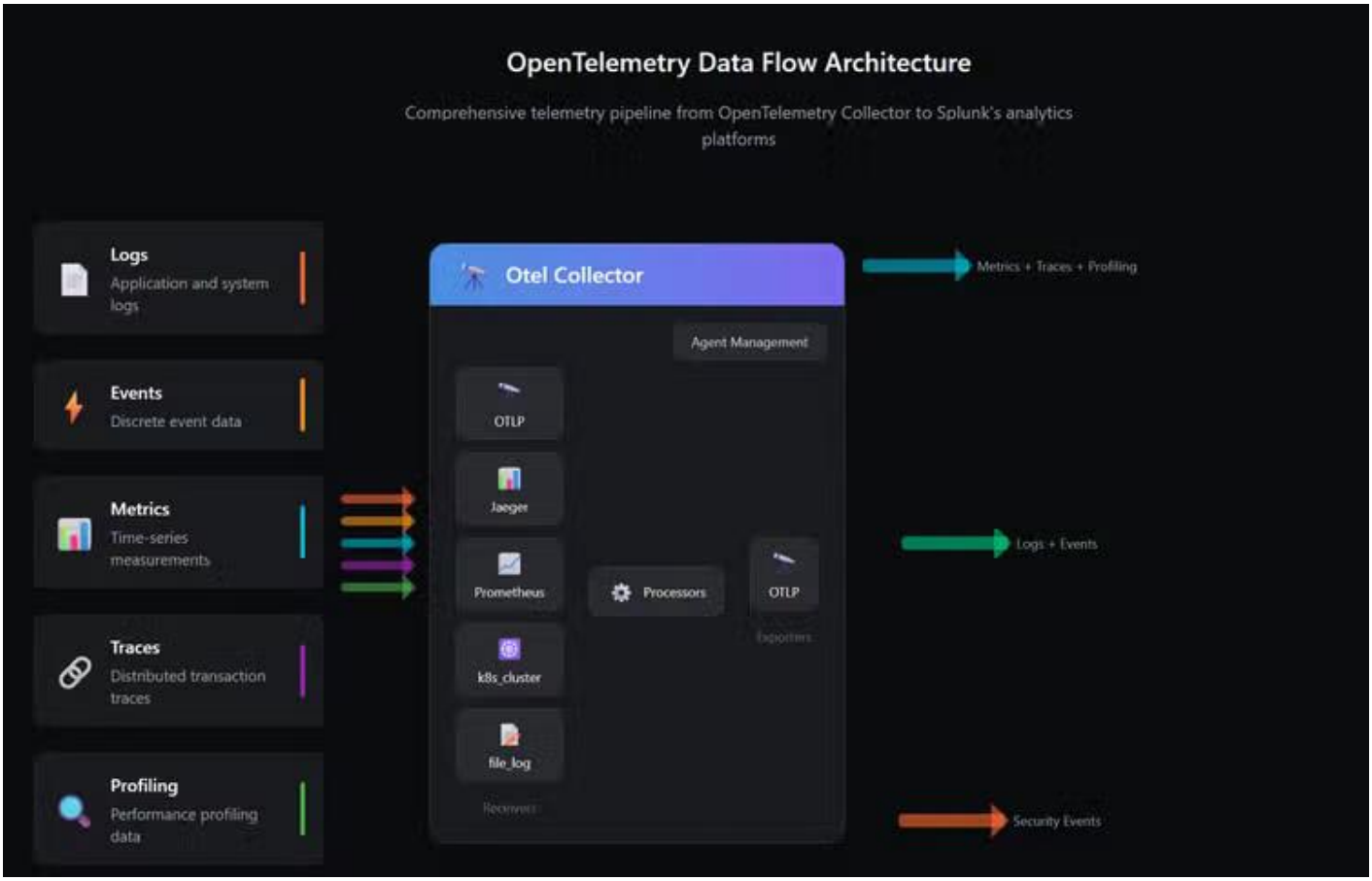


Figure 2: Workflow

Solution Architecture

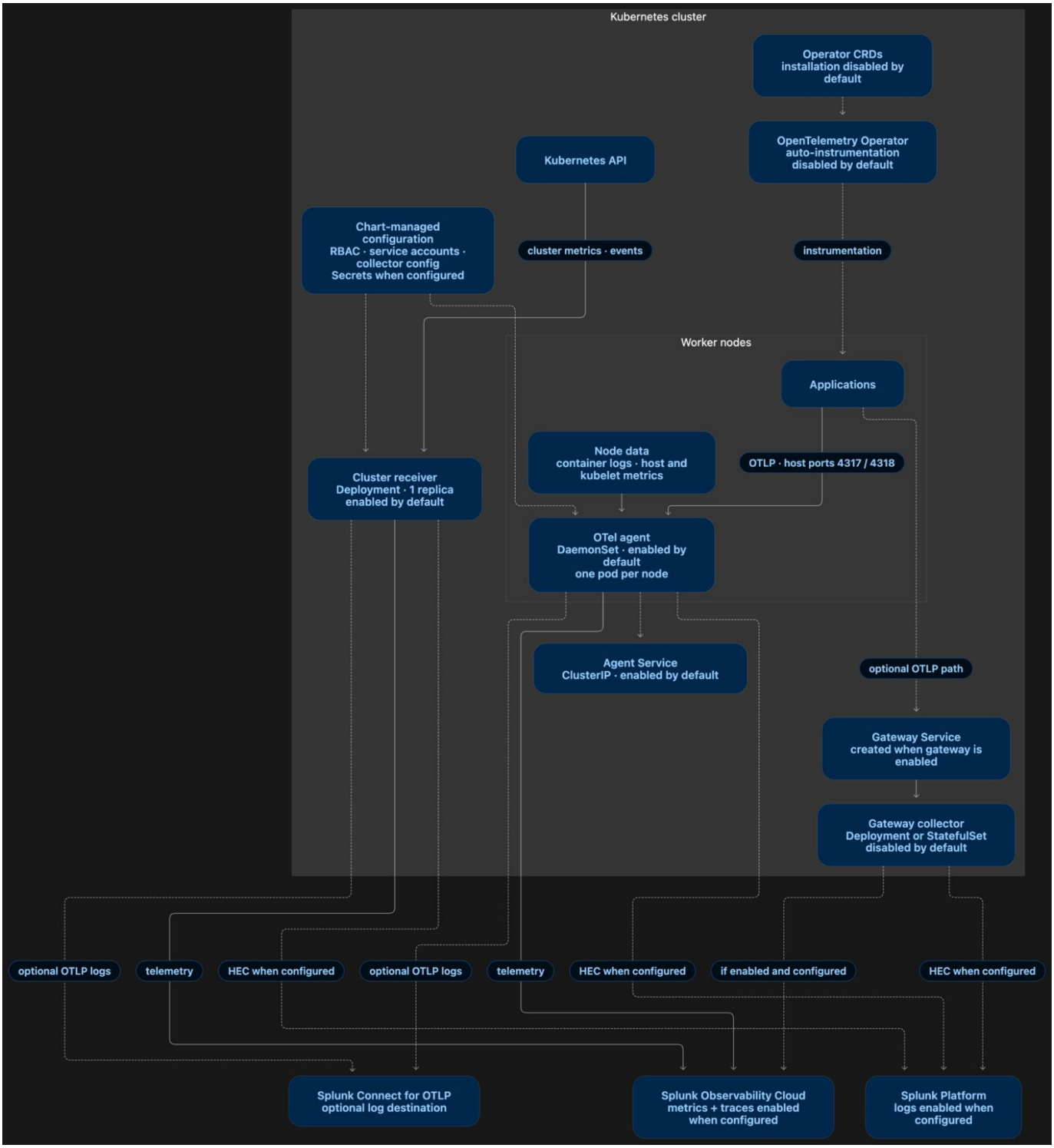


Figure 3: Splunk OpenTelemetry Collector Helm Chart

At its core, the Supervisor control plane provides essential services, such the Harbor container registry, the Contour ingress controller, and the Argo CD GitOps utility. For more information on vSphere Supervisor services and installation, visit: https://techdocs.broadcom.com/us/en/vmware-cis/vcf/vsphere-supervisor-services-and-standalone-components/latest.html

This control plane is responsible for the automated deployment and lifecycle management of CNCF-compliant Kubernetes clusters via the vSphere Kubernetes Service (VKS) via Cluster API. The specific version of VKS used in this white paper is 3.6.2.

Table 1: Installed Supervisor Components

Component	Version	Notes
vSphere Kubernetes Service	3.6.2	Allows for CNCF K8s clusters

The example manifests, Helm values, and supporting configuration referenced throughout this paper are available in the accompanying repository: https://github.com/vmware/vks-validated-solutions/tree/main/Splunk

VKS Cluster Configuration

The cluster comprised of three control plane nodes and three worker nodes. Storage was provided by vSAN Express Storage Architecture (ESA), using the default RAID 5 storage policy. Transparently, the vSphere CSI driver exposed this policy to an adjacent Kubernetes storage class, **vsan-esa-default-policy-raid5**.

Control plane nodes were configured using the best-effort-medium VM class (2 vCPU, 8 GB RAM), while worker nodes used the guaranteed-xlarge VM class (4 vCPU, 32 GB RAM), which fully reserves its CPU and memory.

Table 2: Kubernetes Cluster Configuration Summary

Component	Configuration	Notes
Kubernetes Control Plane	3 Replicas — 2x vCPU 8GB RAM vSAN RAID 5	VM Class: “best-effort-medium” Storage Class: “vsan-esa-default-policy-raid5”
Kubernetes Worker Nodes	3 Replicas — 4x vCPU 32GB RAM vSAN RAID 5	VM Class: “guaranteed-xlarge” Storage Class: “vsan-esa-default-policy-raid5”
Kubernetes Release	v1.35.5	
OS Image	Ubuntu 24.04	Kernel version 6.8.0-90-generic

The VKS cluster configuration is defined declaratively in the YAML file below. As previously outlined, applying this manifest to the Supervisor triggers the creation of the cluster, which is then managed and reconciled by the Cluster API.

```
# vks-cluster.yaml

apiVersion: cluster.x-k8s.io/v1beta1
kind: Cluster
metadata:
  name: kubernetes-cluster-kmnr
  namespace: supervisor-namespace
spec:
  clusterNetwork:
    pods:
      cidrBlocks:
        - 192.168.156.0/20
    services:
      cidrBlocks:
```

```
      - 10.96.0.0/12
  serviceDomain: cluster.local
topology:
  class: builtin-generic-v3.6.0
  version: v1.35.5---vmware.1-vkr.1
  variables:
    - name: kubernetes
      value:
        certificateRotation:
          enabled: true
          renewalDaysBeforeExpiry: 90
    - name: vmClass
      value: best-effort-medium
    - name: storageClass
      value: vsan-esa-default-policy-raid5
  controlPlane:
    replicas: 3
    metadata:
      annotations:
        run.tanzu.vmware.com/resolve-os-image: os-name=ubuntu, os-version=24.04
  workers:
    machineDeployments:
      - class: node-pool
        name: kubernetes-cluster-kmnr-nodepool-nve9
        replicas: 3
        metadata:
          annotations:
            run.tanzu.vmware.com/resolve-os-image: os-name=ubuntu, os-version=24.04
        variables:
          overrides:
            - name: vmClass
              value: guaranteed-xlarge
            - name: volumes
              value:
                - name: vol-5be3
                  mountPath: /var/lib/containerd
                  storageClass: vsan-esa-default-policy-raid5
```

Kubernetes Packages

VKS clusters (via the VKr release) include a default set of core packages, such as Antrea for networking and Pinniped for authentication.

Table 3: Kubernetes Cluster Packages

Package	Version	Notes
Antrea	2.4.4	VKS core package
Gateway API	1.3.0	VKS core package
Guest-cluster auth-service	1.4.4	VKS core package
Metrics Server	0.8.0	VKS core package
Pinniped	0.41.0	VKS core package
Secretgen Controller	0.19.2	VKS core package
vSphere CPI	1.34.0	VKS core package

vSphere CSI	3.6.0	VKS core package
Cert-Manager	1.18.2	VKS add-on
Splunk OpenTelemetry Collector	0.163.0	Observability

Solution Validation

Install Splunk OpenTelemetry Collector Helm Chart on Kubernetes

1. Add the chart repository

```
helm repo add splunk-otel-collector-chart \
  https://signalfx.github.io/splunk-otel-collector-chart
```

Pin the chart version used by this example. Version 0. 163.0 was current when the example was prepared:

```
CHART_VERSION=0.163.0
```

The chart project warns that minor releases can include breaking changes, so change this version deliberately after reviewing the release notes and rendered manifests.

2. Create the namespace

Create a namespace.yaml file:

```
apiVersion: v1
kind: Namespace
metadata:
  name: splunk-otel
  labels:
    app.kubernetes.io/part-of: splunk-otel-collector
    pod-security.kubernetes.io/enforce: privileged
    pod-security.kubernetes.io/enforce-version: latest
    pod-security.kubernetes.io/audit: restricted
    pod-security.kubernetes.io/audit-version: latest
    pod-security.kubernetes.io/warn: restricted
    pod-security.kubernetes.io/warn-version: latest
```

Create the namespace:

```
kubectl apply -f namespace.yaml
```

Note: do not replace this with Helm's --create-namespace: the PSA labels must exist before the chart submits its workloads.

3. Set the cluster values

```
cp cluster-values.example.yaml cluster-values.yaml
```

Edit cluster-values.yaml:

- clusterName is required for VKS and should uniquely identify the cluster.
- splunkObservability.realm is the realm assigned to the Splunk organization.
- environment is a deployment lifecycle value such as test or production. It should not be set to vks; VKS is the platform, not the deployment environment.

Leave distribution unset. The Splunk chart has no vks distribution value and treats an unset value as Kubernetes/other.

4. Create the access-token Secret

The chart expects a pre-existing Secret key named splunk_observability_access_token.

```
read -rsp 'Splunk Observability access token: ' SPLUNK_ACCESS_TOKEN
printf '\n'

kubectl --namespace splunk-otel create secret generic \
  splunk-otel-credentials \
  --from-literal="splunk_observability_access_token=${SPLUNK_ACCESS_TOKEN}" \
  --dry-run=client \
  --output=yaml |
kubectl apply -f -

unset SPLUNK_ACCESS_TOKEN
```

The token is neither written to a values file nor passed to Helm. For a production GitOps workflow, create the same Secret key with the organization's normal secret manager.

5. Check the rendered resources

This checks both chart rendering and whether the current VKS identity can submit the resulting objects:

```
helm template splunk-otel-collector \
  splunk-otel-collector-chart/splunk-otel-collector \
  --version "${CHART_VERSION}" \
  --namespace splunk-otel \
  --values vks-values.yaml \
  --values cluster-values.yaml |
kubectl apply --dry-run=server -f -
```

6. Install

```
helm upgrade --install splunk-otel-collector \
  splunk-otel-collector-chart/splunk-otel-collector \
  --version "${CHART_VERSION}" \
  --namespace splunk-otel \
  --values vks-values.yaml \
  --values cluster-values.yaml \
  --values vks-loopback-control-plane-values.yaml \
  --atomic \
  --timeout 10m
```

There is no need to set `gateway.enabled=false`; false is already the chart default.

Validate Splunk OpenTelemetry for Kubernetes

```
helm status splunk-otel-collector --namespace splunk-otel
kubectl --namespace splunk-otel get daemonsets,deployments,pods
```

```
kubectl --namespace splunk-otel get events \
  --sort-by=.metadata.creationTimestamp
For collector startup or export errors:
kubectl --namespace splunk-otel logs \
  --selector app.kubernetes.io/instance=splunk-otel-collector \
  --all-containers \
  --prefix \
  --tail=100
```

Conclusion

Running the Splunk OpenTelemetry Collector on vSphere Kubernetes Service gives organizations a consistent, repeatable way to extend Splunk Observability Cloud to Kubernetes workloads on VMware Cloud Foundation. VKS provides the CNCF-certified Kubernetes runtime, automated cluster lifecycle management, and integrated vSAN storage, while the Splunk OpenTelemetry Collector delivers real-time visibility into the nodes and workloads running in each cluster.

The deployment validated in this paper follows practices suited to production: the Helm chart version is pinned, Pod Security Admission labels are applied before workloads are submitted, the access token is stored in a Kubernetes Secret rather than in values files, and rendered manifests are checked against the cluster with a server-side dry run before installation. Teams can use the example manifests and Helm values in the accompanying repository as a starting point and adapt them to their own clusters and GitOps workflows.



Copyright © 2026 Broadcom. All rights reserved.
The term “Broadcom” refers to Broadcom Inc. and/or its subsidiaries. For more information, go to www.broadcom.com. All trademarks, trade names, service marks, and logos referenced herein belong to their respective companies. Broadcom reserves the right to make changes without further notice to any products or data herein to improve reliability, function, or design. Information furnished by Broadcom is believed to be accurate and reliable. However, Broadcom does not assume any liability arising out of the application or use of this information, nor the application or use of any product or circuit described herein, neither does it convey any license under its patent rights nor the rights of others.
Item No: vmw-bc-wp-tech-uslet-word-2026; Oct-26