Amazon Elastic Kubernetes Service Anywhere on Dell PowerFlex

May 2022

H19148

White Paper

Abstract

This white paper describes the use of Amazon Elastic Kubernetes Service Anywhere on the Dell PowerFlex platform to deliver a seamless and automated operations experience for PowerFlex infrastructure across cloud-native and traditional workloads. This white paper also describes using VMware vSphere to enable connectivity and portability across Amazon Web Services public cloud environments.

Dell Technologies Solutions

PowerFlex Engineering Validated

Scan for audio version

DELL Technologies

Copyright

The information in this publication is provided as is. Dell Inc. makes no representations or warranties of any kind with respect to the information in this publication, and specifically disclaims implied warranties of merchantability or fitness for a particular purpose.

Use, copying, and distribution of any software described in this publication requires an applicable software license.

Copyright © 2022 Dell Inc. or its subsidiaries. Published in the USA 05/22 White Paper H19148.

Dell Inc. believes the information in this document is accurate as of its publication date. The information is subject to change without notice.

Note: This document may contain language from third-party content that is not under Dell's control and is not consistent with Dell's current guidelines for Dell's own content. When such third-party content is updated by the relevant third parties, this guide will be revised accordingly.

Contents

Introduction 4
Dell PowerFlex product overview 6
Solution architecture 9
Installing an Amazon EKS Anywhere cluster on Dell PowerFlex 11
Creating an Amazon EKS Anywhere cluster 18
Validating the cluster deployment 23
Installing the PowerFlex CSI driver 24
Deployment tests and validation 27
Conclusion 37
Configuration details 37
References 39

Introduction

Overview

This white paper describes the integration of PowerFlex CSI driver to provision persistent PowerFlex storage for containerized applications that are deployed on Amazon Elastic Kubernetes Service (EKS) Anywhere. It validates many of the common tasks outlined in the Amazon EKS Anywhere cluster management documentation.

Executive summary

Amazon EKS Anywhere is a new deployment option for Amazon EKS. It enables customers to easily create and operate Kubernetes clusters on-premises to allow easy connectivity and portability to Amazon Web Services (AWS) environments. PowerFlex helps customers deliver a flexible deployment solution that scales as needs change over time with smooth, painless, node-by-node expandability, inclusive of compute and storage, in a unified fabric architecture.

Deploying Amazon EKS Anywhere on Dell Technologies infrastructure streamlines application development and delivery by allowing organizations to easily create and manage on-premises Kubernetes clusters.

The combination of automated Kubernetes cluster management with intelligent, automated infrastructure quickly brings organizations to the next stop in their IT journey. They can provide infrastructure as code and empower their DevOps teams to be the innovation engine for their organization.

The PowerFlex family offers key value propositions for traditional and cloud-native production workloads, deployment flexibility, linear scalability, predictable high performance, and enterprise-grade resilience.

We value your feedback

Dell Technologies and the authors of this document welcome your feedback on the solution and the solution documentation. Contact the Dell Technologies Solutions team by email.

Author: Syed Abrar

Contributors: Anthony Foster

Note: For links to additional documentation for this solution, see the PowerFlex section on the Dell Technologies Info Hub.

Objective

PowerFlex is a fully integrated, preconfigured, and tested software-defined infrastructure system that is an ideal platform for Amazon EKS Anywhere. Amazon EKS Anywhere helps customers automate cluster management, reduce support costs, and make multi-cluster management easier (regardless of where the cluster is running), while driving operational simplicity through Kubernetes orchestration.

Audience

This white paper is intended for sales engineers, field consultants, IT administrators, customers, and anyone that is interested in configuring and deploying a Kubernetes

cluster with the PowerFlex CSI driver to dynamically provision persistent volumes (PV) in an Amazon EKS Anywhere Kubernetes cluster.

The audience of this white paper must have a working knowledge of containers, Kubernetes, VMware vSphere, PowerFlex, and Amazon EKS Anywhere.

Terminology

Table 1. Terminology

Term	Definition
AGR	Aggregate
CA	Certificate Authority
CSI	Container Storage Interface
CRD	Custom Resource Definition
EKS A	Elastic Kubernetes Service Anywhere
ITOM	IT Operations Management
LCM	Life Cycle Management
MDM	Meta Data Manager
PV	Persistent Volume
PVC	Persistent Volume Claim
SDC	Storage Data Client for PowerFlex
SDS	Storage Data Server for PowerFlex
SLA	Service Level Agreement
SSD	Solid-State Disk for PowerFlex
TLS	Transport Layer Security
ToR	Top of Rack
VLAN	Virtual Local Area Network
VM	Virtual Machine

Dell PowerFlex product overview

Dell PowerFlex family

Dell PowerFlex is a software-defined infrastructure platform that reduces operational and infrastructure complexity. It empowers organizations to move faster by delivering flexibility, elasticity, and simplicity with extraordinary predictable performance for mission-critical workloads, while also providing resiliency at scale. The Dell PowerFlex family provides a foundation that combines compute and high-performance storage resources in a managed, unified fabric.

Dell PowerFlex delivers transformational value

- Delivers stringent SLAs effortlessly

  Dell PowerFlex uses software to unlock the full potential of rapid advances in industry-standard hardware and deliver extreme SLA outcomes. It aggregates resources across a broad set of nodes, unlocking massive input, output, and throughput performance while minimizing latency. Its self-balancing architecture eliminates any hotspots and ensures consistency and simplicity over time. You can scale the system while linearly scaling performance from a minimum of four nodes to thousands of nodes, on-demand and without disruption. With its self-healing architecture, Dell PowerFlex can handle outages, upgrades, and maintenance without downtime, resulting in 99.9999 percent availability.

- Flexible and dynamic architecture

  Dell PowerFlex delivers transformational agility that enables organizations to rapidly respond to changing business needs. It offers flexibility to mix and match storage, compute, and hyperconverged nodes in a dynamic deployment, allowing you to scale storage and compute resources together or independently, one node at a time.

- Shared platform for heterogeneous workloads

  Dell PowerFlex supports a broad range of operating environments simultaneously such as bare metal operating systems, hypervisors, and container platforms with a unified underlying infrastructure platform and management. You can also support heterogeneous workloads with varying requirements on a flexible shared infrastructure platform and modernize your application architectures on your schedule.

- Extensive automation for predictability and simpler workflows

  Dell PowerFlex offers full-stack IT Operations Management (ITOM) and Life Cycle Management (LCM) with Dell PowerFlex Manager. It provides extensive automation capabilities with Dell PowerFlex Manager REST APIs and custom Ansible modules to integrate with your infrastructure, application, and DevOps workflows. Dell PowerFlex Manager enables automated deployments and expansions with minimal time for the IT team, letting them focus on other strategic initiatives.

- Broad ecosystem of workload solutions

  Dell PowerFlex is designed to deliver superior outcomes at any scale for the most demanding mission-critical environments. It is optimized for a wide range of validated workload solutions ranging from traditional relational databases and modern cloud-native NoSQL databases to throughput-intensive analytics workloads.

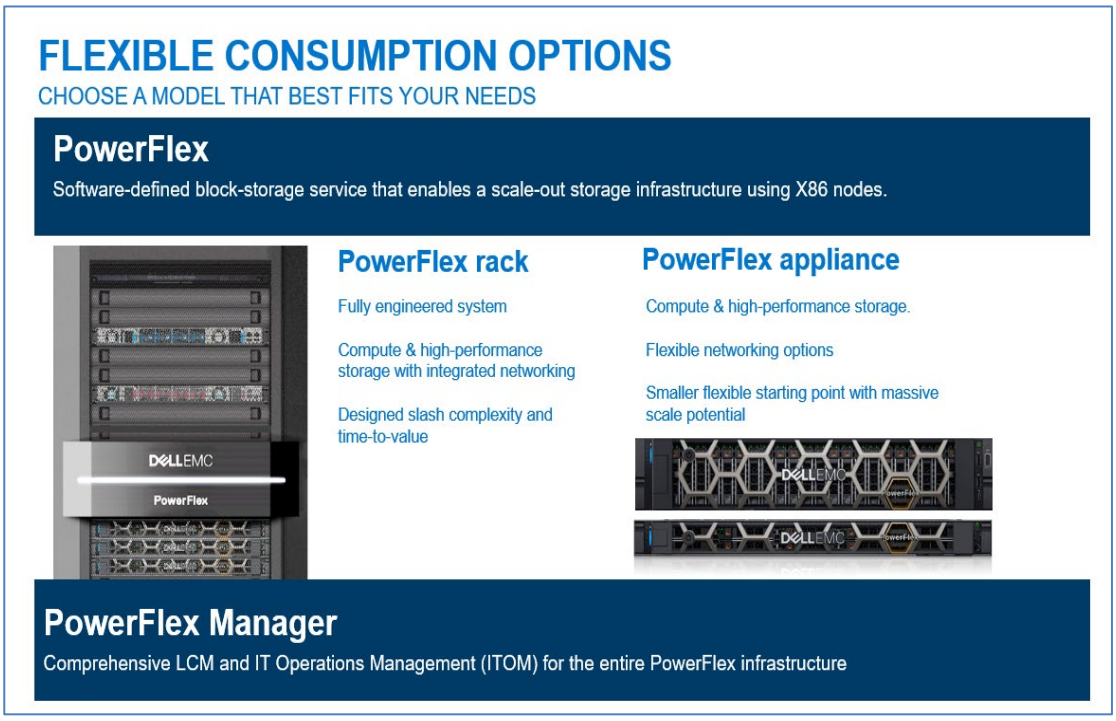


Figure 1. PowerFlex family

Dell PowerFlex software

The PowerFlex software provides software-defined storage services. It delivers high performance, highly resilient block storage service that can scale to thousands of nodes.

PowerFlex Manager simplifies the infrastructure management and orchestration with comprehensive ITOM and LCM capabilities that span compute and storage infrastructure, from BIOS and firmware to nodes, software, and networking. It also enables flexible APIs and extensive automation.

Dell PowerFlex consumption options

Dell PowerFlex is available with multiple consumption options to help you meet your project and data center requirements. Dell PowerFlex appliance and Dell PowerFlex rack provide customers with the flexibility to choose a deployment option to meet their exact requirements.

Dell PowerFlex rack

Dell PowerFlex rack is a fully engineered system, with integrated networking that enables customers to simplify deployments and accelerate time to value. The preconfigured and flexible network options, ingrained security controls, end-to-end life cycle management, secure call home, and single call support are some of the many key features that help customers streamline day-2 operation, deliver agility and resilience, and maintain a security posture.

Dell PowerFlex appliance

Dell PowerFlex appliance offers customers a smaller configuration of four nodes, while enabling them to use their existing network infrastructure.

With Dell PowerFlex, customers deploy to match their initial needs and easily expand with massive scale potential, without having to compromise on performance and resiliency.

Flexible consumption-based billing options

Dell PowerFlex is available through APEX custom solutions by the APEX Flex on Demand and APEX Datacenter Utility for customers looking to adopt consumption-based OpEx models.

APEX Flex on Demand

APEX Flex on Demand allows you to pay for technology as you use it and provides immediate access to buffer capacity. Your payment adjusts to match your usage.

APEX Datacenter Utility

APEX Datacenter Utility provides the leading product portfolio from Dell Technologies that is coupled with your choice of professional services and support to manage your data center and its operations. A single invoice provides monthly payments that are based on a predictable rate and vary based on your usage.

Solution architecture

This section describes how Amazon EKS Anywhere is deployed on a Dell PowerFlex HCI cluster. You can deploy this solution on the Dell PowerFlex family products.

Note: The solution that is described in this section is validated in the engineering lab with the specific hardware specification shown in Figure 2.

The Dell PowerFlex HCI cluster includes four physical servers (each server includes two Intel Xeon Gold 6248R 24-core processors), 192 GB RAM, and five 1.5 TB SSDs. ESXi 7.0.2 is installed on each of the four nodes, which are running PowerFlex SDS and SDC components to provide compute and storage resources. A single protection domain is provisioned from the SSDs on these four SDS systems. A single storage pool and persistent storage volumes are created and assigned to the Kubernetes pods with CSI driver.

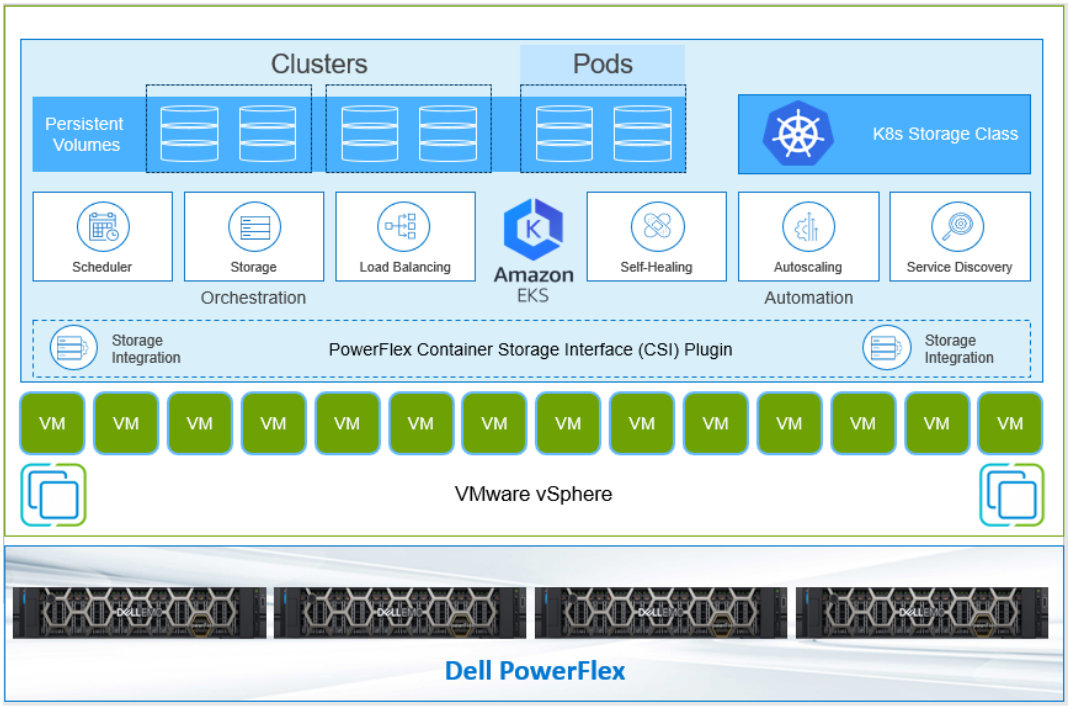


Figure 2. Logical architecture of the Amazon EKS Anywhere environment on Dell PowerFlex

Pre-requisites to deploy an Amazon EKS Anywhere service include:

- A PowerFlex environment running VMware vSphere 7.0 or later.
- Sufficient capacity to deploy eight to ten Amazon EKS VMs.
- A network in the vSphere workload cluster with a DHCP service. The workload VMs must be connected to this network.

- Amazon EKS administrative VMs require Internet connectivity to download manifests, OVAs, and the Amazon EKS distribution.

Initial deployments can start with four PowerFlex nodes and scale to meet the expansion requirements of storage, compute, or both for scalability.

After the Amazon EKS admin VM is deployed, a command is issued that deploys the workload clusters and creates associated CRD instances on the workload cluster. A single instance is prepared and with some integrated scripting, the system can orchestrate the complex deployment. This scripting greatly simplifies the process when compared to a traditional Kubernetes deployment.

When the initial deployment is completed, Amazon provides a test workload that you can use to validate the environment.

The validated design adds the PowerFlex software-defined storage together with the CSI plug-in to a typical Amazon EKS Anywhere deployment. This design makes it possible to present PowerFlex storage capabilities directly to Amazon EKS. With the CSI plug-in, it is possible to extend volumes through Amazon EKS, as well as snapshot volumes.

This design allows IT departments to move toward developer-oriented processes. Developers can work with storage natively. This architecture greatly reduces administrative efforts and saves time. This architecture also eliminates the need to learn different storage management tools and operational processes.

Beyond storage control in Amazon EKS Anywhere, the results of these operations can be viewed in the PowerFlex management interface. These operations provide an end-to-end view of the environment and allow traditional IT administrators to use familiar tools to manage and monitor their environment. This approach makes it easy for an entire IT organization to move towards a more developer-centric environment.

Installing an Amazon EKS Anywhere cluster on Dell PowerFlex

Overview

This section describes the steps to install Amazon EKS Anywhere cluster on Ubuntu and to configure the PowerFlex CSI driver on a PowerFlex cluster.

Requirements

The following table describes the prerequisites for building an EKS Anywhere cluster:

Table 2. Requirements for Amazon EKS Anywhere cluster on Dell PowerFlex family

Name	Description	Reference
VMware vSphere 7+	vSphere 7+ environment running vCenter	https://docs.vmware.com/en/VMware-vSphere/index.html
Ubuntu	Admin VM All the operations and management of Amazon EKS Anywhere is managed from the admin VM.	
GOVC	GOVC for VMware	https://github.com/vmware/govmomi/tree/master/govc
Eksctl	Eksctl installed on Ubuntu	https://github.com/weaveworks/eksctl/releases/latest
Eksctl -Anywhere	Install Eksctl -Anywhere installed on Ubuntu	brew install aws/tap/eks-anywhere
Docker CE	Docker CE Installed on Ubuntu	https://docs.docker.com/engine/install/ubuntu/
Kubectl	Kubectl to interact with Kubernetes cluster.	https://kubernetes.io/docs/tasks/tools/install-kubectl/
Ubuntu	Ensure that the nodes are accessed using SSH and the required ports must be opened before the cluster installation.	
Dell PowerFlex CSI	Dell PowerFlex CSI is used from the Workstation VM.	https://github.com/dell/csi-powerflex

Set up Ubuntu administrative machine

Run the following commands from the admin VM to set up the Ubuntu administrative machine:

```
sudo apt update

sudo apt install -y docker.io

sudo usermod -a -G docker $USER

wget https://distro.eks.amazonaws.com/kubernetes-1-19/releases/4/artifacts/kubernetes/v1.19.8/bin/linux/amd64/kubectl

mkdir -p $HOME/bin

chmod +x kubectl

mv kubectl $HOME/bin/

echo 'export PATH=$PATH:$HOME/bin' >> ~/.bashrc
```

Log out and log in back into the system to obtain the correct group permissions and path.

Install Amazon EKS Anywhere CLI tool

To install the CLI tool, run the `eks-a` command, extract the prebuilt binaries from the provided archive. Provide the eks-a binary run permission and move it to the `$HOME/bin` folder.

```
curl "https://github.com/weaveworks/eksctl/releases/latest/download/eksctl_$(uname -s)_amd64.tar.gz" \

    --silent --location \

    | tar xz -C /tmp

sudo mv /tmp/eksctl /usr/local/bin/

Copy

Install the eksctl-anywhere plugin.

chmod +x ./eks-a

mv ./eks-a $HOME/bin/
```

Import an OVA/OVF template to vSphere

Amazon EKS Anywhere deployment is provided as an OVA file. Download the OVA from the AWS website and then follow the steps to deploy it.

vCenter OVF template deployment steps

1. Right-click your data center and select **Deploy OVF Template.**

   The following figure shows the vSphere Client for the environment:

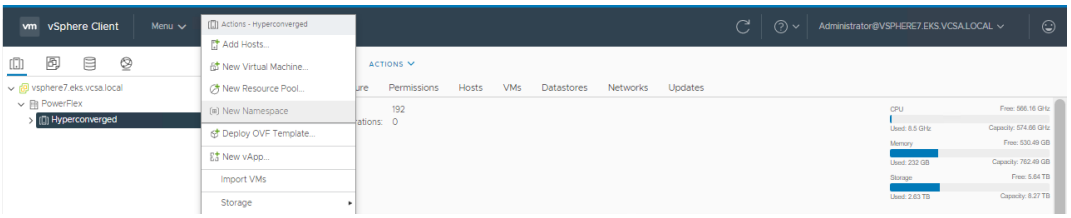


Figure 3. vSphere OVF template

2. Select an OVF template using the URL or by selecting a local OVF file and clicking **Next,** as shown in the following figure. If you cannot select an OVF template using the URL, download the file and use the Local file option.

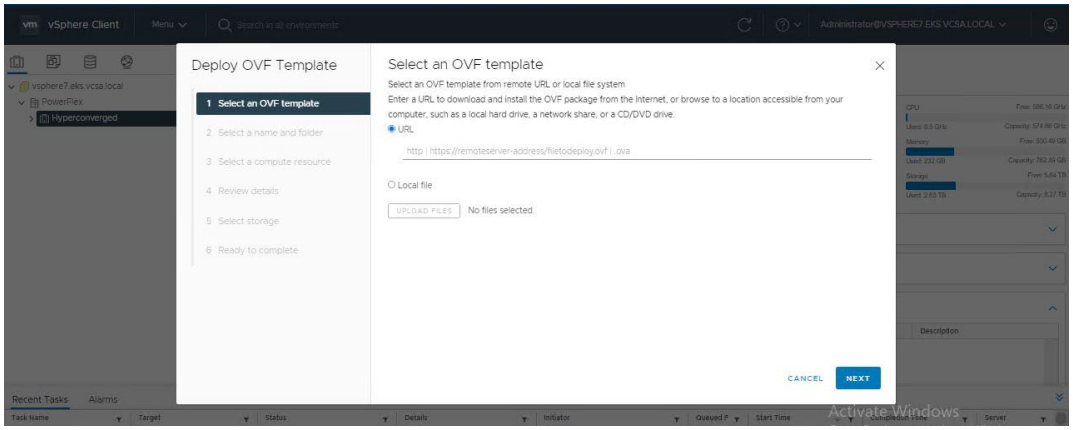


Figure 4. Downloading OVF template

3. Select a folder in which you want to deploy your OVF package and click **Next,** as shown in the following figure:

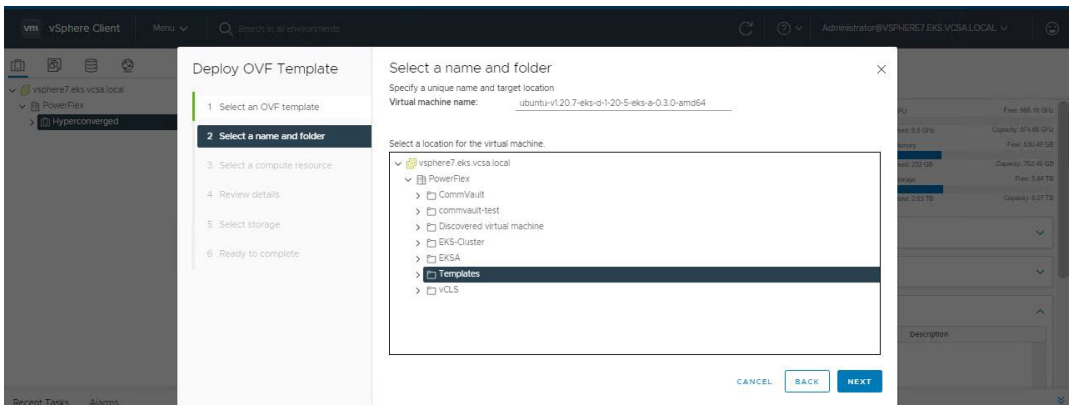


Figure 5. Deploying OVF template

Note: You cannot have two OVF templates with the same name in one directory. For workload VM templates, leave the Kubernetes version in the template name for reference. A workload VM template supports at least one prior Kubernetes major version.

4. Select a compute resource.
5. Review the details and click **Next.**
6. Accept the agreement and click **Next**.

7. Select the appropriate storage and click **Next**.
8. Select the destination network, click **Next,** and then click **Finish.**
9. Take a snapshot of the VM, as shown in the following figure. Right-click the imported VM and select **Snapshots** > **Take snapshot**.

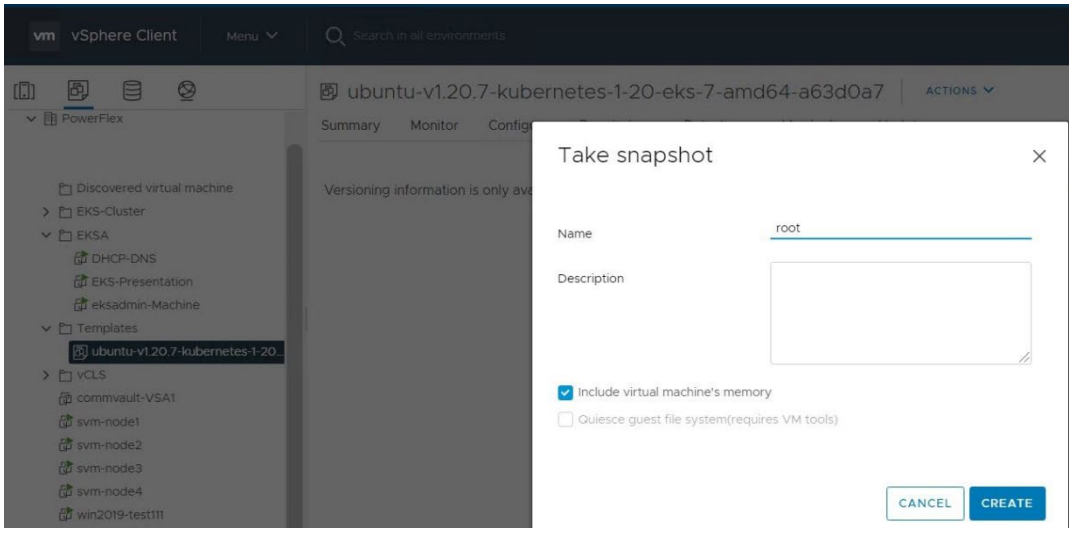


Figure 6. Taking a snapshot

10. Name your snapshot and click **Create.**

    Snapshots for the imported VM are displayed under the Snapshots tab for the VM, as shown in the following figure:

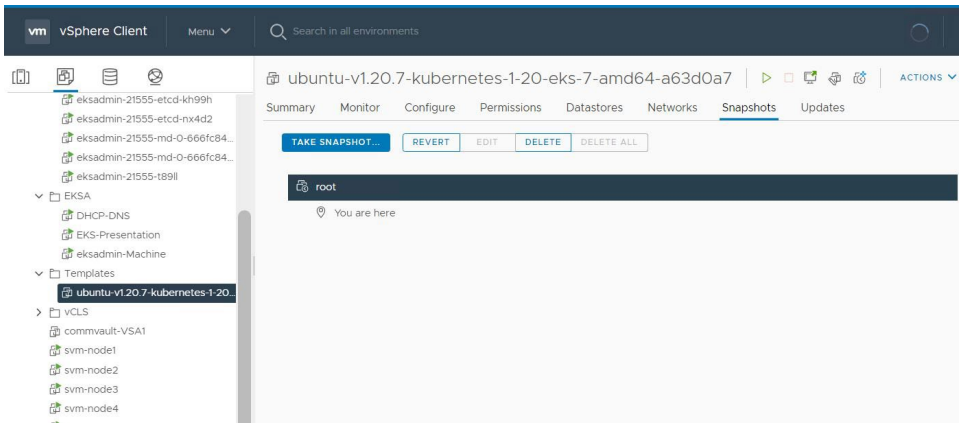


Figure 7. Managing a snapshot

11. Right-click the imported VM and select **Template > Convert to Template**, as shown in the following figure:

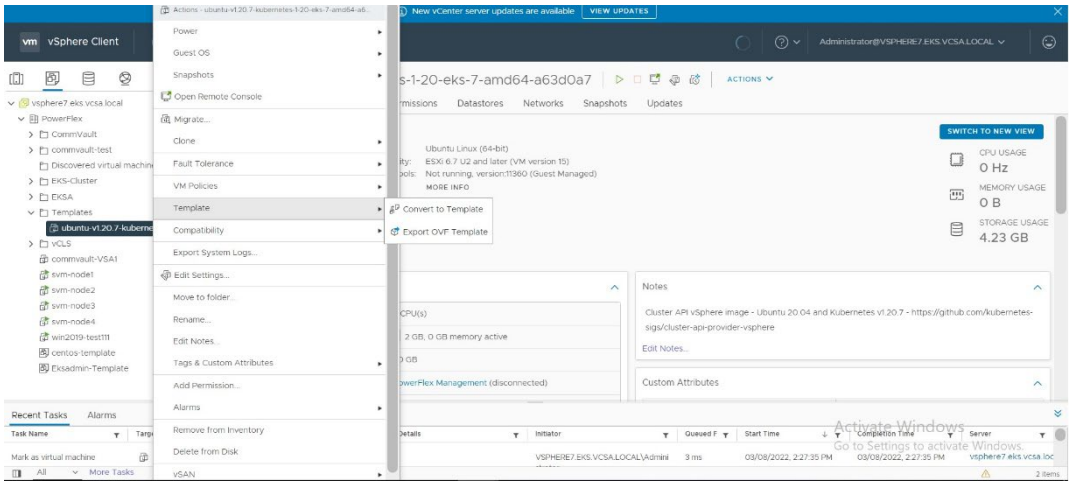


Figure 8. Converting to template

Tag the OVA

You need to create tag categories and associate them with the OVA templates.

Tag to indicate operating system Family

1. Select the newly created template and go to **Summary** > **Tags**, as shown in the following figure.

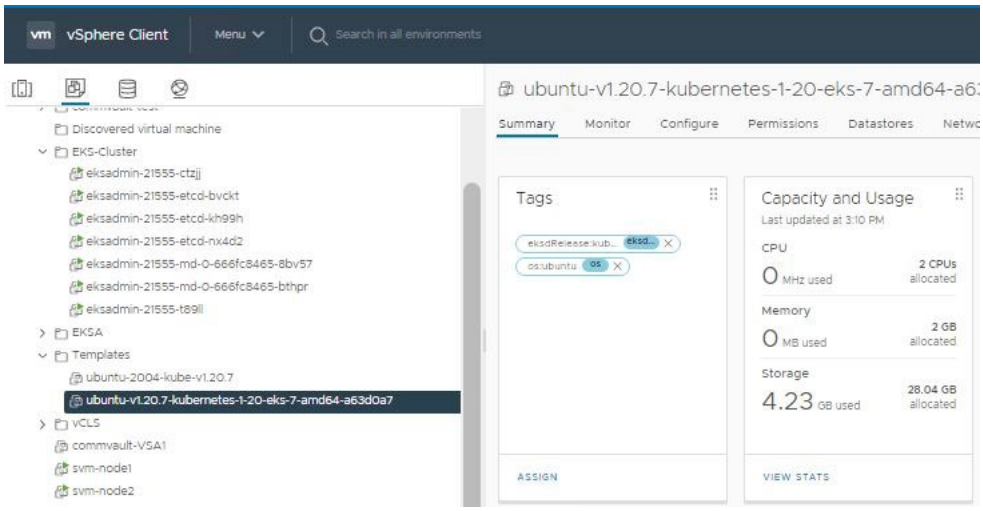


Figure 9. Selecting the new template

2. Click **Assign** and then click **Add Tag** to create a tag and attach it.

   Name the tag: os:ubuntu as shown in the following figure.

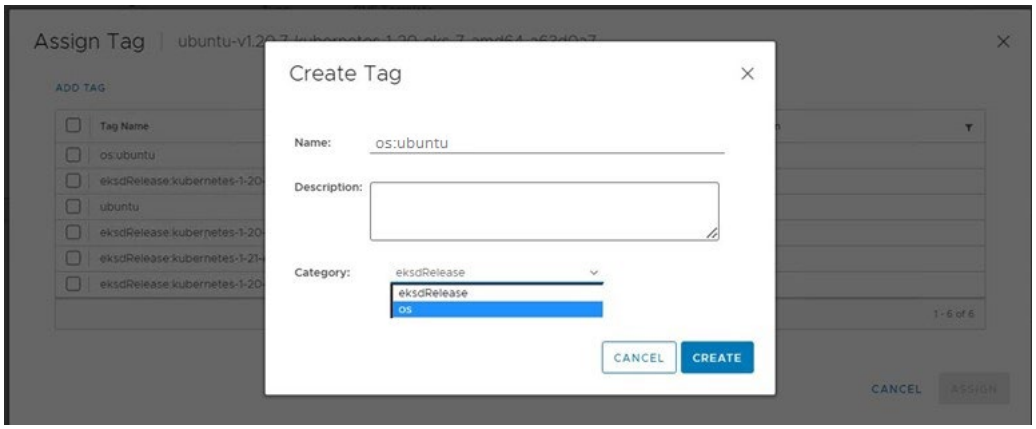


Figure 10. Creating a tag

The following figure shows the Assign Tag window:

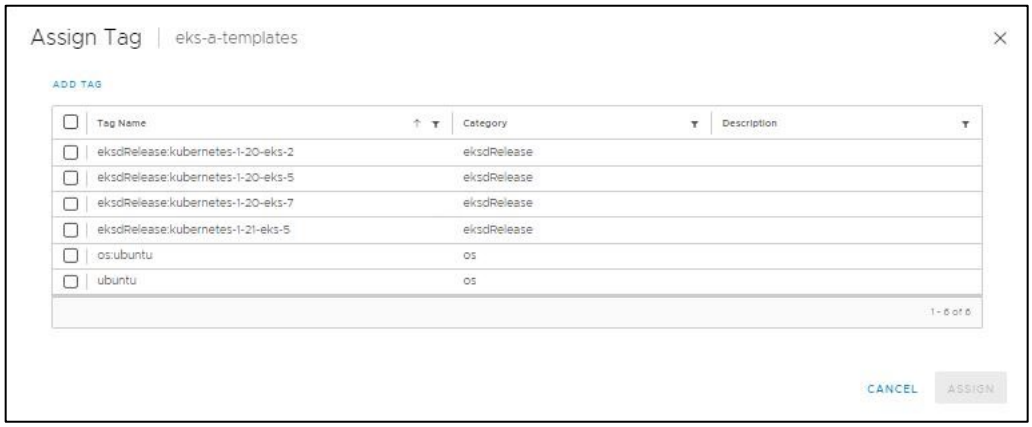


Figure 11. Assigning a tag

Tag to indicate eksd release

1. Select the new template and go to **Summary** > **Tags**, as shown in the following figure:

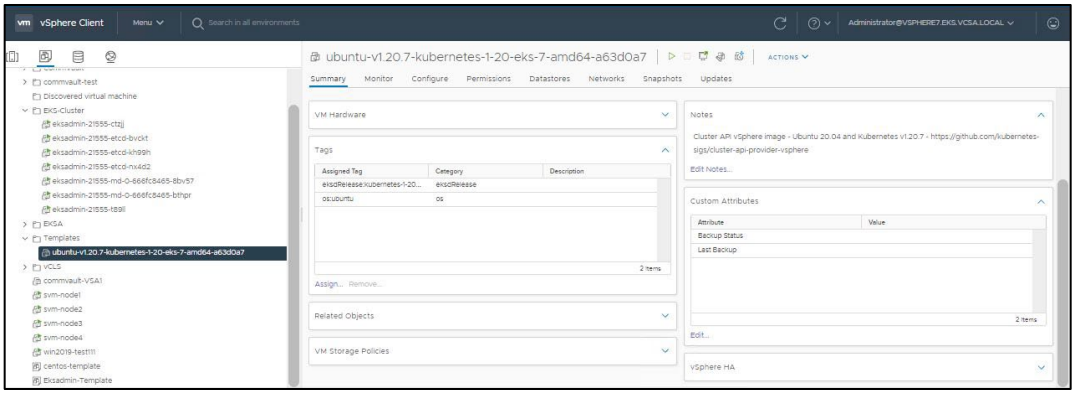


Figure 12. Selecting the new template

2. Click **Assign** and then click **Add Tag** to create a new tag, as shown in the following figure.

If you are adding an eksdRelease tag for the first time, create the category first. Click **Create New Category** and name it eksdRelease.

The following figure shows the Create Tag window where you can create a new category:



Figure 13. Creating a tag

The following figure shows the **Assign Tag** window:

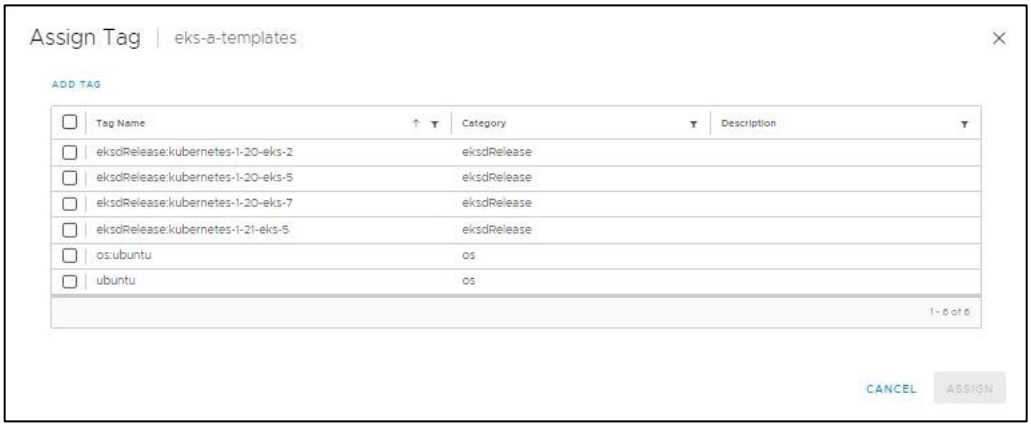


Figure 14. Assigning a tag

3. Name the tag `eksdRelease`, for example, `eksdRelease:kubernetes-1-20-eks-5` for the 1.20 ova.

Note: If the tags are not applied as shown, eks -a template validation fails and CLI operations stop.

GOVC

GOVC is an open source command-line utility for performing administrative actions on a VMware vCenter or vSphere environment. The CLI is an alternative to the UI and well suited for automation tasks.

To install GOVC on the _admin vm_, see Steps to Install GOVC.

After installation, verify that you can connect to the vCenter with GOVC, as shown in the following figure:



Figure 15. GOVC status

Creating an Amazon EKS Anywhere cluster

Overview

Use the vSphere details and EKS-A cluster information to create a cluster configuration.

Use the `eks-a` command to create your cluster based on the provided configuration from your admin VM.

Cluster creation

1. Edit the `$CLUSTER_NAME.yaml` file to enter the values for your environment.

   Cluster names must start with an upper or lowercase letter and can contain only letters, numbers, or dashes, as shown in the following example:

```
# Name your cluster or use this for a random name

export CLUSTER_NAME="${USER}-${RANDOM}"

eks-a generate clusterconfig $CLUSTER_NAME -p vsphere >
$CLUSTER_NAME.yaml
```

   See the following sample YAML file:

```
apiVersion: anywhere.eks.amazonaws.com/v1alpha1
kind: Cluster
metadata:
  name: eksadmin-52112
spec:
  clusterNetwork:
    cni: cilium
    pods:
      cidrBlocks:
      - 192.168.0.0/16
    services:
      cidrBlocks:
      - 10.96.0.0/12
  controlPlaneConfiguration:
    count: 2
    endpoint:
      host: "192.168.151.60"
    machineGroupRef:
      kind: VSphereMachineConfig
      name: eksadmin-52112-cp
  datacenterRef:
    kind: VSphereDatacenterConfig
    name: eksadmin-52112
  externalEtcdConfiguration:
    count: 3
    machineGroupRef:
      kind: VSphereMachineConfig
      name: eksadmin-21555-etcd
  kubernetesVersion: "1.20"
  workerNodeGroupConfigurations:
  - count: 2
    machineGroupRef:
      kind: VSphereMachineConfig
      name: eksadmin-52112
```

```
---
apiVersion: anywhere.eks.amazonaws.com/v1alpha1
kind: VSphereDatacenterConfig
metadata:
  name: eksadmin-52112
spec:
  datacenter: "PowerFlex"
  insecure: false
  network: "PowerFlex Data1"
  server: "192.168.151.34"
  thumbprint:
"5C:65:FB:1E:A5:DE:53:98:3E:C1:56:5C:FC:D5:10:FA:8D:04:00:3B
"


---
apiVersion: anywhere.eks.amazonaws.com/v1alpha1
kind: VSphereMachineConfig
metadata:
  name: eksadmin-52112-cp
spec:
  datastore: "EKS_Datastore"
  diskGiB: 25
  folder: "EKS-Cluster"
  memoryMiB: 8192
  numCPUs: 2
  osFamily: ubuntu
  resourcePool: "*/Resources"
  users:
  - name: ec2-user
    sshAuthorizedKeys:
    - ssh-rsa
AAAAXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
---
apiVersion: anywhere.eks.amazonaws.com/v1alpha1
kind: VSphereMachineConfig
```

```
    metadata:
      name: eksadmin-52112
    spec:
      datastore: "EKS_Datastore"
      diskGiB: 25
      folder: "EKS-Cluster"
      memoryMiB: 8192
      numCPUs: 2
      osFamily: ubuntu
      resourcePool: "*/Resources"
      users:
      - name: ec2-user
        sshAuthorizedKeys:
        - ssh-rsa AAAAxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx


    ---
    apiVersion: anywhere.eks.amazonaws.com/v1alpha1
    kind: VSphereMachineConfig
    metadata:
      name: eksadmin-52112-etcd
    spec:
      datastore: "EKS_Datastore"
      diskGiB: 25
      folder: "EKS-Cluster"
     memoryMiB: 8192
      numCPUs: 2
      osFamily: ubuntu
      resourcePool: "*/Resources"
      users:
      - name: ec2-user
        sshAuthorizedKeys:
        - ssh-rsa
    AAAAxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
    ---
```

2. After you run the YAML file, apply your configuration with EKS A to create the cluster. This process takes about 20 minutes to complete depending on your environment.
3. Run the following command to set the environment variables:

```
# Get USERNAME and PASSWORD from
your vSphere admin

export VSPHERE_USERNAME=
administrator@eksa.vsphere.local

export
VSPHERE_PASSWORD=*********
```

4. Run the following command to start the deployment on the admin VM:

```
eksctl anywhere create cluster -f eksadmin-52112.yaml
```

The following figure shows the cluster creation:

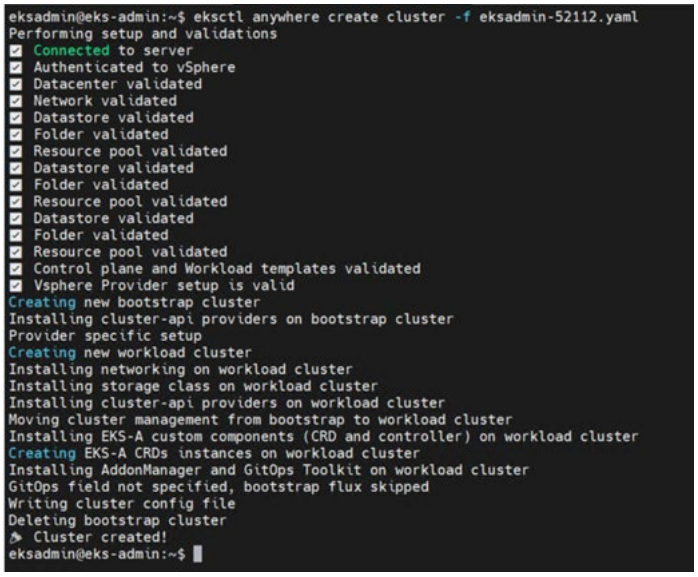


Figure 16. Cluster creation

5. After the deployment of the Amazon EKS Anywhere cluster, verify the cluster status by running the following command:

```
Kubectl get po -A -l control-plane=controller-manger
```

The following figure shows the cluster status:

```
eksadmin@eks-admin:~/csi-powerflex/test/helm$ kubectl get po -A -l control-plane=controller-manager
NAMESPACE                           NAME                                                     READY
STATUS    RESTARTS   AGE
capi-kubeadm-bootstrap-system       capi-kubeadm-bootstrap-controller-manager-7f8865f9c4-5hpb7       2/2
Running   10         6d2h
capi-kubeadm-control-plane-system   capi-kubeadm-control-plane-controller-manager-5ff686db97-v87bj   2/2
Running   2          6d2h
capi-system                         capi-controller-manager-658f6487fd-lhqcm                         2/2
Running   9          6d2h
capi-webhook-system                 capi-controller-manager-845c55b489-92d7q                         2/2
Running   0          2d1h
capi-webhook-system                 capi-kubeadm-bootstrap-controller-manager-68b8cc9759-lfctg       2/2
Running   0          2d1h
capi-webhook-system                 capi-kubeadm-control-plane-controller-manager-7dc88f767d-4bfh5   2/2
Running   0          6d2h
capi-webhook-system                 capv-controller-manager-5586796865-z7qks                         2/2
Running   1          6d2h
capv-system                         capv-controller-manager-848d7ff8db-xc4xl                         2/2
Running   2          6d2h
etcdadm-bootstrap-provider-system   etcdadm-bootstrap-provider-controller-manager-67fd58cc85-87mws   2/2
Running   3          2d1h
etcdadm-controller-system           etcdadm-controller-controller-manager-78744fc78b-8qzb6           2/2
Running   9          6d2h
```

Figure 17. Verify the EKS Anywhere cluster status

Validating the cluster deployment

The following steps validate the cluster deployment:

1. Run the following command to verify that your cluster is working properly:

   ```
   kubectl apply -f
   “https://anywhere.eks.amazonaws.com/manifests/hello-eks-
   a.yaml”
   ```

2. Run the following command to see the new pod running in your cluster:

   ```
   kubectl get pods -l app=hello-eks-a
   ```

3. There is a default web page that is served from the container. Run the following command to forward the deployment port to your local machine:

   ```
   kubectl port-forward deploy/hello-eks-a 8000:80
   ```

4. Run the following command to open a browser on your ___admin__ system or use curl localhost:8000 to view the page example application, as shown in the following figure:

   ```
   curl localhost:8000
   ```

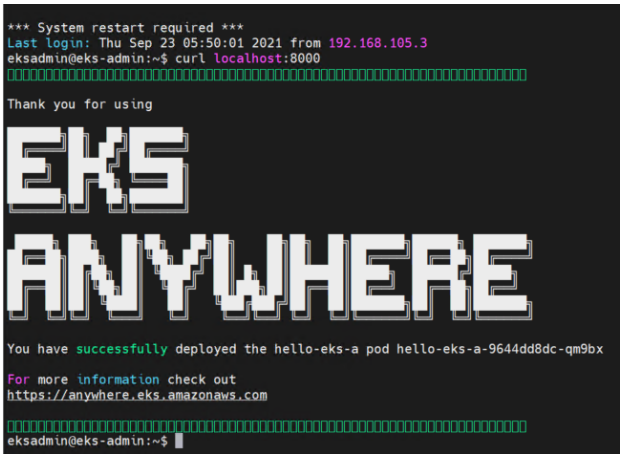


Figure 18. Hello EKS-Anywher pod

Installing the PowerFlex CSI driver

Overview

The CSI driver for PowerFlex is a plug-in that is installed into Kubernetes to provide persistent storage using the underlying PowerFlex storage system. The CSI driver for PowerFlex and Kubernetes communicates using the CSI protocol. The CSI driver for PowerFlex supports PV capabilities, dynamic and static PV provisioning, and snapshot capabilities.

The Helm chart installs the CSI driver for Dell PowerFlex using a shell script. This script installs the CSI driver container image along with the required Kubernetes sidecar containers.

The controller section of the Helm chart installs the following components in a single deployment in the `vxflexos` namespace:

- CSI driver for Dell PowerFlex
- Kubernetes Provisioner, which provisions the volumes
- Kubernetes Attacher, which attaches the volumes to the containers
- Kubernetes Snapshotter, which provides snapshot support
- Kubernetes External Resizer, which resizes the volume

The node section of the Helm chart installs the following components in a daemon set in the `vxflexos` namespace:

- CSI driver for Dell PowerFlex
- Kubernetes Node Registrar, which handles the driver registration

Prerequisites

Before you install the CSI Driver for PowerFlex, ensure that you perform the following tasks:

- Install Kubernetes v1.18 +.
- Configure Docker for bi-directional mount (MountFlags=shared).
- Install Helm 3.
- Enable Zero Padding on PowerFlex.
- Install PowerFlex SDC on all Kubernetes nodes.
- Create a Kubernetes secret for PowerFlex credentials.
- Obtain the Helm values for options like PowerFlex system name or ID, default gateway, and MDM IP addresses, and default storage pool.
- Ensure that the volume snapshot requirements are met.

For more information, see GitHub.

Install CSI drives for PowerFlex

1. Run the following command to download the installation source files from GitHub:

   ```
   $ git clone https://github.com/dell/csi-vxflexos
   ```

2. Run the following command to create a namespace called `vxflexos`:

   ```
   $ kubectl create namespace vxflexos
   ```

3. Run the following script to collect information from the PowerFlex SDC: `get_vxflexos_info.sh`.

4. Copy the `csi-vxflexos/values.yaml` script into a `myvalues.yaml` file in the same directory as the `csi-install.sh` script.

5. Edit the `myvalues.yaml` file to set the parameters such as file system types, volume name prefix, and controller count for the installation.

6. Create a `config.json` file for driver configuration. This file contains information such as the PowerFlex system IP details and credentials.

7. Run the following command to proceed with the installation:

   ```
   $ sh csi-install.sh --namespace vxflexos –values
   myvalues.yaml
   ```

8. Run the following command to check the `vxflexos` namespace for running pods to verify it is deployed correctly:

   ```
   $ kubectl get pods -n vxflexos
   ```

   For more information about CSI driver installation, see GitHub.

9. Storage classes can be created through kubectl commands. For more information about creating storage classes, see Storage Classes.

   ```
   eksadmin@eks-admin:~/csi-powerflex/samples/storageclass$
   kubectl apply -f storageclass.yaml
   ```

   ```
   eksadmin@eks-admin:~/csi-powerflex/samples/storageclass$
   kubectl apply -f storageclass-xfs.yaml
   ```

10. Ensure that the PowerFlex CSI driver is running on the EKS A Cluster, as shown in the following status example:

```
eksadmin@eks-admin:~/csi-powerflex/test/helm$ kubectl get ns
NAME                                STATUS   AGE
capi-kubeadm-bootstrap-system       Active   6d2h
capi-kubeadm-control-plane-system   Active   6d2h
capi-system                         Active   6d2h
capi-webhook-system                 Active   6d2h
capv-system                         Active   6d2h
cert-manager                        Active   6d2h
default                             Active   6d2h
eksa-system                         Active   6d2h
etcdadm-bootstrap-provider-system   Active   6d2h
etcdadm-controller-system           Active   6d2h
kube-node-lease                     Active   6d2h
kube-public                         Active   6d2h
kube-system                         Active   6d2h
vxflexos                            Active   5h18m
eksadmin@eks-admin:~/csi-powerflex/test/helm$ kubectl get pods -n vxflexos
NAME                                  READY   STATUS    RESTARTS   AGE
vxflexos-controller-7667fff648-dtkwx  5/5     Running   0          5h16m
vxflexos-controller-7667fff648-lz5xk  5/5     Running   0          5h16m
vxflexos-node-4s9cl                   2/2     Running   0          5h16m
vxflexos-node-gkv2c                   2/2     Running   0          5h16m
eksadmin@eks-admin:~/csi-powerflex/test/helm$ kubectl get sc
NAME                 PROVISIONER                RECLAIMPOLICY   VOLUMEBINDINGMODE      ALLOWVOLUMEEXPANSION   AGE
standard             csi.vsphere.vmware.com     Delete          Immediate              false                  6d2h
vxflexos             csi-vxflexos.dellemc.com   Delete          WaitForFirstConsumer   true                   24h
vxflexos-xfs (default)   csi-vxflexos.dellemc.com   Delete          WaitForFirstConsumer   true                   24h
eksadmin@eks-admin:~/csi-powerflex/test/helm$
```

Figure 19. CSI pod status

11. Test the deployment workflow of a simple pod on PowerFlex storage. For more information, see Test PowerFlex CSI Driver.

12. Verify that the persistent volume is created in PowerFlex cluster using the PowerFlex UI, as shown in the following figure:

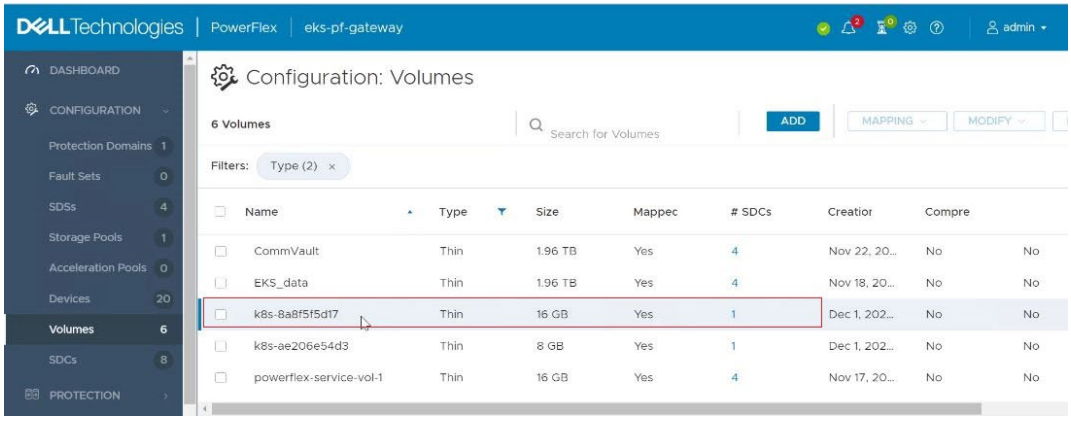


Figure 20. PowerFlex UI

Deployment tests and validation

Overview

This section provides a detailed description of the tests that were performed to validate the Amazon EKS Anywhere cluster hosted on PowerFlex.

Use case 1: Provision PVC volumes with PowerFlex CSI driver

This use case describes provisioning the PVC volumes with PowerFlex CSI driver:

- MySQL database is deployed in the lab environment to validate persistent volumes with Dell PowerFlex CSI driver. The objective is to create a persistent volume referencing a disk in the environment.
- The activity is monitored through the PowerFlex storage UI.
- The YAML file that is shown in the following example is used to deploy the MySQL Database Pod with a persistent storage claim.

```
apiVersion: v1
kind: Service
metadata:
  name: mysql
spec:
  ports:
  - port: 3306
  selector:
    app: mysql
  clusterIP: None
---
apiVersion: apps/v1
kind: Deployment
metadata:
  name: mysql
spec:
  selector:
    matchLabels:
      app: mysql
  strategy:
    type: Recreate
  template:
    metadata:
      labels:
        app: mysql
    spec:
      containers:
      - image: mysql:5.6
        name: mysql
        env:
          # Use secret in real usage
        - name: MYSQL_ROOT_PASSWORD
          value: password
        ports:
        - containerPort: 3306
          name: mysql
        volumeMounts:
        - name: mysql-persistent-storage
          mountPath: /var/lib/mysql
      volumes:
      - name: mysql-persistent-storage
        persistentVolumeClaim:
          claimName: mysql-pv-claim
```

- The following YAML sample is used for deploying a Pod with a persistent volume claim.

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: mysql-pv-claim
spec:
  storageClassName: vxflexos
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 8Gi
```

- The `kubectl get pvc` command retrieves the status of the claim, as shown in the following figure:



Figure 21. PVC Provisioned

- Verify the volumes in the PowerFlex Storage UI, as shown in the following figure:

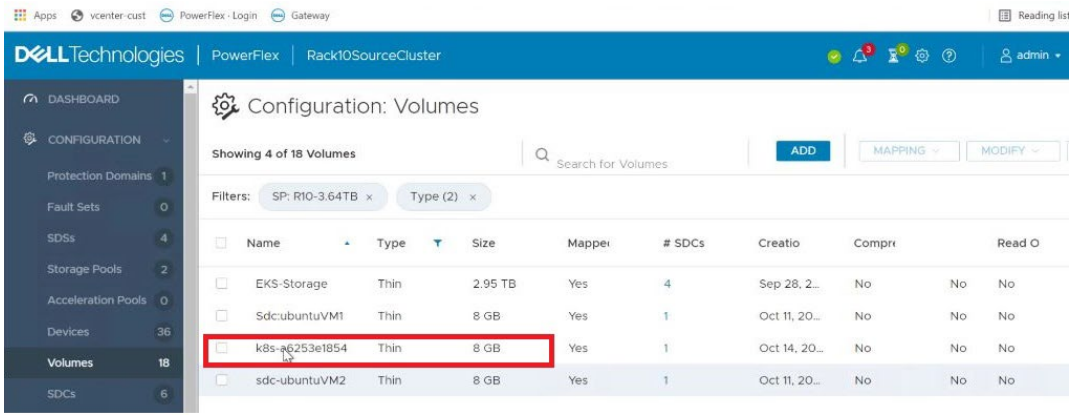


Figure 22. Dell PowerFlex storage UI

Use case 2: Extend PVC volumes with PowerFlex CSI

This use case describes the extension of PVC volumes with PowerFlex CSI:

- In the lab environment, a MySQL pod with a PVC volume of 8 GB is created.
- PVC is extended with the `kubectl patch pvc` command.

- The activity is monitored from the view volume section of the PowerFlex UI, as shown in the following figure:

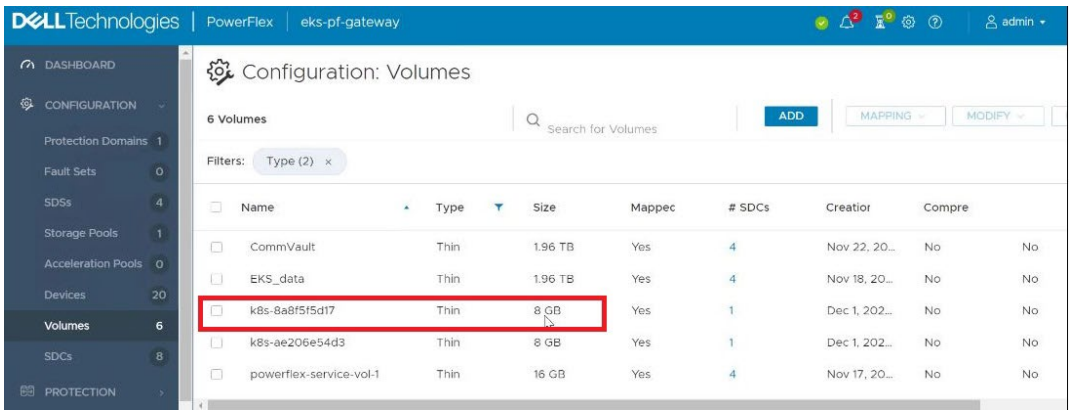


Figure 23. PowerFlex PVC volume

- The `kubectl get pvc` command provides the status of the claim and the `kubectl patch pvc` command extends the claim, as shown in the following example:

```
eksadmin@eksadmin:~/csi-powerflex/mysql$ kubectl get pvc
NAME                STATUS   VOLUME                   CAPACITY   ACCESS MODES   STORAGECLASS   AGE
mysql-pv-claim      Bound    k8s-8a8f5f5d17   8Gi        RWO            vxflexos       139m
mysql-pv-claim-snap1   Bound    k8s-ae206e54d3   8Gi        RWO            vxflexos       133m
eksadmin@eksadmin:~/csi-powerflex/mysql$ kubectl patch pvc  mysql-pv-claim  -p '{"spec":{"resources":{"requests":{"s
torage":"16Gi"}}}}'
persistentvolumeclaim/mysql-pv-claim patched
eksadmin@eksadmin:~/csi-powerflex/mysql$
```

Figure 24. PVC claim extended

- Verify the PVC expansion on the PowerFlex Storage UI, as shown in the following figure:

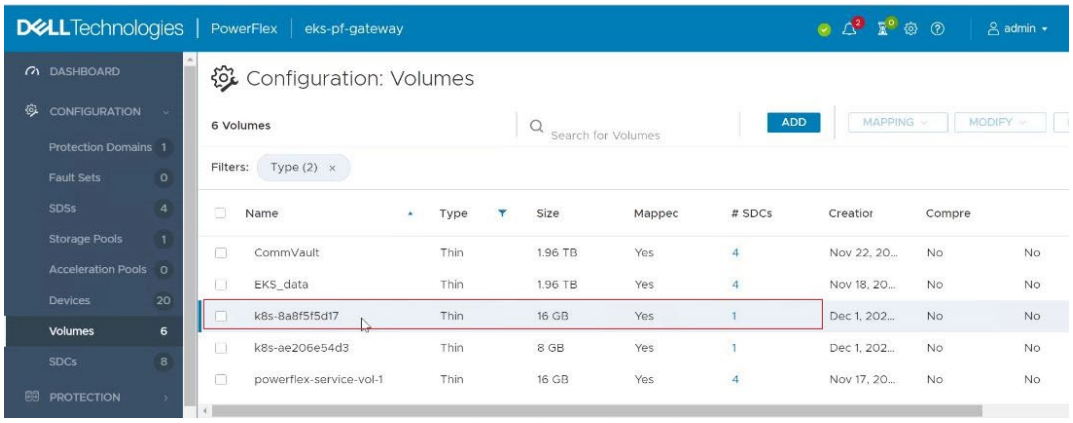


Figure 25. Dell PowerFlex storage volume

Use case 3: Snapshot with PowerFlex CSI

This use case describes how to take a screenshot with PowerFlex CSI:

- Pod 1 is deployed with MySQL database, and a PVC volume is provisioned in the lab environment.
- A sales database is created by logging in to the MySQL instance
- Fresh data is inserted into the sales database
- A snapshot of Pod 1 is created
- MySQL Database is deployed in Pod 2
- A snapshot of Pod 1 to Pod 2 is attached.
- Database queries validate the attached snapshot on Pod 2.
- The following YAML file is used to deploy the MySQL Database Pod 1 with the PVC:

```
apiVersion: v1
kind: Service
metadata:
  name: mysql
spec:
  ports:
  - port: 3306
  selector:
    app: mysql
  clusterIP: None
---
apiVersion: apps/v1
kind: Deployment
metadata:
  name: mysql
spec:
  selector:
    matchLabels:
      app: mysql
  strategy:
    type: Recreate
  template:
    metadata:
      labels:
        app: mysql
    spec:
      containers:
      - image: mysql:5.6
        name: mysql
        env:
          # Use secret in real usage
        - name: MYSQL_ROOT_PASSWORD
          value: password
        ports:
        - containerPort: 3306
          name: mysql
        volumeMounts:
        - name: mysql-persistent-storage
          mountPath: /var/lib/mysql
      volumes:
      - name: mysql-persistent-storage
        persistentVolumeClaim:
          claimName: mysql-pv-claim
```

- The following YAML file is used to provision the PVC:

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: mysql-pv-claim
spec:
  storageClassName: vxflexos
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 8Gi
```

- The following figure shows the deployed MySQL Pod 1 with PVC:

```
eksadmin@eksadmin:~/csi-powerflex/mysql$ kubectl apply -f mysql-pv.yaml
persistentvolumeclaim/mysql-pv-claim created
eksadmin@eksadmin:~/csi-powerflex/mysql$ kubectl apply -f mysql-deployment.yaml
service/mysql created
deployment.apps/mysql created
eksadmin@eksadmin:~/csi-powerflex/mysql$ kubectl get pods
NAME                     READY   STATUS              RESTARTS   AGE
hello-eks-a-9644dd8dc-fccm8   1/1     Running             0          4d23h
mysql-68579b78bb-qgwhx        0/1     ContainerCreating   0          7s
eksadmin@eksadmin:~/csi-powerflex/mysql$ kubectl get pvc
NAME             STATUS   VOLUME           CAPACITY   ACCESS MODES   STORAGECLASS   AGE
mysql-pv-claim   Bound    k8s-8a8f5f5d17   8Gi        RWO            vxflexos       29s
eksadmin@eksadmin:~/csi-powerflex/mysql$ kubectl get pods
NAME                     READY   STATUS    RESTARTS   AGE
hello-eks-a-9644dd8dc-fccm8   1/1     Running   0          4d23h
mysql-68579b78bb-qgwhx        1/1     Running   0          27s
eksadmin@eksadmin:~/csi-powerflex/mysql$
```

Figure 26. Pod 1 MySQL Deployment with PVC volume

- Logged in to the MySQL instance and created a database that is named `sales`, as shown in the following figure:

```
mysql> CREATE TABLE Information (firstname VARCHAR(20),lastname VARCHAR(20),gender CHAR(1),grade INT(10), dob DATE);

Query OK, 0 rows affected (0.01 sec)

mysql>  INSERT INTO Information VALUES ('Amanda','Williams','f','10','1999-03-30');
Query OK, 1 row affected (0.01 sec)

mysql>  INSERT INTO Information VALUES ('Peter','Williams','m','9','1998-03-15');
Query OK, 1 row affected (0.01 sec)

mysql> INSERT INTO Information VALUES ('Cristie','Wills','f','8','1999-02-05');
Query OK, 1 row affected (0.00 sec)

mysql> SELECT * FROM Information;
+-----------+----------+--------+-------+------------+
| firstname | lastname | gender | grade | dob        |
+-----------+----------+--------+-------+------------+
| Amanda    | Williams | f      |    10 | 1999-03-30 |
| Peter     | Williams | m      |     9 | 1998-03-15 |
| Cristie   | Wills    | f      |     8 | 1999-02-05 |
+-----------+----------+--------+-------+------------+
3 rows in set (0.00 sec)

mysql> exit
Bye
pod "mysql-client" deleted
eksadmin@eksadmin:~/csi-powerflex/mysql$
```

Figure 27. Create Sales database

- The values are inserted to the sales database, and a snapshot is created for the MySQL Pod 1.

- The following YAML file is used to create a snapshot of the PVC:

```
apiVersion: snapshot.storage.k8s.io/v1
kind: VolumeSnapshot
metadata:
  name: mysql-pv-claim-snap1
  namespace: default
spec:
  volumeSnapshotClassName: vxflexos-snapclass
  source:
    persistentVolumeClaimName: mysql-pv-claim
```

- The following YAML file is used to deploy the MySQL Database Pod 2 with PVC:

```
apiVersion: v1
kind: Service
metadata:
  name: mysql-2
spec:
  ports:
  - port: 3307
  selector:
    app: mysql
  clusterIP: None
---
apiVersion: apps/v1
kind: Deployment
metadata:
  name: mysql-2
spec:
  selector:
    matchLabels:
      app: mysql
  strategy:
    type: Recreate
  template:
    metadata:
      labels:
        app: mysql
    spec:
      containers:
      - image: mysql:5.6
        name: mysql
        env:
          # Use secret in real usage
        - name: MYSQL_ROOT_PASSWORD
          value: password
        ports:
        - containerPort: 3307
          name: mysql
        volumeMounts:
        - name: mysql-persistent-storage
          mountPath: /var/lib/mysql
      volumes:
      - name: mysql-persistent-storage
        persistentVolumeClaim:
          claimName: mysql-pv-claim-snap1
```

- The following YAML file is used for provisioning the persistent volume claim:

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: mysql-pv-claim-snap1
spec:
  storageClassName: vxflexos
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 8Gi
```

- During the PVC provisioning, Pod 2 is provisioned using the Pod 1 Snapshot as follows:

```
eksadmin@eksadmin:~/csi-powerflex/mysql/2nd-Pod$ kubectl apply -f 2-mysql-pv.yaml
persistentvolumeclaim/mysql-pv-claim-snap1 created
eksadmin@eksadmin:~/csi-powerflex/mysql/2nd-Pod$ kubectl apply -f 2-mysql-deployment.yaml
service/mysql-2 created
deployment.apps/mysql-2 created
eksadmin@eksadmin:~/csi-powerflex/mysql/2nd-Pod$ kubectl get pvc
NAME                  STATUS   VOLUME          CAPACITY   ACCESS MODES   STORAGECLASS   AGE
mysql-pv-claim        Bound    k8s-8a8f5f5d17   8Gi        RWO            vxflexos       5m32s
mysql-pv-claim-snap1  Bound    k8s-ae206e54d3   8Gi        RWO            vxflexos       16s
eksadmin@eksadmin:~/csi-powerflex/mysql/2nd-Pod$
```

Figure 28. Pod 2 Deployment

- Database queries are run on Pod 2 to validate database entries as shown in the following figure:

```
mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| #mysql50#lost+found |
| mysql              |
| performance_schema |
| sales              |
+--------------------+
5 rows in set (0.00 sec)

mysql> use sales;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> SELECT * FROM Information;
+-----------+----------+--------+-------+------------+
| firstname | lastname | gender | grade | dob        |
+-----------+----------+--------+-------+------------+
| Amanda    | Williams | f      |    10 | 1999-03-30 |
| Peter     | Williams | m      |     9 | 1998-03-15 |
| Cristie   | Wills    | f      |     8 | 1999-02-05 |
+-----------+----------+--------+-------+------------+
3 rows in set (0.00 sec)

mysql>
```

Figure 29. Attached Pod 1 PVC to Pod 2

- The sales database with the entries can be retrieved on Pod 2.

Conclusion

This white paper demonstrates how Amazon EKS Anywhere can be used with the Dell PowerFlex family. Organizations get on-premises Kubernetes operational tooling that is consistent with Amazon EKS. This combination brings together both the data center and cloud to simplify the management of both. PowerFlex CSI enables expanded storage capabilities for developers inside Amazon EKS Anywhere and allows you to use similar tools to manage your environment.

Configuration details

Storage-only and compute-only nodes

The following tables provide the configuration details of the Dell PowerEdge R640 server storage-only and compute-only nodes:

Table 3. HCI nodes

Hardware	Configuration
CPU Cores	2 x Intel Xeon Gold 6248 R CPU @ 3.00 GHz
Memory	192 GB (12 x 16 GB, 2933 MHz)
NIC	2 x Mellanox ConnectX-4 LX 25 GbE SFP Adapter
	1 x Intel Ethernet 10G 4P X710/I350 rNDC
Storage	BOSS S1 Controller 2 x 240 GB SATA SSD
	Dell HBA330 controller 8 x 1.9 TB SAS SSD
Operating system	Embedded operating system
Hypervisor	ESXi 7.0.2
Dell PowerFlex	3.6.0.2

Amazon EKS Anywhere configuration details

The following tables provide the configuration details of an EKS Anywhere management VM:

Table 4. EKS Anywhere admin VM details

Components	Items	Details
Virtual hardware	Virtualized hardware	vSphere 7.0.2
	CPU	4 vCPU
	RAM	16 GB
	Hard disk	40 GB
	NIC	VMware Virtual NIC
Software	Operating system	Ubuntu 20.0.4.3 LTS

Components	Items	Details
	Container Runtime	Docker 20.10.8
	GOVC	v0.26.1
	EKS Anywhere	v0.5
	Kubernetes	v1.19
	EKS-A CLI	0.74.0
	HELM	V3.7.1
	Dell CSI	v2.0
Network Access	• public.ecr.aws • api.github.com • eks-anywhere-beta.s3.amazonaws.com (to download the manifests and OVAs) • distro.eks.amazonaws.com (to download EKS Distro)	Network access To download manifests, OVAs, and EKS distro.

Table 5. EKS Anywhere workload cluster details

Components	Items	Details
Hardware	Virtualized hardware	vSphere 7.0.2
	CPU	2 vCPU
	RAM	8 GB
	Hard disk	20 GB
	NIC	VMware Virtual NIC
Software	Operating system	Ubuntu 20.0.4.3 LTS
	HELM	V3.7.1
	SDC	v3.6.0.2
	Dell CSI	v2.0

References

Dell Technologies documentation

The following Dell Technologies documentation provides additional and relevant information. Access to these documents depends on your login credentials. If you do not have access to a document, contact your Dell Technologies representative.

- PowerFlex Overview — Video
- PowerFlex Specification Sheet
- PowerFlex Solutions Documents — InfoHub
- PowerFlex Networking Best Practices and Design Considerations
- PowerFlex - Storage definitions
- Dell CSI Drivers - Dell CSI Git Repo

Amazon EKS Anywhere documentation

The following Amazon EKS documentation provides additional and relevant information:

- EKS Anywhere Overview
- EKS Anywhere Installation
- Compare EKS Anywhere and EKS
- EKS Anywhere Cluster creation workflow
- EKS Anywhere Cluster management
- EKS Anywhere Cluster Troubleshooting