
Tanzu Kubernetes Grid 1.5.4

Implementation and Run Book

Version History

Date	Ver.	Author	Description	Reviewers
18/08/2022	Draft	Kalyaan	Implementation/Run Book	Krishnaiah Kunta

Contents

1. Terminology.....	4
2. Requirements.....	5
2.1 Virtual Infrastructure Requirements	5
2.2 Network Infrastructure Requirements	6
2.3 Users Roles and Responsibilities.....	6
2.4 BootStrap Machine	7
2.5 Firewall Requirements	8
3. Extension Installation Location.....	9
4. Setting up TKG Environment.....	10
4.1 Setting up BootStrap Server	10
4.2 Creating Storage Policy.....	14
4.3 Deploy and Configure Management Cluster	15
4.3.1 Verifying Management Cluster Deployment	24
4.4 Deploy and Configure Workload Cluster	25
4.4.1 Connecting to Workload Cluster Deployed using Admin kubeconfig	27
4.4.2 Connecting to Workload Cluster Deployed using Non-admin Config	28
4.4.3 Creating StorageClass in Workload Cluster	29
5. Installing and Configuring the Packages/Extensions	31
5.1 Deploying Cert-Manager Extension	34
6. Generic Day-2 Operation Commands.....	38

1. Terminology

Acronym/Word/Phrase	Definition/Meaning
Bootstrap Server	Machine to be used for bootstrapping of the management clusters by provisioning a KIND (Kubernetes in Docker) Cluster
TKG	Tanzu Kubernetes Grid is an enterprise-ready Kubernetes runtime that streamlines operations across a multi-cloud infrastructure.
K8s	Abbreviation for Kubernetes.
Management Cluster	Management cluster is the first components deployed in the Tanzu Kubernetes Grid installation on which the cluster API runs, which is used for provisioning and lifecycle management of Tanzu Kubernetes clusters (Workload clusters)
Tanzu Kubernetes Cluster Plan	Cluster plan is the blueprint that defines the configuration of the Kubernetes clusters. Default plans available are, Dev and Prod
Tanzu Kubernetes Cluster	These are the Kubernetes clusters deployed using the management cluster. The containerized application workloads would be hosted on these Kubernetes clusters
Tanzu Kubernetes Grid Installer	The Tanzu Kubernetes Grid installer is a graphical wizard that you start up by running the <code>tanzu management-cluster create --ui</code> command
Control Plane Node	The kubernetes control plane VMs that host API server, the controller manager, the scheduler and etcd components.
Worker Node	Nodes which host the pods/containers
Cluster API	The open-source project for API for TKG
VM	Virtual Machine

CRD	Custom Resource Definition
CIDR	Classless inter-domain routing
YAML	Yet Another Markup Language - a markup language, like JSON
Prometheus	Prometheus is a system and service monitoring system
Grafana	Grafana is a visualization tool which gets the data from the Prometheus.
Fluent-bit	Fluent Bit is a fast, lightweight log processor and forwarder
Ingresss	An API object that manages external access to the services in a cluster, typically HTTP. Ingress may provide load balancing, SSL termination and name-based virtual hosting

2. Requirements

2.1 Virtual Infrastructure Requirements

Sno	Component	Comments
1	vSphere 6.7U3 or Higher	Requirement for TKGS 1.5.4
2	ESXi Hosts	Standalone or Cluster with Minimum 2 hosts
3	License	Enterprise License
4	Networking	DVS to be configured with required port-groups/vLAN to enable DHCP Communication
5	Storage	Required storage to be provisioned on the hosts on which the TKG clusters would be provisioned

2.2 Network Infrastructure Requirements

Sno	Component	Comments
1	DHCP Server	Configured with Option 3 and Option 6 to ensure Gateway and DNS details are passed to the client along with the IP address
2	IP Segments	One IP segment for each cluster to be deployed
3	Static IP for Kube-VIP	One static IP for each cluster from the DHCP Range of each cluster
4	NTP Server	Required for TLS authentication within Tanzu Kubernetes Grid clusters (configured either using DHCP option 42 or using overlay file)
5	Load Balancer	AVI Load balancer (NSX Advance) to be Deployed and Configured

2.3 Users Roles and Responsibilities

User with required permissions to deploy Tanzu Kubernetes Grid. Ex: tkg-user

vSphere Object	Required Permission
Cns	Searchable
Datastore	Allocate space Browse datastore Low level file operations
Global (if using Velero for backup and restore)	Disable methods Enable methods Licenses
Network	Assign network
Profile-driven storage	Profile-driven storage view
Resource	Assign virtual machine to resource pool
Sessions	Message Validate session

Virtual machine	Change Configuration > Add existing disk Change Configuration > Add new disk Change Configuration > Add or remove device Change Configuration > Advanced configuration Change Configuration > Change CPU count Change Configuration > Change Memory Change Configuration > Change Settings Change Configuration > Configure Raw device Change Configuration > Extend virtual disk Change Configuration > Modify device settings Change Configuration > Remove disk Change Configuration > Toggle disk change tracking* Edit Inventory > Create from existing Edit Inventory > Remove Interaction > Power On Interaction > Power Off Provisioning > Allow read-only disk access* Provisioning > Allow virtual machine download* Provisioning > Deploy template Snapshot Management > Create snapshot* Snapshot Management > Remove snapshot*
vApp	Import

2.4 BootStrap Machine

The bootstrap machine is the laptop, host, or server that you deploy management and workload clusters from, and that keeps the Tanzu and Kubernetes configuration files for your deployments.

Component	Version
OS	RHEL/Ubuntu/Windows/MacOS
CPU	2 vCPU
Memory	6GB
Docker	20.10.16
TanzuCli	1.5.4
Kubectl	1.22.9
Ytt	0.37
Kbld	0.31
Imgpkg	0.22
Kapp	0.42

2.5 Firewall Requirements

Source	Destination	Port	Protocol	Service Description
Management and Workload Cluster CIDR	Avi Controller	443	TCP	(NSX ALB only) Allow Avi Kubernetes Operator (AKO) and AKO Operator (AKOO) access to Avi Controller
Workload Cluster Network CIDR	Management Cluster Network CIDR	31234	TCP	Allow Pinniped concierge on workload cluster to access Pinniped supervisor on management cluster
Workload Cluster Network CIDR	Management Cluster HAProxy IP	6443	TCP	Allow workload cluster to register with management; with NSX ALB, can override port 6443 with VSPHERE_CONTROL_PLANE_ENDPOINT_PORT
Management Cluster CIDR	Workload Cluster HAProxy IP	6443; 5556	TCP	Allow management cluster to configure workload cluster; with NSX ALB, can override port 6443 with VSPHERE_CONTROL_PLANE_ENDPOINT_PORT
Management and Workload Cluster CIDR	Avi Controller	443	TCP	(NSX ALB only) Allow Avi Kubernetes Operator (AKO) and AKO Operator (AKOO) access to Avi Controller
Workload Cluster Network CIDR	Management Cluster Network CIDR	31234	TCP	Allow Pinniped concierge on workload cluster to access Pinniped supervisor on management cluster
Workload Cluster Network CIDR	Management Cluster HAProxy IP	6443	TCP	Allow workload cluster to register with management; with NSX ALB, can override port 6443 with VSPHERE_CONTROL_PLANE_ENDPOINT_PORT
Management Cluster CIDR	Workload Cluster HAProxy IP	6443; 5556	TCP	Allow management cluster to configure workload cluster; with NSX ALB, can override port 6443 with VSPHERE_CONTROL_PLANE_ENDPOINT_PORT
Management and Workload Cluster CIDR	NTP Servers	123	UDP	Allow components to sync current time
Management and Workload Cluster CIDR	DNS Servers	53	UDP	Allow components to look up machine addresses
Management and Workload Cluster CIDR	vCenter IP	443	TCP	Allow components to access vCenter to create VMs and Storage volumes
Management and Workload Cluster CIDR	Harbor IP	443	TCP	Allow components to retrieve container images
Browser	OIDC/LDAP endpoint	31234	HTTPS	Pinniped using Nodeport in vSphere
Browser	OIDC/LDAP endpoint	30167	HTTPS	Dex using Nodeport in vSphere
Platform	Platform	6081	UDP	CNI ingress rule for Geneve
Platform	Platform	10349	TCP	CNI ingress rule for Antrea

Bootstrap Server	Managenet cluster CIDR, Workload Cluster	22, 6443	TCP	Connectivity from bootstrap server to API server of management and workload clusters
Bootstrap Server	OIDC/LDAP endpoint	31234	HTTPS	Pinniped using Nodeport in vSphere
Bootstrap Server	OIDC/LDAP endpoint	30167	HTTPS	Dex using Nodeport in vSphere

3. Extension Installation Location

As part of this deployment, we have deployed the below extensions.

Sno	Extension	Function	Location
1	Harbor	Image Registry	Shared Service Cluster

4. Setting up TKG Environment

4.1 Setting up Bootstrap Server

Bootstrap server could either be a desktop/laptop/server running, Linux, Windows or MAC OS, from which we would be deploying the Tanzu Kubernetes management and workload clusters.

The Following CLI tools to be installed on the bootstrap server:

Docker:

For Docker installation, please refer to the docker installation guide as per the selected Operating system for the bootstrap server.

Note:

1. If the bootstrap server is running on Linux and using a non-root user for the deployment purpose, add the non-root user to the docker group. This is required for tanzucli to interact with docker
2. Ensure Bootstrap server is configured for Cgroup v1

Perform the below steps to validate the cgroup version:

```
docker info | grep -i cgroup
```

Expected output:

```
Cgroup Driver: cgroupfs  
Cgroup Version: 1
```

If the cgroup is set to version:2, run the below command to set to version 1

```
system.unified_cgroup_hierarchy=0
```

kubectl & tanzucli

- a. Download the kubectl cli and Tanzucli from the below URL:

```
https://customerconnect.vmware.com/en/downloads/details?downloadGroup=TKG-141&productId=988&rPId=82536
```

- b. Ensure the version is set to 1.5.4.0 and navigate to the kubectl//Tanzucli, plug-in sections and download the respective plug-in as per the OS installed on the bootstrap server
- c. Create a folder on the bootstrap server and unpack the downloaded CLI tar file to this folder.

```
tar -xvf tanzu-cli-bundle-linux-amd64.tar
```

```
tar -xvf kubectl-linux-v1.22.9+vmware.1.gz
```

- d. Navigate to the cli subfolder under the tanzu folder created in above step
- e. Install the Tanzucli binary to /usr/local/bin

```
sudo install core/v1.5.4/tanzu-core-linux_amd64 /usr/local/bin/Tanzu
```

- f. Verify the installation/version using the command

```
tanzu version
```

- g. Now install the kubectl cli, using below commands:

```
chmod ugo+x kubectl-linux-v1.22.9+vmware.1
```

```
sudo install kubectl-linux-v1.22.9+vmware.1 /usr/local/bin/kubectl
```

- h. Verify the installation/version using the command

Kubectl version

TanzuCli Plug-ins:

1. Navigate to the Tanzu folder, where the tar file of the tanzuccli and kubescli has been downloaded and then to the cli folder inside it
2. Install the plug-ins using the below commands:

```
tanzu plugin install --local cli all
```

3. To list the plug-ins installed

```
tanzu plugin list
```

Carvel Tools (Imgpkg, ytt, kbld, kapp)

Carvel provides a set of reliable, single-purpose, composable tools that aid in application building, configuration, and deployment to Kubernetes.

1. **ytt** - A command-line tool for templating and patching YAML files. You can also use ytt to collect fragments and piles of YAML into modular chunks for easy re-use.
2. **kapp** - The applications deployment CLI for Kubernetes. It allows you to install, upgrade, and delete multiple Kubernetes resources as one application.
3. **kbld** - An image-building and resolution tool.

4. **imgpkg** - Tool that enables Kubernetes to store configurations and the associated container images as OCI images, and to transfer these images

To install the above carvel tools, navigate to the cli folder within the Tanzu folder, where the tar file for cli have been downloaded and follow below commands to install them.

Intalling YTT:

1. Unzip the downloaded zip file
`gunzip ytt-linux-amd64-v0.37.0+vmware.1.gz`
2. Make the file executable
`chmod ugo+x ytt-linux-amd64-v0.37.0+vmware.1`
3. Move the executable file to /usr/local/bin/
`mv ./ytt-linux-amd64-v0.37.0+vmware.1 /usr/local/bin/ytt`

Installing kapp:

1. Unzip the downloaded zip file
`unzip kapp-linux-amd64-v0.42.0+vmware.1.gz`
2. Make the file executable
`chmod ugo+x kapp-linux-amd64-v0.42.0+vmware.1`
3. Move the executable file to /usr/local/bin/
`mv ./kapp-linux-amd64-v0.42.0+vmware.1 /usr/local/bin/kapp`

Installing kbld:

1. Unzip the downloaded zip file
`gunzip kbld-linux-amd64-v0.31.0+vmware.1.gz`
2. Make the file executable
`chmod ugo+x kbld-linux-amd64-v0.31.0+vmware.1`
3. Move the executable file to /usr/local/bin/

```
mv ./kbld-linux-amd64-v0.31.0+vmware.1 /usr/local/bin/kbld
```

Installing imgpkg:

1. Unzip the downloaded zip file

```
gunzip imgpkg-linux-amd64-v0.22.0+vmware.1.gz
```
2. Make the file executable

```
chmod ugo+x imgpkg-linux-amd64-v0.22.0+vmware.1
```
3. Move the executable file to /usr/local/bin/

```
mv ./imgpkg-linux-amd64-v0.22.0+vmware.1 /usr/local/bin/imgpkg
```

4.2 _Creating Storage Policy

Storage Policy needs to be created in vSphere, using which the “Storage Class” construct would be created in the Kubernetes environment. These storage classes, would be used to create the persistent volumes to be used by pods in Kubernetes clusters.

Multiple storage classes could be created for each Kubernetes cluster as per the requirements.

1. Create tag for the datastore, which would be part of the storage policy
 - Right-click the datastore you want to tag and select Tags and Custom Attributes > Assign Tag.
 - Click Add Tag and specify the tag's properties.
 - Provide the name of the tag, description, and category
2. In the vSphere Client, open the Create VM Storage Policy wizard.
 - Click Menu > Policies and Profiles.
 - Under Policies and Profiles, click VM Storage Policies.
 - Click Create VM Storage Policy.
 - Provide name and description

- Under, policy structure, select, “Enable rules for “VSAN” storage and “Enable tagbased placement rules”, check boxes and click on next.
- Select the required availability, storage rules, and tags for the vSAN configuration

Under the availability tab, select the “Failures to Tolerate” set to “1” and Failure tolerance method to RAID1

- Select, Storage rules tab and set the values as in the below screenshot.
- Select, “Advance Policy Rules” and set the number of stripes to 1, object space reservation to, “Thin provisioning”, disable the option for object checksum and Force Provisioning.
- Select the “Tags” tab and select the name of the tag created previously for the vSAN datastore.
- In the storage compatibility list, select the vSAN datastore
- Click on Next and finish.

We will be using this storage policy to deploy storage class in the workload clusters at a later stage.

4.3 Deploy and Configure Management Cluster

The management cluster is a Kubernetes cluster that runs Cluster API operations on a specific cloud provider to create and manage workload clusters on that provider. The management cluster is also where you configure the shared and in-cluster services that the workload clusters use

Prerequisites

1. Bootstrap server configured with required Cli tools
2. DHCP Configured with required Options
3. IP segment configured in DHCP propagated to ESXi host part of the cluster
4. Distributed port groups created with correct vLAN of the management cluster
5. VM folder created in vCenter for hosting the management cluster components
6. Required OVA image to deploy the control plane and worker node, imported in vCenter
7. User with required permissions created and mapped to all the components in vCenter (Clusters, hosts, networks, storage etc)
8. LDAP details require for cluster authentication for users

Management Cluster could be deployed in two ways:

1. GUI – Run the tanzu Kubernetes grid installer, which opens a wizard interface for step-by-step deployment
2. CLI – Requires a Yaml file to be created with required inputs to deploy the cluster and deployed using cli commands.

Deploying using GUI:

1. Login to the bootstrap server
2. Export the registry name, certificate as environment variable.
3. Generate the SSH key pair to be used to connect to the cluster nodes
 - a. Ensure keygen tool is available on the bootstrap server from where the tanzucli command would be executed

```
ssh-keygen -t rsa -b 4096 -C "email@example.com"
```

- At the prompt, enter the file in which to save the key (/root/.ssh/id_rsa), either accept this default location or type a different location to save the key file
- Provide the password for the key pair
- Add private key to ssh agent on the bootstrap server machine and enter the password provided in above step

```
ssh-add ~/.ssh/id_rsa
```

- Restart the ssh agent service

```
eval 'ssh-agent -s'
```

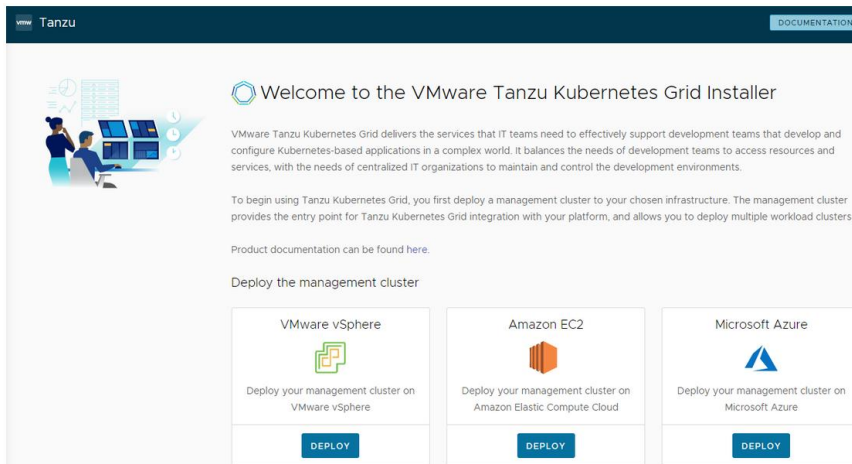
4. Initiate the GUI installer wizard

```
tanzu management-cluster create --ui
```

Note: If the bootstrap server doesn't have UI, the installer wizard interface could be redirected to another system with UI using below command

```
tanzu management-cluster create --ui --bind 127.0.0.1:8080 --browser none
```


- a. In first screen, select the underlying platform on which to deploy the management cluster



- b. Enter the details of the vCenter server

1. IaaS Provider Validate the vSphere provider account for Tanzu

1 You must click "CONNECT" to verify the vSphere credentials before moving to next step.

VCENTER SERVER ⓘ :DCCBSVC01.rbi1.rbi.org.in USERNAME tkgadmin@rbi1.rbi.org.in PASSWORD

SSL THUMBPRINT VERIFICATION ⓘ
☐ Disable Verification

CONNECT

DATACENTER ⓘ

SSH PUBLIC KEY ⓘ

BROWSE FILE

- c. Select and name of the datacenter and enter the SSH Public key Generated in previous step.

- f. Enter the name for the management cluster. If no name is entered, a random unique name would be generated.
- g. Select the box for “Machine health check”
- h. Select the box for “Audit Logging”
- i. Select the Control plane Endpoint provider, this could be either a Kube-Vip or NSX Advance Load Balancer.


In this deployment, we are using Kube-Vip.

- j. Enter a static IP under the Control plane endpoint field. This is a mandatory field when Kube-vip is selected as control plane endpoint provider.

Note: This IP address needs to be in the same IP segment as the DHCP range used for the management cluster but not part of this DHCP range.

2. Management Cluster Settings
Production cluster selected: 3 node control plane


Management Cluster Settings



Development

Single control plane node. Recommended for a development environment.

INSTANCE TYPE ⓘ



Production

Three control plane nodes. Recommended for a production environment.

INSTANCE TYPE ⓘ large (cpu: 4, ram: 16 GB, d ⓘ

MANAGEMENT CLUSTER NAME (OPTIONAL) ⓘ
drdcekprdmgmtcls

MACHINE HEALTH CHECKS ⓘ
☒ Enable

CONTROL PLANE ENDPOINT PROVIDER ⓘ
Kube-vip

CONTROL PLANE ENDPOINT ⓘ
172.24.86.197

WORKER NODE INSTANCE TYPE ⓘ
large (cpu: 4, ram: 16 GB, d ⓘ

ENABLE AUDIT LOGGING ⓘ
☒ Enable

NEXT

- k. Skip the NSX Advance Loadbalancer configuration section, as we would be doing over the top deployment of NSX ALB on the clusters i.e deploy AKO in each cluster, post deployment of the cluster.
- l. In the resources page, select the Folder, Datastore and cluster/host to which to deploy the management cluster nodes

5. Resources Resource Pool: /DRDCCBSK8SPRDDC/host/DRDCCBSK8SPRDCLS/Resources, VM Folder: /DRDCCBSK8SPRDDC/vm/TKG-MGMTCLS-PROD, Datastore: /DRDCCBSK8SPRDDC/datastore /DRDCCBSK8SPRDCLSSDRS01/DRDCCBSK8SPRDCLS-DS01

Specify the Resources [?](#)

VM FOLDER [?](#) DATASTORE [?](#)

/DRDCCBSK8SPRDDC/vm/TKG-MGMTCLS-PROD PRDDC/datastore/DRDCCBSK8SPRDCLSSDRS01/DRDCCBSK8SPRDCLS-DS01

CLUSTERS, HOSTS, AND RESOURCE POOLS [?](#)

☒ DRDCCBSK8SPRDCLS

☐ DRDCCBSVMPRDCLS

[NEXT](#)

- m. In the Kubernetes network settings page, select the distributed port group configured with the vLAN for Management cluster and provide the cluster service CIDR and Cluster POD CIDR ranges.

Note: The default CNI would be selected as Antrea

6. Kubernetes Network Network: /DRDCCBSK8SPRDDC/network/DRDC-TKG-MGMTCLS-638

Kubernetes Network Settings [?](#)

CNI Provider: Antrea

NETWORK NAME [?](#) CLUSTER SERVICE CIDR [?](#) CLUSTER POD CIDR [?](#)

/DRDCCBSK8SPRDDC/netv 100.64.0.0/13 100.96.0.0/11

Proxy Settings

☐ Enable Proxy Settings

[NEXT](#)

- n. In the, "Configure Identity Management" section, Set the toggle button to enable the Identity management Settings and select, "LDAP" as the identity provider.

Optionally Specify Identity Management with OIDC or LDAPS

☒ Enable Identity Management Settings

- o. Provide the details of the LDAP server, port and other search strings and filters for users and groups.

While most of the fields are mentioned as optional, it is recommended to provide all the fields in order to ensure correct user/groups resolutions using the verify settings button in the UI interface.

LDAPS Identity Management Source

LDAPS Endpoint ①	BIND DN (OPTIONAL) ①	BIND PASSWORD (OPTIONAL) ①
<input type="text"/>	<input type="text"/>	<input type="password"/>
User Search Attributes		
BASE DN (OPTIONAL) ①	FILTER (OPTIONAL) ①	USERNAME (OPTIONAL) ①
<input type="text"/>	<input type="text"/>	<input type="text"/>
Group Search Attributes		
BASE DN (OPTIONAL) ①	FILTER (OPTIONAL) ①	NAME ATTRIBUTE (OPTIONAL) ①
<input type="text"/>	<input type="text"/>	<input type="text"/>
USER ATTRIBUTE (OPTIONAL) ①	GROUP ATTRIBUTE (OPTIONAL) ①	
<input type="text"/>	<input type="text"/>	
ROOT CA (OPTIONAL) ①	VERIFY LDAP CONFIGURATION (OPTIONAL) ①	
<input type="text"/>	<input type="checkbox"/>	
		<input type="button" value="VERIFY LDAP CONFIGURATION..."/>
<input type="button" value="NEXT"/>		

- p. In the, “Select Base OS Image” section, select the Kubernetes OVA image previously downloaded from MYVMWARE site for TKG 1.5.4 version and imported to the cloud provider i.e the vCenter Server in this case.

Note: For TKG 1.5.4 the Management clusters run on Kubernetes 1.22.9 version.

The photon flavor of the Kubernetes version or the Ubuntu version could be used for the deployment. In this Deployment, we have used the Ubuntu version of Kubernetes OVA

- q. Skip the TMC Registration section as TMC is not used in this deployment.
- r. Skip the CIEP participation and click on next
- s. Review the configuration selected thus far and if any modifications are to be performed, click on the “Edit Configuration” button to navigate back to the wizard.
- t. On reaching the “Review Configuration” screen, the wizard generates the configuration file for the management cluster deployment using the option selected in the wizard and saves it to the location, “~/config/tanzu/tkg/clusterconfigs”
- u. At this stage, we could click on, “ Deploy Management Cluster” button to initiate the cluster deployment or use the CLI command displayed to initiate the cluster deployment from the CLI of the bootstrap server using the configuration file created by the wizard in the location mentioned in above step.

Deploying Management Cluster using CLI:

To deploy the Management cluster using the CLI, we could either use the clusterconfig yaml file generated using the UI wizard which is saved in the root folder of the bootstrap server (“~/config/tanzu/tkg/clusterconfigs”) or manually create a yaml file to create the cluster with the required fields.

Sample Cluster Configuration file for vSphere Endpoint:

```
#! -----
#! Basic cluster creation configuration
#! -----

CLUSTER_NAME: mgmt-cluster
CLUSTER_PLAN: prod
INFRASTRUCTURE_PROVIDER: vsphere
ENABLE_CEIP_PARTICIPATION: false
ENABLE_AUDIT_LOGGING: true
CLUSTER_CIDR: 100.96.0.0/11
SERVICE_CIDR: 100.64.0.0/13
IDENTITY_MANAGEMENT_TYPE: ldap

LDAP_BIND_DN:
LDAP_BIND_PASSWORD:
LDAP_GROUP_SEARCH_BASE_DN:
LDAP_GROUP_SEARCH_FILTER:
LDAP_GROUP_SEARCH_GROUP_ATTRIBUTE:
LDAP_GROUP_SEARCH_NAME_ATTRIBUTE:
LDAP_GROUP_SEARCH_USER_ATTRIBUTE:
LDAP_HOST:
LDAP_ROOT_CA_DATA_B64:
LDAP_USER_SEARCH_BASE_DN:
LDAP_USER_SEARCH_FILTER:
LDAP_USER_SEARCH_NAME_ATTRIBUTE:
LDAP_USER_SEARCH_USERNAME:

#! -----
#! vSphere configuration
#! -----

VSPHERE_SERVER:
VSPHERE_USERNAME:
VSPHERE_PASSWORD:
VSPHERE_DATACENTER:
VSPHERE_RESOURCE_POOL:
```

```
VSPHERE_DATASTORE:
VSPHERE_FOLDER:
VSPHERE_NETWORK: VM Network
VSPHERE_CONTROL_PLANE_ENDPOINT: # Required for Kube-Vip
VIP_NETWORK_INTERFACE: "eth0"
VSPHERE_TEMPLATE:
VSPHERE_SSH_AUTHORIZED_KEY:
VSPHERE_STORAGE_POLICY_ID: ""
VSPHERE_TLS_THUMBPRINT:
VSPHERE_INSECURE: false
DEPLOY_TKG_ON_VSPHERE7: false
ENABLE_TKGS_ON_VSPHERE7: false
```

```
#! -----
#! Node configuration
#! -----
```

```
SIZE:
CONTROLPLANE_SIZE:
WORKER_SIZE:
OS_NAME: ""
OS_VERSION: ""
OS_ARCH: ""
VSPHERE_NUM_CPUS:
VSPHERE_DISK_GIB:
VSPHERE_MEM_MIB:
VSPHERE_CONTROL_PLANE_NUM_CPUS:
VSPHERE_CONTROL_PLANE_DISK_GIB:
VSPHERE_CONTROL_PLANE_MEM_MIB:
VSPHERE_WORKER_NUM_CPUS:
VSPHERE_WORKER_DISK_GIB:
VSPHERE_WORKER_MEM_MIB:
```

```
ENABLE_MHC:
ENABLE_MHC_CONTROL_PLANE: true
ENABLE_MHC_WORKER_NODE: true
MHC_UNKNOWN_STATUS_TIMEOUT: 10m
MHC_FALSE_STATUS_TIMEOUT: 20m
```

```
#! -----
#! Image repository configuration
#! -----
```

```
TKG_CUSTOM_IMAGE_REPOSITORY:
TKG_CUSTOM_IMAGE_REPOSITORY_SKIP_TLS_VERIFY:
TKG_CUSTOM_IMAGE_REPOSITORY_CA_CERTIFICATE:
```

Once the yaml file is created as per the above sample configuration details, run the below command to initiate the management cluster creation.

```
tanzu management-cluster create --file path/to/cluster-config-file.yaml -v 10 &> tmp/mgmt.log &
```

– v 10 enables verbose mode output for the command executed

&> tmp/mgmt.log & - Redirects the output of the command executed to the log file specified.

This enables the command to run in the background, even in case of the session being disconnected.

4.3.1 Verifying Management Cluster Deployment

- a. On the bootstrap server, to list the available management clusters, run the below command:

```
tanzu login
```

or

```
tanzu cluster list --include-mangement-cluster
```

- b. To get the details of the management cluster:

```
tanzu management-cluster get
```

```
[sysadmin1301@DRDC-CKPRDSTS clusterconfig]$ tanzu mc get
NAME                                NAMESPACE STATUS  CONTROLPLANE WORKERS  KUBERNETES  ROLES  PLAN
drdoekprdmgmtcls                    tkg-system running 3/3      3/3      v1.22.9+vmware.1 management prod

Details:
NAME                                READY SEVERITY REASON SINCE MESSAGE
drdoekprdmgmtcls                    True  5m24s
ClusterInfrastructure - VSphereCluster/drdoekprdmgmtcls True  5m24s
ControlPlane - KubeadmControlPlane/drdoekprdmgmtcls-control-plane True  5m24s
3 Machines...
Workers
MachineDeployment/drdoekprdmgmtcls-md-0 True  5m26s
Machine/drdoekprdmgmtcls-md-0-6ff888f467-tn2zn True  5m26s
MachineDeployment/drdoekprdmgmtcls-md-1 True  5m26s
Machine/drdoekprdmgmtcls-md-1-774fb9c4dc-76x5w True  5m26s
MachineDeployment/drdoekprdmgmtcls-md-2 True  5m26s
Machine/drdoekprdmgmtcls-md-2-7cc5c959d5-fd5wf True  5m26s

Providers:
NAMESPACE NAME TYPE PROVIDERNAME VERSION WATCHNAMESPACE
capi-kubeadm-bootstrap-system bootstrap-kubeadm BootstrapProvider kubeadm v1.0.1
capi-kubeadm-control-plane-system control-plane-kubeadm ControlPlaneProvider kubeadm v1.0.1
capi-system cluster-api CoreProvider cluster-api v1.0.1
capv-system infrastructure-vmware InfrastructureProvider vmware v1.0.3
[sysadmin1301@DRDC-CKPRDSTS clusterconfig]$
```

- c. Retrieve the kubeconfig file for the management cluster to which to connect to.

If Identity provider has been configured for the cluster, a regular non-admin kubeconfig file could be exported without the `--admin` option which enables logging to the management cluster using the ldap credentials.

If identity provider is not configured on the management cluster, then an admin kubeconfig, could be used, which gives complete access to the management cluster

1. Generate “admin” kubeconfig file

```
tanzu management-cluster kubeconfig get -admin
```


2. Generate non-admin kubeconfig to authenticate with ldap credentials

```
tanzu management-cluster kubeconfig get --export-file [file name]
```

- d. Set the context to the management cluster

```
kubectl config use-context [context name]
```

- e. Execute the kubectl commands to examine the pods, deployments etc.. on management cluster

```
Kubectl get pods --KUBECONFIG=[path to kubeconfig file exported]
```

Note: We could either pass the kubeonfig file path in the above command or export the kubeconfig file as environment variable

4.4 Deploy and Configure Workload Cluster

Tanzu Kubernetes clusters or workload load clusters are the Kubernetes clusters in which the application workloads/pods run.

Prerequisites to deploy workload cluster

- a. DHCP scope is created for assignment of ip addresses to the nodes part of the workload cluster
- b. Obtain a static ip in same segment as DHCP scope created for workload cluster but not in part of same range of ips
- c. Required network port groups, folders, resource pools, created in vSphere environment
- d. Ensure the availability of the Kubernetes OVA in the vSphere environment to be used for deploying the workload cluster.
- e. Required CLI tools(Tanzucli, kubectl etc..) are installed on the bootstrap server and are of compatible versions as per the management cluster version
- f. Management cluster is deployed, and context is set to the respective management cluster

Most of the configurations for the workload cluster would be same as the management cluster. Hence, to deploy the workload cluster a copy of the manifest (yaml file) generated by UI or created manually for deploying the management cluster, could be used to modify it as per workload cluster requirements and create the cluster using this modified yaml file.

1. Once the cluster configuration yaml file has been created as per the requirements of the workload cluster to be deployed, execute the below command to create the workload cluster.

If using the copy of the yaml file generated while deploying management cluster, modify the below fields as per the workload cluster requirements

Field	Comments
CLUSTER_NAME:	should be unique for each workload cluster
VSPHERE_SERVER:	Same of all workload clusters (workload cluster hosted in same vCenter in different vSphere cluster)
VSPHERE_USERNAME:	Username to connect to vCenter server
VSPHERE_PASSWORD:	password for username used in above field
VSPHERE_DATACENTER:	Name of the Datacenter as displayed in vCenter server
VSPHERE_RESOURCE_POOL:	Name of the vSphere Cluster, unique for prod and non-prod environments
VSPHERE_DATASTORE:	Specify the vsan datastore for the respective cluster, selected in above field
VSPHERE_FOLDER:	Unique folder to be created for each kubernetes cluster to be deployed
VSPHERE_NETWORK:	Unique port group to be configured on the distributed switch/NSX-T
VSPHERE_CONTROL_PLANE_ENDPOINT:	Unique static IP for each kubernetes cluster to be used as KUBE-VIP
VSPHERE_TEMPLATE	kubernetes OVA template imported into vcenter, as per the TKG version
VSPHERE_TLS_THUMBPRINT:	TLS/SSL thumbprint of the vCenter server
VSPHERE_NUM_CPUS:	#to be used if same cpu config to be used for all control plane and worker nodes
VSPHERE_DISK_GIB:	#to be used if same disk config to be used for all control plane and worker nodes
VSPHERE_MEM_MIB:	#to be used if same memory config to be used for all control plane and worker nodes
VSPHERE_CONTROL_PLANE_NUM_CPUS:	CPU config for control plane nodes as per requirements
VSPHERE_CONTROL_PLANE_DISK_GIB:	Disk config for control plane nodes as per requirements
VSPHERE_CONTROL_PLANE_MEM_MIB:	Memory config for control plan nodes as per requirements
VSPHERE_WORKER_NUM_CPUS:	CPU config for worker plane nodes as per requirements

VSPHERE_WORKER_DISK_GIB:	Disk config for worker plane nodes as per requirements
VSPHERE_WORKER_MEM_MIB:	Memory config for worker plane nodes as per requirements

Below is the sample workload cluster configuration used for eduloan workload cluster deployed in prod environment

1. If creating the cluster configuration yaml file manually, as described in the management cluster deployment and configuration section, ensure all the required fields are part of the configuration file and are properly indented, prior to running the command to create the cluster.
2. Ensure you are in the correct management cluster context, prior to creating the workload cluster
3. Run the below command to create the workload cluster

```
tanzu cluster create --file [path of workload cluster yaml file]
```

Note: You may use -v 10, flag in the above command to enable verbose output of the deployment and also redirect the output to file and send the deployment to run in the background using, &> /tmp/cluster.log &

4.4.1 Connecting to Workload Cluster Deployed using Admin kubeconfig

Post successful deployment of the workload cluster, we could connect to the cluster using the admin kubeconfig to first validate the deployment i.e the status of the control plane nodes and worker nodes and the pod part of the cluster, prior to providing access to the devops users to deploy the application pods.

1. To get the admin kubeconfig for workload cluster, we execute similar command as we did previously for management cluster:

```
tanzu cluster kubeconfig get [clustername] --export-file [path to location to save the admin kubeconfig file] --admin
```

2. Once the admin kubeconfig has been generated, we could either export it as environment variable or specify it explicitly in the commands executed against the workload cluster

```
export KUBECONFIG=[ path to the admin kubeconfig file generated in above step]
```

or

```
kubectl get nodes -A KUBECOFNIG=[path to the admin kubeconfig file generated in above step]
```

4.4.2 Connecting to Workload Cluster Deployed using Non-admin Config

To provide workload cluster access to the devop users to authenticate with their ldap credentials and deploy the applications, perform the below steps:

1. Generate a user kubeconfig file

```
Tanzu cluster kubeconfig get [clustername] --export-file [path to save user kubeconfig file]
```

2. Create a clusterrole for users in the workload cluster by connecting to it using admin kubeconfig or list the inbuilt cluster roles

- Generate the admin kubeconfig of the workload cluster, if not already generated

```
Tanzu cluster kubeconfig get [clustername] --export-file [path to file location] --admin
```

- Connect to the workload cluster using the admin kubeconfig generated
- Switch context to the workload cluster admin context

```
kubectl config use-context [clustername]-admin@[clustername]
```

3. List the cluster roles available by default in the workload cluster

```
Kubectl get clusterroles
```

4. Create the cluster role binding using the role – cluster-admin, available on the workload cluster, to provide admin level access to the devops users on this specific cluster

Generic syntax of the yaml file to create the cluster role binding:

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
```

```
name: [name of the rolebinding]
subjects:
  kind: group
  name: [AD user group]
  apiGroup: rbac.authorization.k8s.io
roleRef:
  kind: ClusterRole
  name: cluster-admin [This one of the default role part of the inbuilt roles]
  apiGroup: rbac.authorization.k8s.io
```

5. Apply the yaml file created in above step on the workload cluster

Kubectl apply -f [rolebinding.yaml]

6. Provide the kubeconfig exported in step1 to the devops users, Once they login to the workload cluster using this kubecofig file, they would be redirected to the login screen, to enter their ldap credentials.

If the user is part of the AD group specified in the rolebinding file, the user would be authentication and authorized to execute the commands on the workload cluster.

4.4.3 Creating StorageClass in Workload Cluster

Storage Class object enables cluster users to dynamically create persistent volume claims (PVC) and Persistent Volume (PV) objects with different storage types and rules.

StorageClass objects include a provisioner field identifying the internal or external service plug-in that provisions PVs, and a parameters field that associates the Kubernetes storage class with storage options defined at the infrastructure level, such as VM Storage Policies in vSphere.

In this deployment, as we are using, vSphere environment as cloud provider to host the Kubernetes environment, we would be using vSphere Cloud Native Storage (CNS) storage type, which leverages, `csi.vsphere.vmware.com`, (Container Storage Interface) plug-in.

Prerequisites:

1. Create storage policy in vCenter for vSAN (Storage used for this deployment)
2. Deploy workload Cluster

Steps:

1. Create the required yaml file to deploy the storageclass in the workload cluster

Sample yaml file:

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: TKG-PRD-SC
parameters:
  storagePolicyName: [storagepolicy created in vCenter]
  datastoreurl: [path of the datastore]
  csi.storage.k8s.io/fstype: ext4
provisioner: csi.vsphere.vmware.com
reclaimPolicy: Retain
VolumeBindingMode: Immediate
```

2. Once the yaml file has been created, connect to the workload cluster using admin kubeconfig
3. Apply the yaml file created using the kubectl command
Kubectl apply -f [storageclass.yaml]
4. Once the storage class yaml file has been applied in the workload cluster, to confirm if it is applied correctly, run the below command
Kubectl get sc [name of the storage class]

This storage class created could be used by devops users in their application manifest file for creating persistent volumes for the application pods.

5. test the functionality of the storageclass created, we could create a test PVC on the workload cluster from the admin context, prior to handing over to the devops users.

To create the test pvc, the below, sample pvc.yaml file could be used:

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: test
spec:
  accessModes:
    - ReadWriteOnce
  storageClassName: [name of storageclass created previously using above steps]
resources:
  requests:
    storage: 1Gi
```

6. Create the test pvc by applying the above yaml file in the workload cluster

```
kubectl apply -f pvc.yaml
```

7. To confirm the successful creation of the pvc/pv

```
Kubectl get pvc,pv -A
```

8. You should see the state of the PVC as bounded and a PV created of the specified size in the yaml file.

5. Installing and Configuring the Packages/Extensions

A package/extension is a collection of related software that supports or extends the core functionality of the Kubernetes cluster in which the package is installed. Packages are grouped into *package* repositories in the Tanzu Command Line Interface (CLI)

Tanzu Kubernetes Grid includes two types of packages:

1. **Core packages**
2. **User-managed packages**

Core-Packages:

Automatically installed and managed by Tanzu Kubernetes Grid. These packages are located in the tanzu-core package repository.

Tanzu-core and Tanzu-standard package repositories are present in each of the cluster deployed.

The following table lists the core-packages and the cluster in which they are installed:

Package	Package repository	Package namespace	Cluster type	Package description
ako-operator	tanzu-core	tkg-system	Management	(vSphere only) Provides VMware NSX Advanced Load Balancer. This package is installed if NSX Advanced Load Balancer is enabled.
Antrea	tanzu-core	tkg-system	Management and workload	Enables pod networking and enforces network policies for Kubernetes clusters. If Antrea is selected as the CNI provider, this package is installed in every cluster.
Calico	tanzu-core	tkg-system	Management and workload	Enables pod networking and enforces network policies for Kubernetes clusters. If Calico is selected as the CNI provider, this package is installed in every cluster.
kapp-controller	tanzu-core	tkg-system	Management and workload	Manages packages. kapp-controller is installed in every cluster.
load-balancer-and-ingress-service	tanzu-core	tkg-system	Management and workload	(vSphere only) Provides L4+L7 load balancing for applications running in clusters created by Tanzu Kubernetes Grid; used for north-south traffic. This package is installed in every cluster if NSX Advanced Load Balancer is enabled.
metrics-server	tanzu-core	tkg-system	Management and workload	Provides Metrics Server. This package is installed in every cluster.
pinniped	tanzu-core	tkg-system	Management and workload	Provides user authentication. If identity management is enabled, this package is installed in every cluster. If you disable identity management when you create the management cluster, you can reenable it later. For more information, see Enable Identity Management in an Existing Deployment.
tanzu-addons-manager	tanzu-core	tkg-system	Management	Manages the lifecycle of core packages. tanzu-addons-manager is installed in every management cluster.
vsphere-cpi	tanzu-core	tkg-system	Management and workload	(vSphere only) Provides the vSphere Cloud Provider Interface. This package is installed in every cluster.
vsphere-csi	tanzu-core	tkg-system	Management and workload	(vSphere only) Provides the vSphere Cloud Storage Interface. This package is installed in every cluster.

To get the configuration file used to deploy the core packages in the cluster:

3. Get the secret of the add-on/package

Ex: To get the details of antrea package in a cluster named, test deployed in tkg-system namespace

```
kubectl get secret test-antrea-addon -n tkg-system
```

4. Locate the version tag for antrea

```
kubectl get clusters test -n tkg-system --show-labels
```

Make a note of the tanzuKubernetesRelease value in the output

```
kubectl get tkr TKR-VERSION
```

TKR-VERSION is the tanzuKubernetesRelease values from the previous command

5. In the output, locate the version tag under packages/core/antrea

```
imgpkg pull -b [registry]/packages/core/antrea:v0.13.3_vmware.1-tkg.1 -o antrea
```

6. To retrieve the configuration of the add-on deployed in the cluster by name, test in tkg-system namespace

```
kubectl get secret example-mgmt-cluster-antrea-addon -n tkg-system -o jsonpath={.data.values\\.yaml} | base64 -d > values.yaml
```

User-managed packages:

Manually installed and managed by administrators. These packages are located in the tanzu-standard package repository.

Here is the list of available user-managed packages that could be installed in workload clusters.

Function	Package	Package repository
Certificate management	cert-manager	tanzu-standard
Container networking	multus-cni	tanzu-standard
Container registry	Harbor	tanzu-standard
Ingress control	Contour	tanzu-standard
Log forwarding	fluent-bit	tanzu-standard

Monitoring	Grafana	tanzu-standard
Monitoring	Prometheus	tanzu-standard
Service discovery	external-dns	tanzu-standard

To install the user-managed plug-ins, Tanzu package plug-in will be used. This needs to be installed on the bootstrap server from which we would be connecting to the cluster and executing the commands.

To list all available add-ons, run the below command on the workload cluster:

```
tanzu package available list
```

In this deployment we would be deploying, cert-manager fluentbit, prometheus in each individual TKG clusters.

1. Cert-manager manages the certificates for all the add-ons/packages installed in the workload cluster.
2. Prometheus is an open-source monitoring and alerting tool, which collects metrics from TKG clusters and triggers alerts based on the rule expressions defined.
3. Fluentbit is a log processor and forwarder that enable collection of logs and data from different sources like, supervisor cluster, TKG cluster and forward to multiple endpoints like, Elastic search, Kafka, Splunk or Http endpoints.

5.1 Deploying Cert-Manager Extension

Cert Manager provides automated certificate management. It already runs by default in management clusters when cluster is created. On workload cluster, cert-manager needs to be deployed manually.

It is required by contour, external-dns, harbor, multus-cni, and prometheus and Grafana, add-ons and needs to be deployed prior to deploying any of these add-ons/packages on the workload clusters.

Steps:

1. Connect to the Kubernetes cluster using either admin config or user config with required roles and permissions
2. List all the available add-ons/packages on the workload cluster

tanzu package available list -A
3. Find the version of the available cert-manager package

tanzu package available list cert-manager.tanzu.vmware.com -A
4. Install the cert-manager package in cert-manager, namespace

```
tanzu package install cert-manager --package-name cert-manager.tanzu.vmware.com --  
namespace cert-manager --version 1.1.0+vmware.1-tkg.2 --create-namespace
```

5. To confirm successful installation of the cert-manager, either of the below commands could be executed.

```
tanzu package installed list -A
```

Output of the above command would list the version of cert-manager installed and the status as reconcile succeeded and namespace in which it is installed, if the installation is successful.

```
kubectl get apps -A
```

Output of this command displays the namespace, age of deployment and description with the Reconcile status. Reconcile succeeded under the description indicates a successful deployment of the cert-manager.

6. To view the running status of the pods associated with the cert-manager

```
kubectl get pods -n [namespace]
```

```
kubectl get pods -n cert-manager
```

Configuring backup using Velero

Velero is an open-source community standard tool for backing up and restoring Kubernetes cluster objects and persistent volumes, which we would be leveraging in this deployment for backup and restore of all the clusters deployed.

In case of issues with the cluster or crash of the cluster, the backup could be used to restore the contents of the backup to a new cluster along with the extensions/add-ons and internal API objects.

Note: You must create a new cluster to restore to; you cannot restore a cluster backup to an existing cluster.

Prerequisites:

- Deploy the workload cluster
- Install velero CLI on local client machine
- Create S3 buckets to save the backup of the clusters

Steps:

1. Navigate to the TKG 1.5.4 download page

<https://customerconnect.vmware.com/en/downloads/details?downloadGroup=TKG-140&productId=988&rPid=82536>

2. Download the velero CLI, as per the operating system on the client machine.
3. Unzip the tar file downloaded and move it to /usr/local/bin location and make it an executable file.

```
chmod +x /usr/local/bin/velero
```

4. Connect to the workload cluster on which to install the velero server component
5. Execute the below command to install the velero server component on the cluster

- a. Obtain the name of the s3 bucket/object store created for the cluster
- b. Get the credentials required to connect to the object store and save the backups

Velero install --image [path to image in the registry] --plugins [path to velero plugin for aws in the registry] --use-restic --provider aws --bucket [bucket name] --secret-file [path to location of user credentials to access the bucket] --use-volume-snapshot=false --backup-location-config region=us-east-1,s3ForcePathStyle="true",s3Url=[url of the object store],insecureSkipTLSVerify=true

6. Apply the restic yaml file to integrate with velero

```
Kubectl apply -f [path to restic yaml file]
```

7. To verify the velero deployment

```
Kubectl logs deployment/velero -n velero
```

8. To create the backup of the cluster

- a. Connect to the cluster to be backed up
- b. Ensure velero cli is installed on the machine from which the connection was made to the cluster.
- c. Run the below command to take backup of the cluster

Velero backup create [cluster name]

- d. To verify the backup created from cli

Velero backup describe [name of the backup]

We could also login to the object store UI interface and check for the backup objects created in the s3 bucket.

6. Generic Day-2 Operation Commands

1. List available Management clusters

- a. From bootstrap machine, run the below command

```
tanzu login
```

This would list all the available management clusters. Select the required management cluster to get the details of that cluster

2. Get the details of the selected management cluster

```
tanzu management-cluster get
```

3. To scale the control planes node or worker nodes of management cluster

```
tanzu cluster scale [MANAGEMENT CLUSTER-NAME] --controlplane-machine-count 3 --worker-machine-count 4 --namespace [namespace]
```

Note:

- a. Either of the controlplane machine count or worker node count could be scaled or both using the above command.
- b. The total count of control plane should always be an odd number

4. Deleting Management Clusters

- a. Delete management cluster with no workload clusters managed by it

Login to the cluster to be deleted

```
tanzu management-cluster delete [cluster name] --yes
```

--yes: will skip validation for deleting the cluster

- b. If there are workload cluster managed by the management cluster to be deleted

- Delete all the workload clusters first by connecting to management cluster, prior to deleting the management cluster
 - a. `tanzu cluster delete [workload cluster name]`
 - b. `tanzu management-cluster delete [management cluster name]`
- Force delete the management cluster, which would delete all the workload cluster as well
 - a. `tanzu management-cluster delete [management cluster name] --force`

Note: Use this option, only if option1 does not work

7. To list the workload cluster

```
tanzu cluster list
```

Here is the possible status for the clusters:

Clusters can be in the following states:

- **creating:** The control plane is being created
- **createStalled:** The process of creating control plane has stalled
- **deleting:** The cluster is in the process of being deleted
- **failed:** The creation of the control plane has failed
- **running:** The control plane has initialized fully
- **updating:** The cluster is in the process of rolling out an update or is scaling nodes
- **updateFailed:** The cluster update process failed
- **updateStalled:** The cluster update process has stalled
- **No status:** The creation of the cluster has not started yet

8. To list the management clusters along with workload clusters

```
tanzu cluster list --include-management-cluster
```

9. To create the admin kubeconfig for workload cluster to connect to cluster when identity management is not enabled for the cluster

```
tanzu cluster kubeconfig get [cluster name] --admin
```

10. To export admin kubeconfig to a file for workload cluster, to be shared with other administrators to connect to the cluster with admin context

```
tanzu cluster kubeconfig get [cluster name] --admin --export-file [path to save the file]
```

11. To export a non-admin kubeconfig file to connect to cluster using the Ldap authentication

```
tanzu cluster kubeconfig get [cluster name] --export-file [path to save the file]
```

12. To connect to the workload cluster using non-admin kubeconfig and authenticating with ldap

```
kubectrl get pods -A --kubeconfig [non-admin kubecofig]
```

This would redirect the user to a browser on the local machine, to login to the cluster using the ldap username and password

Once the authentication is successful and then appropriate cluster role is mapped for the user, the output of the above command would be displayed on the screen.

If user receives the error: "Error from server (Forbidden): pods is forbidden: User "<user>" cannot list resource "pods" in API group "" at the cluster scope:", Ensure the required cluster role and role binding is created for the user on the cluster.

If users jumphost/local machine does not have browser, perform one of the following tasks, prior to running the commands against the cluster to authenticate

- a. Export the below environment variable

```
export TANZU_CLI_PINNIPED_AUTH_LOGIN_SKIP_BROWSER=true
```

This would prevent the login process from opening the login prompt in local browser

- b. Edit the non-admin kubeconfig to add the switch: -- -skip-broswer, save the file and close it

Now execute the command to authenticate to the cluster using non-admin kubeconfig

```
kubectrl get pods -A --kubeconfig my-cluster-credentials
```

- login url would be displayed on the terminal, copy this url to a machine which has browser and provide the user credentials.
- You will see a message that the identity provider could not send the authentication code because there is no localhost listener on your workstation.
- Copy the URL of the authenticated session from the URL field of the browser
- On the machine on which the authentication was initiated i.e the machine without the browser, perform the below step
- Open the terminal or create a duplicate session of the existing terminal session and run the command:


```
curl -L '<copied_URL>' # URL copied from above step
```

- Once the above command is successfully executed, you would see the following message on the terminal

```
you have been logged in and may now close this tab
```

- Close the terminal now and execute the command to list the pods again, since the authentication was successful, it would display the output of the command

Note: For the output to be displayed, ensure the user is provided with appropriate cluster role and cluster role binding

13. Connecting to the workload cluster as a standard devops user from local desktop

- a. Obtain the management cluster kube-vip which manages the workload cluster and the workload cluster details

```
tanzu login --endpoint https://Mgmt cluster-Kube-vip:6443 --name [workload cluster]
```

- b. Login to the cluster using the ldap credentials
- c. Obtain the kubeconfig of the workload cluster

```
tanzu cluster kubeconfig get [workload cluster name]
```

- d. Switch context to workload cluster

```
kubect1 config use-context tanzu-cli-[kubeconfig]@[workload cluster]
```

14. To list the worker nodes and control plane nodes of the cluster

```
Kubect1 get nodes -o wide
```

15. To list all the pods running in the cluster

```
Kubect1 get pods -A
```

16. Horizontal scaling of the workload cluster

```
tanzu cluster scale cluster_name --controlplane-machine-count 5 --worker-machine-count 6
```

Note: If cluster is deployed in a specific namespace, append: -n [namespace] to above command

17. Shutdown of Tanzu Kubernetes grid instance

- a. Obtain the details of the etcd database of management cluster

```
kubectl --kubeconfig /etc/kubernetes/admin.conf get pods `kubectl --kubeconfig /etc/kubernetes/admin.conf get pods -A | grep etc | awk '{print $2}'` -n kube-system -o=jsonpath='{.spec.containers[0].command}' | jq
```

- b. For each control plane node of part of the cluster, run the below commands to backup the etcd database

```
etcdctl snapshot save LOCAL-BACKUP --endpoints=ENDPOINTS --cacert=CA --cert=CERT --key=KEY
```

- c. Verify the backup

```
etcdctl --write-out=table snapshot status LOCAL-BACKUP
```

- d. To collect the below cluster information:

```
tanzu cluster list --include-management-cluster > CLUSTER-INFO-1
```

```
kubectl config get-contexts >> CLUSTER-INFO-1
```

```
kubectl config use-context tkg-mgmt-vsphere-20211111074850-admin@tkg-mgmt-vsphere-20211111074850 >> CLUSTER-INFO-1
```

```
kubectl get nodes -o wide >> CLUSTER-INFO-1
```

```
kubectl config use-context mycluster1-admin@mycluster1 >> CLUSTER-INFO-1
```

```
kubectl get nodes -o wide >> CLUSTER-INFO-1
```

```
cat CLUSTER-INFO-1
```

- e. Drain all application pods from the workload clusters

- f. Stop all virtual machines on vCenter in the following order:

- Shut down management cluster control plane nodes
- Shut down management cluster worker nodes
- Shut down workload cluster control plane nodes
- Shut down workload cluster worker nodes

- g. Start all virtual machines on vCenter in the following order

- Start workload cluster control plane nodes
 - Start workload cluster worker nodes
 - Start management cluster control plane nodes
 - Start management cluster worker nodes
- h. Run the cluster information commands as listed in step: “d”
- i. Compare the output of cluster information in step: ‘d’ and step ‘h’

18. Upgrading Tanzu Kubernetes Grid Instance

- a. Upgrade the Tanzucli version to the latest version
 - Delete the tkg-compatibility file in the location (unless changed)

`~/.config/tanzu/tkg/compatibility/tkg-compatibility.yaml`
 - Install the latest tanzucli and kubectl version on the bootstrap server where the previous version was installed
- b. Download and import the latest Kubernetes OVA image as per the Kubernetes grid version into the vCenter Server
- c. Assign the user created with appropriate permissions to deploy the cluster, onto the imported ova file
- d. Upgrade the management clusters

Note:

Ensure the required images for the latest version are made available in the registry used by the

- List the management clusters

`tanzu login`

`tanzu management-cluster get`

 or

`tanzu cluster list --include-management-cluster`

- Run the upgrade command

```
tanzu management-cluster upgrade
```

Note:

There is a default timeout of 30 minutes for the execution of upgrade process. If it execution times-out, you may increase the timeout value to value greater than 30minute as per requirement

```
tanzu management-cluster upgrade --timeout 45m0s
```

- Once the upgrade is completed, check the version by listing the management cluster

```
tanzu cluster list --include-management-cluster
```

- Once the cluster is upgraded, push the latest add-on/package repository to the cluster

```
tanzu package repository add tanzu-standard --url REPOSITORY-URL -n  
tanzu-package-repo-global --create-namespace
```

e. Upgrade the workload clusters

- Connect to the management cluster which manages the target cluster
- Run the below command to get the list of available Kubernetes version which could be used for the upgrade

```
tanzu kubernetes-release get
```

- Run the upgrade command

```
tanzu cluster upgrade CLUSTER-NAME
```

Note: You cannot skip minor versions when upgrading your `tkr` version

To upgrade to specific version of Kubernetes

`tanzu cluster upgrade [cluster-name] --tkr [Kubernetes version]`

19. Management cluster log's location

`kubectl logs deployment.apps/capv-controller-manager -n capv-system manager --
kubeconfig </path/to/kubeconfig>`

Note: kubeconfig location could be found in the output of the, `tanzu management-cluster create` command

20. To check the workload cluster deployment logs

Set the context to the management cluster and check the logs for the capv-controller manager deployment in management cluster, which is responsible for deployment of workload clusters

`kubectl logs deployments/capv-controller-manager -n capi-system manager`

21. To delete failed instance of management cluster deployment

- List the management cluster and check for the status, if the cluster is listed, delete using below command

`tanzu management-cluster delete [management cluster name]`

Note: If the kind cluster is not deleted post deletion of the management cluster, perform the below steps

`docker ps -a
docker kill container_ID`

- If the management cluster is not listed, delete the kind cluster

`kind get clusters`

`kind delete cluster --name [cluster name from above command
output]`

- In the vCenter, check for any VM's left behind part of the management cluster creation process, if any found, power-off the VM and delete it.

22. SSH to the cluster control plane and worker nodes

```
ssh -i [rsa key] capv@[node ip]
```